Covert Learning: How to Learn with an Untrusted Intermediary*

Ran Canetti
Boston University
canetti@bu.edu

Ari Karchmer
Boston University
arika@bu.edu

June 8, 2021

Abstract

We consider the task of learning a function via oracle queries, where the queries and responses are monitored (and perhaps also modified) by an untrusted intermediary. Our goal is twofold: First, we would like to prevent the intermediary from gaining any information about either the function or the learner's intentions (e.g. the particular hypothesis class the learner is considering). Second, we would like to curb the intermediary's ability to meaningfully interfere with the learning process, even when it can modify the oracles' responses.

Inspired by the works of Ishai et al. (Crypto 2019) and Goldwasser et al. (ITCS 2021), we formalize two new learning models, called *Covert Learning* and *Covert Verifiable Learning*, that capture these goals. Then, assuming hardness of the Learning Parity with Noise (LPN) problem, we show:

- Covert Learning algorithms in the agnostic setting for parity functions and decision trees, where a polynomial time eavesdropping adversary that observes all queries and responses learns nothing about either the function, or the learned hypothesis.
- Covert Verifiable Learning algorithms that provide similar learning and privacy guarantees, even in the presence of a polynomial-time adversarial intermediary that can modify all oracle responses. Here the learner is granted additional random examples and is allowed to abort whenever the oracles responses are modified.

Aside theoretical interest, our study is motivated by applications to the outsourcing of automated scientific discovery in drug design and molecular biology. It also uncovers limitations of current techniques for defending against model extraction attacks.

*Supported by the DARPA SIEVE program, Agreement Nos. HR00112020020 and HR00112020021.

Contents

1 Introduction

A motivating scenario. Imagine a biologist, Alice, who wishes to learn a model—within some class of hypothesized models—for the relationship between the structure of a molecule and its "activity" (e.g. whether or not the molecule binds to a certain protein). Alice plans to conduct a variety of lab experiments in order to learn her model.

However, in Alice's lab all experiments are public: they are observable by anyone. Can Alice design experiments so that only she will learn her model? Furthermore, can Alice design the experiments so that they will not leak her initial hypotheses on the possible models, which encode Alice's innovative, secret list of molecule features that are likely to influence activity? In fact, can Alice design the experiments so that no one else but her learns *anything at all* from her experiments?

To complicate things further, suppose that after starting the experiments, Alice is notified that she has been exposed to COVID-19 and has to quarantine at home; she has no choice but to delegate the recording of the results from her experiments to an untrusted colleague, Bob. Thus, in addition to concealing her learned model, hypothesized class of models, and any information about the molecular relationship, Alice needs a way to verify the results reported by Bob. In summary, Alice needs a learning algorithm that will carry the following (informal) guarantees:

- *Learning*: If Bob reports the results correctly, then Alice is guaranteed to acquire some satisfactory model for the studied molecular relationship.
- *Verifiability*: Even if Bob behaves maliciously, Alice is guaranteed to acquire a satisfactory model, *as long as she does not decide to reject Bob's report.*
- *Hypothesis-hiding*: Bob does not learn anything about the model Alice has learned or about Alice's hypothesized class of models.
- *Concept-hiding*: Bob does not learn anything about the molecular relationship that he did not already know.

The learning requirement mimics classic learning-theoretic formalisms. In particular, it naturally corresponds to agnostic learning with membership queries: the molecular relationship corresponds to a concept, Alice's experiments correspond to queries to the concept at arbitrary points, and Alice's task of finding a model within a class of models corresponds to learning a hypothesis out of a given hypothesis class (e.g. polynomial size decision trees).

Put in these terms, our work is focused on the following questions: Can we devise agnostic learning algorithms in the membership query model that satisfy the above verifiability and hiding guarantees? If so, then for which hypothesis classes, and under what computational assumptions? In fact, how should we even define these (so far informal) goals?

Before proceeding to present our contributions, we note that this work has been inspired by the works of Ishai et al. [IKOS19] and Goldwasser et al. [GRSY20] that consider related models. We elaborate on these works and the relationships in Section 1.3.

1.1 Our Contributions

We define and construct learning algorithms that satisfy the above requirements. We first present our definitions (Section 1.1.1), then state our results (Section 1.1.2), then overview our techniques (Section 1.1.3).

1.1.1 New Learning Models: Covert and Verifiable Learning

We propose two new learning models: the basic *Covert Learning* model, which considers a passive adversary only, and the *Covert Verifiable Learning* model, which considers an intermediary who may observe queries and even modify responses.

The Covert Learning model. Our model is grounded in the learning with membership queries setting, where a learner is allowed to directly query the concept, with an added twist: every query and response obtained by the learner is also obtained by a computationally bounded adversary. The high level goal is for the learner to construct queries that are useful to herself, but are completely unintelligible to any adversary.

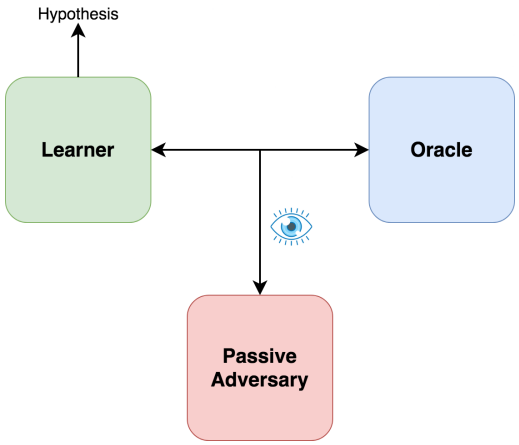

Figure 1: A Covert Learning scenario. The learner interacts with the concept by making queries to an oracle that implements access to the concept at arbitrary points. Meanwhile, an adversary attempts to deduce information about the learner's hypothesis or about the concept itself, given a view: the set of queries and responses obtained by the learner.

One may be tempted to formulate this property by requiring that the adversary gains nothing from the interaction between the learner and the concept. However, this would be too much to demand, since the adversary does (at the very least) learn the responses to the learner's queries. We thus somewhat relax the hiding property to say that the adversary learns nothing *except for some number of random examples from the concept*. In other words, the view of the adversary can be *simulated* in probabilistic polynomial time (p.p.t.), given only random examples from the concept. This in particular means that the notion of Covert Learning is meaningful only when the learning task at hand is computationally hard in the traditional PAC learning model, where a concept must be learned from random examples only.

A bit more formally, Let $\mathcal{X}$ be a set, and consider a distribution D over $\mathcal{X} \times \{0,1\}$. We will call a sample $(x,y) \sim D$ an example, where x is an *input* and y is a *label*, and call D a *concept*. Let $\mathcal{H}$ denote a *hypothesis class*, which is a subset of functions $h : \mathcal{X} \to \{0,1\}$. A learning algorithm under the Covert Learning model is tasked with finding an hypothesis $h \in \mathcal{H}$ that best approximates the concept D on unobserved examples $(x,y) \sim D$. This notion is captured by a *loss function* (w.r.t. to a concept). For example: $\mathcal{L}_D(h) = \Pr_{(x,y)\sim D}[h(x) \neq y]$. The learning goal of the Covert Learning model is then the requirement that the learner outputs $h \in \mathcal{H}$ such that $\mathcal{L}_D(h) \leq \mathcal{L}_D(\mathcal{H}) + \epsilon = \inf_{h\in\mathcal{H}} \mathcal{L}_D(h) + \epsilon$ with high probability, and we will call such an h ϵ-good. In order to achieve this goal, the learner is given access to a (possibly probabilistic) oracle that labels a queried input $x_j \in \mathcal{X}$ with a corresponding y_j. The novelty

of the Covert Learning model is the guarantee that—in addition to the learning goal—no information about the hypothesis class or the concept is leaked to a passive adversary, except some random examples from the concept. *This guarantee holds even when the adversary has access to extraneous information on the concept.*

Definition 1.1. ***Covert Learning*** *(informal version of Definition 2.7). A covert learning algorithm—for a collection of hypothesis classes and with respect to a class of concepts and a loss function—is an algorithm that, for any concept in the class and accuracy parameters ϵ, δ, takes as input a target hypothesis class in the collection, and interacts with an oracle that labels queries to the concept such that the following are true:*

- ***Completeness.*** *The learning algorithm outputs an ϵ-good hypothesis for the concept with probability $1 - \delta$.*
- ***Privacy.*** *There exists a p.p.t. simulation algorithm that, given access to additional random examples from the concept, generates a distribution of queries and responses which is computationally indistinguishable from that of the real interaction. The simulation algorithm should function without further access to the oracle, or knowledge of the target hypothesis class within the collection.*

On hypothesis-hiding. In addition to hiding the learned concept, the above definition also requires that a covert learning algorithm hides the initial hypothesis class. Let us motivate this requirement. Indeed, when operating in a setting where the concept is included in a fixed class and can be learned fully, there is little motivation for hypothesis-hiding. However, in the more realistic setting of agnostic learning—where no assumptions are made about the concept—one resorts to learning the best approximation to the concept that is contained in some chosen hypothesis class. Clearly, the choice of hypothesis class is crucial in determining the value of the resulting approximation. Therefore, the chosen hypothesis class reflects the learner's prior beliefs about the concept, and is *itself* valuable information in need of protection. Indeed, the main motivation of *tolerant testing*[1] is to decide if a class of hypotheses contains a good approximator to an unknown concept. Concretely, the learner could be motivated to hide the results of a tolerant testing procedure that were received as advice. Alternatively, relating back to the motivating scenario, the specific domain knowledge that Alice has might influence her choices of experiments, which could in turn reveal information about her sensitive domain knowledge. Alice may be motivated to conceal her sensitive domain knowledge.

