Electronic Colloquium on Computational Complexity, Report No. 41 (2019)

Quantum hardness of learning shallow classical circuits

Srinivasan Arunachalam[*] Alex B. Grilo [†] Aarthi Sundaram [‡]

Abstract

In this paper, we study the quantum learnability of constant-depth classical circuits under the uniform distribution and in the distribution-independent framework of PAC learning. In order to attain our results, we establish connections between quantum learning and quantum-secure cryptosystems. We then achieve the following results.

1. **Hardness of learning AC^0 and TC^0 under the uniform distribution.** Our first result concerns the concept class TC^0 (resp. AC^0), the class of constant-depth and polynomial-sized circuits with unbounded fan-in majority gates (resp. $\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$ gates). We show that if there exists no quantum polynomial time (resp. sub-exponential time) algorithm to solve the Learning with Errors (LWE) problem, then there exists no polynomial time quantum learning algorithm for TC^0 (resp. AC^0) under the uniform distribution (even with access to quantum membership queries). The main technique in this result uses explicit pseudo-random generators that are believed to be quantum-secure to construct concept classes that are hard to learn quantumly under the uniform distribution.

2. **Hardness of learning TC^0_2 in the PAC setting.** Our second result shows that if there exists no quantum polynomial-time algorithm for the LWE problem, then there exists no polynomial-time quantum-PAC learning algorithm for the class TC^0_2, i.e., depth-2 TC^0 circuits. The main technique in this result is to establish a connection between the quantum security of public-key cryptosystems and the learnability of a concept class that consists of decryption functions of the cryptosystem.

This gives a strong conditional negative answer to one of the "*Ten Semi-Grand Challenges for Quantum Computing Theory*" raised by Aaronson [Aar05], who asked if AC^0 and TC^0 can be PAC-learned in quantum polynomial time.

[*]Center for Theoretical Physics, MIT. Funded by the MIT-IBM Watson AI Lab under the project *Machine Learning in Hilbert space*. arunacha@mit.edu

[†]CWI and QuSoft, Amsterdam, The Netherlands. Supported by ERC Consolidator Grant 615307-QPROGRESS. alexg@cwi.nl

[‡]Joint Center for Quantum Information and Computer Science, University of Maryland, USA. Supported by the Department of Defense. aarthims@umd.edu

ISSN 1433-8092

1 Introduction

Machine learning is a diverse field of research with many real-world applications and has received tremendous attention in the past decade. From a theoretical perspective, since the seminal paper of Valiant [Val84], there has been a lot of theoretical effort in considering different learning models that formalize what we mean by *learning* and understanding which problems can (or cannot) be *efficiently* learned within these models.

In recent years, due to the considerable development of quantum computation (both on the theoretical and experimental fronts), there has been an increased focus on understanding the tasks for which quantum computers can offer speedups. Machine learning tasks have emerged as a candidate in this respect. To this end, results in *quantum learning theory* aim to identify learning problems for which quantum computers *provably* provide a (significant) advantage.

More concretely, in learning theory, the goal is to devise a learning algorithm (or learner) for a set of functions which is called a *concept class*. The functions in the concept class $\mathcal{C}$ are referred to as *concepts*. In this paper, we will consider (without loss of generality) concepts that are Boolean functions $c : \{0,1\}^n \to \{0,1\}$. The learner is provided with *examples* of the form $(x, c(x))$, where c is an *unknown* concept lying in $\mathcal{C}$ and x is picked uniformly at random from $\{0,1\}^n$ and the goal of the learner is to *learn* c, i.e., it should output a hypothesis h that is close to c.[1] We say that a *learner* $\mathcal{A}$ *learns* $\mathcal{C}$, if for every $c \in \mathcal{C}$, $\mathcal{A}$ learns c. In learning theory, the intent is to devise *efficient* learning algorithms for an interesting concept class $\mathcal{C}$, i.e., the learner should use few examples and not too much time in order to learn $\mathcal{C}$.

In *quantum* learning theory, the goal is still to learn concept classes $\mathcal{C}$, but now with the access to quantum resources. Bshouty and Jackson [BJ99] introduced a quantum learning model wherein the learner is a *quantum* algorithm and is provided with *quantum examples* of the form

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x, c(x)\rangle$$

for some concept $c \in \mathcal{C}$. In order to see why quantum examples generalize classical examples, observe that a quantum learner could choose to measure the quantum example in the computational basis, which results in a classical example $(x, c(x))$ for a uniformly random $x \in \{0,1\}^n$. The advantage of quantum learning usually comes from the fact that one can perform arbitrary unitary operations on these quantum examples, enabling one to improve sample or time complexity for learning the concept class $\mathcal{C}$.

The first example of a quantum advantage for a learning problem was showed by Bernstein and Vazirani [BV97]. They showed how to learn the concept class of linear functions $\mathcal{C} = \{c(x) = \sum_i s_i x_i \mod 2 : s \in \{0,1\}^n\}$ with a constant number of quantum examples. Classically, in order to learn this concept class efficiently, it is necessary and sufficient to obtain $\widetilde{\Theta}(n)$ examples. With these many examples a classical learner can use Gaussian elimination and learn the unknown target concept in polynomial time.

Subsequently, Bshouty and Jackson [BJ99] showed that the class of Boolean functions that can be represented as polynomial-sized DNF formulas can be learned in quantum polynomial

[1] More formally, this is referred to as the uniform-distribution learning setting. In learning theory there are several variations of this model that we skip for the sake of simplicity. See Section 1.1 for a brief introduction and Section 2.4 for more details.

time.[2] One crucial ingredient for their quantum learning algorithm was the ability to perform Fourier sampling using quantum examples (which we discuss in Section 1.2). Classically, Verbeurgt [Ver90] showed that DNFs can be learned in quasi-polynomial time using classical examples and this has remained the state-of-the art for the last thirty years! This emphasizes the power of quantum examples for learning. There have been many other instances (which we discuss in Section 1.2) where quantum examples give an advantage for quantum learning algorithms.

A natural follow-up question to the quantum-efficient learnability of DNF formulas, i.e., depth-2 circuits, is:

> Can the concept class of *shallow, i.e., constant-depth, classical circuits* be learned more *efficiently* using *quantum resources* as compared to classical resources?

In particular, are there efficient (i.e., polynomial-time) quantum learning algorithms for Boolean functions that can be represented by AC^0 circuits, i.e., constant-depth circuits with *unbounded* fan-in $\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}$ gates? More ambitiously, can we quantum-efficiently learn TC^0, i.e., constant-depth circuits with majority gates.[3] This question was raised by Aaronson [Aar05] as one of the *"Ten Semi-Grand Challenges for Quantum Computing Theory"*.

In this work we address this question and give evidence that efficient quantum algorithms *do not exist* for learning AC^0 and TC^0. More concretely, under the assumption that the Learning with Errors problem (LWE) [Reg09] cannot be solved in polynomial time by quantum computers, the class of TC^0 functions cannot be learned in quantum polynomial time. Also, under the less-standard assumption that LWE cannot be solved in sub-exponential time by quantum computers, we show that polynomial-time learning algorithms for AC^0 do not exist. LWE is one of the leading candidates for public-key post-quantum cryptography and it is believed to be hard to solve, even for quantum computers. For instance, the current best-known quantum algorithms for it run in exponential time [BKW03, AG11].

1.1 Learning models

In this section, we first recall the definitions of the classical and quantum learning models. For a detailed introduction to these models, we refer the reader to Section 2.4.

Classical learning model. In the Probably Approximately Correct (PAC) model of learning [Val84], a *concept class* $\mathcal{C}$ is a subset of Boolean functions, i.e., $\mathcal{C} \subseteq \{c : \{0,1\}^n \to \{0,1\}\}$ and every element $c : \{0,1\}^n \to \{0,1\}$ in $\mathcal{C}$ is referred to as a *concept* . The goal of the learning algorithm $\mathcal{A}$ is to learn an unknown *target concept* $c \in \mathcal{C}$ given *labeled examples* of the form $(x, c(x))$ where x is drawn from an *unknown* distribution $D : \{0,1\}^n \to [0,1]$. We say that a learning algorithm $\mathcal{A}$ *learns* $\mathcal{C}$ if it satisfies the following: for every $c \in \mathcal{C}$ and every distribution $D : \{0,1\}^n \to [0,1]$, with probability at least $2/3$, $\mathcal{A}$ outputs a hypothesis $h : \{0,1\}^n \to \{0,1\}$ that satisfies $\Pr_{x\sim D}[h(x) = c(x)] \geq 1 - \varepsilon$. The advantage of the learner over a random guess is $\beta := \frac{1}{2} - \frac{\varepsilon}{2}$ and β is called the *bias* of the learner. The learner $\mathcal{A}$ *properly* learn the concept class $\mathcal{C}$ if its output hypothesis is always in the concept class, i.e., $h \in \mathcal{C}$. Else, $\mathcal{A}$ is an improper learner for $\mathcal{C}$. Similarly, $\mathcal{A}$ is a *weak learner* if $\beta = n^{-c}$ for some constant $c > 0$. In this paper all our lower bounds will be for *weak improper learners* (which makes the lower bounds stronger).

[2]A DNF formula is a disjunction of conjunctions of variables and their negations.

[3]A majority gate on n bits outputs 1 if $\lfloor n/2 \rfloor + 1$ bits evaluate to 1 and 0 otherwise.

The sample complexity of a learning algorithm $\mathcal{A}$ is the worst-case number of labeled examples it uses and the time complexity of $\mathcal{A}$ is the worst-case running time (where the worst-case is taken with respect to the hardest concept $c \in \mathcal{C}$ and distribution D). The sample/time complexity of a concept class $\mathcal{C}$ is the sample/time complexity of the most efficient learning algorithm for $\mathcal{C}$.

In this work, we also consider learning models that relax the PAC learning framework in two ways. First we allow the learner to make *membership queries* to a target concept c, i.e., $\mathcal{A}$ is allowed to ask "what is $c(x)$" for an arbitrary x of its choice. Second, instead of the learner succeeding under *every* distribution D, we consider the learnability of $\mathcal{C}$ when D is fixed and known to the learner. We say $\mathcal{A}$ learns *PAC learns* $\mathcal{C}$ under D with membership queries if: for every $c \in \mathcal{C}$, with probability $\geq 2/3$, $\mathcal{A}$, takes labeled examples and makes membership queries, and outputs a hypothesis h such that $\Pr_{x\sim D}[h(x) \neq c(x)] \leq \varepsilon$.

Quantum learning model. Bshouty and Jackson [BJ99] introduced the model of quantum-PAC learning, which naturally generalizes the classical PAC model. Here, a *quantum* learning algorithm $\mathcal{A}$ has to learn the unknown concept $c \in \mathcal{C}$ given *quantum* examples of the form

$$\sum_x \sqrt{D(x)}|x, c(x)\rangle,$$

where $D : \{0,1\}^n \to [0,1]$ is an unknown distribution. The goal of the quantum learner and the notion of sample and time complexities are analogous to the classical model.

As described in the classical setting, we also consider the model where the quantum learning algorithm is given access to an oracle $O_c : |x, b\rangle \to |x, b \oplus c(x)\rangle$ that allows it to make *quantum membership queries*. Additionally, instead of requiring the learner to succeed for *all* distributions D, we also consider quantum learners that learn $\mathcal{C}$ under a fixed distribution D. For quantum learning algorithms, a natural choice of D is the uniform distribution over $\{0,1\}^n$, and in this case a quantum example is given by $\frac{1}{\sqrt{2^n}}\sum_x |x, c(x)\rangle$. We discuss the utility of such examples in the next section. We say a learner *uniform quantum-PAC learns* $\mathcal{C}$ with membership queries if: for every $c \in \mathcal{C}$, with probability $\geq 2/3$, $\mathcal{A}$, uses uniform quantum examples and quantum membership queries to output a hypothesis h such that $\Pr_{x\sim D}[h(x) \neq c(x)] \leq \varepsilon$.

1.2 Strengths of quantum examples under the uniform distribution

One of the main tools in quantum learning theory, when considering learning under the uniform distribution, is the ability to efficiently perform *Fourier sampling*. In order to explain it, we first introduce the following. For a Boolean function $c : \{0,1\}^n \to \{-1,1\}$, the Fourier coefficients of c are given by $\widehat{c}(S) = \frac{1}{2^n}\sum_S c(x)(-1)^{x\cdot S}$. The *Fourier distribution* $\{\widehat{c}(S)^2 : S \in \{0,1\}^n\}$ is given by the squared Fourier coefficients of c and they satisfy $\sum_S \widehat{c}(S)^2 = 1$. Fourier sampling refers to the task of *sampling* from the Fourier distribution $\{\widehat{c}(S)^2\}_S$.

An advantage of having uniform quantum examples is that, using standard ideas from quantum information theory, a quantum learner can efficiently perform the operation

$$\frac{1}{\sqrt{2^n}}\sum_x |x, c(x)\rangle \to \sum_S \widehat{c}(S)|S\rangle$$

given $O(1)$ copies of $\frac{1}{\sqrt{2^n}}\sum_x |x, c(x)\rangle$. Measuring the resulting state allows a quantum learner to obtain a sample from the Fourier distribution. Hence, using uniform quantum examples, one can sample from the Fourier distribution $\{\widehat{c}(S)^2\}_S$. Classically, we do not know how to perform this sampling process (or even approximately sample) efficiently, since the Fourier coefficients

$\widehat{c}(S)$ depends on 2^n values of x. Therefore, one avenue to obtain quantum speedups with uniform quantum examples arises from the use of Fourier sampling.

Indeed, quantum Fourier sampling has been profitably used in many applications. This idea was first used in the aforementioned paper of Bernstein and Vazirani [BV97], where they observe that the Fourier support of linear functions is concentrated on a single point. Therefore, unknown linear functions can be learned using just one uniform quantum example. Fourier sampling was later used by Bshouty and Jackson [BJ99] to show that DNFs can be learned quantum-efficiently. A classical analogue of this question is a long-standing open question.[4] Kanade et al. [KRS18] extended the result of Bshouty and Jackson and showed how to learn DNFs quantum-efficiently even under product distributions.

In fact, a notorious bottleneck in classical learning theory is obtaining a polynomial time learning algorithm for the class of $O(\log n)$-juntas[5] which are a subset of polynomial-sized DNFs. By contrast, Atıcı and Servedio [AS09] showed that $O(\log n)$-juntas can be learned quantum-efficiently under the uniform distribution. In fact the time-efficient learnability of $O(\log n)$-juntas can be used to time-efficiently quantum learn the concept class NC^0 under the uniform distribution (for a proof of this, see Appendix A).[6] Subsequently, Arunachalam et al. [ACLW18] showed that the concept class of k-Fourier-sparse Boolean functions (which includes both $(\log k)$-juntas and linear functions) can be learned query-efficiently given quantum examples.

One thing common to all these results was the application of the Fourier sampling to learn a concept class under the uniform distribution, which was key for the large quantum speedup.

Related works. In the context of learning, apart from the uniform distribution setting, some works have focused on understanding the power of quantum learning under *arbitrary distributions* [SG04, AS05, AW18]. In particular, [AW18] showed that quantum examples do not provide an advantage over classical examples for PAC learning.

Recently, both shallow circuits and the Learning with Errors problem have been used in different contexts to understand the capabilities of quantum computation. Grilo et al. [GKZ18] showed that polynomially many *quantum examples* suffice to solve the LWE problem in quantum polynomial time, while this remains classically a hard problem when given just *classical* examples. Bravyi et al. [BGK18] exhibited a problem which can be solved using shallow quantum circuits but requires logarithmic depth classical circuits (with bounded fan-in) to solve. Bene Watts et al. [BKST19] improved their result by removing the "bounded fan-in" assumption. In another context, under the assumption that the Learning with Errors problem is hard for quantum computers (given just classical examples), Mahadev [Mah18] demonstrated a classical protocol to classically verify the result of an efficient quantum computation.

1.3 Our results

In this paper we address two natural questions that arise from the work of Bshouty and Jackson [BJ99], which showed the quantum-efficient learnability of depth-2 circuits.

The first question is, can the work of [BJ99] be extended to learn depth-3 circuits, or more generally, constant-depth polynomial-sized circuits (i.e., TC^0 and AC^0) in quantum polynomial time?

[4]With classical membership queries, DNF formulas can be learned in classical polynomial time [Jac97].

[5]A k-junta is a Boolean function on n bits, whose output only depends on k out of the n input bits.

[6]NC^0 is the concept class of constant-depth circuits consisting of fan-in 2 AND, OR, NOT gates.

Classically, in a seminal result, Linial, Mansour and Nisan [LMN93] constructed an $n^{O(\log n)}$-time learning algorithm for AC^0 by approximately learning the Fourier spectrum of an unknown AC^0 circuit. Subsequently, Kharitonov [Kha93] showed that their learning algorithm is optimal assuming that factoring cannot be solved in quasi-polynomial time. Since factoring can be solved in *quantum* polynomial time with Shor's algorithm [Sho97], the lower bound of Kharitonov doesn't apply to quantum learners. Moreover, since Fourier sampling is easy quantumly (under the uniform distribution), it seems plausible that one could efficiently learn important properties of the Fourier spectrum of AC^0 circuits (similar to the work of Linial et al. [LMN93]). This could possibly result in efficient quantum learning algorithms for AC^0 under the uniform distribution (similar to many results discussed in the previous section).

The second question is, can the work of [BJ99] be extended to the class of depth-2 *threshold* circuits (known as TC^0_2), i.e., can TC^0_2 be learned quantum-efficiently? We notice that these threshold circuits (i.e., $\mathsf{TC}^0, \mathsf{TC}^0_2$) are *practically* very relevant since constant-depth polynomial-sized feed-forward neural networks with weights (of the neurons) bounded by some polynomial in the input size, can be implemented as circuits in TC^0. If there were efficient quantum algorithms for learning TC^0, then it is plausible that quantum computers could have given an enormous advantage in approximately learning the weights for neural networks.

In this paper, we give a conditional negative answer to both questions. In particular, we show that under the assumptions that support the security of current post-quantum cryptography: (i) TC^0 cannot be learned efficiently by quantum computers under the uniform distribution and AC^0 cannot be learned in sub-exponential time on quantum computers; (ii) TC^0_2 cannot be PAC learned efficiently by quantum computers. We summarize these results in the table below.[7]

No polynomial-time learner in this model	For the complexity class	Assuming
Uniform-distribution PAC (with membership queries)	AC^0	No sub-exponential time algorithm for LWE
	TC^0	$\mathsf{LWE} \notin \mathsf{BQP}$
Distribution-free PAC	TC^0_2	$\mathsf{LWE} \notin \mathsf{BQP}$

These results give a strong conditional refutation to a question of Aaronson [Aar05]. Aaronson asked if, TC^0 and AC^0 can be *quantum-PAC learned* in polynomial time? Our first result gives a conditional negative answer for the case when we fix the *uniform distribution* and we additionally allow the learner to make quantum membership queries. Our second result gives a conditional refutation to the PAC learnability of TC^0 even when restricted to depth-2 threshold circuits.

In order to achieve our results, we follow a strategy proposed by Valiant [Val84], who showed the hardness of *proper* learning the class of polynomial-sized circuits based on the security of cryptographic objects. This strategy was subsequently improved upon to give conditional *improper* learning lower bounds and used to prove the classical hardness of various concept classes [Kha92, Kha93, KV94, KS09]. These results have the following structure: assuming there exists an efficient learning algorithm for some concept class $\mathcal{C}$, there exists an adversary that is able to break some cryptographic construction using the learning algorithm as a subroutine. Here, the adversary provides the resources to the learning algorithm based on the cryptographic primitive it is trying to break. This implies that if the cryptographic construction is secure, then such a learning algorithm cannot exist.

[7] BQP, or *bounded-error quantum polynomial time*, is the class of decision-problems that be solved in polynomial time on a quantum computer with bounded-error.

In this paper, we quantize these well-studied classical proof-of-hardness techniques. The difficulties in quantizing such results are three-fold. First, many of the classical hardness of learning results rely on cryptographic primitives whose security is based on the hardness of factoring. As stated previously, this would not hold in the quantum regime due to Shor's quantum polynomial-time factoring algorithm [Sho97]. Second, the fact that adversaries can efficiently create classical examples from some distribution D does not imply that *quantum examples* can be created according to the same distribution. An important issue we run into in this case is solving the index-erasure problem, which is known to be hard to solve on quantum computers [AMRR11, LR19]. Finally, some of the hardness results implicitly use the fact that the learning algorithm they are considering is classical and the proof techniques do not follow through in the quantum setting. For example, Kharitonov [Kha93] uses collision arguments to bound the amount of information retrieved by the learner, but this approach does not work quantumly. We discuss these issues in further detail in the next section.

In subsequent sections, we delineate the connections between the hardness of quantum learning and the security of certain cryptographic primitives – specifically, quantum-secure pseudo-random generators and public-key encryption schemes. Next, we sketch how to use these connections to show hardness of quantum learning for some interesting concept classes.