As a matter of fact, digging deeper into real world applications of learning with membership queries reveals further motivation for hypothesis-hiding, even when the concept is known to be from a fixed class (and therefore may be learned fully). In some specific practical applications (see Section 1.2 for more details), arbitrarily synthesized membership queries are difficult or expensive (in some measure) to obtain. For example, conducting a biological assay using an unstable compound. As is the case, and despite the fact that a concept may be known to be contained in a fixed class, the learner might voluntarily submit itself to an agnostic learning setting (i.e., settle for a hypothesis from a less expressive, easier to learn class, that does *not* contain the full set of potential concepts). Doing so is motivated by either the desire to reduce the *total* number of membership queries needed, or avoid making contrived or artificial queries (e.g. the inclusion of a highly unstable chemical in the biological assay).

[1] In tolerant testing, a generalization of property testing, the goal is distinguish the case where a function is "close" to a class of functions, or "far." A further generalization is the problem of estimating the distance of a function to a certain class of functions.

The Covert Verifiable Learning (CVL) model. The Covert Verifiable Learning model considers the case where, in addition to observing all queries and responses, the adversary (henceforth, the adversarial *intermediary*) also actively modifies the oracle's responses. Still, we require the learner to either detect the modifications and abort, or else come up with a good approximation of the actual concept represented by the oracle (which may in and of itself be an arbitrary function).

To make this requirement meaningful—namely, to allow the learner to meaningfully distinguish between responses that were modified by the adversarial intermediary and those that were not—we give the learner access to some number of ground truth random examples from the concept. (See Figure 2). We consider three variants of the CVL model, depending when the adversarial intermediary learns these additional random examples: In the weakest variant, the queries remain completely hidden throughout. In the intermediate model, we consider the case where the examples become known *once the learning process completes.* Finally, we consider our strongest variant, where these examples are publicly known *in advance.*

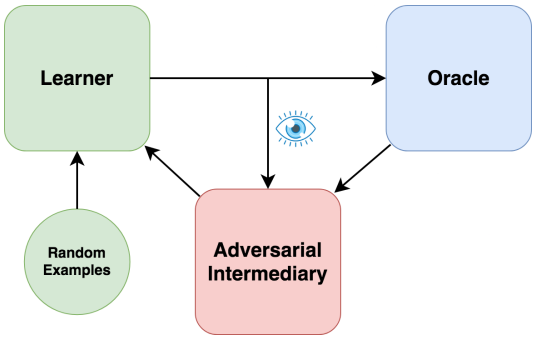

Figure 2: The "intermediate model" Covert Verifiable Learning scenario. A learner, given a set of random examples of a concept, accesses supplementary data on the concept using an oracle in the presence of an adversarial intermediary. While attempting to deduce information about the concept or the learner's hypothesis, the adversarial intermediary may tamper with the oracle responses (both to help steal information and to simply deceive the learner). Whereas, the learner aims to output a hypothesis that models the concept.

In more detail, the Covert Verifiable Learning model requires that, like Covert Learning, the output of the learner is a hypothesis $h \in \mathcal{H}$ that such that (w.r.t. the concept D) $\mathcal{L}_D(h) \leq \mathcal{L}_D(\mathcal{H}) + \epsilon$ with high probability, *but only when the adversarial intermediary simply observes and does not tamper with oracle responses.* The Covert Verifiable Learning model then augments the Covert Learning model by requiring that, for any adversarial intermediary that tampers with the oracle, the output of the learner is an $h \in \mathcal{H}$ that such that $\mathcal{L}_D(h) > \mathcal{L}_D(\mathcal{H}) + \epsilon$ with low probability, *assuming that the learner did not reject the interaction all together.*

The concept-hiding and hypothesis-hiding guarantees should still hold—albeit with an adversarial intermediary. To capture this stronger requirement, we adapt the simulation-based privacy of Covert Learning to embrace the active nature of the adversarial intermediary. Basically, we require that for any adversarial intermediary, there is a simulator that can interact with the adversarial intermediary such that no computationally bounded adversary be able to tell whether the adversarial intermediary is interacting with the actual learner or with the simulator. As in Covert Learning, the simulator will access random examples from the concept, but operate with no further knowledge about the concept and no knowledge of the learner's hypothesis class. Depending on the variant we consider, the adversary may have access to the learner's random examples (recall that in the intermediate setting, they leak subsequent the interaction).

Definition 1.2. ***Covert Verifiable Learning*** *(informal version of Definition 3.3) A covert verifiable learning algorithm—for a collection of hypothesis classes and with respect to a class of concepts and a loss function—is a learning algorithm that, for any concept in the class and accuracy parameters ϵ, δ, takes as input a target hypothesis class in the collection, a set of random examples from the concept, and interacts with an oracle that labels queries on the concept such that the following are true:*

- ***Completeness.*** *If the adversarial intermediary acts honestly (i.e. no oracle responses are corrupted), then the learning algorithm outputs an ϵ-good hypothesis for the concept with probability $1 - \delta$.*
- ***Soundness.*** *For any computationally bounded adversarial intermediary who tampers with oracle responses, if the learning algorithm does not reject then it outputs a hypothesis which is not ϵ-good with probability at most δ.*
- ***Privacy (intermediate model).*** *For any adversarial intermediary, there exists a simulator such the following two random variables are indistinguishable to an external adversarial entity which chooses the concept, the target hypothesis class, and accuracy parameters:*

  Real execution: *The output of the intermediary from a real interaction with the learning algorithm and the oracle, along with the set of random examples that the learner received in this interaction (the intermediary does not see the random examples).*

  Ideal execution: *The output of the simulator, along with the set of random examples that the learning algorithm received in the interaction. The simulator is given access to the set of random examples that were known to the learning algorithm, plus an additional set of random examples. However, the simulator can neither have further access to the concept nor have knowledge of the target hypothesis class.*

  If the output of the real execution does not include the random examples given to the learning algorithm, then we say that the algorithm is a covert verifiable learning algorithm with fully private examples.

  If the random examples given to the learning algorithm are also given to the intermediary, then we say that the protocol is a public covert verifiable learning algorithm.

For simplicity, we don't give the intermediary the ability to modify the queries. Indeed, an intermediary that is able to modify the learner's queries is arguably able to learn the function to begin with.

1.1.2 Overview of Results

As discussed, meaningful covert learning algorithms can exist only for learning problems where learning from random examples is hard, whereas learning with membership queries is feasible. However, it is not a priori clear that meaningful covert learning algorithms exist at all. In fact, to the best of our knowledge, for all known learning algorithms in the membership query model, an external observer can learn the function by just observing the queries and responses. This holds even when no efficient learning algorithms are known in the traditional PAC model (for instance, consider the algorithm of Kushilevitz and Mansour for decision trees [KM93], which is thought to be hard in the traditional PAC model [Blu03, OS07]).

This works constructs polynomial time, covert learning algorithms for salient learning tasks within the two new learning models. First, we consider the problem of Covert Learning for noisy parity

functions. In this problem, a secret n-bit parity function is generated by drawing an n-bit vector k, where each bit is sampled i.i.d. from a Bernoulli random variable with mean $1/\sqrt{n}$, and defining the parity function to be $f(x) = \langle x, k \rangle$. An example (x, y) is generated from a concept D^k_{LPN} which draws a uniformly random input x, and returns $y = f(x) \oplus 1$ with probability $1/\sqrt{n}$, and $y = f(x)$ otherwise. By the low-noise LPN assumption [Ale03], learning the hidden vector parity function from examples $(x, y) \sim D^k_{\mathsf{LPN}}$ is not possible in polynomial time. On the other hand, oracle queries to D^k_{LPN} make the problem tractable. Let $D_{\mathsf{LPN}} = \{D^k_{\mathsf{LPN}} \mid k \in \{0,1\}^n\}$. To this end, we define a hypothesis class $\mathcal{H}_T$ as the set of all parity functions on a subset of $T \subseteq [n]$. We show:

Theorem 1.3. *(Informal version of Theorem 2.15) Assuming hardness of the low-noise LPN assumption, there is a covert learning algorithm for the collection $\mathcal{C} = \{\mathcal{H}_T \mid T \subseteq [n]\}$, with respect to the concept class D_{LPN} and the loss function $\mathcal{L}_D$.*

Switching gears slightly, we demonstrate that our Covert Learning algorithm for noisy parities (up to syntactic modifications) suffices as a cryptographic key exchange protocol. Our key exchange protocol may be transformed into a public key encryption by following the KEM paradigm. The resulting protocol is similar in spirit to the public-key encryption scheme of Alekhnovich [Ale03] (over the binary field, from low-noise LPN, pseudorandom public keys and ciphertexts), but Alekhnovich encrypts a single bit directly. On the other hand, our key exchange communicates many bits at a time, and the resulting public-key encryption scheme allows sending messages of nearly $\sqrt{n}$ bits at once.

Next, we consider the following concept class. Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{-1,1\}$. $D_{\mathcal{F}}$ is a concept class indexed by $f \in \mathcal{F}$, where for any $D_f \in D_{\mathcal{F}}$, an example $(x, y) \sim D_f$ is generated by first sampling an input x uniformly at random, and then a returning $(x, f(x))$.

The first problem we consider is that of learning the "heavy" Fourier coefficients of a function. In this problem, the goal of a learner (given a function $f : \{0,1\}^n \to \{-1,1\}$) is to find the set of all k such that $\mathbb{E}_x[f(x)\chi_k(x)] \geq \tau$, where $\tau \geq 1/poly(n)$ is a given parameter and $\chi_k(x) = (-1)^{\langle k,x \rangle}$. We denote by $\hat{f}^{\geq \tau}_b$ the aforementioned set of k with the added stipulation that $|k| \leq b$. Achieving this goal using only examples $(x, y) \sim D_f$ is known to be as hard many longstanding open problems in computational learning theory, such as PAC learning DNF formulas, even when it is only required to find k such that $|k| = O(log\ n)$ [Blu03, Jac97, OS07]. On the other hand, membership queries make the problem tractable [GL89]. With this in mind, we define a hypothesis class $\mathcal{H}^b_T = \{\chi_k \mid k_i = 0 \implies i \notin T, |k| \leq b\}$, where $T \subseteq [n]$, and a loss function $\mathcal{L}^{\tau,b} : \mathcal{P}([n]) \to [0,1]$ given by

$$\mathcal{L}^{\tau,b}(T) = \Pr_{k \sim \hat{f}^{\geq \tau}_b}\left[\chi_k \in T\right]$$

where $k \sim \hat{f}^{\geq \tau}_b$ is a uniformly random sample $k \in \hat{f}^{\geq \tau}_b$ and $\mathcal{P}(S)$ denotes the powerset of a set S (we also require that $|T| \leq poly(n)$). We show:

Theorem 1.4. *(Informal version of Theorem 2.26) Let $\mathcal{F}$ be the class of all n-bit boolean functions. Assuming sub-exponential hardness of the standard LPN problem, there is a covert learning algorithm for the collection $\mathcal{C} = \{\mathcal{H}^b_T \mid T \subseteq [n]\}$, with respect to the concept $D_{\mathcal{F}}$ and the loss function $\mathcal{L}^{\tau,b}$ and for $b \leq O(log\ n)$, $\tau \geq 1/poly(n)$.*

In the problem of agnostically learning decision trees, a learner is given access to $D_f \in D_{\mathcal{F}}$ and tasked with finding (close to) the best decision tree that minimizes some loss function with respect to

D_f. This learning problem, too, is thought to be difficult in the traditional PAC model, but is known to be efficiently learnable with membership queries [KM93, Blu03]. Building on top of the covert learning algorithm for $O(log\ n)$-degree Fourier coefficients, we show:

Theorem 1.5. *(Informal version of Theorem 2.34) Assuming sub-exponential hardness of the standard LPN problem, there is a covert learning algorithm for the collection of all subsets of functions computable by poly(n) size decision trees with respect to the concept class $D_{\mathcal{F}}$ and the loss function $\mathcal{L}_D$.*

Unsatisfied with only the covert learning algorithms, we demonstrate how to transform our covert learning algorithms into covert verifiable learning algorithms. We do so both according to the intermediate setting and the stronger public variant. Specifically, in the intermediate setting we show:

Theorem 1.6. *(Informal version of Theorem 3.4) Assuming sub-exponential hardness of the standard LPN problem, there is a covert verifiable learning algorithm for the collection $\mathcal{C} = \{\mathcal{H}_T^b \mid T \subseteq [n]\}$, with respect to the concept $D_{\mathcal{F}}$ and the loss function $\mathcal{L}^{\tau,b}$, and for $b \leq O(log\ n)$, $\tau \geq 1/poly(n)$.*

Theorem 1.7. *(Informal version of Theorem 3.8) Assuming sub-exponential hardness of the standard LPN problem, there is a covert verifiable learning algorithm for the collection of all subsets of functions computable by poly(n) size decision trees with respect to the concept class $D_{\mathcal{F}}$ and the loss function $\mathcal{L}_D$.*

In the public variant, we prove:

Theorem 1.8. *(Informal version of Theorem 3.10) Let $s\text{-}\mathcal{DNF}_n$ be the class of all $f : \{0,1\}^n \to \{-1,1\}$ computable by a size s DNF formula. Assuming sub-exponential hardness of the standard LPN problem, there is a public covert verifiable learning algorithm for the collection $\mathcal{C} = \{\mathcal{H}_T^b \mid T \subseteq [n]\}$, with respect to the concept class $D_{s\text{-}\mathcal{DNF}_n}$ and loss function and the loss function $\mathcal{L}^{\tau,b}$, for $s \leq poly(n)$, $b \leq O(log\ n)$, and $\tau \geq 1/poly(n)$.*

In particular, the result of Theorem 3.10 gives the first verifiable PAC learning protocol without any private examples, even in the model of [GRSY20] which does not consider privacy. We also show a Covert Verifiable Learning algorithm with fully private examples, that obtains strong verifiability and hiding. Let $s\text{-}\mathcal{JUNTA}$ be the class of all $f : \{0,1\}^n \to \{-1,1\}$ computable by an s-junta. An s-junta is an n-bit boolean function with $n - s$ irrelevant variables. A variable is irrelevant if changing its value in the input never changes the output of the function.

Theorem 1.9. *(Informal version of Theorem 3.20) There is a covert verifiable learning algorithm with fully private examples with respect to the concept class $D_{s\text{-}\mathcal{JUNTA}}$, with statistical soundness and perfect privacy, where $s \leq O(log\ n)$.*

1.1.3 Algorithmic Ideas

We give high level descriptions of the algorithmic techniques. More formal overviews precede the constructions in the body of the paper.

Covert Learning of noisy parities. Our Covert Learning algorithm for learning noisy parities employs a "masked query" technique which works as follows. To mask a query $q \in \{0,1\}^n$, the learner starts by requesting n uniformly random examples from the oracle. Then, by taking the inputs of those random examples and drawing a random LPN secret, a "mask" is produced by multiplying the random inputs with the secret, and corrupting the resulting vector with independent random noise for each entry. Each query desired by the learner is then "masked" by adding the resulting sequence of LPN samples. In other words, each query is one-time-padded with an LPN instance. By the LPN assumption, a single masked query is pseudorandom. Moreover, the joint distribution for a set of masked queries is pseudorandom. The learner proceeds by sending the set of masked queries to the oracle, and upon receiving the results, decodes each one using the LPN secrets, the random examples, and by leveraging natural homomorphic properties provided by the LPN problem (with low noise). The simulation algorithm works by simply sampling queries from the uniform distribution, and pairing them with uniformly random results. We reduce the hardness of distinguishing the simulated transcript from the real transcript to solving the low-noise LPN problem.

Covert Learning of low-degree Fourier coefficients and decision trees. The covert learning algorithms for low-degree Fourier coefficients and decision trees use the same "masked query" technique as the covert learning algorithm for noisy parities. In particular, we use our "masked query" technique to run Goldreich-Levin queries on the (arbitrary) function in question. In contrast to the noisy parity setting, each individual query is not correctly decoded. Instead, the entire set of results is aggregated in a way resembling the original technique of Goldreich and Levin [GL89]. This allows us to then recover heavy Fourier coefficients belonging to $O(log\ n)$-degree parity functions. Due to the noise of the masking, the technique fails to extract higher degree coefficients. Once the set of $O(log\ n)$-degree parities that have noticeable Fourier coefficients is known, we employ standard techniques to produce a hypothesis which is the sign of a low-degree polynomial. We give a Fourier-based analysis that obtains agnostic learning guarantees on the hypothesis for the class of polynomial size decision trees. To demonstrate privacy, we adopt a variant of the LPN assumption that works over sparse secrets. The variant is due to [YZ16], whose hardness is implied by a sub-exponential hardness assumption on the standard LPN problem. We then construct a simulator that returns uniformly random examples of the function. We reduce a putative distinguisher between a real execution and a simulated execution to solving the variant of the LPN problem.