1.3.1 Pseudo-random generators vs. quantum learning

Pseudo-random generators (PRG) are cryptographic objects that "stretch" random strings. More concretely, a PRG is a deterministic algorithm G that takes an n-bit string as input and outputs an $\ell(n)$-bit string for $\ell(n) > n$, which we refer to as the pseudo-random string. Further, G satisfies the condition that: if the seed x is picked uniformly at random from $\{0,1\}^n$, then no efficient adversary can distinguish $G(x)$ from uniformly random $\ell(n)$-bit strings, with negligible advantage over a random guess.[8] If polynomial time *quantum* adversaries cannot distinguish pseudo-random strings from uniformly random strings, then we say that the PRG is *quantum-secure*. We informally state our first result below (see Theorem 4.2 for a full statement).

Result 1 *If there is a quantum-secure PRG G, then there exists a concept class $\mathcal{C}_G$ (i.e., $\mathcal{C}_G$ depends on G) such that $\mathcal{C}_G$ does not have an efficient uniform weak quantum-PAC learner with membership queries.*

Kharitonov [Kha93] established the connection between PRGs and classical learning by constructing a circuit class such that the PRG is computed by the circuit class. He then proceeds to show that if a learning algorithm for such a concept class exists, then it is possible to break the PRG. Our main technical contribution here is to quantize Kharitonov's result. While the structure of our proof is largely inspired by his proof, we would like to reiterate that the quantization is not straightforward. For instance, the crux of his proof, which is based on determining the probability of collision in the classical examples, does not make sense in the quantum regime. Clearly, each quantum example contains information about every $x \in \{0,1\}^n$ but efficiently accessing this information simultaneously for all x is information-theoretically impossible. Furthermore, unlike with its classical counterpart, we need to account for a quantum learner's added ability to sample from the Fourier distribution of the target concept c.

We now sketch a proof of Result 1. For a PRG $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$, where $\ell(n)$ is a polynomial in n, define a concept class

$$\mathcal{C}_G = \{c_z : \{0,1\}^n \to \{0,1\} \mid c_z(x) = z_{x \pmod{\ell(n)}}, z \in \mathsf{range}(G)\},$$

[8]By *negligible advantage*, we mean: for every $c > 0$ the advantage is at most $1/n^c$.

where $x \in \{0,1\}^n$ is viewed as an integer in $\{0,1,\ldots,2^n-1\}$ and $\mathrm{range}(G) = \{G(y) : y \in \{0,1\}^n\}$.

By contradiction, let us assume there exists an efficiently uniform-PAC quantum learning algorithm for $\mathcal{C}_G$. Using this efficient learner, we construct a quantum distinguisher for the PRG G. Since G is assumed to be quantum-secure, this contradicts our assumption and proves the result. More concretely, let z be an $\ell(n)$-bit string and suppose that the distinguisher has to determine if z is a pseudo-random string or a uniformly random string. Consider a distinguisher that simulates a quantum learning algorithm $\mathcal{A}$ by preparing copies of the following state that will serve as a quantum example for $\mathcal{A}$

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle|z_{x \pmod{\ell(n)}}\rangle.$$

Since $\ell(n)$ is a polynomial (in n), this quantum state can be prepared efficiently by the quantum distinguisher. $\mathcal{A}$ uses the copies of these quantum examples and outputs a hypothesis h that approximates c_z. The distinguisher then picks a random challenge $x^* \in \{0,1\}^n$ and outputs 1 if and only if $h(x^*) = z_{x^* \pmod{\ell(n)}}$. One technical aspect that we show here is that, if the learning algorithm has bias $\beta(n)$, then the distinguisher has an advantage $\geq \frac{\beta(n)}{2}$ in distinguishing pseudo-random strings from a uniformly random string. In particular, if there is a polynomial time weak quantum learning algorithm for $\mathcal{C}$ under the uniform distribution, then there exists a polynomial-time quantum distinguisher $\mathcal{D}$ that satisfies

$$\left| \Pr_{x \in \{0,1\}^n}[\mathcal{D}(G(x))] - \Pr_{y \in \{0,1\}^{\ell(n)}}[\mathcal{D}(y)] \right| \geq \frac{1}{n^c}, \tag{1}$$

for some $c > 0$. In order to prove that a quantum learner with advantage $\beta(n)$ implies a distinguisher with advantage $\frac{\beta(n)}{2}$, we provide two different proofs – using distinct techniques and imposing two (incomparable) dependencies on the parameters. The first proof supplies an intuitive argument based on an information-theoretic approach while the second is a more involved hybrid-method based argument. In terms of the trade-offs between the PRG's stretch and the sample/query complexity of the learning algorithm, the former is better suited to algorithms using a super-linear number of queries while the latter to those using a sub-linear number of queries. This allows for a more fine-grained analysis of the query complexity of various concept classes. In fact, using the second proof-technique, we consider PRGs arising from the $\mathsf{Subset-sum}$ problem (which is another problem believed to be hard to solve on a quantum computer for some parameters) and show that: assuming that the $\mathsf{Subset-sum}$ PRG is quantum-secure, then every quantum learning algorithm for AC^0 needs to make $\Omega(\sqrt{n})$ queries. We define the $\mathsf{Subset-sum}$ problem, the corresponding PRG and the proof of the query lower bound in Appendix B.

1.3.2 Public-key encryption schemes vs. quantum learning

A public-key encryption scheme consists of a triple of algorithms (Key-generator, Enc, Dec). Key-generator is a randomized algorithm that on input 1^λ (where λ is a security parameter) outputs a tuple $(\mathrm{K_{pub}}, \mathrm{K_{priv}})$, where $\mathrm{K_{pub}}$ is a publicly known key used to encrypt messages and $\mathrm{K_{priv}}$ is a private key used to decrypt messages. Enc is a deterministic algorithm that receives as input the public key $\mathrm{K_{pub}}$, some randomness r and a message $b \in \{0,1\}$ and outputs $\mathrm{Enc}(\mathrm{K_{pub}}, r, b)$, which we denote by $\mathrm{Enc}_{\mathrm{K_{pub}}}(r,b)$ for simplicity. Dec receives as input the private key $\mathrm{K_{priv}}$ and $\mathrm{Enc}_{\mathrm{K_{pub}}}(r^*, b^*)$ and outputs $c \in \{0,1\}$ (we write $\mathrm{Dec}_{\mathrm{K_{priv}}}$ in order be explicit about the dependence of Dec on $\mathrm{K_{priv}}$). The public-key encryption scheme is said to be correct if: for uniformly random values r^* and b^* $\mathrm{Dec}_{\mathrm{K_{priv}}}\left(\mathrm{Enc}_{\mathrm{K_{pub}}}(r^*, b^*)\right) \neq b^*$ with negligible probability, i.e., at most n^{-c} for some $c > 0$. An encryption scheme is (quantum) secure if, given K_{pub} and $\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)$, a (quantum) polynomial time adversary can output b^* with at most a negligible advantage over a random guess.

We connect quantum-secure public-key encryption schemes to the hardness of learning as follows (see Theorem 5.1 for a full statement).

Result 2 *Let S be a quantum-secure public-key cryptosystem. If $\mathcal{C}_{\mathsf{S}}$ is the concept class containing the decryption functions of the cryptosystem S, then there is no efficient weak quantum-PAC learner for $\mathcal{C}_{\mathsf{S}}$.*

The works of Kearns and Valiant [KV94] and Klivans and Sherstov [KS09] provide a connection between public-key encryption schemes and learning functions. They showed that if there exists a PAC learning algorithm for a concept class that contains the decryption function $\text{Dec}_{\mathsf{K}_{\text{priv}}}$, then it is possible to predict b^* from K_{pub} and $\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)$ with a $1/\text{poly}(n)$ bias. They prove this by simulating the learning algorithm as follows: the distinguisher prepares examples of the form $(r, \text{Enc}_{\mathsf{K}_{\text{pub}}}(r, b))$ for (uniformly) random r and b. Using the guarantees of the classical PAC learning algorithm and the correctness of the encryption scheme, they show that the hypothesis h output by the learner satisfies

$$\Pr_{r^*, b^*}\left[h\Big(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)\Big) = b^*\right] \geq \frac{1}{2} + \frac{1}{n^c},$$

for some $c > 0$. In this paper, we quantize their argument, but the situation is much more intricate than in the classical case. Classically, r and b can be picked uniformly at random at each step in order to create a new training example $(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r, b), b)$. Quantumly, however, we do not know of an efficient way to create a quantum example $\frac{1}{\sqrt{2|\mathcal{R}|}} \sum_{r,b} |\text{Enc}_{\mathsf{K}_{\text{pub}}}(r, b)\rangle|b\rangle$, where $\mathcal{R}$ is the space of the possible randomness. Notice that a straightforward way of preparing this state involves solving the index-erasure problem [AMRR11, LR19], which is conjectured to be a hard problem to solve on a quantum computer. See Section 5 for more details.

Instead, we first define a distribution D as follows: pick $\text{poly}(n)$-many uniformly random (r, b) and let D be the uniform distribution over $\{\text{Enc}_{\mathsf{K}_{\text{pub}}}(r, b)\}$ where the set ranges over the $\text{poly}(n)$-many observed (r, b). Our hope is to run the quantum learner on this distribution D (which we are allowed to since we assumed that it is a quantum-PAC learner). However, we run into an issue which [KS09] need not worry about. Let $\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)$ be the challenge string that the distinguisher needs to correctly decrypt to b^*. Observe that $\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)$ need not even lie in the support of D, so running a quantum learner on the distribution D might not even help the distinguisher in predicting b^*. In contrast, in the simulation argument of Klivans and Sherstov [KS09] the pair $(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*), b^*)$ is always in the uniform distribution, since r and b are picked uniformly at random to create the classical example $(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r, b), b)$.

Ideally, we would like to use our PAC learner on a distribution D' for which $(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*), b^*)$ is the support of D'. This would enable the distinguisher to use the guarantees of a quantum learner when run on D'. The challenge here is that the distinguisher would need to find such a D' without prior knowledge of $\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*)$ and b^*! We circumvent this issue with the following observation: if two distributions are sufficiently close to each other, then the learner should perform "essentially equivalently" on both distributions. In particular, we use a training distribution D that is close enough to the testing distribution D' containing $(\text{Enc}_{\mathsf{K}_{\text{pub}}}(r^*, b^*), b^*)$ so that the learning algorithm is unable to distinguish them.

We now provide more details here. Suppose we have a quantum-secure public-key cryptosystem with a (randomized) encryption function $\mathsf{Enc}_{\mathsf{K}_{\text{pub}}, r} : \{0,1\} \to \{0,1\}^n$ and decryption function

$\mathsf{Dec}_{\mathsf{K}_{\mathsf{priv}}} : \mathcal{E} \to \{0,1\}^n$, where $\mathcal{E}$ is the set of all valid encryptions and $(\mathsf{K}_{\mathsf{pub}}, \mathsf{K}_{\mathsf{priv}})$ is the output of the Key-generation algorithm. Assume that the challenge string is $\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r^*, b^*)$ for a uniformly random r^*, b^* and an adversary for this cryptosystem has to correctly guess b^*.

In order to construct such an adversary, first define the concept class $\mathcal{C} = \{\mathsf{Dec}_{\mathsf{K}_{\mathsf{priv}}} : \mathsf{K}_{\mathsf{priv}}\}$, the set of all decryption functions (one for each private key). Furthermore, assume that there is a weak quantum-PAC learner for $\mathcal{C}$ that uses L examples, i.e., a polynomial-time quantum learning algorithm $\mathcal{A}$ which receives L quantum examples $\sum_x \sqrt{D(x)}|x, c(x)\rangle$ (for an unknown distribution D and concept $c \in \mathcal{C}$) and outputs a hypothesis h that is close to the concept c.

As discussed previously, we now define a meaningful distribution D on which the distinguisher runs the quantum learner: consider a set S with L^3 tuples (r_i, b_i) that are chosen uniformly at random from their respective domains; let D be the uniform distribution over $\{\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r,b) : (r,b) \in S\}$. The adversary behaves as follows: run the quantum-PAC learner $\mathcal{A}$ under the distribution D and provide it L quantum examples of the form

$$|\psi\rangle = \frac{1}{\sqrt{L^3}} \sum_{(r,b)\in S} |\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r,b)\rangle|b\rangle.$$

When $\mathcal{A}$ outputs the hypothesis h, the adversary outputs $h\Big(\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r^*, b^*)\Big)$ as its guess for b^*. Notice that with overwhelming probability, S does not contain the tuple (r^*, b^*) corresponding to the challenge. So there is no guarantee on the value of $h\Big(\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r^*, b^*)\Big)$. In order to overcome this, consider a quantum example state

$$|\psi'\rangle = \frac{1}{\sqrt{L^3+1}} \left(|\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r^*, b^*)\rangle|b^*\rangle + \sum_{(r,b)\in S} |\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r,b)\rangle|b\rangle \right).$$

We show that as the learner uses only L quantum examples, $|\psi\rangle^{\otimes L}$, the output statistics of every quantum learning algorithm, when run on $|\psi\rangle$ and $|\psi'\rangle$, is very similar. In fact, we show that the distribution on the hypothesis set is almost the same in each case. Now, using the performance guarantees of a quantum-PAC learning algorithm and the closeness of the distributions between the hypothesis sets, we conclude that $h\Big(\mathsf{Enc}_{\mathsf{K}_{\mathsf{pub}}}(r^*, b^*)\Big)$ equals b^* with probability at least $\frac{1}{2} + \frac{1}{\mathrm{poly}(n)}$. This contradicts the quantum-secure assumption on the cryptosystem.

1.3.3 Conditional hardness of learning TC^0 and AC^0

The first consequence of Result 1 is to give a strong conditional *negative answer* to the question of Aaronson [Aar05] regarding the quantum learnability of TC^0 circuits.

Result 3 *If the Learning with Errors problem cannot be solved in quantum polynomial time, then there is no polynomial-time uniform weak quantum-PAC learner for* TC^0 *with membership queries.*

As previously mentioned, the Learning with Errors problem (LWE) is a cornerstone of current post-quantum cryptosystems and is widely believed to be hard for quantum computers. The starting point for proving Result 3 is the pseudo-random generator presented by Banerjee, Peikert and Rosen [BPR12]. They show that their PRG is quantum secure under the assumption that the *Learning with Rounding* (LWR) problem cannot be solved by quantum computers efficiently. We do not define the LWR problem here (see Section 3.2.2 for details), but point out that LWR is at least as hard as LWE. The PRG $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ proposed in [BPR12] satisfies the following properties: (i) for every $x \in \{0,1\}^n$, every bit of $G(x)$ can be computed by a TC^0 circuit, (ii) the stretch function $\ell(n)$ can be an arbitrary polynomial in n, and (iii) suppose there exists a *distinguisher* $\mathcal{D}$ for G, i.e., a polynomial-time algorithm $\mathcal{D}$ that satisfies

$$\left| \Pr_{x\in\{0,1\}^n}[\mathcal{D}(G(x))] - \Pr_{y\in\{0,1\}^{\ell(n)}}[\mathcal{D}(y)] \right| \geq \frac{1}{n^c} \tag{2}$$

for some $c > 0$, then there exists a polynomial-time algorithm that solves LWE. Notably, when $\mathcal{D}$ is a *quantum* distinguisher for the PRG G, it implies the existence of a *quantum* polynomial-time algorithm for LWE. Therefore, the PRG G is quantum-secure and accessible in TC^0. We can then use our connection between PRGs and hardness of quantum learning (Result 1) to prove Result 3. Using a similar approach (albeit, with notable subtleties), we also prove the hardness of learning AC^0 under less standard assumptions.

Result 4 *If the Learning with Errors problem cannot be solved in quantum sub-exponential time, then there is no polynomial-time uniform weak quantum-PAC learner for* AC^0 *with membership queries.*

In order to better understand this assumption, we remark that the current best known classical algorithms for LWE require exponential time [LLL82, BKW03, Wag02, AG11] and any straightforward quantization of these results gives only a polynomial speedup. Therefore, a sub-exponential time algorithm for LWE would result in a threat to the security of LWE-based cryptography and be a breakthrough for the algorithms and cryptanalysis communities.

We now point out a key difference involved in proving Result 4, as compared to Result 3. An inherent problem in considering the known LWE-based PRGs to prove Result 4 is that, any circuit computing a given pseudo-random bit (for an arbitrary seed) needs to compute a matrix-vector product. While this can be done directly through a TC^0 circuit, it cannot be computed using an AC^0 circuit. In fact, it is *a priori* unclear if we could devise a PRG having polynomial stretch whose hardness is based on a quantum-hard problem while also being computable in AC^0.

Similar to Kharitonov [Kha93], we overcome this issue by considering PRGs with stretch super-polynomial in the seed length. Specifically, we obtain a pseudo-random string whose size is polynomial in n while the seed length is poly-logarithmic in n. We show that an arbitrary bit of such a pseudo-random string can be computed by an AC^0 circuit (with a fixed seed). However, since the stretch of the PRG is sub-exponential in the seed length, a stronger assumption is needed to guarantee its security. To this end, by assuming that LWE cannot be solved in sub-exponential time by quantum computers, the PRG that we consider here is quantum-secure. With this new PRG, we can repeat the arguments of Result 3 and derive the polynomial-time hardness of quantum learning for AC^0.

Using similar proof techniques, we also obtain an incomparable (conditional) lower bound for learning *quasi-polynomially-sized* AC^0 circuits. Assume that Learning with Rounding cannot be solved in quantum sub-exponential time, then quasi-polynomial size AC^0 circuits cannot be learned in quantum quasi-polynomial time, under the uniform distribution. By contrast, Linial et al. [LMN93] showed that, classically, *polynomial-sized* AC^0 circuits can be learned in quasi-polynomial time. To find a matching lower bound for quantum learning is left for future work.

It is worth noting that our result implies also conditional hardness results for *classical* learning of TC^0 and AC^0, under the assumption that solving LWE is hard for classical computers (instead of factoring in the case of Kharitonov [Kha93]). Additionally, we provide *explicit* proofs of many lemmas and theorems in [Kha93], which were omitted in the conference version of that paper.[9]

[9]We have been unable to find a full version of this paper.

1.3.4 Conditional hardness of PAC learning TC_2^0

Our third hardness result is the following.

Result 5 *If the Learning with Errors problem cannot be solved in quantum polynomial time, then there is no polynomial-time weak quantum-PAC learner for* TC_2^0.

In contrast to our first hardness result for TC^0, we stress that the quantum learners in this result are *quantum-PAC learners*. The main idea to prove this result is to consider the LWE-based public-key cryptosystem proposed by Regev [Reg09] (see Section 3.1.2). Klivans and Sherstov [KS09] considered this cryptosystem and showed that the decryption functions in this cryptosystem can be implemented by circuits in TC_2^0. We can then use our connection between quantum learning and quantum-secure cryptosystems (Result 2) to derive Result 5.

As previously mentioned, understanding the learnability of TC_2^0 (and TC^0), apart from being theoretically important, also sheds light on the question: can quantum computers help learn the weights of neural networks faster? It is well-known [MSS91, GHR92] that constant-depth polynomial-sized feed-forward neural networks, where the weights of neurons are bounded by a polynomial in the input size, can be implemented by TC^0 circuits. So our conditional negative results on learning TC^0 gives an indication that, quantum resources do not give an exponential advantage in learning the weights of such neural networks (assuming LWE is quantum-hard).

1.4 Open questions

This work raises a number of interesting open questions, which we list below.

Learning AC_d^0. What is the smallest d for which we can prove that quantum learning AC_d^0 is (conditionally) hard? Bshouty and Jackson [BJ99] gave a quantum polynomial-time algorithm for AC_2^0 and for some universal constant $d' \geq 3$ (independent of the input-size of the $\mathsf{AC}_{d'}^0$ circuit), our work rules out polynomial-time learning algorithms for $\mathsf{AC}_{d'}^0$, assuming there exists no sub-exponential time quantum algorithm for the LWE problem. Classically, using the constant-depth construction of PRGs by Naor and Reingold [NR04] in Kharitonov's [Kha93] result, one can show that AC_5^0 is hard to learn (assuming factoring is hard on a classical computer).

Improving the (conditional) lower bound for learning AC^0. Can we show that there exists no quasi-polynomial-time algorithm for learning AC^0? Specifically, can we show a conditional lower bound that matches the upper bound of Linial, et al. [LMN93]? If this were true, then (assuming the conditional lower bound) quantum examples and membership queries would not give any advantage in learning AC^0 circuits. One way to approach this problem would be to find a quantum-secure PRG that achieves a super-polynomial stretch with respect to seed size while still being computable in AC^0. We believe that applying our techniques to such a PRG would help provide a tight lower bound that matches the upper bound of [LMN93].

Uniform learning of TC_2^0. The conditional hardness results of TC^0 under the uniform distribution and hardness of quantum-PAC learning TC_2^0 do not rule out the possibility that TC_2^0 admits polynomial-time quantum learning algorithms under the uniform distribution. It is an open question to show if TC_2^0 can be learned quantum-efficiently under the uniform distribution or if we can show a conditional hardness result.

Hardness of (quantum) learning quantum shallow circuits Linial et al. [LMN93] showed the quasi-polynomial time learnability of shallow *classical* circuits. Correspondingly, what would be the time/sample complexity of learning shallow *quantum* circuits with a quantum learner?