Augmenting the covert learning algorithms with verifiability. In order to engineer the verifiability guarantee into our covert learning algorithms, we use one main technique which works as follows. We take the the covert learning algorithms and wrap them with an outer loop, which at each iteration randomly decides to do a "learning" phase, where the covert learning algorithm is executed, or do a"test" phase. In a test phase, the algorithm sends a subset of the privately held queries to the oracle. Naturally, if the intermediary modifies any responses in this case, then the algorithm will detect that. Crucially, the distribution of queries in the learning phase (pseudorandom) is computionally indistinguishable from the distribution of queries in the test phase (uniformly random), due to the masking technique (and the LPN assumption). Therefore, this allows us to formalize the notion that no computationally bounded adversarial intermediary can reliably lie on the learning phase but not the testing phase—it would entail breaking the indistinguishability of the distributions of queries of the two phases and therefore the LPN hardness assumption.

When considering verifiability in the Public Covert Verifiable Learning setting (recall, here the learner does not have any private examples to leverage), the above technique does not immediately work.

However, we can modify it in a simple way as follows. As before, the learning phase consists of executing a covert learning algorithm. The testing phase is instead conducted by taking the public random examples and applying the same masking technique as used on the learning queries. Now, the test phase and the learning phase are still computationally indistinguishable to the adversarial intermediary, but the queries of the testing phase cannot be linked back to the public random examples. The learner can then decide if the intermediary is lying on the masked public examples by using the secret keys of the masks to unlock and measure a correlation between the oracle's responses on the masked public examples and the public labels. Like above, the adversarial intermediary generating an "acceptable" correlation, while reliably lying on the learning phase, would entail breaking the indistinguishability of the distributions of queries of the two phases and therefore the LPN hardness assumption.

Unconditional Covert Verifiable Learning with fully private examples. We design a Covert Verifiable Learning with fully private examples algorithm for $O(log\ n)$-juntas. The algorithm works by requesting random hamming neighbors of the privately held uniformly random example set, and using them to find all the $O(log\ n)$ relevant variables. Clearly, this means that the distribution of the membership queries is also uniform, despite the joint distribution being far from the concatenation of independent and uniformly random distributions. Since the adversary cannot see one component of the joint distribution, this suffices to give perfect privacy. By planting other private random examples (those which did not have random hamming neighbors requested), we may also prove that the protocol achieves statistical soundness against computationally unbounded adversarial intermediaries.

1.2 Real World Applications

Outsourcing of drug design and discovery. The drug design and discovery process begins by searching a massive space of chemical compounds for an "active" compound [LPP04, DAG06, DGRDR08]. A compound is called active (with respect to some biological structure) if it produces a reaction under some biological test (e.g. whether or not a molecule or compound binds with a protein). Quickly finding (and optimizing) active compounds among a massive search space is a primary goal of the drug discovery process.

The recent trend of drug companies delegating elements of the R&D process to well-equipped and specialized third parties who can carry out the necessary biological experiments on behalf of the drug companies has greatly enhanced the efficiency of the drug discovery process (for more information, see [Cla11] and the references therein). However, currently the outsourcing of experiments carries within it the risk of exposure of both the experimental design and experimental results. Indeed, much of the proprietary knowledge and intellectual property underpinning pharmaceutical science is generated in this way, but only until relatively recently was it not conducted in-house [Cla11].

One of the famous methods for carrying out drug discovery is *Quantitive Structure-Activity Relationship* modelling, or QSAR (for more information, see [DAG06] and the references therein). The QSAR methodology attempts to identify a relationship between compound activity and compound structure. As noted in [BHZ19], a compound or molecule may be described using a predefined set of features, which may then be linked to positive classification if it is active, and a negative classification if inactive. A membership query can be simulated by assembling a compound according to the specific attributes defined by the algorithm's query and submitting it to face some biological test. Thus, the process of privately and verifiably delegating QSAR modelling can be distilled to the following covert verifiable learning setting: Drug company A contracts a private lab to gather relevant data labelled by a function f, with the end goal of learning a model that provides a good approximation to f. In this case, A may want to prevent the private lab from:

1. Reselling or releasing the data (queries to f) to a competing drug company B, after collecting the data for A.

2. Leaking to B that A is interested in a certain type of model, or certain trade secrets like cutting edge domain knowledge that is revealed by the design of the queries.

3. Charging more money for arbitrary, complex or "high value" data (that is, data needed to learn expressive models like polynomial size DNF formulas).

4. Cutting costs by providing faulty data.

Using a covert verifiable learning algorithm in this setting achieves each of the above points, while maintaining the usual learning guarantees of the plain learning with membership queries setting. In particular, the concept-hiding guarantee prevents (1), as the queries requested by A are essentially useless to any other (computationally bounded) party. Meanwhile, hypothesis-hiding (for a relevant collection of hypothesis classes) counters (2) and (3), as ability to efficiently do either would clearly violate the guarantee. Ultimately, the verifiability requirement also prohibits a private lab from (4).

We note that decision trees are one of the standard ways used in QSAR modelling to obtain a relationship between molecule features and activity. Thus, the decision tree learning algorithms in this work are highly relevant.

Outsourcing of automated scientific discovery. Aside from the above drug discovery example, there are many more potential applications for the secure outsourcing of automated scientific discovery. Indeed, learning with membership queries [Ang88] has jump-started the exploration of its many diverse applications [BHZ19], in various areas of scientific discovery including functional genomics and molecular biology [KWJ+04], whole-genome shotgun sequencing [ABK+04], program synthesis [BDCVY17], VLSI testing and many others. In general, in any setting where a query can be synthesized and automatically answered (e.g. experiments such as lab tests), learning with membership queries may be a fruitful technique for automated scientific discovery [BHZ19]. In the same way as in the drug discovery example, many of the aforementioned applications can benefit from privacy and verifiability guarantees.

Attacks on MLaaS systems. Another notable application of Covert (Verifiable) Learning is in the context of model extraction attacks on machine-learning-as-a-service (MLaaS). In MLaaS, a machine learning company often deploys a trained ML model for (paying) clients to use, given access to an interface that implements query access to the model. In a model extraction attack, an adversary interacts with the query interface, attempting to obtain enough information about the underlying ML model to reverse engineer it. The relation of this attack threat to the learning with membership queries setting is clear, though (to the best of our knowledge) has just begun to be formalized in [CCG+20]. Due to the obvious financial and security motives of preventing proprietary MLaaS models from being abused—and specifically reverse-engineered—many defenses have been recently proposed. In one main type of defense that has been proposed, MLaaS providers monitor the queries submitted by a user, to decide when a client is benign (i.e. using the query interface in an honest way), or malicious (is attempting to reverse engineer the model) (e.g. [KMAM18], [JSMA19]). A cat-and-mouse sequence of attacks and defenses has ensued, while no rigorous guarantees have been obtained (for neither cat nor mouse). The Covert (Verifiable) Learning model provides a framework for studying the viability of query monitoring defenses. In fact, membership query learning algorithms under the (aptly named) Covert Learning model can be seen to circumvent such defenses. Indeed, the Covert Learning hiding guarantees prevent any computationally bounded passive adversary (in this case, any efficient extraction monitors)

from using a learner's (extractor's) queries to gather information about the concept (the MLaaS model), or the learner's resulting hypothesis (the extractor's reverse engineering). This raises concerns about the efficacy of query monitor defenses.

1.3 Related Work

Two recent works explore models related to ours and have influenced this work.

1.3.1 Cryptographic Sensing

Ishai et al. study a related scenario, called Cryptographic Sensing [IKOS19]. Like the present work, Cryptographic Sensing focuses on the goal of sensing (or, learning) properties of a physical object, while keeping these same properties secret to any passive adversary who does not have access to the internal randomness of the sensing algorithm. However, Cryptographic Sensing does not consider our notion of hypothesis-hiding, nor does it consider active intermediaries and verifiability. Furthermore, [IKOS19] chiefly focuses on *exact* learning of the object, where the aim is to decode the object exactly with non-noisy queries, and hiding is achieved for any high-entropy object. In contrast, our focus is on agnostic learning, where membership queries may return noisy responses. As a result, our model allows learning parities, whereas [IKOS19] only obtain learning algorithms for linear functions over larger moduli or over the integers. Another effect of noisy membership queries is that they allow concept-hiding even when there is a large and public labeled data set (the latter would rule out hiding a linear function in the noiseless setting of [IKOS19]). Indeed, our simulation-based definition will allow us to consider hiding in relation to auxiliary information about the concept, in a strong, zero-knowledge-like way. For more details on Cryptographic Sensing, see Section A.1.