Hardness of quantum learning from other assumptions Finally, we leave as an open question the possibility of proving the hardness of quantum learning from other complexity-theoretic assumptions. We now point to some potential directions:

- Recent results have proved Strong Exponential Time Hypothesis (SETH)-based hardness of the Gap-SVP [AS18] problem and the Subset – sum [ABHS19] problem (actually, they prove the limit of a specific approach to solve the Subset – sum problem). These hardness results do not imply any hardness for learning problems directly and we wonder if they can be tightened in order to make it possible.
- Oliveira and Santhanam [OS17] established a connection between learning theory, circuit lower bounds and pseudo-randomness. Is it possible to quantize such connections?
- Daniely and Schwartz [DS16] showed a complexity-theoretic hardness of PAC learning DNFs. Could we also show hardness for quantum-PAC learning DNFs as well?

Organization

In Section 2 we state some required lemmas, cryptographic primitives and formally define the classical and quantum learning models. In Section 3 we discuss the Learning with Errors problem and its variants along with the pseudo-random generators constructed from these problems. In Section 4 we describe the connection between quantum-secure PRGs and quantum learning under the uniform distribution. In Section 5 we describe the connection between quantum-secure public-key cryptosystems and quantum-PAC learning. Finally, in Section 6 we describe our main results showing the hardness of learning $\mathsf{AC}^0, \mathsf{TC}^0, \mathsf{TC}^0_2$.

Acknowledgments

S.A. thanks Andrea Rocchetto and Ronald de Wolf for discussions during the initial stage of this work and Robin Kothari for introducing him to this problem. S.A. and A.S. did part of this work during the Workshop on Quantum Machine Learning at the Joint Centre for Quantum Information and Computer Science, University of Maryland. S.A. did part of this work at QuSoft, CWI and was supported by ERC Consolidator Grant QPROGRESS. We thank Anurag Anshu and Prabhanjan Ananth for helpful discussions. We thank Robin Kothari and Ronald de Wolf for many helpful comments which improved the presentation of this paper.

2 Preliminaries

2.1 Notation and basic claims

We define some widely used definitions below. For $k \in \mathbb{N}$, we let $[k] := \{0, \ldots, k-1\}$. All logarithms will be taken with respect to the base 2. A function $\mu : \mathbb{N} \to \mathbb{R}$ is said to be *negligible* in a parameter $\lambda \geq 1$, which we denote $\mathrm{negl}(\lambda)$, if it satisfies the following:

$$\text{for every integer } t > 0, \text{ there exists an integer } K_t > 0 \text{ such that for all } \lambda > K_t, \text{ we have } |\mu(\lambda)| < \lambda^{-t}.$$

Similarly, a function $\eta : \mathbb{N} \to \mathbb{R}$ is said to be *non-negligible* in λ if there exists an integer $t > 0$ and $K_t > 0$ such that $\eta(\lambda) \geq \lambda^{-t}$ for all $\lambda > K_t$. For a distribution $D : \{0,1\}^n \to [0,1]$, we write $x \sim D$ to say that x is drawn according to the distribution D. We say there is a *non-negligible advantage* in distinguishing D from another distribution D' if for every poly(n)-time adversary Adv (i.e., algorithm) we have

$$\left| \Pr_{x_1,\ldots,x_L \sim D}[\mathsf{Adv}(x_1,\ldots,x_L) = 1] - \Pr_{x_1,\ldots,x_L \sim D'}[\mathsf{Adv}(x_1,\ldots,x_L) = 1] \right| \geq \eta(n),$$

where $\eta(n)$ is a non-negligible function and L is a polynomial in n. For simplicity, we say that a function $f(n) = \mathrm{poly}(n)$, if there exist constants $a, b > 0$ such that $n^a < f(n) < n^b$.

For function $f : \{0,1\}^n \to \{0,1\}^m$ and for all $1 \leq k \leq n$, we say that $g : \{0,1\}^n \to \{0,1\}^{2^k}$ is a k-extension of f if

$$\forall x \in \{0,1\}^n, \;\; g(x) = \begin{cases} f(x) \mod 2^{2^k} & \text{if } k \leq \log m, \\ f(x) * 2^{2^k - m} & \text{if } \log m < k \leq n, \end{cases}$$

where $f(x)$ is viewed as an integer in $[2^m]$ and $*$ denotes Boolean multiplication. In other words, g either truncates the output of f to 2^k bits when $2^k \leq m$ or trivially pads the output of f up to 2^k bits with trailing 0s when $2^k > m$.

A *half-space* in n dimensions is a Boolean function $f : \{0,1\}^n \to \{0,1\}$ of the form

$$f(x) = \left[\!\left[\sum_i a_i x_i \geq \theta \right]\!\right]$$

where $a_1, \ldots, a_n$ and θ are fixed integers and $[\![\cdot]\!]$ denotes the indicator function which evaluates to 1 if and only if $\sum_i a_i x_i \geq \theta$. The *intersection of k half-spaces* is a function of the form $g(x) = \bigwedge_{i=1}^k f_i(x)$ where the f_is are half-spaces and $\wedge$ is the AND function. A *polynomial threshold function* (PTF) of degree d is a Boolean function of the form $f(x) = [\![p(x) \geq 0]\!]$, where p is a degree-d polynomial with integer coefficients. Note that a half-space is a degree-1 PTF. A PTF f, defined as $f(x) = [\![p(x) \geq 0]\!]$, is called *light* if the sum of the absolute value of the coefficients in p is at most polynomial in n.

The following claim is a well-known fact in quantum information theory.

Fact 2.1 *Let the binary random variable* $\mathbf{b} \in \{0,1\}$ *be uniformly distributed and* $|\psi_0\rangle, |\psi_1\rangle$ *be two quantum sates. Suppose that an algorithm is given* $|\psi_{\mathbf{b}}\rangle$ *(for unknown b) and is required to guess whether* $\mathbf{b} = 0$ *or* $\mathbf{b} = 1$. *It will guess correctly with probability at most* $\frac{1}{2} + \frac{1}{2}\sqrt{1 - |\langle\psi_0|\psi_1\rangle|^2}$.

Note that if we can distinguish $|\psi_0\rangle$ and $|\psi_1\rangle$ with probability $\geq 1-\xi$, then $|\langle\psi_0|\psi_1\rangle| \leq 2\sqrt{\xi(1-\xi)}$. If $\xi = \frac{1}{2} - \tau$ (for some $\tau > 0$) and we can distinguish $|\psi_0\rangle$ and $|\psi_1\rangle$ with probability $\geq \frac{1}{2} + \tau$, then

$$\||\psi_0\rangle - |\psi_1\rangle\|_2 = \sqrt{2 - 2\langle\psi_0|\psi_1\rangle} \geq \sqrt{2 - 2\sqrt{1 - 4\tau^2}} \geq \sqrt{2 - 2(1 - 2\tau^2)} = 2\tau, \tag{3}$$

where we used $|\langle\psi_0|\psi_1\rangle| \leq 2\sqrt{\xi(1-\xi)} \leq 2\sqrt{1-4\tau^2}$ (for $\xi = \frac{1}{2} - \tau$) in the second inequality and $\sqrt{1-\alpha} \leq 1 - \alpha/2$ (for $\alpha > 0$ in the second inequality).

2.2 Information theory and communication complexity

We describe some basic concepts in information theory that we use later. Given a probability distribution $D : \mathcal{X} \to [0,1]$, the entropy of a random variable $X \sim D$ is given by

$$\mathbf{H}(X) = -\sum_{x \in \mathcal{X}} \Pr_{X \sim D}[X = x] \log\left(\Pr_{X \sim D}[X = x] \right).$$

The *binary entropy* of $\varepsilon \in [0,1]$ is defined as $\mathbf{H}_b(\varepsilon) = -\varepsilon \log \varepsilon - (1-\varepsilon)\log(1-\varepsilon)$. Moreover, $\mathbf{H}_b(\varepsilon)$ can be upper bounded as follows.

Fact 2.2 *For all $\varepsilon \in [0,1/2]$ we have the binary entropy $\mathbf{H}_b(\varepsilon) \leq O(\varepsilon \log(1/\varepsilon))$, and from the Taylor series expansion of $\mathbf{H}_b(\varepsilon)$, we have*

$$1 - \mathbf{H}_b(1/2+\varepsilon) \leq 2\varepsilon^2/\ln 2 + O(\varepsilon^4).$$

Given a probability distribution $D : \mathcal{X} \times \mathcal{Y} \to [0,1]$, and the random variables $(X,Y) \sim D$, the conditional entropy of X given Y is

$$\mathbf{H}(X|Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} \Pr_{(X,Y)\sim D}[(X,Y)=(x,y)] \log\left(\frac{\Pr_{(X,Y)\sim D}[(X,Y)=(x,y)]}{\Pr_{X\sim D}[X=x]}\right).$$

Given a probability distribution $D : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to [0,1]$, the random variables $(X,Y,Z) \sim D$, the mutual information between X and Y given Z is

$$\mathbf{I}(X:Y|Z) = \mathbf{H}(X|Z) - \mathbf{H}(X|Y,Z).$$

The following fact about conditional entropy and prediction errors will be useful for us.

Lemma 2.3 (Fano's inequality) *Let X be a random variable taking values in $\{0,1\}$ and Y be a random variables taking values in $\mathcal{Y}$. Let $f : \mathcal{Y} \to \{0,1\}$ be a* prediction *function, which predicts the value of X based on an observation of Y. Suppose $\varepsilon = \Pr[f(Y) \neq X]$ is the probability of error made by the prediction function, then $\mathbf{H}(X|Y) \leq \mathbf{H}_b(\varepsilon)$.*

We now briefly describe communication complexity. For details, we refer the reader to [Tou15, KLGR16]. Here, we are interested in the setup with two parties Alice (denoted A) and Bob (denoted B). A (resp. B) receives input X (resp. Y) such that $(X,Y) \sim D$ for a publicly known probability distribution D. A and B then follow some protocol π in which they exchange quantum information back-and-forth. Finally, B outputs a random variable Z. The quantum communication complexity of the protocol $\mathsf{QCC}(\pi)$ is the number of qubits communicated in the protocol π.

Touchette [Tou15] defined the notion of quantum information complexity for a protocol, denoted $\mathsf{QIC}(\pi)$, which is rather subtle and out of the scope of this work. In [Tou15, Theorem 1], Touchette showed that for all protocols π, we have $\mathsf{QIC}(\pi) \leq \mathsf{QCC}(\pi)$. Similarly, Kerenidis et al. [KLGR16, Theorem 1] showed that $\mathsf{QIC}(\pi)$ is at most the classical information complexity of the protocol $\mathsf{CIC}(\pi)$, whose definition we omit here. Also, it is not hard to see that if B outputs some value Z, then

$$\mathbf{I}(Z:X|Y) \leq \mathsf{CIC}(\pi).$$

Putting together [Tou15, Theorem 1] and [KLGR16, Theorem 1] along with the inequality above, we obtain the following corollary.

Corollary 2.4 *Given a quantum communication protocol π between two parties A and B whose inputs are X and Y, respectively, drawn from a distribution D. Let Z be the output of B. Then,*

$$\mathbf{I}(Z:X|Y) \leq QCC(\pi).$$

2.3 Cryptographic primitives

Definition 2.5 (One-way functions) *A deterministic function $f : \{0,1\}^n \to \{0,1\}^m$ is a* one-way function *if there is a polynomial-time algorithm that computes f and for every constant $c > 0$ and every polynomially bounded adversary Adv, we have*

$$\Pr_{x\in\{0,1\}^n}[\mathsf{Adv}(f(x)) \in f^{-1}(f(x))] \leq \frac{1}{n^c},$$

for sufficiently large n. If Adv *is allowed to be a quantum polynomial-time algorithm, then f is called a* quantum one-way *function.*

Definition 2.6 (Pseudo-random generators) *A deterministic function $G : \{0,1\}^s \to \{0,1\}^{\ell(s)}$ is a pseudo-random generator if the function $\ell : \mathbb{N} \to \mathbb{N}$ satisfies $\ell(s) > s$ for all s and every probabilistic algorithm* Adv *that runs in* poly(s)*-time and for every constant $c > 0$ we have*

$$\left| \Pr_{x \in \{0,1\}^s}[\mathsf{Adv}(G(x)) = 1] - \Pr_{y \in \{0,1\}^{\ell(s)}}[\mathsf{Adv}(y) = 1] \right| < \frac{1}{s^c},$$

where the probability is taken over uniform $x \in \{0,1\}^s$ and $y \in \{0,1\}^{\ell(s)}$. Additionally, s is the seed length*, $\ell(s)$ is the* stretch function *and $\ell(s) - s$ is the* stretch *of the pseudo-random generator G.*

In this work, we also consider quantum polynomial-time distinguishers. In short, when we say that a pseudo-random generator is *secure*, we mean that it satisfies Definition 2.6 and it is *quantum-secure* if it satisfies a variation of Definition 2.6 where Adv is a polynomial-time quantum algorithm.

We point out that such a distinction is important. For example, the security of the Blum, Blum and Shub pseudo-random generator [BBS86] is based on the assumption that factoring is hard for polynomial-time algorithms. While this assumption is reasonable for classical algorithms, and therefore this PRG would be *secure*, factoring can be solved in quantum polynomial time [Sho97], and thus this PRG is *not quantum-secure*.

2.4 Learning models

2.4.1 Classical distribution-independent learning

We begin by introducing the classical Probably Approximately Correct (PAC) model of learning which was introduced by Leslie Valiant [Val84]. A *concept class* $\mathcal{C}$ is a collection of Boolean functions $c : \{0,1\}^n \to \{0,1\}$, which are often referred to as *concepts*. In the PAC model, a learner $\mathcal{A}$ is given access to a *random example oracle* $\mathrm{EX}(c, D)$ where $c \in \mathcal{C}$ is an *unknown* target concept (which the learner is trying to learn) and $D : \{0,1\}^n \to [0,1]$ is an *unknown* distribution. At each invocation of $\mathrm{EX}(c, D)$ the oracle returns a *labelled example* $(x, c(x))$ where x is drawn from the distribution D.[10] Then $\mathcal{A}$ outputs a hypothesis h and we say that $\mathcal{A}$ is an (ε, δ)-*PAC learner* for a concept class $\mathcal{C}$ if it satisfies the following:

> for every $\varepsilon, \delta \in [0,1]$, for all $c \in \mathcal{C}$ and distributions D, when $\mathcal{A}$ is given ε, δ and access to the $\mathrm{EX}(c, D)$ oracle, with probability $\geq 1 - \delta$, $\mathcal{A}$ outputs a hypothesis h such that
> $$\Pr_{x \sim D}[h(x) \neq c(x)] \leq \varepsilon$$

The learner's *advantage* over a random guess is given by $\beta = \frac{1}{2} - \frac{\varepsilon}{2}$ and $2\beta = 1 - \varepsilon$ is called the *bias* of the learner.

The *sample complexity* of $\mathcal{A}$ is the maximum number of invocations of the $\mathrm{EX}(c, D)$ oracle which the learner makes when maximized over all $c \in \mathcal{C}$ and all distributions D. Finally the (ε, δ)-PAC *sample complexity of* $\mathcal{C}$ is defined as the minimum sample complexity over all $\mathcal{A}$ that (ε, δ)-PAC learn $\mathcal{C}$. The *time complexity* of (ε, δ)-PAC learning $\mathcal{C}$ is the minimum number of *time steps* of an algorithm $\mathcal{A}$ that (ε, δ)-PAC learns $\mathcal{C}$ (where the minimum is over all $\mathcal{A}$ that (ε, δ)-PAC learn $\mathcal{C}$). In the *weak learning setting*, we say that $\mathcal{A}$ (ε, δ)–*weakly learns* (resp. strongly learns) $\mathcal{C}$ if $\varepsilon = \frac{1}{2} - n^{-c}$

[10]Note that the oracle $\mathrm{EX}(c, D)$ doesn't take any input and simply returns a labelled example.

(resp. $\varepsilon = 1/3$) for some constant $c > 0$ and input size n. Freund et al. [FSA99] showed that weak-PAC learning $\mathcal{C}$ is equivalent to strong-PAC learning.

2.4.2 Quantum distribution-independent learning

The quantum model of PAC learning was introduced by Bshouty and Jackson [BJ99]. Instead of having access to an $\mathrm{EX}(c, D)$ oracle, here a *quantum-PAC learner* has access to a $\mathrm{QEX}(c, D)$ oracle

$$\mathrm{QEX}(c, D) : |0^n, 0\rangle \to \sum_x \sqrt{D(x)}|x, c(x)\rangle,$$

and we leave the $\mathrm{QEX}(c, D)$ oracle undefined on other basis states. We refer to the state produced by $\mathrm{QEX}(c, D)$ as a *quantum example*, which is a coherent superposition over classical labeled examples. A quantum-PAC learner is given access to copies of quantum examples and performs a POVM (positive-valued-operator measurement), where each outcome of the POVM corresponds to a hypothesis. Similar to the classical distribution-independent learning setting, the *quantum sample complexity* of an algorithm $\mathcal{A}$ is the maximum number of invocations of the $\mathrm{QEX}(c, D)$ oracle which the learner makes, when maximized over all $c \in \mathcal{C}$ and all distributions D. The (ε, δ)-*quantum* PAC *sample complexity of* $\mathcal{C}$ is defined as the minimum quantum sample complexity over all $\mathcal{A}$ that (ε, δ)-quantum-PAC learn $\mathcal{C}$.

2.4.3 Uniform distribution learning

The classical PAC model of learning places a strong requirement on learners, i.e., the learner needs to (ε, δ)-PAC learn $\mathcal{C}$ for *every unknown distribution* D. In classical learning theory, there has been a lot of work in understanding a weaker model of learning – when D is restricted to the uniform distribution on $\{0, 1\}^n$ (which we denote as $\mathcal{U}$). In this restricted model, a classical learner is given access to $\mathrm{EX}(c, \mathcal{U})$ (known to the learner) which generates $(x, c(x))$ where x is sampled according to the uniform distribution $\mathcal{U}$. An algorithm $\mathcal{A}$ is said to (ε, δ)-learn $\mathcal{C}$ under $\mathcal{U}$ if it satisfies the following:

> for every $\varepsilon, \delta \in [0, 1]$, for all $c \in \mathcal{C}$, when $\mathcal{A}$ is given ε, δ and access to the $\mathrm{EX}(c, \mathcal{U})$ oracle, with probability $\geq 1 - \delta$, $\mathcal{A}$ outputs a hypothesis h such that $\Pr_{x \sim D}[h(x) \neq c(x)] \leq \varepsilon$

The sample complexity and time complexity of learning $\mathcal{C}$ under the uniform distribution is defined similar to Section 2.4.1 when we fix $D = \mathcal{U}$.

One can similarly consider the case when a quantum learner is given access to $\mathrm{QEX}(c, \mathcal{U})$

$$\mathrm{QEX}(c, \mathcal{U}) : |0^n, 0\rangle \to \frac{1}{\sqrt{2^n}} \sum_x |x, c(x)\rangle.$$

We leave QEX undefined on other basis states and assume the learner *does not* have access to the inverse of $\mathrm{QEX}(c, \mathcal{U})$. The quantum sample complexity and time complexity of learning a concept class $\mathcal{C}$ under the uniform distribution is defined similar to Section 2.4.2 when we restrict $D = \mathcal{U}$. A powerful advantage of being given access to $\mathrm{QEX}(c, \mathcal{U})$ is *Fourier sampling*. We do not introduce Fourier sampling here and refer the interested reader to [AW17, Section 2.2.2].

2.4.4 Learning with membership queries

The classical model of PAC learning places yet another strong requirement on learners, i.e., the learner is only given access to labelled examples generated by the oracle $\mathrm{EX}(c, D)$ (for an unknown

$c \in \mathcal{C}$ and distribution D). Angluin [Ang87] relaxed this requirement and introduced the model of learning with *membership queries*. In this scenario, in addition to $\mathrm{EX}(c,D)$, a learner is given access to a *membership oracle* $\mathrm{MQ}(c)$ for the unknown target concept c, which takes as input $x \in \{0,1\}^n$ and returns $c(x)$. An algorithm $\mathcal{A}$ is said to (ε,δ)-learn $\mathcal{C}$ under D with membership queries if it satisfies the following:

> for every $\varepsilon,\delta \in [0,1]$, for all $c \in \mathcal{C}$, when $\mathcal{A}$ is given ε,δ and access to $\mathrm{EX}(c,D)$, $\mathrm{MQ}(c)$ oracles, with probability $\geq 1-\delta$, $\mathcal{A}$ outputs a hypothesis h such that $\Pr_{x\sim D}[h(x) \neq c(x)] \leq \varepsilon$.

We abuse notation by saying the sample complexity of $\mathcal{A}$ is the maximum number of invocations of an oracle[11] when maximized over all $c \in \mathcal{C}$ under distribution D. The sample complexity and time complexity of learning $\mathcal{C}$ under D given membership queries is defined similar to Section 2.4.1 (where the classical learner now has access to $\mathrm{MQ}(c)$ in addition to $\mathrm{EX}(c,D)$).

One can similarly consider the case when a quantum learner is additionally given access to a quantum membership query oracle $\mathrm{QMQ}(c)$

$$\mathrm{QMQ}(c) : |x,b\rangle \to |x, b \oplus c(x)\rangle,$$

for $x \in \{0,1\}^n$ and $b \in \{0,1\}$. The quantum learner is allowed to perform arbitrary unitary operations in between applications of the $\mathrm{QMQ}(c)$ oracle. In this paper, we will also view as $c \in \{0,1\}^{2^n}$, described by the truth-table of $c : \{0,1\}^n \to \{0,1\}$. The quantum sample complexity and time complexity of learning $\mathcal{C}$ under D given quantum membership queries is defined similar to Section 2.4.2 (where the classical learner now has access to $\mathrm{QMQ}(c)$ in addition to $\mathrm{QEX}(c,D)$). More generally, when we say a learning algorithm is allowed to make *queries to a string* $c \in \{0,1\}^N$, it means that the algorithm is given access to the oracle $\mathrm{QMQ}(c) : |x,b\rangle \to |b \oplus c_x\rangle$ for $x \in [N]$ and $b \in \{0,1\}$. The query complexity of such an algorithm is the number of invocations of $\mathrm{QMQ}(c)$.

Learning with membership queries under the uniform distribution. Finally, one can combine the learning models in this section with Section 2.4.3 and consider (quantum) learners which are given access to (quantum) membership queries and (quantum) labelled examples when the underlying distribution D is restricted to the uniform distribution $\mathcal{U}$. We say $\mathcal{A}$ is an (ε,δ)-*uniform PAC learner for* $\mathcal{C}$ with membership queries if $\mathcal{A}$ (ε,δ)-learns $\mathcal{C}$ under $\mathcal{U}$ with membership queries. Similarly we can define a (ε,δ)-*uniform quantum-PAC learner for* $\mathcal{C}$ with quantum membership queries. In this paper we will consider such learners in the weak learning setting (for which $\varepsilon = \frac{1}{2} - n^{-c}$ for some $c > 0$) and let $\delta = 1/3$. For simplicity we will omit the (ε,δ)-dependence when referring to weak classical-PAC or quantum-PAC learners. The sample complexity and time complexity of such learners is defined similar to Section 2.4.1, 2.4.2 (wherein the classical learner now has access to $\mathrm{EX}(c,\mathcal{U})$ and $\mathrm{MQ}(c)$ and the quantum learner has access to $\mathrm{QEX}(c,\mathcal{U})$ and $\mathrm{QMQ}(c)$).

In order to further understand the theoretical aspects of quantum machine learning, we refer the reader to [SP18, AAD+15, AW17].

2.5 Circuits and neural networks

In this paper we will be concerned with the class of shallow or constant-depth Boolean circuits that consist of AND, OR, NOT and Majority gates. We define these classes formally now.

[11] Note that an invocation of either $\mathrm{EX}(c,D)$ $\mathrm{MQ}(c)$ counts as one application of the oracle.

2.5.1 The circuit classes AC^0 and TC^0

An AC^0 circuit on n bits consists of AND, OR and NOT gates whose inputs are $x_1,\ldots,x_n,\overline{x_1},\ldots,\overline{x_n}$. Fan-in to the AND and OR gates are unbounded. The size of the circuit (i.e., the number of gates in the AC^0 circuit) is bounded by a polynomial in n and the depth of the circuit is a constant (i.e., independent of n). We can further assume that the gates at level i of the circuit have all their inputs coming from the $(i-1)$-th level and all the gates at the same level are AND or OR gates. The class of Boolean functions that can be expressed by such a depth-d circuit is written as AC^0_d and $\mathsf{AC}^0 = \bigcup_{d\geq 0} \mathsf{AC}^0_d$.

A depth-d threshold circuit, denoted TC^0_d, is similar to an AC^0_d circuit, except that the circuit is also allowed to have Majority gates $\mathrm{MAJ} : \{0,1\}^n \to \{0,1\}$, where the $\mathrm{MAJ}(x) = 1$ if and only if $\sum_i x_i \geq n/2$. Note that TC^0 contains AC^0 since AND, OR gates can be written in terms of the majority gate as follows: $\mathsf{AND}(x_1,\ldots,x_n) = \mathrm{MAJ}(0^{n-1},x_1,\ldots,x_n)$ and $\mathsf{OR}(x_1,\ldots,x_n) = \mathrm{MAJ}(1^{n-1},x_1,\ldots,x_n)$ for every $x \in \{0,1\}^n$. Finally, denote $\mathsf{TC}^0 = \bigcup_{d\geq 0} \mathsf{TC}^0_d$.

2.5.2 Neural networks and TC^0

One motivation for learning the concept class TC^0 is that it presents a theoretical way to model neural networks with polynomially bounded weights. Although we do not deal with neural networks in this paper, we briefly mention their connection to TC^0 circuits. A *feed-forward neural network* can be modeled as an acyclic directed graph where the nodes consist of *neurons*. We do not define neurons here since it is beyond the scope of our paper, instead one can think of neurons as real-valued function associated with weights. In order to make the connection to TC^0 we will consider a subclass of neural networks called feed-forward neural networks. A sequence of works [MTT61, Mur71, Par90, MSS91, GHR92] showed that constant-depth polynomial-sized neural networks with neuron-weights bounded by polynomial in the input size,[12] can be implemented as a circuit in TC^0. Equivalently, learning the concept class TC^0 broadly translates to the ability of an algorithm that approximately learns the weights of neural networks.

3 Hardness assumptions and cryptographic constructions

In this section we give a detailed introduction to lattice-based problems and some cryptographic primitives built using them. Our motivation in doing this, firstly, is to highlight the subtleties inherent in the hardness results and security proofs associated with these lattice-based problems and primitives respectively. Secondly, the details also help in explaining the subtleties that are reflected in our utilization of these primitives for showing the hardness of quantum learning.

Let χ be a distribution over $\mathbb{Z}$ and assume that we can efficiently sample from χ.[13] Let $B > 0$. We say that χ is B-bounded if $\Pr_{e\sim\chi}[|e| > B] \leq \mu(n)$, where e, B are at most n bits long and μ is a function that is negligible in n. In other words, a distribution χ is B-bounded if, with high probability, the magnitude of e when drawn according to χ is at most B.

Throughout this section, we use the following notation. For $\mathbf{a},\mathbf{s} \in \mathbb{Z}_d^q$, let $\mathbf{a}\cdot\mathbf{s}$ be defined as $\sum_{j\in[d]} a_j s_j \mod q$. For $q \geq p \geq 2$, we define $\lfloor\cdot\rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$ as $\lfloor x\rceil_p = \lfloor (p/q)\cdot x\rceil$ where $\lfloor\cdot\rceil$ denotes the

[12]Here the size of the neural network is the number of neurons in the network.

[13]The distribution χ is dependent on input size n, but we drop the n-dependence for notational simplicity.

closest integer. A *discrete Gaussian* is the standard normal Gaussian distribution on the real line when restricted to take integer values.

3.1 Learning with Errors (LWE)

Now, we can define the decision version of the *Learning with Errors (*LWE*)* problem.

Definition 3.1 (LWE$_{d,q,\chi,m}$) *The (decision) Learning with Errors problem with dimension d, modulus q and a B-bounded distribution χ over $\mathbb{Z}$ is defined as follows: On input m independent samples $\{(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^d \times \mathbb{Z}_q\}_i$, where the $\mathbf{a}_i$ are uniformly sampled from $\mathbb{Z}_q^d$ and $\mathbb{Z}_q$ respectively, distinguish (with non-negligible advantage) between the following two cases:*

- *(*LWE*-samples) The b_is are* noisy products *with respect to a fixed secret $\mathbf{s}$ distributed uniformly in $\mathbb{Z}_q^d$, i.e., b_is are of the form $b_i = \mathbf{a}_i \cdot \mathbf{s} + e_i \pmod q$ where $e_i \in \mathbb{Z}$ is sampled according to χ conditioned on $|e_i| \le B$.*
- *(Uniform samples) For every i, b_i is uniformly sampled from $\mathbb{Z}_q$ and is independent of $\mathbf{a}_i$.*

When the number of samples m is arbitrary (i.e., not bounded in terms of the dimension d), we will simply denote the problem as LWE$_{d,q,\chi}$.

Consider an efficient distinguisher $\mathcal{D}$ as defined in Definition 2.6 that distinguishes a distribution D from being uniformly random. Clearly, if D is the distribution generated by LWE-samples, then the distinguisher $\mathcal{D}$ would also serve as an efficient algorithm to solve the (decision) LWE problem. We formally define this below.

Definition 3.2 (Distinguishers for LWE$_{d,q,\chi,m}$) *An algorithm $\mathcal{D}$ is a distinguisher for the (decision)* LWE$_{d,q,\chi,m}$ *problem if, given m independent samples from $\mathbb{Z}_q^d \times \mathbb{Z}_q$, it distinguishes m* LWE*-samples $\{\mathbf{a}_i, b_i^{(1)}\}_i$ from m uniformly random samples $\{\mathbf{a}_i, b_i^{(2)}\}_i$ with non-negligible advantage i.e., there exists a non-negligible function $\eta : \mathbb{N} \to \mathbb{R}$ such that*

$$\left| \Pr_{(\mathbf{a}_i, b_i^{(1)})} [\mathcal{D}(\{\mathbf{a}_i, b_i^{(1)}\}_i) \text{ outputs } 1] - \Pr_{(\mathbf{a}_i, b_i^{(2)})} [\mathcal{D}(\{\mathbf{a}_i, b_i^{(2)}\}_i) \text{ outputs } 1] \right| \ge \eta(d).$$

When $\mathcal{D}$ is a $\mathrm{poly}(d,m)$*-time probabilistic (resp. quantum) algorithm, it is said to be an* efficient *classical (resp. quantum) distinguisher.*

3.1.1 Hardness of LWE

For a suitable choice of parameters, it is believed that LWE is a hard problem to solve. This is based on believed the worst-case hardness of lattice-based problems such as GapSVP$_\gamma$ (decision version of the shortest vector problem) or SIVP (shortest independent vectors problem) [Reg09, Pei09]. We do not introduce these problems or discuss their hardness here; an interested reader can refer to Peikert's survey [Pei16] on the topic. The following theorems will be important for us in this paper.

Theorem 3.3 (Quantum Reduction [Reg09, Pei09]) *Let $d, q \ge 1$ and $\alpha \in (0,1)$ be such that $\alpha q \ge 2\sqrt{d}$. Let $D_{\mathbb{Z}_q,\alpha}$ denote the discrete Gaussian distribution over $\mathbb{Z}_q$ with standard deviation αq. Then there exists a quantum reduction from worst-case hardness of the d-dimensional* GapSVP$_{\widetilde{O}(d/\alpha)}$ *problem to the* LWE$_{d,q,D_{\mathbb{Z}_q,\alpha}}$ *problem.*[14]

[14]In order to view LWE$_{d,q,D_{\mathbb{Z}_q,\alpha}}$ in the framework of Definition 3.1, we can set B as a suitable function of αq, possibly a constant multiple of αq or even $\mathrm{poly}(\alpha q)$ and this ensures that $\chi := D_{\mathbb{Z}_q,\alpha}$ is B-bounded.

We remark that this theorem statement combines two distinct steps – (1) showing the hardness of the *search version* of the LWE problem (which asks to find the secret $\mathbf{s}$ when given m independent LWE-samples) from GapSVP and (2) proving the sample preserving equivalence of the search and decision versions of LWE [Reg09, Pei09]. Subsequent works succeeded in de-quantizing the reduction from [Reg09] except that the dependence on the dimension changed. Initially, the classical reduction in [Pei09] from $\mathsf{GapSVP}_{\widetilde{O}(d/\alpha)}$ to $\mathsf{LWE}_{d,q,D_{\mathbb{Z}_q,\alpha}}$ required that $q \geq 2^{d/2}$. Later, Braverski et al. [BLP+13] built on this result and other techniques from fully homomorphic encryption obtained a reduction with a polynomial modulus.

Theorem 3.4 (Classical Reduction [BLP+13]) *Let $d, q \geq 1$ and $\alpha \in (0,1)$ be such that $q = \mathrm{poly}(d)$ and $\alpha q \geq 2d$. Let $D_{\mathbb{Z}_q,\alpha}$ be the discrete Gaussian distribution over $\mathbb{Z}_q$ with standard deviation αq. Then there exists a classical reduction from the worst-case hardness of the d-dimensional $\mathsf{GapSVP}_{\widetilde{O}(d/\alpha)}$ problem to the $\mathsf{LWE}_{d^2,q,D_{\mathbb{Z}_q,\alpha}}$ problem.*

The corollary below provides a suitable choice for parameters that lead to hard LWE instances.

Corollary 3.5 *Let $d' \in \mathbb{N}$ and α, q be parameters such that $\alpha = 1/\sqrt{d'}$ and $\alpha q \geq 2\sqrt{d'}$. Let $\chi = D_{\mathbb{Z}_q,\alpha}$, the discrete Gaussian distribution over $\mathbb{Z}_q$ with standard deviation αq. If there exists a polynomial-time quantum distinguisher for $\mathsf{LWE}_{d,q,\chi,m}$, then there exists a polynomial-time quantum algorithm for the $\sqrt{d'}$-dimensional $\mathsf{GapSVP}_{\widetilde{O}(d'^{1.5})}$ problem.*

It is believed that there exists no polynomial-time quantum algorithms for the d-dimensional $\mathsf{GapSVP}_{\widetilde{O}(d^3)}$ problem [GG00], which implies that there are no quantum polynomial-time algorithms for LWE with the parameters as stated in Corollary 3.5.

3.1.2 Public-key encryption scheme based on LWE

In this section we describe the public-key cryptosystem proposed by Regev [Reg09] whose security is based on the hardness of LWE.

Definition 3.6 (LWE – PKE [Reg09]) *The $\mathsf{LWE}-\mathsf{PKE}_{d,q,m}$ public-key encryption scheme consists of:*

Key-generation: *Pick $\mathbf{s} \in \mathbb{Z}_q^d$ and $A \in \mathbb{Z}_q^{m\times d}$ uniformly at random from the respective supports. Draw $e \in \mathbb{Z}_q^d$ from the distribution χ, as defined in the LWE problem. Let $\boldsymbol{b} = A\mathbf{s} + e$ and output $\mathrm{K}_{\mathrm{priv}} = \boldsymbol{s}$ and $\mathrm{K}_{\mathrm{pub}} = (A, b)$.*

Encryption: *To encrypt a bit c using $\mathrm{K}_{\mathrm{pub}} = (A, \boldsymbol{b})$, pick $S \subseteq [m]$ uniformly at random and the encryption is $\left(\mathbf{1}_S^T \cdot A, \mathbf{1}_S^T \cdot \boldsymbol{b} + c\lfloor \frac{q}{2} \rfloor\right)$, where $\mathbf{1}_S \in \{0,1\}^m$ is defined as $\mathbf{1}_S(i) = 1$ if and only if $i \in S$*

Decryption: *In order to decrypt the ciphertext (a, b) using $\mathrm{K}_{\mathrm{priv}} = \boldsymbol{s}$, output 0 if $b - a^T\boldsymbol{s} \pmod q \leq \lfloor \frac{q}{4} \rfloor$, otherwise output 1.*

Theorem 3.7 *Let $d \in \mathbb{N}$, $\varepsilon > 0$ be a constant, $q = \mathrm{poly}(d)$ and $m \geq (1+\varepsilon)(d+1)\log q$ be polynomially bounded in d. Then, the probability of decryption error for $\mathsf{LWE}-\mathsf{PKE}_{d,q,m}$ is $2^{-\omega(d^2/m)}$. Moreover, an adversary can distinguish an encryption of 0 from an encryption of 1 in polynomial time with non-negligible advantage over a random guess iff there is a polynomial-time distinguisher for $\mathsf{LWE}_{d,q,\chi,m}$.*

The following property of LWE – PKE was proven by Klivans and Sherstov [KS09].

Lemma 3.8 (Lemma 4.3 [KS09]) *The decryption function of LWE – PKE can be computed by light degree-2 PTFs.*

3.2 Learning with Rounding (LWR)

While the hardness results for LWE suggest that it is a good candidate for quantum-secure cryptographic primitives, the amount of randomness that LWE schemes need to generate samples make it unsuitable for the construction of pseudo-random generators. For this reason, we consider the Learning with Rounding problem (LWR) which was introduced by Banerjee, Peikert and Rosen [BPR12] and can be seen as a deterministic variant of LWE.

Definition 3.9 ($\mathsf{LWR}_{d,q,p,m}$) *The (decision) Learning with Rounding problem with dimension d, modulus q and rounding modulus $p < q$ is defined as follows: on input m independent samples $\{(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^d \times \mathbb{Z}_p\}_i$ where the $\mathbf{a}_i$'s are sampled uniformly from $\mathbb{Z}_q^d$, distinguish (with non-negligible advantage) between the following two cases:*

- *(LWR-samples) There is a fixed secret $\mathbf{s}$ uniformly sampled from $\mathbb{Z}_q^d$ such that the b_is are* rounded products *with respect to $\mathbf{s}$, i.e., $b_i = \lfloor \mathbf{a}_i \cdot \mathbf{s} \rceil_p$ for every i.*
- *(Uniform samples) For every i, b_i is uniformly sampled from $\mathbb{Z}_p$ and is independent of $\mathbf{a}_i$.*

Again, when the number of samples is arbitrary, the corresponding problem is denoted $\mathsf{LWR}_{d,q,p}$. Similar to Definition 3.2 of distinguishers for $\mathsf{LWE}_{d,q,\chi,m}$ one could analogously define classical and quantum distinguishers for the $\mathsf{LWR}_{d,q,p,m}$ problem as well.

3.2.1 Hardness of LWR

There have been many results studying the hardness of the LWR problem. We first discuss a result of Banerjee et al. [BPR12] which is simple-to-state and shows the hardness of the LWR problem based on the hardness of the LWE problem. Although their reduction is sub-optimal in terms of the parameters, we later state a result of Bogdanov et al. [BGM+16] which improves upon their result with better dependence on the parameters.

Theorem 3.10 *([BPR12, Theorem 3.2]) Let $d > 0$ and χ be an efficiently sampleable B-bounded distribution over $\mathbb{Z}$. Fix $q \geq p \cdot B \cdot d^{\omega(1)}$. Then, for any distribution over secrets $\mathbf{s} \in \mathbb{Z}_q^d$, solving $\mathsf{LWR}_{d,q,p}$ is as hard as solving $\mathsf{LWE}_{d,q,\chi}$ for the same distribution over secrets.*

One advantage of the reduction by Banerjee et al. [BPR12] is that this result holds for an unbounded number of samples but the *caveat* is that q is *required* to be super-polynomial in d. The reduction proceeds by showing that a distinguisher for $\mathsf{LWR}_{d,q,p}$ can act as a distinguisher for $\mathsf{LWE}_{d,q,\chi}$. Specifically, they prove that this occurs with high probability for a fixed secret $\mathbf{s}$, if the distribution of $\mathsf{LWR}_{d,q,p}$-samples is statistically close to the distribution of $\mathsf{LWE}_{d,q,\chi}$-samples. In other words, they require that except for a few "bad cases", most likely $\lfloor \mathbf{a}_i \cdot s \rceil_p = \lfloor \mathbf{a}_i \cdot s + e_i \rceil_p$ when e_i is sampled from χ. However for this to hold with high probability, the LWE error e_i needs to be very small in magnitude relative to q/p. Since the error e_i scales as $1/\mathrm{poly}(d)$ for hard instances of LWE, when the modulus q scales super-polynomially in d, the bad cases where $\lfloor \mathbf{a}_i \cdot s \rceil_p \neq \lfloor \mathbf{a}_i \cdot s + e_i \rceil_p$ occur with negligible probability.

We now connect the hardness of LWR to the hardness of LWE. Suppose there exists an efficient distinguisher for $\mathsf{LWR}_{d,q,p,m}$ that distinguishes LWR-samples from uniform samples with an advantage δ. Then, the using the reduction from Theorem 3.10, we can construct an efficient distinguisher for the $\mathsf{LWE}_{d,q,\chi,m}$ problem for $q \geq d^{\omega(1)}$, with an advantage $\delta - O(mBp/q)$. Unfortunately, this does not suffice to, *a priori*, prove the *quantum hardness* for LWR since there are no known reductions from worst-case hard lattices problems to $\mathsf{LWE}_{d,q,\chi}$ when q scales super-polynomially in d.

However, the result of Banerjee et al. [BPR12] has been improved upon in recent years by a number of works [AKPW13, BGM+16, AA16] with more sophisticated analysis. In these results it suffices to pick $q \geq \text{poly}(d)$ and in the regime where $q = \Theta(\text{poly}(d))$, we can relate the hardness of LWR from that of LWE. We now state the hardness reduction from [BGM+16] as it introduces the least number of new parameters.

Lemma 3.11 (LWR hardness from [BGM+16]) *Let $B > 0$, χ be a B-bounded distribution and $d \in \mathbb{N}$. Let $q, p, m = \text{poly}(d)$ such that $q \geq 2mBp$. Let $d' \in \mathbb{N}$ such that $d' > d/\log q$. If there exists an efficient distinguisher for the $\mathsf{LWR}_{d,q,p,m}$ with advantage $\varepsilon \geq 1/\text{poly}(d)$, then there exists an efficient distinguisher for the $\mathsf{LWE}_{d',q,\chi,m}$ problem that has advantage $\varepsilon' \geq O((\varepsilon/qm)^2)$.*

This lemma now allows us to connect distinguishers for LWR to the distinguishers for LWE.