1.3.2 PAC-verification

Goldwasser et al. initiated the study of PAC-verification [GRSY20], which aims to answer questions about the complexity of verifying machine learning models via interactive proofs. Among other scenarios, they consider the task where a prover, having learned a concept (perhaps via membership queries), wishes to convey the learned model to a distrusting third party (a verifier) that has only random examples from the concept. In this setting, they obtain a protocol for PAC-verification for the heavy Fourier coefficients (of any degree) of arbitrary functions. Their protocol is statistically sound in a model that corresponds to our CVL with fully private random examples model. That is, the prover has no access to the random examples available to the verifier. For more details about PAC-verification, see Section A.2. We note, however, that Goldwasser et al. do not consider (or obtain) any hiding requirements—neither concept hiding nor hypothesis hiding. Furthermore, while our covert verifiable learning algorithms offer only computational soundness, some of them provide soundness even in the setting where the random examples are known to the prover in advance.

1.3.3 Other Related Models

Delegation of computation. Though bearing some resemblance to the traditional cryptographic task of delegation of computation, our setting focuses on the specific task of learning. In this respect, we are focused on good *outcomes*, that is, guarantees on the efficacy of the learned hypothesis. In contrast, delegation of computation provides guarantees on the correctness of the computation steps themselves, and provides no guarantees on the learned hypothesis. For example, the delegation of computation model does not address the use of incorrect or poisoned data.

The PAC+MQ model. The power of membership queries in the agnostic setting was studied by Feldman in [Fel09]. Feldman defines an agnostic PAC+MQ learning model and, assuming existence of one-way functions, shows a particular learning problem that is computationally hard to learn in the agnostic PAC model with uniformly random examples, while efficiently learnable in the agnostic PAC+MQ model. Essentially, the agnostic PAC+MQ model augments the agnostic PAC learning model (Definition C.2) with query access to a membership oracle for the concept. It is possible to view our Covert Learning algorithms as working in a model that is between PAC and PAC+MQ, where the membership queries cannot be synthesized arbitrarily (as in PAC+MQ), but must be generated in a way that can be emulated by a simulation algorithm.

r-local membership queries. Another learning model lying between PAC and PAC+MQ was introduced in [AFK13]. There, *r-local* membership queries are permitted, in that any membership query must have have hamming distance r from an example received from the concept D. This requirement forces the membership queries to "look" like they are distributed according to D, but it falls short of our model. In contrast, we require that the membership queries, *in conjunction* with the examples from D, can be emulated by a simulation algorithm.

Differentially private learning. The study of differentially private learning was initiated in [KLN+11]. Roughly, the work of [KLN+11] asks what can hypothesis classes can learned by an algorithm whose output does not depend too heavily on any one specific training example. In essence, differentially private learning is concerned with maintaining the privacy of sensitive *training data* used by the learner. In contrast, our notion of privacy is orthogonal, as it pertains to the secrecy (with respect to third parties) of the underlying concept, and the hypothesis of the learner itself.

2 Covert Learning

In this section, we concentrate on the basic Covert Learning setting, which considers only eavesdropping attacks. We give a formal definition of Covert Learning. Next, we demonstrate how to construct a Covert Learning algorithm for a noisy parity learning problem as a warm-up. Then we extend the techniques used in the warm-up and show a Covert Learning algorithm for learning the $O(log\ n)$-degree Fourier coefficients of any function $f : \{0,1\}^n \to \{-1,1\}$. Using this algorithm as a subroutine, we obtain a Covert Learning algorithm for functions computable by polynomial size decision trees.

2.1 Preliminaries

We briefly recall here the standard terminology and notation which we use throughout the paper.

Definition 2.1. *A concept D_n is a joint distribution over an input domain $\mathcal{X}_n$ and label domain $\mathcal{Y}_n$.*

Definition 2.2. *A hypothesis class $\mathcal{H}_n$ is a set of functions $\mathcal{H}_n = \{h : \mathcal{X}_n \to \mathcal{Y}_n\}$.*

We call a sampled $(x, y) \sim D_n$ an *example*, where x is the *input* and y is the *label*. We use $\mathcal{X}_n = \{0,1\}^n$, and either $\mathcal{Y}_n = \{0,1\}$ or $\mathcal{Y}_n = \{-1,1\}$. We will use the term *concept class* denoted by $\mathcal{D}_n$ to signify a set of concepts (which are joint distributions over the input domain $\{0,1\}^n$ and label domain $\{0,1\}$).

Definition 2.3. *A concept oracle $\mathcal{O}_{D_n}$ for a concept D_n is a (probabilistic) oracle with the property that on query $z \in \{0,1\}^n$, $\mathcal{O}_{D_n}(z) = y$ with probability $\Pr_{D_n}[(x,y)|x = z]$, and $y \oplus 1$ otherwise.*

Finally, we very often use the notation to denote random variables of n-bit vectors.

Definition 2.4. *β_μ^n denotes the distribution over an n-bit vector where each of the bits is drawn i.i.d. from a Bernoulli random variable with mean μ.*

2.2 Definition of Covert Learning

In defining Covert Learning, we wish to require that the transcript of the interaction between a learner and a membership oracle reveals no information to a passive adversary about either:

- the concept, or
- the learner's chosen hypothesis class, or any auxiliary information that the learner has on the concept prior to the interaction.

Furthermore, these requirements should hold even when there is auxiliary information (in the form of random examples from the concept) available to the adversary.

As a starting point, we consider the learning with membership queries model, where the learner is given query access to a probabilistic oracle that responses queries about a concept (a concept oracle $\mathcal{O}_{D_n}$ for the concept D_n). The learner's goal is to find a hypothesis, out of some given class of hypotheses $\mathcal{H}_n$, that best approximates the concept D_n with respect to a loss function. For example, a loss function $\mathcal{L}_{D_n}(h) = \Pr_{(x,y)\sim D_n}[h(x) \neq y]$. This gives us a baseline model for accuracy guarantees in the learning with access to membership queries setting. However, we diverge from this model in an important way. Rather than define learning with respect to a single, fixed hypothesis class (as is common in learning theory), we use a *collection* of hypothesis classes. This will provide a natural way to model the desire to hide auxiliary information on the concept, as well as the chosen hypothesis class.

In more detail, we fix a collection of hypothesis classes $\mathcal{C}_n$, and require accuracy guarantees *for every* hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$: the learning algorithm will receive as input a description of a specific target hypothesis class within the collection, along with accuracy parameters $\epsilon, \delta > 0$. Then, the learning algorithm will *agnostically* learn the target hypothesis class with respect to a given loss function. For example, using the above example loss function, the learner will try to find an $h \in \mathcal{H}_n$ such that $\Pr_{(x,y)\sim D_n}[h(x) \neq y] \leq \inf_{h\in\mathcal{H}_n} \Pr_{(x,y)\sim D_n}[h(x) \neq y] + \epsilon$, with probability at least $1 - \delta$. That is, the algorithm should output a hypothesis—within the given target class—that best approximates the concept (up to the given accuracy parameters and a distribution over inputs). The input to the learner naturally models the *intent* of the learner, by capturing the particular choice of hypothesis class within the collection, and any auxiliary information used to select the class (e.g. the results of a tolerant testing algorithm or specific domain knowledge).

Finally, we will require that the transcript of the communication between the learner and the concept oracle does not leak any knowledge to an eavesdropper, in the following sense: we require that there exists a p.p.t. algorithm (a simulator) that generates an (ideal) simulated transcript of the (real) interaction between the learner and the concept oracle, with access to random examples from the concept, but not further access to the concept oracle. Furthermore, the simulator should operate without knowledge of the learner's target hypothesis class. The simulated transcript should be indistinguishable from a

real transcript, even to a (polynomial time) adversary that has access to auxiliary information on the concept.

More formally, we define two distributions, $\{\mathsf{real}_{\mathcal{A}}^{\mathcal{O}_{\mathcal{D}_n}}\}$ and $\{\mathsf{ideal}_{Sim}\}$ as follows.