Corollary 3.12 *Let $B > 0$ and $d \in \mathbb{N}$. Also, let $p, q, m = \text{poly}(d)$ such that $q \geq 2mBp$. Let $d' > d/\log q$, $\alpha \in \{2\sqrt{d'}/q, \ldots, B/6q\}$ and $\chi = D_{\mathbb{Z}_q,\alpha}$. If there exists a quantum distinguisher for the $\mathsf{LWR}_{d,q,p,m}$ problem that runs in time polynomial in d, then there exists a quantum distinguisher for the $\mathsf{LWE}_{d',q,\chi,m}$ problem with run-time also polynomial in d.*

3.2.2 PRGs from the LWR problem

We define a pseudo-random generator based on the LWR problem following the construction in [BPR12]. In the rest of this paper, for the sake of simplicity, we will assume p and q are powers of 2 ensuring that $\log p$ and $\log q$ are integral values.

Definition 3.13 (LWR−PRG) *The $\mathsf{LWR}_{d,q,p,m}$ problem for $q, p, m = poly(d)$ and $p|q$ defines a pseudo-random generator $\mathsf{LWR{-}PRG}_{d,q,p,m}$ as follows: on input a uniformly random (public) matrix $A \in \mathbb{Z}_q^{d\times m}$ and a secret seed $\mathbf{s} \in \mathbb{Z}_q^d$, the PRG outputs a pseudo-random vector in $\mathbb{Z}_p^m$, defined by $G_A : \mathbb{Z}_q^d \to \mathbb{Z}_p^m$ where $G_A(\mathbf{s}) = \lfloor A^t \cdot \mathbf{s} \rceil_p$. The seed length of this PRG is $s := d \log q$ bits and its stretch function is given by $\ell := m \log p$ bits.*

Note that, for a suitable choice of q, p and m that are polynomially bounded in d, it is possible to achieve an arbitrary polynomial for the stretch function, i.e., $\ell = O(\text{poly}(d))$.

Lemma 3.14 *Let $d \in \mathbb{N}$. Let $q, p, m = \text{poly}(d)$ such that $p < q$. Then, the $\mathsf{LWR{-}PRG}_{d,q,p,m}$ is a quantum-secure PRG.*

Proof. The proof derives the security of the LWR−PRG from the conjectured quantum hardness of LWE and GapSVP for a suitable choice of parameters. Let $A \in \mathbb{Z}_q^{d\times m}$ denote the random public matrix used to instantiate the PRG. By way of contradiction, suppose that the $\mathsf{LWR{-}PRG}_{d,q,p,m}$ G_A is not quantum-secure. Let ℓ denote the stretch function of G_A. Then, there exists a polynomial-time quantum distinguisher $\mathcal{D}$ that can determine if the bit sequence $z = z_0 z_1 \ldots z_{\ell-1}$ is an output of G_A or is a uniformly sampled bit string. From Definition 3.13, $\ell = m \log p$ and so one can view the bit sequence z as m numbers $\{b_i\}_i$ where each $b_i \in \mathbb{Z}_p$. Using this m-number sequence and the matrix A used by the PRG G_A, we can construct m samples $\{\mathbf{a}_i, b_i\}_i$ where $\mathbf{a}_i$ is the ith column of A and b_i is the ith number in the m-number sequence. Clearly, when z is an output of G_A, these samples correspond to m-LWR samples. Otherwise, they correspond to m uniform samples from $\mathbb{Z}_p$. In this way, $\mathcal{D}$ becomes a distinguisher that can differentiate between m LWR-samples and m uniformly sampled numbers from $\mathbb{Z}_p$. Then, from Corollaries 3.5 and 3.12, we can use $\mathcal{D}$ to construct polynomial-time quantum algorithms for the corresponding LWE and GapSVP problems. This contradicts the original assumption on the LWE hardness [Reg09, Pei09], and thereby contradicts the assumption that $\mathsf{LWR{-}PRG}_{d,q,p,m}$ is not quantum-secure. This concludes the proof of the lemma. □

A corollary of this lemma is that the parameters of the LWR−PRG can be chosen such that it is quantum-secure for every polynomial stretch.

Lemma 3.15 *Let $d \in \mathbb{N}$ and $q, p = \text{poly}(d)$. Also, let $s := d \log q$. Then, for every polynomial $\ell(s) > s$ there exists an m and for every uniformly sampled $A \in \mathbb{Z}_q^{d\times m}$, there is an $\mathsf{LWR{-}PRG}_{d,q,p,m}$ G_A that outputs the bit sequence $z_0 z_1 \cdots z_{\ell(s)-1}$ and is a quantum-secure pseudo-random generator. Moreover, given an index $0 \leq i < \ell(s)$, the output bit z_i can be computed by a TC^0 circuit.*

Proof. For every choice of $\ell(s)$, set $m = \ell(s)/\log p$. Now consider the $\mathsf{LWR{-}PRG}_{d,q,p,m}$ G_A whose stretch is exactly $m \log p = \ell(s)$. Then from Lemma 3.14, G_A is quantum-secure.

Additionally, given an index $i \in \{0,\ldots,\ell(s)-1\}$, we can choose $j \in \{0,\ldots,m\}$ and $t \in \{0,\ldots,\log p\}$ be such that they satisfy the following three equalities

$$i = j\log p + t, \quad j = \lfloor i/\log p \rfloor \quad \text{and} \quad t \equiv i \pmod{\log p}. \tag{4}$$

Note that since $i \leq \ell(s) = m\log p$, it follows that $j \leq m$. In order to compute the output bit z_i, take the j^{th} column of the matrix A and calculate the binary representation of $\lfloor \langle A_j, \mathbf{s}\rangle \rceil_p$. By definition, the t^{th} bit of this binary number equals z_i.

Observe that all the operations performed to determine j, t and b_i are arithmetic operations either taken modulo p or modulo q. The inner product can be viewed as a sum of d binary scalar products modulo q. Since $q = \text{poly}(d)$, the d-dimensional inner product $\langle A_j, \mathbf{s}\rangle$ can be performed by a TC^0 circuit on d bits. As p and q are assumed to be powers of 2, the $\lfloor\cdot\rceil_p$ task effectively reduces to just truncating some of the most significant digits of the inner product. Using the fact that, the inner product, the arithmetic operations and rounding modulo p can be efficiently computed by TC^0 circuits for $p, q, m \leq \text{poly}(d)$ [RT92, HAB02], it follows that, given an index i, z_i can be computed by a TC^0 circuit on $\widetilde{O}(d)$ bits. □

One problem that prevents us from computing an arbitrary bit of the quantum-secure LWR−PRG in a smaller circuit class, say AC^0, lies in the inner product operation where we need to add d numbers modulo q. However, using the observation that arithmetic operations on $O(\log n)$ bit numbers can be computed by an AC^0 circuit [MT98] helps us overcome this issue.[15] For this reason, we consider the LWR−PRG with a shorter seed length i.e., by setting $d = \text{polylog}(n)$ and considering $\text{poly}(n)$-sized AC^0 circuits. This resembles the technique used by Kharitonov [Kha93] to show that some instances of the BBS PRG can be implemented in AC^0.

Lemma 3.16 *Consider an AC^0 circuit on n bits. Let d, q, p be such that $d = \log^\gamma n$ for some constant $\gamma > 2$, $p = \text{poly}(d)$ and $q = n^{O(\log n)}$ where $p \mid q$ and p is a power of 2. Let $s(n) := d\log q$. Then, for every polynomial $\ell(n) > s(n)$, there exists an m and for every uniformly sampled $A \in \mathbb{Z}_q^{d\times m}$, there exists an $\mathsf{LWR{-}PRG}_{d,q,p,m}$ $G_A : \{0,1\}^{s(n)} \to \{0,1\}^{\ell(n)}$ such that: given $z \in \mathsf{range}(G_A)$ and $i \in \{0,\ldots,\ell(n)-1\}$, the bit z_i can be computed by a $\text{poly}(n)$-sized AC^0 circuit.*

[15] Basic arithmetic operations can be expressed by circuits containing AND, NOT, OR and MAJ gates. MAJ gates with $O(\text{polylog}\, n)$-bit fan-in can be computed by AC^0 circuits.

Proof. The proof follows the ideas used in the second part of the proof of Lemma 3.15. We show that an arbitrary output bit of the PRG G_A can be computed by an AC^0 circuit. As before, set $m := \ell(n)/\log p$. Note that with $q = n^{O(\log n)} = 2^{O(d^{2/\gamma})}$ and $m = \ell(n)/\log p = O(\mathrm{poly}(n)) = 2^{O(d^{1/\gamma})}$, both q and m grow super-polynomially with respect to d. In order to compute the bit z_i, observe that the operation $\lfloor\langle A_j, \mathbf{s}\rangle\rceil_p$ requires the ability to multiply and add numbers from $\mathbb{Z}_q$ and round it to a value in $\mathbb{Z}_p$. It is known that addition, subtraction and multiplication of $O(\mathrm{polylog}\, n)$-bit numbers can be performed in AC^0 [MT98]. This would let us determine the indices j and t, defined as in Eq. 4, in AC^0. Since the length of numbers in $\mathbb{Z}_q$ can be bounded by $O(\log^2 n)$ bits and the vectors are of length $O(\mathrm{polylog}\, n)$, the inner product $\langle A_j, \mathbf{s}$ can also be calculated in AC^0.[16] Since p is a power of 2, in order to compute $\lfloor\cdot\rceil_p$, we would only need to consider the last $\log p = O(\log d) = O(\log\log n)$ digits to obtain the number modulo p and this can efficiently be calculated in AC^0. Hence, the $\mathsf{LWR-PRG}$ G_A with $O(\mathrm{polylog}\, n)$-sized dimension can be implemented using an AC^0 circuit. □

4 Quantum-secure PRGs vs. quantum learning

In this section we prove our main theorem which shows the connection between efficient quantum learning and quantum algorithms which serve as distinguishers for pseudo-random generators. We give two proofs of the main theorem, an information-theoretic argument and a hybrid-method argument. The information-theoretic argument is fairly intuitive and follows ideas similar to the original work of Kharitonov [Kha93]. The second proof technique uses the hybrid-method argument (introduced by Bennett et al. [BBBV97]) and gives a different (incomparable) dependence on the parameters of the PRG and the learning algorithm.

The first step while considering a quantum learning algorithm is to define a suitable concept class which the learner aims to learn. For $n, n' > 0$, and $f : \{0,1\}^n \to \{0,1\}^{n'}$, we define a generic concept class $\mathcal{C}_f$ which depends on f. For simplicity of notation, we drop the dependence on f and just denote the class as $\mathcal{C}$ – the function used to create $\mathcal{C}$ will be clear from context. It is formally defined as follows:

Definition 4.1 (Concept Class) *Let $G_n : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ be a function where $\ell(n) > n$. For $k > 1$, let $G_{n,k}$ denote the k-extension of G_n. Define the concept class $\mathcal{C} := \bigcup_{k=1}^n \mathcal{C}_k$ where*

$$\mathcal{C}_k := \{c_z : \{0,1\}^n \to \{0,1\} \mid c_z(x) = z_{x \pmod{2^k}},\ \ z \in \mathsf{range}(G_{n,k})\ \} \tag{5}$$

The use of the k-extension allows us define the concept class independent of any assumption on the function G. We also remark that x in the expression $z_{x \pmod{2^k}}$ is viewed as an integer in $[2^n]$. Now, one can connect an efficient learner for concept class $\mathcal{C}$ to the existence of a distinguisher for G_n (i.e., identify if a Boolean string is part of the output sequence of G_n or is uniformly random) using the main theorem stated below.

Theorem 4.2 *Let $n \geq 2$. For $\ell(n) > n$, let $G_n : \{0,1\}^n \to \{0,1\}^{\ell(n)}$. For every $k > 1$, let $G_{n,k} : \{0,1\}^n \to \{0,1\}^{2^k}$ denote the k-extension of G_n. For this G_n, let $\mathcal{C}$ be the concept class as defined in Definition 4.1. Suppose there exists a $t(n)$-time uniform quantum-PAC learner for $\mathcal{C}$ (given access to q quantum membership queries and uniform quantum examples) with bias $\beta(n)$. If*

$$\ell(n) > \min\left\{\frac{n \cdot \mathsf{q}}{\beta(n)^2}, \frac{4\mathsf{q}^2}{\beta(n)^2}\right\}, \tag{6}$$

[16]The complexity of adding $(\log n)$ n-bit integers is in AC^0 as sketched in [Fil15]

then there exists a $(t(n)+\mathsf{q}(n)\ell(n))$-time quantum algorithm $\mathcal{D}:\{0,1\}^{\ell(n)}\to\{0,1\}$ that satisfies:

$$\left|\Pr_{x\in\{0,1\}^n}[\mathcal{D}(G_n(x))=1]-\Pr_{y\in\{0,1\}^{\ell(n)}}[\mathcal{D}(y)=1]\right|\geq\frac{\beta(n)}{2},\tag{7}$$

where the probability is taken over uniform $x\in\{0,1\}^n$ and $y\in\{0,1\}^{\ell(n)}$.

A key part of the proof for this theorem hinges on bounding how well a quantum algorithm predicts a random bit of an unknown uniformly random string z. We assume that the algorithm is allowed to a query some bit of the string (in superposition) at most q times and call this performing a membership query to z. Then, we can upper bound the probability of predicting a random bit of z as stated below.

Lemma 4.3 *Let $n>1$ and $z\in\{0,1\}^{\ell(n)}$ be a uniformly random string. Consider a quantum algorithm that makes q quantum membership queries to z. Given a uniformly random question $x\in\{0,1\}^n$, the probability of the quantum algorithm predicting $z_{x \pmod{\ell(n)}}$ correctly is*

$$\Pr_{x\in\{0,1\}^n}[h_x=z_{x \pmod{\ell(n)}}]\leq\frac{1}{2}+\min\left\{\sqrt{\frac{n\cdot\mathsf{q}}{\ell(n)}},\frac{2\mathsf{q}}{\sqrt{\ell(n)}}\right\},\tag{8}$$

where h_x is the output of the algorithm given the question x.

We now make a remark on the two bounds in the lemma above. Firstly note that for $\mathsf{q}=\Omega(n)$, we have $\sqrt{\frac{n\cdot\mathsf{q}(n)}{\ell(n)}}<\frac{2\mathsf{q}}{\sqrt{\ell(n)}}$. Generally, when proving the hardness of learning concept classes, the number of membership queries (i.e., q) could be an arbitrary polynomial in n, in which case it always suffices to consider the upper bound of $\frac{1}{2}+\sqrt{\frac{n\cdot\mathsf{q}(n)}{\ell(n)}}$ in Eq. 8. So, a critical reader might wonder the need to prove the upper bound of $\frac{1}{2}+\frac{2\mathsf{q}}{\sqrt{\ell(n)}}$ in Eq. 8. Later, we will use the second bound for a more *fine-grained* analysis to give a *lower bound* on the query complexity of polynomial-time quantum learning algorithms. More concretely, in Appendix B.3, we use the second bound in Eq. 8 to show that efficient quantum learning algorithms for AC^0 needs to make $\Omega(\sqrt{n})$ quantum membership queries based on the hardness of the $\mathsf{Subset-sum}$ problem. In order to prove this lower bound, we use a PRG that can be computed in AC^0 on n bits but has only logarithmic stretch [IN96] (i.e., the stretch function is $n+O(\log n)$), and in this case the term in the first bound in Eq. 8 does not give us anything meaningful.

We now proceed to the proof of Lemma 4.3.

Proof of Lemma 4.3. This proof combines two cases namely, showing that for some x drawn uniformly from $\{0,1\}^n$, $\Pr_{x\in\{0,1\}^n}[h_x=z_{x \pmod{\ell(n)}}]$ is: (a) at most $\frac{1}{2}+\sqrt{\frac{n\cdot\mathsf{q}}{\ell(n)}}$ using an information-theoretic approach; and (b) at most $\frac{1}{2}+\frac{2\mathsf{q}}{\sqrt{\ell(n)}}$ using the hybrid method. Clearly, if both of these bounds hold, then by combining them and using the smaller of the two, Eq. 8 also holds. We now prove each case separately.

(a) Information-theoretic proof: Without loss of generality, assume $\beta(n)=\sqrt{\frac{n\cdot\mathsf{q}}{\ell(n)}}$. On input some index x, view h_x as the output of a "hypothesis function" $h:\{0,1\}^n\to\{0,1\}$, such that $h_x=h(x)$. By contradiction, let us assume that the hypothesis function h satisfies

$$\Pr_{x\in\{0,1\}^n}\left[z_{x \pmod{\ell(n)}}=h(x)\right]>\frac{1}{2}+\frac{\beta(n)}{2},\tag{9}$$

where x is drawn uniformly from $\{0,1\}^n$. In this case, the mutual information between the random string z and the truth-table of the hypothesis h is given by

$$\mathbf{I}(z:h) = \sum_{x\in[\ell(n)]} \mathbf{I}(z_x:h) = \sum_{x\in[\ell(n)]} (\mathbf{H}(z_x) - \mathbf{H}(z_x|h)) = \ell(n) - \sum_{x\in[\ell(n)]} \mathbf{H}(z_x|h), \tag{10}$$

where the first and last equality used the independence of the z_xs since $z \in \{0,1\}^{\ell(n)}$ was uniformly random. By Fano's inequality (in Lemma 2.3), it follows that $\mathbf{H}(z_x|h) \le \mathbf{H}_b(p_x)$, where $\mathbf{H}_b(\cdot)$ is the binary entropy function and p_x is the probability of error on guessing z_x by an arbitrary estimator whose input is $h:\{0,1\}^n \to \{0,1\}$. By assumption of h in Eq. 9, it follows that $\frac{1}{\ell(n)}\sum_x p_x < \frac{1}{2} - \frac{\beta(n)}{2}$. Using this, we then have that

$$\sum_x \mathbf{H}(z_x|h) \le \max_{\{p_x\}} \sum_x \mathbf{H}_b(p_x),$$

where the maximization is over $\{p_x\}$ in the set $\left\{p_x : \frac{1}{\ell(n)}\sum_x p_x \le \frac{1}{2} - \frac{\beta(n)}{2}\right\}$. Since $\mathbf{H}_b$ is a concave function, the maximum is obtained when all the p_xs are equal (this also follows from Jensen's inequality). Given our upper-bound on the sum $\sum_x p_x$, it follows that $\sum_x \mathbf{H}_b(p_x)$ is maximized when $p_x = \frac{1}{2} - \frac{\beta(n)}{2}$ for every x. In this case, we have

$$\max_{\{p_x\}} \sum_x \mathbf{H}_b(p_x) \le \sum_x \mathbf{H}_b\left(\frac{1}{2} - \frac{\beta(n)}{2}\right) = \ell(n)\cdot \mathbf{H}_b\left(\frac{1}{2} - \frac{\beta(n)}{2}\right) \le \ell(n)\left(1 - \frac{2\beta(n)^2}{\ln 2}\right), \tag{11}$$

where we use Fact 2.2 in the last inequality. Putting together Eq. 10 and 11, we obtain

$$\mathbf{I}(z:h) > \frac{2\ell(n)\beta(n)^2}{\ln 2} > 2\cdot n\cdot \mathsf{q}. \tag{12}$$

We now upper bound the mutual information between the output hypothesis h and the uniformly random string z. One way to view the quantum algorithm is as a protocol where a quantum membership query to z is a message from the algorithm to an oracle hiding z and the oracle's output is a message from the oracle to the algorithm. In this case, using Corollary 2.4 the mutual information between the output hypothesis h and the random string z can be upper-bounded by the communication complexity of the protocol,

$$\mathbf{I}(h:z) \le (n+1)\cdot \mathsf{q}. \tag{13}$$

However, since $n \ge 2$ and by the assumption on $\beta(n)$, the lower bound in Eq. 12 contradicts Eq. 13, which in turn contradicts our assumption in Eq. 9.

(b) Hybrid method proof: Let the random index x for which the algorithm outputs h_x be denoted as x^*. Since $x^* \in [2^n]$ is picked uniformly at random and the algorithm has to predict $z_{x^* \pmod{\ell(n)}}$, we could without loss of generality assume that a algorithm is instead given $x^* \in [\ell(n)]$ uniformly at random and has to predict z_{x^*}. So from here on, we let $x^* \in [\ell(n)]$.

For $x^* \in [\ell(n)]$ and $b \in \{0,1\}$, let $z \in \{0,1\}^{\ell(n)}$ be a uniformly random string conditioned on $z_{x^*} = b$. One way to view the task is: given query access to $O_{z_{x^*}=b}$ for uniformly random $x^* \in [\ell(n)]$, $b \in \{0,1\}$ and $z \in \{0,1\}^{\ell(n)}$ (conditioned on $z_{x^*} = b$), an algorithm needs to distinguish between $O_{z_{x^*}=0}$ and $O_{z_{x^*}=1}$. Here we show that a q-query algorithm has success probability of at most $\frac{1}{2} + \frac{2\mathsf{q}}{\sqrt{\ell(n)}}$ in distinguishing between these two oracles. In order to prove this, we use an argument based on the hybrid method. A standard q-membership query quantum algorithm has the following structure

$$U_{\mathsf{q}+1} O_z U_{\mathsf{q}} O_z \cdots U_2 O_z U_1 |0\rangle,$$

where $z \in \{0,1\}^{\ell(n)}$ and the $|0\rangle$ register includes the workspace of the quantum algorithm and the action of O_z is given by $O_z : |x,b,w\rangle \to (-1)^{b\cdot z_x}|x,b,w\rangle$.