Definition 2.5. *Let $\mathcal{D}_n$ be a concept class, and $\mathcal{C}_n$ a collection of hypothesis classes. We define $\{\mathsf{real}_{\mathcal{A}}^{\mathcal{O}_{\mathcal{D}_n}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) selects $\epsilon, \delta > 0$, a hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and a concept $D_n \in \mathcal{D}_n$.*
2. *A set of examples $\mathcal{S}$ is drawn from D_n.*
3. *A learner $\mathcal{A}$ receives ϵ, δ and $\mathcal{H}_n$, and begins interacting with the concept by querying the oracle $\mathcal{O}_{D_n}$ on examples of his choosing, receiving back responses for each queried example. $\mathcal{A}$ tries to agnostically learn $\mathcal{H}_n$ using this oracle. Denote the queries and responses as $\mathsf{transcript}_{\mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n,\epsilon,\delta)}$.*
4. *Output*
$$\left(\mathcal{H}_n, \epsilon, \delta, \mathsf{transcript}_{\mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n,\epsilon,\delta)}\right)$$

Definition 2.6. *Let Sim be a p.p.t. algorithm, which takes as input a set of random examples to a concept, and a length parameter which denotes the number of queries requested by the learner in the real interaction. We define $\{\mathsf{ideal}_{Sim}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) selects $\epsilon, \delta > 0$, a hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and a concept $D_n \in \mathcal{D}_n$.*
2. *A set of examples $\mathcal{S}$ is drawn from D_n.*
3. *A p.p.t. simulator Sim receives $\mathcal{S}, \ell$, and "interacts" with the $\mathcal{O}_{D_n}$ and outputs the set queries and responses denoted as $\mathsf{transcript}_{Sim(\mathcal{S},\ell)}$. Here ℓ is the number of queries that the learner requests in the real interaction.*
4. *Output*
$$\left(\mathcal{H}_n, \epsilon, \delta, \mathsf{transcript}_{Sim(\mathcal{S},\ell)}\right)$$

We note that the size of the random example set obtained by the simulator is given as a parameter of the definition of Covert Learning. Formally,

Definition 2.7. ***Covert Learning.*** *Let $\mathcal{C}_n$ be a collection of hypothesis classes, let $\mathcal{D}_n$ be a concept class, let $\mathcal{O}_{\mathcal{D}_n}$ be a class of oracles indexed by $D_n \in \mathcal{D}_n$, and let $\mathcal{L}$ be a loss function. $\mathcal{A}$ is a $(m(n), \alpha)$-covert learning algorithm for $\mathcal{C}$ with respect to $\mathcal{D}_n$, $\mathcal{O}_{\mathcal{D}_n}$ and $\mathcal{L}$ if for every $\epsilon, \delta > 0$, $\mathcal{A}$ satisfies the following:*

- *Completeness. For every distribution $D_n \in \mathcal{D}_n$, and every $\mathcal{H}_n \in \mathcal{C}_n$, the random variable $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta)$ satisfies*
$$\Pr_h\left[\mathcal{L}(h) \leq \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon\right] \geq 1 - \delta$$
*The loss function may depend on the distribution D_n. For **proper** Covert Learning, the output of $\mathcal{A}$ must be an element of $\mathcal{H}$, i.e. $h \in \mathcal{H}_n$.*

- *Privacy. There exists a p.p.t. simulator Sim such that:*

$$\left\{\mathsf{real}_{\mathcal{A}}^{\mathcal{O}_{\mathcal{D}_n}}\right\} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

where $\overset{c}{\approx}$ denotes computational indistinguishability[2]. We stipulate that the number of random examples given to the simulator is $O(m(n))$.

See Figure 3 for an illustration of the model. Often, we will use the terminology from the computational learning theory literature, and say that a collection of hypothesis classes $\mathcal{C}$ is *α-covertly learnable* if there exists an α-covert learning algorithm for $\mathcal{C}$.

In Appendix B, we give a relaxations of the Covert Learning definition, namely concept-hiding learning (which focuses on the concept-hiding guarantee).

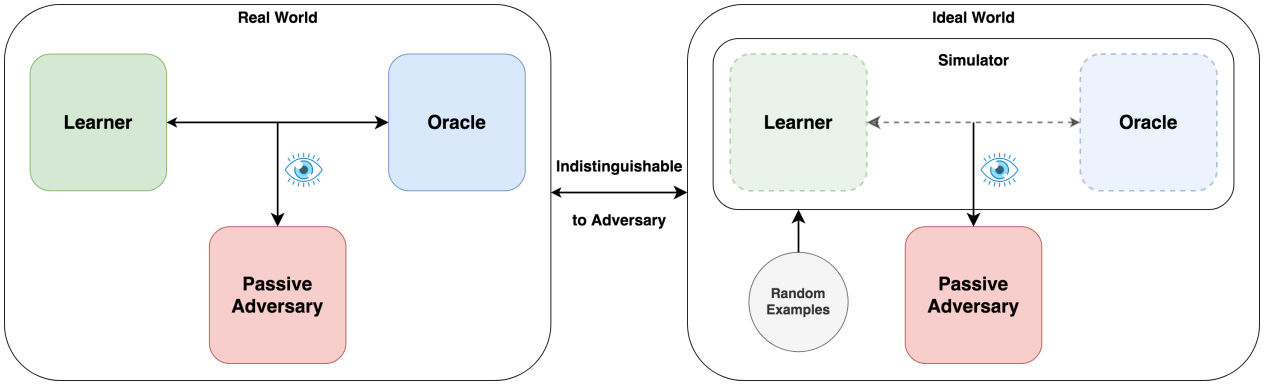

Figure 3: Privacy of Covert Learning. The "real world," where the adversary views the learner access the oracle, should be indistinguishable from the "ideal world," where the adversary interacts with a simulator that simulates the learner accessing the oracle. The adversary gets to choose the concept which is implemented by oracle (within the given class). Observe that, the simulator is also allowed random examples from the concept, and these are "leaked."

2.2.1 Discussion

About the simulation. We note that the simulation paradigm lends itself well to our setting: It allows formalizing the requirement that sensitive information is not revealed by the interaction, while maintaining the overall usefulness of the interaction. In this case, we formalize the notion that whatever could have been learned by a passive adversary about the concept or learner's hypothesis after the interaction, could have been learned *before* the interaction. Furthermore, we can model the presence of other, unavoidable information leakage (e.g. random examples on the concept).

Our focus is on collections of hypothesis classes that are not efficiently PAC learnable. When every hypothesis class in a collection is efficiently learnable without membership queries (i.e. with random examples only), Covert Learning is considered trivial. This is because in this case the privacy requirement is easily satisfied by a transcript full of random examples (and it does not even rule out leakage, because the adversary can learn the function from them). We thus concentrate on the case where the hypothesis classes within the collection need (or are assumed to need) membership queries to be learned.

[2]For a definition, see Appendix C.

2.3 A Warm-Up: Covert Learning of Noisy Parity Functions

In this section, we concentrate on Covert Learning of parity functions with noise. Indeed, this class of learning problems is broadly assumed to not be efficiently agnostically PAC-learnable in a very strong sense, as per the Learning Parity with Noise (LPN) assumption:

Definition 2.8. ***Search/Decision LPN assumption:*** *For $\mu \in (0, 0.5), n \in \mathbb{N}$, the $(m(n), T(n))$-$\mathsf{DLPN}_{\mu,n}$ assumption states that for every distinguisher $\mathbb{D}$ running in time $T(n)$,*

$$\left| \Pr_{s,\boldsymbol{A},e}[\mathbb{D}(\boldsymbol{A}, \boldsymbol{A}s \oplus e)] = 1 - \Pr_{r,\boldsymbol{A}}[\mathbb{D}(\boldsymbol{A}, r)] = 1 \right| \leq \frac{1}{T(n)}$$

where $s \xleftarrow{\$} \mathbb{Z}_2^n, \boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_2^{m(n)\times n}, e \xleftarrow{\$} \beta_\mu^{m(n)}, r \xleftarrow{\$} \mathbb{Z}_2^{m(n)}$.
For $\mu \in (0, 0.5), n \in \mathbb{N}$, the $(m(n), T(n))-\mathsf{SLPN}_{\mu,n}$ search assumption states that for every inverter $\mathbb{I}$ running in time $T(n)$,

$$\Pr_{s,\boldsymbol{A},e}[\mathbb{I}(\boldsymbol{A}, \boldsymbol{A}s \oplus e) = s] \leq \frac{1}{T(n)}$$

where $s \xleftarrow{\$} \mathbb{Z}_2^n, \boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_2^{m(n)\times n}, e \xleftarrow{\$} \beta_\mu^{m(n)}$.

Remark 2.9. *The search and decisional $\mathsf{LPN}_{\mu,n}$ assumptions are polynomially equivalent, in that an algorithm that breaks one of them can be turned into an algorithm that breaks the other in polynomial time. For more information, consult [Pie12] and the references therein.*

One typical setting of parameters gives the $\mathsf{DLPN}_{1/\sqrt{n},n}$ problem, which is conjectured to be $(O(n), poly(n))$-hard [Ale03]. However, for even super polynomial queries, the best known attacks are not asymptotically better than the $O(n)$ case [YS16]. Furthermore, an important variant was introduced in [ACPS09]. Specifically, it was shown that solving the decisional LPN problem when drawing the secret from the same distribution as the noise vector is as hard as drawing the secret from the uniform distribution. Henceforth, when referring to the $\mathsf{DLPN}_{\mu,n}$ problem, we refer to this setting.