Fix $x^* \in [\ell(n)]$, $z \in \{0,1\}^{\ell(n)}$. For $b \in \{0,1\}$, consider the final states of a q-query algorithm

$$|\psi^{\mathsf{q}}_{z,x^*,b}\rangle = U_{\mathsf{q}+1} O_{z_{x^*}=b} U_{\mathsf{q}} O_{z_{x^*}=b} \cdots U_2 O_{z_{x^*}=b} U_1 |0\rangle.$$

For notational convenience, we will drop the $|\cdot\rangle$ around $\psi^{\mathsf{q}}_{z,x*,b}$ below. By writing out the action of the oracles, it is not hard to see that for every $x^* \in [\ell(n)]$, we have

$$O_{z_{x^*}=1} = O_{z_{x^*}=0} - 2|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W, \tag{14}$$

where $\mathbb{I}_W$ is the identity on the workspace register. We now show that for $t \geq 0$, we have

$$\|\psi^t_{z,x^*,0} - \psi^t_{z,x^*,1}\| \leq \sum_{k=1}^{t} \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\|. \tag{15}$$

First observe that,

$$\begin{aligned}
\|\psi^k_{z,x^*,0} - \psi^k_{z,x^*,1}\| &= \|U_{k+1} O_{x_j=0} \psi^{k-1}_{z,x^*,0} - U_{k+1} O_{x_j=1} \psi^{k-1}_{z,x^*,1}\| \\
&= \|U_{k+1} O_{x_j=0} \psi^{k-1}_{z,x^*,0} - U_{k+1}(O_{x_j=0} - 2|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W)\psi^{k-1}_{z,x^*,1}\| \\
&\leq \|U_{k+1} O_{x_j=0}(\psi^{k-1}_{z,x^*,0} - \psi^{k-1}_{z,x^*,1})\| + \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\| \\
&\leq \|\psi^{k-1}_{z,x^*,0} - \psi^{k-1}_{z,x^*,1}\| + \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\|,
\end{aligned}$$

where the second equality used Eq. 14 and both inequalities used the triangle inequality. Using the above inequalities, we now have

$$\sum_{k=1}^{t} \|\psi^k_{z,x^*,0} - \psi^k_{z,x^*,1}\| \leq \sum_{k=1}^{t} \|\psi^{k-1}_{z,x^*,0} - \psi^{k-1}_{z,x^*,1}\| + \sum_{k=1}^{t} \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\|.$$

Canceling the terms in the summation and observing $\|\psi^0_{z,x^*,0} - \psi^0_{z,x^*,1}\| = 0$ (since these states are independent of the query) gives us Eq. 15.

Consider a q-query algorithm satisfying the following: for a uniformly random $x^* \in [\ell(n)]$ and $z \in \{0,1\}^{\ell(n)}$ conditioned on $z_{x^*} \in \{0,1\}$, with probability at least $1/2+\beta$ the algorithm can distinguish between $\psi^{\mathsf{q}}_{z,x^*,0}$ and $\psi^{\mathsf{q}}_{z,x^*,1}$. Using Eq. 2.1 and Eq. 3, we have

$$\mathbb{E}_{z,x^*}[\|\psi^{\mathsf{q}}_{z,x^*,0} - \psi^{\mathsf{q}}_{z,x^*,1}\|_2] \geq 2\beta. \tag{16}$$

Using Eq. 15, we have

$$\begin{aligned}
\sum_{x^*=1}^{\ell(n)} \|\psi^{\mathsf{q}}_{z,x^*,0} - \psi^{\mathsf{q}}_{z,x^*,1}\| &\leq \sum_{k=1}^{\mathsf{q}} \sum_{x^*=1}^{\ell(n)} \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\| \\
&\leq \sqrt{\ell(n)} \sum_{k=1}^{\mathsf{q}} \sum_{x^*=1}^{\ell(n)} \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\|^2 \leq 4\mathsf{q}\sqrt{\ell(n)}.
\end{aligned} \tag{17}$$

The second inequality was by Cauchy-Schwartz-inequality and the final inequality follows from

$$\begin{aligned}
&\sum_{x^*=1}^{\ell(n)} \|2U_{k+1}|x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1}\|^2 \\
&= 4\sum_{x^*} (\psi^{k-1}_{z,x^*,1})^* \cdot |x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W U^*_{k+1} U_{k+1} |x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1} \\
&= 4\sum_{x^*} (\psi^{k-1}_{z,x^*,1})^* \cdot |x^*,1\rangle\langle x^*,1| \otimes \mathbb{I}_W \cdot \psi^{k-1}_{z,x^*,1} = 4\sum_{x^*}\sum_{w} \langle \psi^{k-1}_{z,x^*,1}|x^*,1,w\rangle^2 \leq 4,
\end{aligned}$$

where the last inequality used the fact that $\psi^{k-1}_{z,x^*,1}$ is a quantum state. Putting together

Eqs. 16 and 17, we have

$$\frac{4\mathsf{q}}{\sqrt{\ell(n)}} \geq \mathbb{E}_{z,x^*}[\| |\psi^{\mathsf{q}}_{z,x^*,0} - \psi^{\mathsf{q}}_{z,x^*,1}\| |] \geq 2\beta,$$

hence the bias in obtaining the correct outcomes is at most $\frac{2\mathsf{q}}{\sqrt{\ell(n)}}$.

□

Now, we can prove Theorem 4.2. It suffices to show that there is some sub-class of concepts in $\mathcal{C}$, say $\mathcal{C}_k$ for some specific k, that is hard to learn thereby making the whole class hard to learn.

Proof of Theorem 4.2. Let $\mathcal{A}$ be a $t(n)$-time uniform quantum-PAC learner that makes at most q quantum queries to a concept $c \in \mathcal{C}$ and outputs a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that

$$\Pr_{x\in\{0,1\}^n}[c(x) = h(x)] \geq \frac{1}{2} + \beta(n),$$

where the probability is over x drawn uniformly from $\{0,1\}^n$. Note that $\mathcal{A}$ can obtain a uniform quantum example by making a single quantum membership query, so without loss of generality we assume $\mathcal{A}$ makes membership queries.

The goal is to use $\mathcal{A}$ to construct a quantum distinguisher $\mathcal{D}$ for the PRG G_n with stretch $\ell(n)$ such that $\mathcal{D}$ satisfies Eq. 7. Notice that, when $k = \log \ell(n)$, $G_{n,k} = G_n$ i.e., there is neither any truncation nor any padding of the output. Hence, in the remainder of the proof, we consider concepts $c \in \mathcal{C}_{\log \ell(n)}$ in which case, $z \in \{0,1\}^{\ell(n)}$. By showing that $\mathcal{C}_{\log \ell(n)}$ is hard to learn, we conclude that $\mathcal{C}$ is hard to learn.

Let $z \in \{0,1\}^{\ell(n)}$ be the input string to a distinguisher $\mathcal{D}$, whose goal is to decide if z is a uniformly random string or if $z \in \mathsf{range}(G_n)$. In order to do this, the distinguisher $\mathcal{D}$ proceeds by first running the quantum learning algorithm $\mathcal{A}$ as follows: when $\mathcal{A}$ makes a quantum membership query, $\mathcal{D}$ performs the following unitary transformation

$$\mathcal{U}_z : |x\rangle|b\rangle \to |x\rangle|b \oplus z_{x \pmod{\ell(n)}}\rangle \quad \text{for every } x \in \{0,1\}^n, b \in \{0,1\}. \tag{18}$$

We remark that $\mathcal{U}_z$ can be performed *efficiently* (i.e., in time $\ell(n)$) by $\mathcal{D}$ since the function $z_{x \pmod{\ell(n)}}$ is cyclic in x, i.e., it only depends on the $\ell(n)$ bits of the input string z. After making q membership queries, $\mathcal{A}$ then outputs a hypothesis h. The distinguisher $\mathcal{D}$ outputs 1 (i.e., z is a pseudo-random string) if and only if $h(x) = z_{x \pmod{\ell(n)}}$ for a uniformly random $x \in \{0,1\}^n$.

The running time of $\mathcal{D}$ is at most $t(n) + \mathsf{q}\cdot\ell(n)$ since $\mathcal{A}$ runs in time $t(n)$ and makes at most $\mathsf{q}(n)$ membership queries, and each unitary transformation $\mathcal{U}_z$ in Eq. 18 used to implement a membership query takes time at most $\ell(n)$. We now show that $\mathcal{D}$ satisfies Eq. 7. Suppose $z \in \{0,1\}^{\ell(n)}$ was the output of a PRG G_n, then the the unitary transformation $\mathcal{U}_z$ exactly corresponds to a quantum membership query for the concept c_z (since we have defined $c_z(x) = z_{x \pmod{\ell(n)}}$ for every x). Hence, we have

$$\Pr_{x\in\{0,1\}^n}[\mathcal{D}(G_n(x)) = 1] = \Pr_{x\in\{0,1\}^n}[c_z(x) = h(x)] \geq \frac{1}{2} + \beta(n), \tag{19}$$

where the inequality follows from the correctness of the quantum learning algorithm $\mathcal{A}$.

On the other hand, if z was a random string, we have from Lemma 4.3, that

$$\Pr_{y\in\{0,1\}^{\ell(n)}}[\mathcal{D}(y) = 1] = \Pr_{x\in\{0,1\}^n}[c_z(x) = h(x)] \leq \frac{1}{2} + \min\left\{\sqrt{\frac{n\cdot\mathsf{q}}{\ell(n)}}, \frac{2\mathsf{q}}{\sqrt{\ell(n)}}\right\} < \frac{1}{2} + \frac{\beta(n)}{2},$$

where the last inequality uses the definition of $\beta(n)$ in the theorem statement.

In this case, the bias of the distinguisher $\mathcal{D}$ is

$$\left|\Pr_{x\in\{0,1\}^n}[\mathcal{D}(G_n(x))=1]-\Pr_{y\in\{0,1\}^{\ell(n)}}[\mathcal{D}(y)=1]\right|\geq\frac{\beta(n)}{2}.$$

This concludes the proof. □

5 Quantum-secure encryption vs. quantum learning

In this section, we prove a theorem relating the security of quantum public-key cryptosystems and the hardness of quantum learning the class of decryptions functions in the cryptosystem.

Kearns and Valiant [KV94] showed the following simple, yet powerful connection between learning and public-key cryptography: consider a *secure* public-key cryptosystem and define a concept class $\mathcal{C}$ as the set of all decryption functions (one for every public and private key). Let us assume that there was an *efficient* PAC learner for $\mathcal{C}$. Given a set of encryptions (which should be viewed as labelled examples), supposing an efficient learner for $\mathcal{C}$ was able to learn an unknown decryption function with non-negligible advantage, then this learner would break the cryptosystem. This contradicts the assumption that the cryptosystem was secure and, in turn, shows the infeasibility of learning $\mathcal{C}$. Kearns and Valiant used this connection to give hardness results for learning polynomial-sized formulas, deterministic finite automata, and constant-depth threshold circuits based on the assumption that Blum integers are hard to factor or that it is hard to invert the RSA function. Our main contribution is to quantize the theorem by Kearns and Valiant [KV94] and draw a relation between quantum learning and security of quantum cryptosystems.

Theorem 5.1 *Consider a public-key cryptosystem which encrypts bits by n-bit strings. Suppose the (randomized) encryption function is given by $e_{\mathsf{K}_{\text{pub}},r}:\{0,1\}\to\{0,1\}^n$ (where $K=(\mathsf{K}_{\text{pub}},\mathsf{K}_{\text{priv}})$ consists of the public and private keys and r is a random string) and decryption function is given by $d_{\mathsf{K}_{\text{priv}}}:\{0,1\}^n\to\{0,1\}$. Let $\mathcal{C}=\{d_{\mathsf{K}_{\text{priv}}}:\{0,1\}^n\to\{0,1\}\}_{\mathsf{K}_{\text{priv}}}$ be a concept class. Let*

$$\varepsilon(n)=\Pr_{K,r}[d_{\mathsf{K}_{\text{priv}}}(e_{\mathsf{K}_{\text{pub}},r}(0))\neq 0 \text{ or } d_{\mathsf{K}_{\text{priv}}}(e_{\mathsf{K}_{\text{pub}},r}(1))\neq 1],$$

be the probability of decryption error (where the probability is taken over uniformly random $K=(\mathsf{K}_{\text{pub}},\mathsf{K}_{\text{priv}})$ and r). If there exists a weak quantum-PAC learner for $\mathcal{C}$ in time $t(n)\geq n$ that satisfies $t(n)\beta(n)=1/n^{\omega(1)}$, then there exists a $t(n)$-time quantum algorithm Adv *that satisfies*

$$\left|\Pr_{K,r^*,b^*}[\mathsf{Adv}(\mathsf{K}_{\text{pub}},e_{\mathsf{K}_{\text{pub}},r^*}(b^*))=b^*]-\Pr_{K,r^*,b^*}[\mathsf{Adv}(\mathsf{K}_{\text{pub}},e_{\mathsf{K}_{\text{pub}},r^*}(b^*))=\overline{b^*}]\right|\geq\frac{1}{\mathrm{poly}(n)},\qquad(20)$$

where the probability is taken over uniformly random K,r^*,b^* over their respective domains.

Proof. For notational simplicity, for a fixed K_{pub}, let $e_{r,b}:=\mathsf{Enc}_{\mathsf{K}_{\text{pub}},r}(b)$ and $\mathcal{E}=\{e_{r,b}:r,b\}$ be the set of all encryptions for the public key K_{pub}. Our goal here is to devise a quantum algorithm Adv that satisfies the following: on input $(\mathsf{K}_{\text{pub}},e^*)$, where $e^*=e_{r^*,b^*}$ is uniformly randomly chosen from $\mathcal{E}$, Adv outputs $\hat{b}$ such that $\hat{b}=b^*$ with $1/\mathrm{poly}(n)$ advantage over a random guess. Adv would clearly serve as our quantum algorithm that satisfies Eq. 20, thereby proving the theorem.

Our approach is to devise an Adv so that the remainder of our reasoning resembles the (classical) proof of this theorem by Klivans and Sherstov [KS09] (who in turn re-derived the proof of Kearns and Valiant [KV94] in their paper). However the quantization of their proof is not straightforward. Classically, it is possible to generate samples from the uniform distribution supported on $\mathcal{E}$ as follows: pick r and b uniformly at random from their respective domains and then output the uniform classical examples $(e_{r,b},b)$. Quantumly, it is not clear how an adversary Adv could create a quantum example

$$\frac{1}{\sqrt{|\mathcal{E}|}}\sum_{r,b}|e_{r,b}\rangle|b\rangle. \tag{21}$$

Notice that if an adversary could prepare the state in Eq. 21, then it could directly pass $t(n)$ quantum examples to the quantum-PAC learner (assumed in the theorem statement) and a similar analysis as in Klivans and Sherstov [KS09] give a quantum algorithm that satisfies Eq. 20. However, one issue while preparing the state in Eq. 21 is the following: the straightforward way to prepare the state Eq. 21 would be create the uniform superposition $\frac{1}{\sqrt{|\mathcal{E}|}}|r\rangle|b\rangle|0\rangle$ and then in superposition prepare $\frac{1}{\sqrt{|\mathcal{E}|}}|r\rangle|b\rangle|e_{r,b}\rangle$. If there was a way to erase the first register, then we would get the state in Eq. 21. However, erasing this register is equivalent to solving the index-erasure problem [AMRR11, LR19] and in general solving the index-erasure problem needs exponential time.

In order to circumvent the issue mentioned above, we now show how to tweak the proof and construct an adversary $\mathsf{Adv}_{\text{real}}$ that correctly decrypt a challenge encryption (i.e., e_{r^*,b^*}) with non-negligible probability. Before constructing $\mathsf{Adv}_{\text{real}}$, we first present an adversary $\mathsf{Adv}_{\text{ideal}}$ that is able to create specific samples (see Eq. 22) that $\mathsf{Adv}_{\text{real}}$ could not create and analyze its success probability. Then we show that the success probability of $\mathsf{Adv}_{\text{real}}$ and $\mathsf{Adv}_{\text{ideal}}$ in predicting the challenge encryption (i.e., e_{r^*,b^*}) are very close.

Let $S_{\text{ideal}} = \{(r_i, b_i)_i\}$ be a fixed set of size $L = 2n^c t(n)$ that contains (r^*, b^*) and let us assume $\mathsf{Adv}_{\text{ideal}}$ has access to S_{ideal}. Then $\mathsf{Adv}_{\text{ideal}}$ can create the quantum example

$$|\psi_{\text{ideal}}\rangle = \frac{1}{\sqrt{L}}\sum_{(r,b)\in S_{\text{ideal}}}|e_{r,b}, b\rangle. \tag{22}$$

Notice that since $\mathsf{Adv}_{\text{real}}$ does not know the decryption of e_{r^*,b^*}, it would most likely not be able to obtain such an S_{ideal} and consequently would not be able to create $|\psi_{\text{ideal}}\rangle$ efficiently. $\mathsf{Adv}_{\text{ideal}}$ then passes on $t(n)$ copies of $|\psi_{\text{ideal}}\rangle$ to the quantum-PAC learning algorithm for $\mathcal{C}$ (which is assumed by the theorem statement). Let $\mathcal{H}(\mathcal{D})$ be the distribution of the hypothesis output by the quantum-PAC learning algorithm when the samples come from the distribution $\mathcal{D}$. For $S = \{(r_i, b_i)\}_i$, let $\mathcal{D}_S$ be the uniform distribution on the training set $\{(e_{r,b}, b) : (r,b) \in S\}$. Suppose the learning algorithm outputs a hypothesis $h \in \text{supp}(\mathcal{H}(\mathcal{D}_S))$, then $\mathcal{A}_{\text{ideal}}$ outputs $h(e_{r^*,b^*})$ as its guess for b^*. We now analyze the probability that output of $\mathsf{Adv}_{\text{ideal}}$ is in fact the correct decryption of e^*:

$$\begin{aligned}
\Pr_{e^*\in U_E}[\mathsf{Adv}_{\text{ideal}}(e^*) = b^*] &= \Pr_{e^*\in U_E}\mathbb{E}_{S\subseteq E}\left[\mathsf{Adv}_{\text{ideal}}(e^*) = b^*\middle| e^*\in S\wedge|S| = L\right]\\
&= \Pr_{e^*\in U_E}\mathbb{E}_{S\subseteq E}\left[\Pr_{h\sim\mathcal{H}(\mathcal{D}_S)}[h(e^*) = b^*]\middle| e^*\in S\wedge|S| = L\right]\\
&= \Pr_{e^*\in U_E}\sum_{S\subseteq E}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S} = S|e^*\in S\wedge|S| = L]\Pr_{h\sim\mathcal{H}(\mathcal{D}_S)}[h(e^*) = b^*]\\
&= \sum_{S\subseteq E}\sum_{e^*\in S}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S} = S||S| = L]\Pr_{W\in U_S}[W = e^*]\Pr_{h\sim\mathcal{H}(\mathcal{D}_S)}[h(e^*) = b^*]\\
&= \sum_{S\subseteq E}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S} = S||S| = L]\Pr_{h\sim\mathcal{H}(\mathcal{D}_S)}\left[\mathbb{E}_{e^*\in S}[h(e^*) = b^*]\right]\\
&\geq \sum_{S\subseteq E}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S} = S||S| = L]\left(\frac{1}{2}+\frac{1}{n^c}\right) = \frac{1}{2}+\frac{1}{n^c}.
\end{aligned} \tag{23}$$

In the first equality, the expectation is taken over uniformly random $S \subseteq E$ conditioned on $|S| = L$

and $e^* \in S$, second equality describes the operations of $\mathsf{Adv}_{\text{ideal}}$, the fourth equality holds because $\Pr_{e^*\in U_E}\sum_{S\subseteq E}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S}=S|e^*\in S\wedge|S|=L] = \frac{1}{|E|}\binom{E-1}{L-1}^{-1} = \binom{E}{L}^{-1}\frac{1}{L} = \sum_{S\subseteq E}\sum_{e^*\in S}\Pr_{\mathbf{S}\subseteq E}[\mathbf{S}=S||S|=L]\Pr_{W\in U_S}[W=e^*]$ and the inequality comes from the guarantees of the quantum learning algorithm.[17] Hence Eq. 23 shows that the ideal quantum algorithm $\mathsf{Adv}_{\text{ideal}}$ satisfies the following: on input a uniformly random $e^* = e_{r^*,b^*}$ from the set E, $\mathsf{Adv}_{\text{ideal}}$ outputs b^* with probability at least $1/2 + n^{-c}$.