In the remainder of this section, we construct a Covert Learning algorithm for the learning parity with noise problem using only the assumption that $\mathsf{DLPN}_{1/\sqrt{n},n}$ is hard itself—the minimal assumption that keeps the problem nontrivial: for any rate of noise bounded away from one half by an inverse polynomial, it easy (using majority voting) to solve $\mathsf{DLPN}_{\mu,n}$ when membership queries are available, and even in the "adversarial" noise case [GL89]. However, this is not enough for Covert Learning. It is clear that running membership query algorithms "in the open" (like Figure 1) may violate all our previously mentioned privacy goals.

2.3.1 The Learning Problem

As a warm-up, we will consider a distributional variant of Covert Learning. Here, the learning and privacy guarantees are required when the concept is drawn from a distribution over the concept class, rather than for every concept in the class. For privacy, this means that the distinguisher will not have the privilege of choosing the concept from the class, but instead it is sampled from the distribution. Our concept class is the following:

Definition 2.10. *Let $\mathcal{X}_n = \{0,1\}^n$. We define the concept class $\mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ to be the family of distributions over $\mathcal{X}_n \times \{0,1\}$ indexed by a $k \in \{0,1\}^n$, that have the following properties,*

- *The input (marginal) distribution over $\mathcal{X}$ of any $D_k \in \mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ is uniform.*
- *For any $D_k \in \mathcal{D}^{\mu,n}_{\mathsf{LPN}}$, the label $y \in \{0,1\}$ of the input x is generated by taking $\langle k, x\rangle$ and flipping the result with probability μ.*

For our learning problem, the concept will be drawn using a distribution over $\mathcal{D}^{\mu,n}_{\mathsf{LPN}}$:

Definition 2.11. *We define a distribution $\mathcal{M}^n_{\mathsf{LPN}}$ over the concept class $\mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ as follows. $D_k \in \mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ is selected by drawing $k \xleftarrow{\$} \beta^n_{1/\sqrt{n}}$.*

The learner will get membership access to the concept by using the following class of oracles:

Definition 2.12. *Let $\mathcal{O}_D$ be a concept oracle for a concept D. Recall the concept oracle that implements "membership query access" to a distribution D over $\mathcal{X} \times \mathcal{Y}$ in the following sense: on a query q to the oracle, a sample from the conditional distribution over $\mathcal{Y}$ is returned, given that $\mathcal{X} = q$.*

Hence, when the concept D_k is drawn from $\mathcal{M}^n_{\mathsf{LPN}}$, the learner will obtain access to $\mathcal{O}_{D_k}$. We will do Covert Learning for the following collection of hypothesis classes:

Definition 2.13. *For $a \in \{0,1\}^n$, let $\ell_a : \{0,1\}^n \to \{0,1\}$, defined by $\ell_a(x) = \langle a, x\rangle$. Let $\mathsf{PARITY}_{A,n} = \{\ell_a \mid i \notin A \implies a_i = 0\}$. Let $\mathcal{C}_{\mathsf{PARITY},n} = \{\mathsf{PARITY}_{A,n} \mid A \subseteq [n]\}$.*

Our Covert Learning task is then as follows. We would like to design a learning algorithm that takes as input any hypothesis class $\mathsf{PARITY}_{A,n} \in \mathcal{C}_{\mathsf{PARITY},n}$ (we will call A the *relevant set*). Then, given access to $\mathcal{O}_{D_k}$, the learning algorithm outputs $\ell_a \in \mathsf{PARITY}_{A,n}$ which minimizes the following loss (with respect to D_k):

Definition 2.14. *Let the loss function $\mathcal{L}_D$ be defined as*

$$\mathcal{L}_D(h) = \Pr_{(x,y)\sim D}[h(x) \neq y]$$

for a concept D.

Meanwhile, the privacy guarantee of Covert Learning should be satisfied. In particular, any information about A or k should be hidden.

2.3.2 The Construction

Overview. We will refer to $\mathcal{D}^{\frac{1}{\sqrt{n}},n}_{\mathsf{LPN}}$ as $\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}$. We construct Covert Learning for $\mathcal{C}_{\mathsf{PARITY},n}$ with respect to $\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}$, $\mathcal{O}_{\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}}$, and $\mathcal{L}_D$, and where learning is considered when the concept is drawn from $\mathcal{M}^n_{\mathsf{LPN}}$. The

Covert Learning algorithm begins by requesting "masked queries" from $\mathcal{O}_{D_k}$. For $x_1, \cdots, x_n \xleftarrow{\$} \beta^n_{1/2}$, let

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \end{bmatrix}$$

Note that, each x_i is a column of $\mathbf{X}$. Furthermore, let $e, s \xleftarrow{\$} \beta^n_{1/\sqrt{n}}$. A masked query $\hat{q} \in \{0,1\}^n$ for query $q \in \{0,1\}^n$ is generated by taking

$$\hat{q} = \mathbf{X}s \oplus e \oplus q$$

In our algorithm, each query q requested by the learner is a unit vector under the above masking, plus requests for the columns of $\mathbf{X}$. Indeed, the i^{th} unit vector is only masked and requested if the i^{th} index is in the relevant set A. Upon receiving the results to the masked unit vectors, denoted by $\mathcal{O}_{D_k}(\hat{q})$, the algorithm decodes each one by taking $\mathcal{O}_{D_k}(\hat{q}) \oplus \langle y, s \rangle$, where $y = (\mathcal{O}_{D_k}(x_1), \cdots, \mathcal{O}_{D_k}(x_n))$. It turns out that,

$$\Pr\left[\mathcal{O}_{D_k}(\hat{q}) \oplus \langle y, s \rangle = \langle k, q \rangle\right] > 0.501$$

And so, our algorithm requests each masked unit vector some constant number of times—the final decoding for each is done by taking the majority bit over the set of results from the duplicate queries. Note that, for a pair duplicate queries (say, two copies of the i^{th} unit vector), the masks are independently generated. Once the decoded results to the masked unit vectors are obtained, the algorithm produces a hypothesis in the natural way: if r_i is the decoded result of the masking query of the i^{th} unit vector, then the output hypothesis is $r(x) = \langle (r_1, \cdots, r_n), x \rangle$.

We now give the construction.

Algorithm: ConstructQueries(s, A)
Let u_i denote the vector with the i^{th} component as 1 and all other components 0.
Draw n uniformly random queries $x_1, \cdots x_n$.
Let $\mathbf{X}$ be the matrix formed by letting x_i be the i^{th} row.
$m = |A|$
for $j \in [m]$ **do**
  $e_j \xleftarrow{\$} \beta^n_{n^{-0.5}}$
end
for $a \in A$ **do**
  $q^a_j = \mathbf{X}^\mathsf{T} s_j \oplus e_j \oplus u_a$
end
$Q = \{q^a_j \mid j \in [m]\}$
$X = \{x_i \mid i \in [n]\}$
output X, Q

Algorithm 1: Algorithm to construct membership queries for CLP

1 **Algorithm:** $\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)$
2 $m = |A|$
3 **for** $i \in [O(log(m/\delta))]$ **do**
4 **for** $j \in [m]$ **do**
5 $s_j \xleftarrow{\$} \beta^n_{n^{-0.5}}$
6 **end**
7 $s = (s_1, \cdots, s_j)$
8 $X, Q = \mathsf{ConstructQueries}(s, A)$
9 *Query the oracle for all the queries in X. Denote the result on x_i as y_i.*
10 **for** *each* $q^a_j \in Q$ **do**
11 *Query the oracle for q^a_j. Denote the result as r^a_j.*
12 $d_{i,j,a} = r^a_j \oplus \langle (y_1, \cdots, y_n), s_j \rangle$
13 **end**
14 **end**
15 **if** $a \in A$ **then**
16 $z_j = \mathsf{majority}_i(d_{i,j,a})$
17 **else**
18 $z_j = 0$
19 **end**
20 $z = (z_1, \cdots, z_j)$
21 Let $\ell_z(x) = \langle x, z \rangle$
22 **output** ℓ_z

Algorithm 2: Covert Learning algorithm for noisy parities

We claim that the above algorithm CLP is a $(poly(n), 1)$-covert learning algorithm for $\mathcal{C}_{\mathsf{PARITY},n}$. We state the theorem first, and spend the rest of this section assembling the proof. We divide the proof into two separate claims, which then combine to prove Theorem 2.15.