We now consider $\mathsf{Adv}_{\text{real}}$. Let

$$|\psi_{\text{real}}\rangle = \frac{1}{\sqrt{L-1}}\sum_{(r,b)\in S_{\text{real}}}|e_{r,b},b\rangle, \tag{24}$$

where $S_{\text{real}} = S_{\text{ideal}}\setminus\{e^*\}$. The idea is that for $L = 2n^c t(n)$, the distance between the two states $|\psi_{\text{ideal}}\rangle$ and $|\psi_{\text{real}}\rangle$ is small. Also notice that the state in Eq. 24 can be constructed efficiently by $\mathsf{Adv}_{\text{real}}$: pick $O(L\cdot t(n))$ many (r_i,b_i)-pairs uniformly at random from their respective domains and set $S_{\text{real}} = \{(r_i,b_i)_i\}$. From these classical values, the learner can create $t(n)$ many copies of the state $|\psi_{\text{real}}\rangle$. We now show that a hypothesis sampled from $\mathcal{H}(\mathcal{D}_{S_{\text{real}}})$ behaves "almost similar" to a hypothesis sampled from $\mathcal{H}(\mathcal{D}_{S_{\text{ideal}}})$. For every $y\in\{0,1\}^n$ and $b\in\{0,1\}$, observe that

$$\begin{aligned}\Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{real}}})}[h(y)=b] &= \Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{real}}})}[h(y)=b] - \Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{ideal}}})}[h(y)=b] + \Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{ideal}}})}[h(y)=b]\\ &\geq \Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{ideal}}})}[h(y)=b] - O\left(\frac{t(n)}{L}\right).\end{aligned}\tag{25}$$

The inequality holds since the statistical distance of the output of an algorithm A in inputs $|\phi\rangle$ and $|\phi'\rangle$ is upper-bounded by $\|\,|\phi\rangle - |\phi'\rangle\,\|^2$, and we have that

$$\|\,|\psi_{\text{ideal}}\rangle^{\otimes t(n)} - |\psi_{\text{real}}\rangle^{\otimes t(n)}\,\|^2 = t(n)\|\,|\psi_{\text{ideal}}\rangle - |\psi_{\text{real}}\rangle\,\|^2 = O\left(\frac{t(n)}{L}\right) \tag{26}$$

by the definition of $|\psi_{\text{ideal}}\rangle$ and $|\psi_{\text{real}}\rangle$ in Eq. 22 and 24 respectively. We now analyze the probability that $\mathsf{Adv}_{\text{real}}$ outputs the right decryption b^*, when given a uniformly random sample $e^*\in E$:

$$\begin{aligned}\Pr_{e^*\in U_E}[\mathsf{Adv}_{\text{real}}(e^*)=b^*] &= \Pr_{e^*\in U_E}\mathbb{E}_{S_{\text{real}}\subseteq E}\left[\mathsf{Adv}_{\text{real}}(e^*)=b^*\Big||S_{\text{real}}|=L-1\right]\\ &= \Pr_{e^*\in U_E}\mathbb{E}_{S_{\text{real}}\subseteq E}\left[\Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{real}}})}[h(e^*)=b^*]\Big||S_{\text{real}}|=L=1\right]\\ &\geq \Pr_{e^*\in U_E}\mathbb{E}_{S_{\text{real}}\subseteq E}\left[\Pr_{h\sim\mathcal{H}(\mathcal{D}_{S_{\text{ideal}}})}[h(e^*)=b^*]\Big||S_{\text{real}}|=L-1\right]-O\left(\frac{t(n)}{L}\right)\\ &\geq \frac{1}{2}+\frac{1}{n^c}-O\left(\frac{t(n)}{L}\right)\geq\frac{1}{2}-O\left(\frac{1}{n^c}\right).\end{aligned}\tag{27}$$

In the equations above, recall that $S_{\text{ideal}} = S_{\text{real}}\cup\{(e^*,b^*)\}$, the first inequality used Eq. 25, the second inequality used the same analysis as in Eq. 23 and the last inequality used $L = 2n^c t(n)$. This shows that $\mathsf{Adv}_{\text{real}}$, on input a uniformly random $e^* = \mathsf{Enc}_{\mathsf{K}_{\text{pub}},r^*}(b^*)$ from the set $\mathcal{E}$, outputs b^* with probability at least $1/2 + O(n^{-c})$.

[17]Note that we have assumed for simplicity that (i) $d_{\mathsf{K}_{\text{priv}}}(e_{\mathsf{K}_{\text{pub}},r}(b)) = b$ with probability 1 and (ii) the quantum learning algorithm outputs a hypothesis with probability 1. If we take into account the decryption probability error $\varepsilon(n)$ and that the learning algorithm succeeded with probability $1-\delta$, then the lower bound in Eq. 23 would be $\frac{1}{2}+(1-\delta)n^{-c}-\varepsilon(n)$ and the remaining part of the analysis remains the same.

It remains to show that Adv satisfies Eq. 20, which is equivalent to

$$|2 \Pr_{K,r^*,b^*}[\mathsf{Adv}(\mathsf{K}_{\mathrm{pub}}, e_{\mathsf{K}_{\mathrm{pub}},r^*}(b^*)) = b^*] - 1|. \tag{28}$$

Letting Adv be $\mathsf{Adv}_{\mathrm{real}}$ in the calculation above, we have

$$|2 \Pr_{K,r^*,b^*}[\mathsf{Adv}(\mathsf{K}_{\mathrm{pub}}, e_{\mathsf{K}_{\mathrm{pub}},r^*}(b^*)) = b^*] - 1| \geq O\left(\frac{1}{n^c}\right) = \frac{1}{\mathrm{poly}(n)},$$

where the inequality used Eq. 27. This concludes the proof of the theorem. □

6 Hardness of learning

We now use Theorems 4.2 and 5.1 and the discussion on LWE-based PRGs in Section 3 to prove the following results.

6.1 Hardness of learning TC^0

For completeness, we formally state Result 3 and follow with its proof.

Theorem 6.1 *Let $n, n' \in \mathbb{N}$ such that $n' = \lceil n/\log n \rceil$. Assuming that the n'-dimensional* LWE *problem is hard for a* $\mathrm{poly}(n)$*-time quantum computer, there is no* $\mathrm{poly}(n)$*-time uniform weak quantum-PAC learner for the concept class of* TC^0 *circuits on* $\widetilde{O}(n)$ *bits.*

Proof. We proceed by a proof of contradiction. First, we construct a concept class $\mathcal{C}$ which is a subset of TC^0. By contradiction, suppose there is an efficient quantum learning algorithm for $\mathcal{C}$, we show that it would imply an efficient quantum distinguisher for the LWR problem for a suitable choice of parameters. Using Corollary 3.12 we can translate the hardness of LWR to provide a polynomial-time quantum distinguisher for an LWE problem, contradicting the assumption that LWE is hard for quantum computers.

We give more details below. Let $a, c > 0$ be constants. Let us assume that there exists a learner $\mathcal{A}$ for $\mathcal{C}$ that uses n^a queries/samples and achieves advantage $\beta(n) = n^{-c}$. Let $d = n$ and pick $p, q > d$ such that $p, q \in O(\mathrm{poly}(d))$, $p < q$ and $p \mid q$. Let us set the seed length $s(n) = d \log q = O(n \log n) = \widetilde{O}(n)$. Then by Lemma 3.15, one can use the pseudo-random generator $G_n : \{0,1\}^{s(n)} \to \{0,1\}^{\ell(n)}$ to construct an instance of the $\mathsf{LWR{-}PRG}_{d,p,q,m}$ for the polynomial stretch function $\ell(n) = \widetilde{\Omega}(n^{a+2c+1})$ by choosing $m = \lceil \ell(n)/\log p \rceil$ (without loss of generality, we can pick $\ell(n)$ to be a power of 2).

Consider the following concept class $\mathcal{C}$ defined as in Definition 4.1. For every $k > 1$, let $G_{n,k} : \{0,1\}^{s(n)} \to \{0,1\}^{2^k}$ denote the k-extension of G_n and define

$$\mathcal{C}_k = \left\{ c_z : \{0,1\}^{s(n)} \to \{0,1\} \mid c_z(x) = z_{x \pmod{2^k}},\ \ z \in \mathrm{range}\left(G_{n,k}\right) \right\} \tag{29}$$

and let $\mathcal{C} = \bigcup_{k=1}^{s(n)} \mathcal{C}_k$. Observe that for every concept $c \in \mathcal{C}$, there exists a TC^0 circuit that computes c. This can be inferred as follows. Since G_n is an $\mathsf{LWR{-}PRG}_{d,p,q,m}$, an arbitrary output bit of $G_n(x)$ can be computed in TC^0 by Lemma 3.15 and the task $x' = x \pmod{2^k}$ can also be performed by a TC^0 circuit (by simply considering the last k digits of x to obtain x'). For $z = G_{n,k}(x)$ for some $1 \leq k \leq n$ and x, observe that computing an arbitrary bit of the k-extension requires either truncating an $\ell(n)$-bit string or outputting 0 when $k > \log n$, both of which can be performed by a TC^0 circuit.

Now consider the learner $\mathcal{A}$ that learns TC^0 in polynomial time. In particular $\mathcal{A}$ is also a learner for $\mathcal{C} \subseteq \mathsf{TC}^0$ and by Theorem 4.2, there exists a polynomial time-quantum distinguisher

$\mathcal{D} : \{0,1\}^{\ell(n)} \to \{0,1\}$ that satisfies the following:

$$\left| \Pr_{x \in \{0,1\}^{s(n)}} [\mathcal{D}(G_n(x)) = 1] - \Pr_{y \in \{0,1\}^{\ell(n)}} [\mathcal{D}(y) = 1] \right| \geq \frac{\beta(n)}{2}, \tag{30}$$

for uniformly chosen x, y. Hence $\mathcal{D}$ serves as a poly(n)-time quantum distinguisher for the $\mathsf{LWR}_{d,p,q,m}$ problem. Using Corollary 3.12, $\mathcal{D}$ also serves as a polynomial-time quantum distinguisher for the $\mathsf{LWE}_{n',q,\chi,m}$ problem since $n' = \lceil n/\log n \rceil > d/\log q$ when $q > d$. This contradicts our assumption and concludes the proof. □

Remark 6.2 (LWE is easy when given quantum examples.) *In the proof of Theorem 6.1, we assume that LWE is hard with* classical *examples, and we use them to construct* quantum *samples for* TC^0*. We notice that this does not contradict the work by Grilo et. al [GKZ18], where they prove that* LWE *is easy when* quantum examples *are provided. In Theorem 6.1 the quantum computer is only provided classical examples to solve the* LWE *problem.*

6.2 Hardness of learning AC^0

The proof for the hardness of learning AC^0 is similar to the proof in the previous section. In order to prove the AC^0 hardness, we make two changes. We first consider "smaller instances" of the LWR−PRG in order to access the output bits of the PRG in AC^0 (instead of TC^0 as we showed in Lemma 3.15). Secondly, we "weaken" the statement of our hardness result by showing that, polynomial time weak quantum learners for AC^0 imply sub-exponential time quantum distinguishers for LWE (instead of polynomial-time quantum distinguishers for LWE as we showed in Theorem 6.1).

We first give reasons as to why the sub-exponential time assumption is justified. The best known algorithms for d-dimensional LWE using various methods from lattice reduction techniques [LLL82], combinatorial techniques [BKW03, Wag02] or algebraic ones [AG11] all require $2^{O(d)}$-time in the asymptotic case. Even by quantizing any techniques, the improvements so far only affect the constants in the exponent and do not provide general sub-exponential time algorithms for LWE. Hence, we are justified in *weakening* our statement of hardness to the assumption that LWE does not have 2^{d^ε}-time algorithms for any small constant $0 < \varepsilon < 1$.

Before going into the proof, we provide some intuition on how the choice of parameters as defined in Lemma 3.16 forces us to impose stronger hardness assumptions – (1) the circuits are poly(n)-sized and the algorithms run in poly(n)-time but, the LWR−PRG we use has smaller dimension i.e., $d = O(\text{polylog}\, n)$. However, from the perspective of seed size d, this means that the algorithms run in time sub-exponential in d. Hence, we need to use the stronger "no sub-exponential time algorithms for LWE" assumption. (2) The reduction in Corollary 3.12 requires all parameters of $\mathsf{LWR}_{d,q,p,m}$ to be polynomially bounded in the dimension d. This is violated by choosing modulus $q = n^{O(\log n)}$ and sample size $m = \text{poly}(n)$. Hence, we use the slightly weaker reduction from Theorem 3.10 as it is *independent* of the number of samples and allows for a super-polynomial modulus.

In order to prove our AC^0 hardness, we modify the concept class from Section 6.1 so that the PRGs can be accessed through an AC^0 circuit. Using the PRG from Lemma 3.16 and following the ideas from Theorem 6.1, we now show the existence of AC^0 circuits that cannot be weakly learned under the uniform distribution in polynomial time using quantum examples/queries. We formally restate Result 4 below and proceed with its proof.

Theorem 6.3 *Consider* AC^0 *circuits on* n *bits and let* $d = O(\mathrm{polylog}\, n)$*. Assuming that quantum algorithms for* d*-dimensional* LWE *instances require* $2^{\Omega(d^{\varepsilon})}$ *time for some* $\varepsilon > 0$*, there is no* poly(n)*-time uniform weak quantum-PAC learner for the concept class of* AC^0 *circuits on* n *bits.*

Proof. Let $a, c > 0$ be constants and let us assume that there exists a weak learner $\mathcal{A}$ for a concept class $\mathcal{C}$, that runs in time $t(n) = O(n^a)$, uses $\mathsf{q}(n) = n^a$ queries/samples and achieves advantage $\beta(n) = n^{-c}$. Pick the parameters $d = \log^{\gamma/\varepsilon} n$ for a constant $\gamma > 2$, $q = n^{O(\log n)} = 2^{O(d^{2\epsilon/\gamma})}$ and $p = \mathrm{poly}(d)$ such that $p \mid q$ and p is a power of 2. Set the seed length $s := d \log q = \widetilde{O}(\log^{2+\gamma/\epsilon} n)$. In order to obtain the stretch $\ell(n) = \Omega(n^{a+2c+1})$, we pick a suitable polynomial such that $m = \ell(n)/\log p = \widetilde{O}(\mathrm{poly}(n)) = 2^{\widetilde{O}(d^{\epsilon/\gamma})}$. Using Lemma 3.16, these parameters define an $\mathsf{LWR{-}PRG}_{d,q,p,m}$ instance, $G_n : \{0,1\}^{s(n)} \to \{0,1\}^{\ell(n)}$ that achieves a super-polynomial stretch with respect to seed size and an arbitrary bit of G_n's output can be computed in AC^0.

We use the concept class from Definition 4.1. For every $k > 1$, let $G_{n,k} : \{0,1\}^{s(n)} \to \{0,1\}^{2^k}$ denote the k-extension of G_n, let $\mathcal{C}_k$ be a set of concepts

$$\mathcal{C}_k = \left\{ c_z : \{0,1\}^{s(n)} \to \{0,1\} \mid c_z(x) = z_{x \pmod{2^k}},\ z \in \mathrm{range}(G_{n,k}) \right\} \tag{31}$$

and let $\mathcal{C} = \bigcup_{k=1}^{s(n)} \mathcal{C}_k$ be the concept class. An arbitrary bit of the k-extension either truncates z or trivially outputs 0 when the index is $> \ell(n)$. These require arithmetic operations on $O(\mathrm{polylog}\, n)$-bits and are computable in AC^0 [MT98]. By extending this argument for all k, observe that every concept $c \in \mathcal{C}$ can be computed with a corresponding AC^0 circuit.

Based on the assumption that a weak quantum learner $\mathcal{A}$ exists for AC^0 circuits, $\mathcal{A}$ can be used to learn $\mathcal{C}$ and using Theorem 4.2, there exists a distinguisher $\mathcal{D} : \{0,1\}^{\ell(n)} \to \{0,1\}$ that satisfies the following condition for uniformly chosen x, y:

$$\left| \Pr_{x \in \{0,1\}^{s(n)}} [\mathcal{D}(G_n(x)) = 1] - \Pr_{y \in \{0,1\}^{\ell(n)}} [\mathcal{D}(y) = 1] \right| \geq \frac{\beta(n)}{2}. \tag{32}$$

Additionally, the distinguisher $\mathcal{D}$ runs in time $T = t(n) + \mathsf{q}(n)\ell(n)$. We can simplify this term to upper bound the running time of $\mathcal{D}$ as follows:

$$\begin{aligned}
t(n) + \mathsf{q}(n)\ell(n) &\leq (\ell(n)+1)\max\{t(n), \mathsf{q}(n)\} \\
&\leq O(\ell(n)\mathsf{q}(n)) && (\text{Since } t(n) = O(\mathsf{q}(n))) \\
&\leq O\left(\frac{\mathsf{q}(n)^2 (n+1)}{\beta(n)^2} \right) && (\text{Using Eq. } 6) \\
&\leq O(n^{2a+2c+1}) && (\text{Since } \mathsf{q}(n) = n^a \text{ and } \beta(n) = n^{-c}) \\
&= 2^{O(\log n)} = 2^{O(d^{\varepsilon/\gamma})} < 2^{O(d^{\varepsilon})} && (\text{For every } \gamma > 2)
\end{aligned}$$

The reduction in Theorem 3.10 implies that (irrespective of the number of samples observed) a distinguisher for the $\mathsf{LWR}_{d,q,p}$ problem can be used as a distinguisher for the $\mathsf{LWE}_{d,q,\chi}$ problem as long as q grows super-polynomially with respect to d.[18] This would give us a distinguisher for d-dimensional LWE that runs in time $2^{O(d^{\varepsilon/\gamma})} < 2^{O(d^{\varepsilon})}$ which contradicts our sub-exponential time hardness assumption for LWE. □

One weakness of our result is we only get a polynomial-time hardness result for learning AC^0, whereas Kharitonov [Kha93] obtains a $\Omega(n^{\mathrm{polylog}\, n})$ lower bound for learning AC^0 assuming factoring is hard. In particular, his lower bound also matches the $O(n^{\mathrm{polylog}\, n})$ sample and

[18]As discussed in Section 3.2, for the LWE distinguisher constructed using Theorem 3.10 to have $1/\mathrm{poly}(n)$ advantage, we need $m/q \leq \mathrm{negl}(n)$ which is satisfied by our choice of q and m.

time complexity upper bound for learning AC^0 circuits as demonstrated by Linial, Mansour and Nisan [LMN93].

Using PRGs based on LWR and our proof technique, we believe that it may not be possible to find a stronger lower bound for polynomial-sized AC^0 circuits. This is because, starting with a quasi-polynomial time learner for some concept class, we require the stretch function to satisfy $\ell(n) = O(n^{\operatorname{polylog} n})$. This means we would need $\widetilde{O}(\ell(n))$ samples for the instance to be quantum-secure which would, in turn, would result in a $\widetilde{O}(\ell(n))$-sized AC^0 circuits. Additionally, it is unclear if the reduction in Theorem 3.10 is robust for this choice of parameters. We leave the question of whether a different choice of quantum-secure PRGs could improve the lower bound open for future work. However, we remark on how further non-standard assumptions on the hardness of LWR may help us achieve super-polynomial lower bounds for super-polynomial sized AC^0 circuits.

Remark 6.4 (Lower bounds for learning quasi-polynomial sized AC^0 circuits) *Assuming there are no sub-exponential time quantum algorithms for the d-dimensional* LWR *problem Then, for $d = O(\operatorname{polylog} n)$, there exists* no *$O(n^{\operatorname{polylog} n})$-time uniform weak quantum-PAC learner for the class of $O(n^{\operatorname{polylog} n})$-sized* AC^0 *circuits on n bits.*

Proving this remark follows the same steps as in the proof of Theorem 6.3 and it is omitted here.

6.3 Hardness of learning TC^0_2

We show here a conditional hardness of quantum learning depth-2 threshold circuits.

Theorem 6.5 *Let $n \in \mathbb{N}$. Assuming that the n-dimensional* LWE *problem is hard for a* $\operatorname{poly}(n)$*-time quantum computer, there is no* $\operatorname{poly}(n)$*-time weak quantum-PAC learner for the concept class of* TC^0_2 *circuits on $\widetilde{O}(n)$-bit inputs.*

The main point of difference between the proof of Theorem 6.5 and the (classical) result of Klivans and Sherstov [KS09] is in the connection between *quantum* learning and public-key *quantum* cryptosystems which we already discussed in the previous section. The remaining part of our proof follows their structure very closely. For brevity, we state a simple lemma from their paper without a proof.

Lemma 6.6 *[KS09, Lemma 4.1] Fix $\varepsilon > 0$ to be a constant. Assume that the intersection of n^ε light half-spaces is weakly quantum-PAC-learnable. Then for every constant $c > 0$, the intersection of n^c light degree-2 PTFs are weakly quantum-PAC learnable.*[19]

We are now ready to prove our main theorem.

Proof of Theorem 6.5. In order to prove the hardness of TC^0_2, we first consider a subclass of TC^0_2, intersection of polynomially-many half-spaces, and prove the conditional quantum hardness of learning this subclass.

Fix $\varepsilon > 0$ to be a constant. Let $\mathcal{C}$ be the concept class of n^ε light half-spaces and $\mathcal{C}'$ be the concept class of degree-2 light PTFs. Let us assume that $\mathcal{C}$ is quantum-PAC learnable. Then by Lemma 6.6, the assumed learnability of $\mathcal{C}$ implies the quantum-PAC learnability of $\mathcal{C}'$. By Lemma 3.8, the decryption function of the LWE-cryptosystem is in $\mathcal{C}'$. Using Theorem 5.1, we

[19]In [KS09, Lemma 4.1], they state the classical version of this lemma. The exact same proof goes through when we replace classical learners by quantum learners.

now relate the quantum-PAC learnability of $\mathcal{C}'$ to the LWE-cryptosystem as follows: suppose that there exists a quantum-PAC learning algorithm for $\mathcal{C}'$, then Theorem 5.1 implies the existence of a distinguisher that can differentiate the encryptions of 0 and 1 in the LWE-cryptosystem. As a consequence, by Theorem 3.7 this would result in a quantum polynomial-time distinguisher for LWE.

In order to conclude the theorem, we show that $\mathcal{C} \subseteq \mathsf{TC}_2^0$, i.e., every polynomial-sized half-space can be written as a depth-2 threshold circuit. In order to see this, first observe that each half-space is already a majority gate (with the inputs suitably negative and replicated based on the coefficients $a_1,\ldots,a_n$) and the top gate $\mathsf{AND}(f_1,\ldots,f_t)$ can be replaced by the majority gate $\mathsf{MAJ}(-t, f_1,\ldots,f_t)$. Putting together our observation in the previous paragraph about the quantum-PAC learnability of $\mathcal{C}$, and the fact that $\mathcal{C} \subseteq \mathsf{TC}_2^0$, we get the theorem statement. □

References

[AA16] Jacob Alperin-Sheriff and Daniel Apon. Dimension-preserving reductions from LWE to LWR. *IACR Cryptology ePrint Archive*, 2016:589, 2016. 22

[AAD+15] J. Adcock, E. Allen, M. Day, S. Frick, J. Hinchliff, M. Johnson, S. Morley-Short, S. Pallister, A. Price, and S. Stanisic. Advances in quantum machine learning, 9 Dec 2015. arXiv:1512.02900. 17

[Aar05] Scott Aaronson. Ten semi-grand challenges for quantum computing theory. `https://www.scottaaronson.com/writings/qchallenge.html`, 2005. , 2, 5, 9

[ABHS19] Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. Seth-based lower bounds for subset sum and bicriteria path. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 41–57, 2019. arXiv:1704.04546. 12

[ACLW18] Srinivasan Arunachalam, Sourav Chakraborty, Troy Lee, and Ronald de Wolf. Two new results about quantum exact learning, 2018. arXiv:1810.00481. 4

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In *International Colloquium on Automata, Languages, and Programming*, pages 403–415. Springer, 2011. 2, 10, 33

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited - new reduction, properties and applications. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference*, pages 57–74, 2013. 22

[AMRR11] Andris Ambainis, Loïck Magnin, Martin Roetteler, and Jérémie Roland. Symmetry-assisted adversaries for quantum state generation. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC*, pages 167–177, 2011. arXiv:1012.2112. 6, 8, 30

[Ang87] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987. 17

[AS05] Alp Atıcı and Rocco Servedio. Improved bounds on quantum learning algorithms. *Quantum Information Processing*, 4(5):355–386, 2005. 4

[AS09] Alp Atıcı and Rocco Servedio. Quantum algorithms for learning and testing juntas. *Quantum Information Processing*, 6(5):323–348, 2009. arXiv:0707.3479. 4, 41

[AS18] Divesh Aggarwal and Noah Stephens-Davidowitz. (gap/s)eth hardness of SVP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 228–238, 2018. arXiv:1712.00942. 12

[AW17] Srinivasan Arunachalam and Ronald de Wolf. Guest column: A survey of quantum learning theory. *SIGACT News*, 48(2):41–67, 2017. arXiv:1701.06806. 16, 17

[AW18] Srinivasan Arunachalam and Ronald de Wolf. Optimal quantum sample complexity of learning algorithms. *Journal of Machine Learning Research*, 19, 2018. Earlier version in CCC'17. arXiv:1607.00932. 4

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh V. Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997. arXiv:quant-ph/9701001. 24

[BBS86] Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, 1986. 15

[BCJ11] Anja Becker, Jean-Sébastien Coron, and Antoine Joux. Improved generic algorithms for hard knapsacks. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 364–385, 2011. 42

[BGK18] Sergey Bravyi, David Gosset, and Robert Koenig. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018. 4

[BGM$^+$16] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. On the hardness of learning with rounding over small modulus. In *Theory of Cryptography - 13th International Conference, TCC 2016-A*, pages 209–224, 2016. 21, 22

[BJ99] Nader H. Bshouty and Jeffrey C. Jackson. Learning DNF over the uniform distribution using a quantum example oracle. *SIAM Journal on Computing*, 28(3):11361153, 1999. Earlier version in COLT'95. 1, 3, 4, 5, 11, 16

[BJLM13] Daniel J. Bernstein, Stacey Jeffery, Tanja Lange, and Alexander Meurer. Quantum algorithms for the subset-sum problem. In *Post-Quantum Cryptography - 5th International Workshop, PQCrypto*, pages 16–33, 2013. 42

[BJMM12] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in 2 n/20: How 1 + 1 = 0 improves information set decoding. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 520–536, 2012. 42

[BKST19] Adam Bene Watts, Robin Kothari, Luke Schaeffer, and Avishay Tal. Exponential separation between shallow quantum circuits and unbounded fan-in shallow classical circuits. In *Proceedings of the 51st Annual ACM Symposium on the Theory of Computing*. ACM, 2019. 4

[BKW03] 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. 2, 10, 33

[BLP$^+$13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584, 2013. 20

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques.*, pages 719–737, 2012. 9, 21, 22

[BV97] Ethan Bernstein and Umesh V. Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997. Earlier version in STOC'93. 1, 4

[DS16] Amit Daniely and Shai Shalev-Shwartz. Complexity theoretic limitations on learning DNF's. In *Proceedings of the 29th Conference on Learning Theory (COLT'16)*, 2016. 12

[Fil15] Yuval Filmus. CS Stack exchange: Sum of $\log n$ n-bit integers is in AC^0. https://cs.stackexchange.com/questions/39693/, 2015. 24

[FP05] Abraham Flaxman and Bartosz Przydatek. Solving medium-density subset sum problems in expected polynomial time. In *STACS 2005, 22nd Annual Symposium on Theoretical Aspects of Computer Science*, pages 305–314, 2005. 42

[Fri86] Alan M. Frieze. On the Lagarias-Odlyzko algorithm for the subset sum problem. *SIAM Journal on Computing*, 15(2):536–539, 1986. 42

[FSA99] Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14:771–780, 1999. 16

[GG00] Oded Goldreich and Shafi Goldwasser. On the limits of nonapproximability of lattice problems. *Journal of Computer and System Sciences*, 60(3):540–563, 2000. 20

[GHR92] Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority gates VS. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992. 11, 18

[GKZ18] Alex Bredariol Grilo, Iordanis Kerenidis, and Timo Zijlstra. Learning with errors is easy with quantum samples, 2018. arXiv:1702.08255. 4, 33

[HAB02] William Hesse, Eric Allender, and David A Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65(4):695–716, 2002. 23

[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. 42

[HJ10] Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 235–256, 2010. 42

[IN96] Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, 1996. Earlier version in FOCS'89. 25, 42, 43

[Jac97] Jeffrey C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55(3):414–440, 1997. Earlier version in FOCS'94. 4

[Kha92] Michael Kharitonov. Cryptographic lower bounds for learnability of boolean functions on the uniform distribution. In *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT*, pages 29–36, 1992. 5

[Kha93] Michael Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, pages 372–381, 1993. 5, 6, 10, 11, 23, 24, 34

[KLGR16] Iordanis Kerenidis, Mathieu Laurière, Francois le Gall, and Mathys Rennela. Information cost of quantum communication protocols. *Quantum Information & Computation*, 16(3&4):181–196, 2016. 14

[KRS18] Varun Kanade, Andrea Rocchetto, and Simone Severini. Learning DNFs under product distributions via μ-biased quantum fourier sampling. *arXiv preprint arXiv:1802.05690*, 2018. 4

[KS09] Adam R. Klivans and Alexander A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. *Journal of Computer and System Sciences*, 75(1):1–12, 2009. Earlier version in FOCS'06. 5, 8, 11, 20, 29, 30, 35

[KV94] Michael J. Kearns and Leslie G. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994. 5, 8, 29

[LLL82] Arjen K. Lenstra, Hendrik W. Lenstra, and László Lovasz. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, Dec 1982. 10, 33, 42

[LMN93] Nati Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993. Earlier in FOCS'89. 5, 10, 11, 12, 35

[LO85] Jeffrey C. Lagarias and Andrew M. Odlyzko. Solving low-density subset sum problems. *J. ACM*, 32(1):229–246, 1985. Earlier version in FOCS'83. 42

[LR19] Nathan Lindzey and Ansis Rosmanis. A tight lower bound for non-coherent index erasure, 2019. arXiv:1902.07336. 6, 8, 30

[Lyu05] Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, 2005. 42

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In *Proceedings of 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 259–267, 2018. arXiv:1804.01082. 4

[MMT11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $2^{0.054n}$. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security*, pages 107–124, 2011. 42

[MSS91] Wolfgang Maass, Georg Schnitger, and Eduardo D. Sontag. On the computational power of Sigmoid versus Boolean threshold circuits. In *32nd Annual Symposium on Foundations of Computer Science*, pages 767–776, 1991. 11, 18

[MT98] Alexis Maciel and Denis Thérien. Threshold circuits of small majority-depth. *Information and computation*, 146(1):55–83, 1998. 23, 24, 34

[MTT61] Saburo Muroga, Iwao Toda, and Satoru Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271:376–418, 1961. 18

[Mur71] Saburo Muroga. Threshold logic and its applications, 1971. John Wiley & Sons. 18

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudorandom functions. *Journal of the ACM*, 51(2):231–262, 2004. Earlier version in FOCS'97. 11

[OS17] Igor Carboni Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness. In *32nd Computational Complexity Conference, CCC*, pages 18:1–18:49, 2017. arXiv:1611.01190. 12

[Par90] Ian Parberry. A primer on the complexity theory of neural networks. *Formal techniques in artificial intelligence*, pages 217–268, 1990. 18

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC*, pages 333–342, 2009. 19, 20, 23

[Pei16] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016. 19

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):34:1–34:40, 2009. Earlier version in STOC'05. 2, 11, 19, 20, 23

[RT92] John H Reif and Stephen R. Tate. On threshold circuits and polynomial computation. *SIAM Journal on Computing*, 21(5):896–908, 1992. 23

[SG04] Rocco Servedio and Steven Gortler. Equivalences and separations between quantum and classical learnability. *SIAM Journal on Computing*, 33(5):1067–1092, 2004. Combines earlier papers from ICALP'01 and CCC'01. quant-ph/0007036. 4

[Sha08] Andrew Shallue. An improved multi-set algorithm for the dense subset sum problem. In *Algorithmic Number Theory*, pages 416–429, 2008. 42

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. Earlier version in FOCS'94. quant-ph/9508027. 5, 6, 15

[SP18] Maria Schuld and Francesco Petruccione. *Supervised Learning with Quantum Computers*. Springer, 2018. 17

[Tou15] Dave Touchette. Quantum information complexity. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, pages 317–326, 2015. arXiv:1404.3733. 14

[Val84] Leslie Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. 1, 2, 5, 15

[Ver90] Karsten A. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the 3rd Annual Workshop on Computational Learning Theory (COLT'90)*, pages 314–326, 1990. 2

[Wag02] David Wagner. A generalized birthday problem. In *Annual International Cryptology Conference*, pages 288–304. Springer, 2002. 10, 33

A Time-efficient learning of NC^0

We prove the following theorem, which is straightforward from known results in quantum learning theory. Since we didn't find an explicit reference to the proof of such an explicit theorem in literature, we make it formal here.

Theorem A.1 *Let $c > 0$ be a constant. Let $\mathcal{C}$ be the concept class of all Boolean functions on n bits which are computable by polynomial number of gates with at most 2 inputs and depth at most d. Then $\mathcal{C}$ can be learned under the uniform distribution with error at most ε, using at most $\widetilde{O}(2^d/\varepsilon)$ quantum examples, $O(2^{2^d})$ classical examples and time $\widetilde{O}(n2^d/\varepsilon + 2^{2^d}\log(1/\varepsilon))$.*

The proof directly follows from Atıcı and Servedio's theorem on learning k-juntas.

Theorem A.2 ([AS09]) *There exists a quantum algorithm for learning k-juntas under the uniform distribution that uses $O((k\log k)/\varepsilon)$ uniform quantum examples, $O(2^k)$ uniform classical examples, and $\widetilde{O}(nk/\varepsilon + 2^k\log(1/\varepsilon))$ time.*

Now we proceed with proving our theorem.

Proof of Theorem A.1. Suppose that $f : \{0,1\}^n \to \{0,1\}$ is computed by a circuit with depth at most d. Since the gates of the circuit have fan-in at most 2, clearly the output of the circuit for f (on an arbitrary input x) depends only on 2^d bits of x. Hence every $f \in \mathcal{C}$ is a 2^d-junta. Using Theorem A.2, it follows that the concept class of 2^d-juntas can be learned using $\widetilde{O}(2^d/\varepsilon)$ quantum examples, $O(2^{2^d})$ classical examples and time $\widetilde{O}(n2^d/\varepsilon + 2^{2^d}\log(1/\varepsilon))$. □

Corollary A.3 *Let $c > 0$ be a constant. Let $\mathcal{C}$ be the concept class of n-bit functions which are computable by polynomial number of gates with fan-in at most 2, depth at most $\log(c\log n)$. Then $\mathcal{C}$ can be learned using at most $\widetilde{O}(n/\varepsilon)$ quantum examples, $O(n^c)$ classical examples and time $\widetilde{O}(n^c/\varepsilon)$.*

Proof. The proof immediately follows from Theorem A.1 by plugging in $d = \log(c\log n)$. □

Observe that, since NC^0 is the class of circuits on n-bits with depth $O(1)$, NC^0 is contained in the concept class $\mathcal{C}$ considered in the corollary above.

B Hardness of learning from hardness of Subset-sum

In this section, we consider the scenario where a breakthrough in algorithm design results in a quantum polynomial-time algorithm for LWE. Then, all our conditional lower bounds for $\mathsf{AC}^0, \mathsf{TC}^0$ and TC^0_2 would be moot. In this case, one possibility is to consider another candidate problem whose hardness with respect to quantum computers is not connected to the hardness of LWE. In this section, we consider the Subset sum problem and show the conditional hardness of learning NC^1. Our main theorems connecting PRGs and quantum learning are robust enough that similar proofs as in the body of the paper carry through.

In this section we discuss constructing PRGs whose security is based on the hardness of the Subset-sum problem. Let us first define this problem.

Definition B.1 (Subset-sum$_{n,p}$) *The Subset-sum problem is the following. Given n numbers $a_1,\ldots a_n$, each $p(n)$ bits long and a number T, find some $S \subseteq \{1,\ldots,n\}$ such that $\sum_{i\in S} a_i = T \pmod{2^{p(n)}}$.*

Even in the worst-case scenario, the hardness of Subset-sum is tightly related to function p: the cases when $p(n) = O(\log n)$ can be solved efficiently by a dynamic programming algorithm, whereas the Subset-sum problem, in general, is NP-harde.

For cryptographic applications, we are interested in the hardness of Subset-sum for random instances, which has been fairly well studied [LO85, Fri86, FP05, Lyu05, Sha08]. Again, the hardness in this case, is intimately related to the function $p(n)$. For $p(n) > n^2$, random instances of the subset-sum problem can be solved efficiently via a reduction to the Shortest-vector problem [LO85, Fri86] followed by using the LLL algorithm for the desired approximation factor [LLL82]. For $p(n) = O(\log^2(n))$, it is possible to divide the input into smaller and efficiently solvable instances (using the dynamic-programming algorithm, for instance), and then the solutions for these smaller problems can be combined to yield a solution for the original problem [FP05]. Finally, there are also sub-exponential time algorithms for the case where $p(n) = n^{\varepsilon}$ for $0 < \varepsilon < 1$ based on the k-set birthday problem [Lyu05, Sha08].

As far as we know, for $p(n) = O(n)$ the Subset-sum is considered to be hard even in the average case scenario and the current, best algorithms for this setup-up still need exponential time [HJ10, MMT11, BCJ11, BJMM12]. Unfortunately, there has not been extensive research on the average-case hardness of Subset-sum in the quantum setting. The first work to consider this case was Bernstein et al. [BJLM13], where they achieve an exponential time quantum algorithm which outperforms the current classical algorithms.

B.1 PRGs from Subset-sum

The Subset-sum problem can be seen as inverting the function

$$f(a_1,\ldots,a_n,S) = \left(a_1,\ldots,a_n,\sum_{i\in S} a_i\right).$$

and its hardness means that f is *one-way*. Impagliazzo and Naor [IN96] then noticed that this property could be used to devise PRGs.[20] For some publicly known values $a_1,\ldots,a_n \in [2^{p(n)}]$, the PRG maps a seed $s \in \{0,1\}^n$ to the integer $\sum_{i:s_i=1} a_i \pmod{2^{p(n)}}$ [IN96]. Choosing $p(n) > (1+c)n$ for some constant $c > 0$ gives a PRG with linear stretch, which can be pushed up to polynomial stretch just by composing this PRG a constant number of times. We also notice that one only needs to compute iterated additions in order to output the pseudo-random bits, which makes this PRG highly parallelizable and therefore, computable in NC^1.

[20] A generalized reduction from one-way functions to PRGs was later shown by Håstad, Impagliazzo, Levin and Luby [HILL99].

Lemma B.2 (Theorem 2.2 of [IN96]) *Let $p(n) > (1+c)n$ for $c > 0$. If the $\mathsf{Subset-sum}_{n,p}$ is quantum one-way, then for every polynomial $\ell(n) > n$, there exists a quantum-secure pseudo-random generator in* NC^1 *that outputs $\ell(n)$ pseudo-random bits.*

Impagliazzo and Naor also provided a second construction of PRGs based on the Subset-sum problem, with only logarithmic stretch, but computable in AC^0. These PRGs in AC^0, instead of fixing the values $a_1,\ldots,a_n$ and then computing the sum of a random subset of values (which cannot be computed in AC^0), generate both $a_1,\ldots,a_n$ and the sum of a random subset at the same time. We refer interested readers to Section 6 of [IN96] for the technicalities of such a construction.

Lemma B.3 *[IN96, Claim 6.1] Let $p(n) = n + \log n$. If $\mathsf{Subset-sum}_{n,p}$ is quantum one-way, then for any constant a and $\ell(n) = n + a\log n$, there is a quantum secure pseudo-random generator in AC^0 that outputs $\ell(n)$ pseudo-random bits.*

Notice that, with these parameters, we cannot achieve polynomial stretch for the pseudo-random string, and devising such a PRG in AC^0 is currently an open-problem.

B.2 Hardness of learning NC^1

As in Section 6.1, we are able to show the following conditional hardness of learning NC^1 circuits based on the hardness of the Subset-sum problem. We omit the proof here.

Theorem B.4 *Let p be a polynomial. Assuming that the problem $\mathsf{Subset-sum}_{n,p}$ is hard for quantum computers, there is no polynomial-time quantum learning algorithm for the concept class NC^1.*

We can prove it by repeating the proof of Theorem 6.1 with the the PRG from Lemma B.2.

B.3 Query lower bound on learning AC^0 based on subset-sum PRGs

Theorem B.5 *Let $p(n) = n + \log n$. Assuming that $\mathsf{Subset-sum}_{n,p}$ is hard for quantum computers, there is no polynomial-time uniform quantum-PAC learner for the concept class AC^0 with query complexity $\mathsf{q}(n) = o(\sqrt{n})$ and advantage $\frac{16\mathsf{q}(n)^2}{n}$.*

Proof. The proof again goes on the same lines as the proof of Theorem 6.1, but since it is more subtle, we now go over the details. Let us assume that there exists a learner $\mathcal{A}$ for $\mathcal{C}$ that uses $\mathsf{q}(n) = o(\sqrt{n})$ queries and samples and achieves advantage $\beta(n) = \frac{16\mathsf{q}(n)^2}{n} = \omega(1)$. Let us consider the PRG G_n from Lemma B.3, with seed length $s(n) = n$ and stretch $\ell(n) = n + \log n$. Consider the following concept class $\mathcal{C}$ defined as

$$\mathcal{C} = \bigcup_{k=1}^{n} \{c_z : \{0,1\}^n \to \{0,1\} \mid c_z(x) = z_{x \pmod{\ell(n))}}, z \in \mathsf{range}(G_n)\}.$$

Observe that for every $c \in \mathcal{C}$, there exists an AC^0 circuit that computes c because the string z, as the output of Subset-sum$_{n,p}$, can be computed in AC^0 (by Lemma B.3), and the test $x' = x \pmod{\ell(s)}$ can also be performed by an AC^0 circuit. Given the assumption that the AC^0 learner $\mathcal{A}$ with the stated parameters exists, by Lemma 4.3 (in particular, using the bound of $\frac{1}{2} + \frac{2T}{\sqrt{\ell(n)}}$) there exists a polynomial-time quantum distinguisher $\mathcal{D} : \{0,1\}^{\ell(n)} \to \{0,1\}$ that satisfies the following:

$$\left| \Pr_{x\in\{0,1\}^n}[\mathcal{D}(G_n(x)) = 1] - \Pr_{y\in\{0,1\}^{\ell(n)}}[\mathcal{D}(y) = 1] \right| \geq \frac{\beta(n)}{2} = \omega(1), \tag{33}$$

for uniformly chosen x, y. Hence $\mathcal{D}$ is an efficient quantum distinguisher for the Subset-sum$_{n,p}$ problem and contradicts our assumption. □