Theorem 2.15. *Assuming $\mathsf{DLPN}_{\mu,n}$ is $(O(n), poly(n))$-hard, $\mathcal{C}_{\mathsf{PARITY},n}$ is $(poly(n), 1)$-covertly learnable with respect to $\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}$, $\mathcal{O}_{\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}}$, and $\mathcal{L}_D$, and where the concept is drawn according to $\mathcal{M}^n_{\mathsf{LPN}}$.*

Proposition 2.16. *For any $n \in \mathbb{N}$, and $\epsilon, \delta > 0$, and $D_k \sim \mathcal{M}^n_{\mathsf{LPN}}$, $\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)$ satisfies*

$$\Pr_{(x,y)\sim D_k}\left[\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)(x) \neq y\right] - \inf_{h \in \mathsf{PARITY}_{A,n}} \Pr_{(x,y)\sim D_k}\left[h(x) \neq y\right] \leq \epsilon$$

with probability at least $1 - \delta$.

Proof. Let $K = \{i \mid k_i = 1\}$ (recall that k denotes the hidden parity from the concept D_k). First, we consider the case that $K \subseteq A$. Let us examine the line 12 of CLP at each iteration. For each $d_{i,j,a}$ we

have that

$$\begin{aligned}
d_{i,j,a} &= \mathcal{O}_{D_k}(q_j^a) \oplus \langle (y_1, \cdots, y_n), s_j \rangle \\
&= \langle q_j^a, k \rangle \oplus r \oplus \langle \mathbf{X}k \oplus r^n, s_j \rangle && \text{(where } r \xleftarrow{\$} \beta_{n^{-0.5}}, r^n \xleftarrow{\$} \beta_{n^{-0.5}}^n\text{)} \\
&= \langle \mathbf{X}^\mathsf{T} s_j \oplus e_j \oplus u_a, k \rangle \oplus r \oplus \langle \mathbf{X}k \oplus r^n, s_j \rangle && \text{(where } u_a \text{ is the } a^{th} \text{ unit vector)} \\
&= \langle u_a, k \rangle \oplus \langle e_j, k \rangle \oplus r \oplus \langle r^n, s_j \rangle \\
&= \langle u_a, k \rangle && \text{(with probability at least 0.501)}
\end{aligned}$$

where the first step follows from the definition of $\mathcal{O}_{D_k}$, the second and third steps use linearity, and the final step is implied by the Piling-up lemma (Lemma C.5), and the fact that the terms $\langle e_j, k \rangle, r, \langle r^n, s_j \rangle$ are each independent from each other. In particular, for sufficiently large n, $\Pr[\langle r^n, s_j \rangle = 1] > 0.56$ and $\Pr[\langle e_j, k \rangle = 1] > 0.56$, while $\Pr[r = 0] \geq 0.99$. Thus for sufficiently large n,

$$\Pr\Big[\langle e_j, k \rangle \oplus r \oplus \langle r^n, s_j \rangle = 0\Big] \geq 0.99\big(0.56^2 + (1 - 0.56)^2\big) > 0.501$$

By standard arguments following from the sampling bound (Lemma C.11) and the union bound (Lemma C.10), we can conclude that iterating (see lines 3-13 of CLP) $O(log(\frac{m}{\delta}))$ times and taking a majority vote for each bit (lines 14-16), suffices to satisfy the statement in Proposition 2.16. In slightly more detail, the sampling bound gives that the fraction of the $O(log(\frac{m}{\delta}))$ trials that are equal to $\langle u_a, k \rangle$ is larger than a half, with probability $1 - \frac{\delta}{m}$. Then, by the union bound, we can conclude that with probability $1 - \delta$ for all bit indices in A the majority vote winner equals $\langle u_a, k \rangle$. As a result, CLP outputs the correct underlying parity function of $\mathcal{O}_{D_k}$. Since any pair of parity functions are $\frac{1}{2}$-far from each other, it follows that the output hypothesis is ϵ-close to the optimal hypothesis in $\mathsf{PARITY}_{A,n}$. Formally,

$$\Pr_{(x,y)\sim D_k}\Big[\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)(x) \neq y\Big] - \inf_{h \in \mathsf{PARITY}_{A,n}} \Pr_{(x,y)\sim D_k}\Big[h(x) \neq y\Big] \leq \epsilon$$

with probability at least $1 - \delta$.

Now, we consider the alternative case, namely, when $K \not\subseteq A$. In this case, it is clear that the output of CLP is always a parity function which is $\frac{1}{2}$-far from the hidden parity. Since the noise is independently generated for each example, it follows that

$$\Pr_{(x,y)\sim D_k}\Big[\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)(x) \neq y\Big] = \inf_{h \in \mathsf{PARITY}_{A,n}} \Pr_{(x,y)\sim D_k}\Big[h(x) \neq y\Big]$$

Thus,

$$\Pr_{(x,y)\sim D_k}\Big[\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n}, \epsilon, \delta)(x) \neq y\Big] - \inf_{h \in \mathsf{PARITY}_{A,n}} \Pr_{(x,y)\sim D_k}\Big[h(x) \neq y\Big] \leq \epsilon$$

with probability 1.

□

Continuing with the proof of Theorem 2.15, we introduce a second claim.

Proposition 2.17. *Assuming that the* $\mathsf{DLPN}_{\frac{1}{\sqrt{n}},n}$ *problem is* $(O(n), poly(n))$*-hard,* CLP *satisfies the privacy guarantee of Covert Learning for* $\mathcal{C}_{\mathsf{PARITY},n}$ *with respect to* $\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}$, $\mathcal{O}_{\mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}}$, *and* $\mathcal{L}_D$.

Proof. We will construct a simulator Sim that satisfies

$$\left\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\right\} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

We first point out that, since the chosen queries are not inherently adaptive, it is possible to run all the iterations of CLP in parallel. We may compress all the messages into one large message that fits into a $O(n \cdot log(\frac{n}{\delta})) \times n$ matrix, and decode the results from there. We argue security in this setting. Consider the following p.p.t. simulation algorithm.

Algorithm: $Sim(\mathcal{S}, \ell)$
Let $(R, r) = (\ell$ uniformly random queries, ℓ uniformly random bits)
output (R, r)

Algorithm 3: Simulation algorithm for privacy of CLP

Observe that the simulator never actually makes any use of the random examples from the concept, so we will actually prove a stronger statement than necessary.

Note that

$$\left\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\right\} = \left\{\left(\mathsf{PARITY}_{A,n}, \epsilon, \delta, \mathsf{transcript}_{\mathsf{CLP}^{\mathcal{O}_{D_k}}(\mathsf{PARITY}_{A,n},\epsilon,\delta)}\right)\right\}$$

where $\mathsf{PARITY}_{A,n}, \epsilon, \delta$ are chosen ahead of time by the adversary, and where $D_k \sim \mathcal{M}_{\mathsf{LPN}}^n$. In a similar fashion, we define the hybrid distribution

$$\mathsf{hybrid} = \left\{(\mathsf{PARITY}_{A,n}, \epsilon, \delta, R, \mathcal{O}_{D_k}(R)) | R \leftarrow \ell \text{ uniformly random queries}\right\}$$

First, we will show that $\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\} \overset{c}{\approx} \mathsf{hybrid}$. We may write $\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\}$ as

$$\left\{(\mathsf{PARITY}_{A,n}, \epsilon, \delta, \mathsf{transcript\text{-}queries}, \mathsf{transcript\text{-}responses})\right\}$$

Observe that then,

$$\left\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\right\} = \left\{\left(\mathsf{PARITY}_{A,n}, \epsilon, \delta, \mathsf{transcript\text{-}queries}, \mathcal{O}_{D_k}(\mathsf{transcript\text{-}queries})\right)\right\}$$

Supposing that there is an algorithm $\mathbb{D}$ that can distinguish $\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\}$ and hybrid, it is clear that there exists an algorithm that distinguishes R and $\mathsf{transcript\text{-}queries}$ by drawing the concept from $\mathcal{M}_{\mathsf{LPN}}^n$, choosing the hypothesis class and accuracy parameters according to $\mathbb{D}$, simulating oracle access internally, and then using $\mathbb{D}$ as a subroutine. This contradicts the LPN assumption, since distinguishing $\mathsf{transcript\text{-}queries}$) from R amounts to solving $O(n \cdot log(\frac{n}{\delta}))$ $\mathsf{DLPN}_{\frac{1}{\sqrt{n}},n}$ instances. Thus $\mathsf{real} \overset{c}{\approx} \mathsf{hybrid}$. It remains to show that

$$\mathsf{hybrid} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

This follows immediately from the same low noise LPN hardness assumption, and taking into account that $\mathcal{O}_{D_k}$ implements low noise LPN examples. Assuming a distinguisher $\mathbb{D}$ for hybrid and $\{\mathsf{ideal}_{Sim}^{\mathcal{M}_{\mathsf{LPN}}^n}\}$, an algorithm to solve the $\mathsf{DLPN}_{\frac{1}{\sqrt{n}},n}$ would be as follows: Select a hypothesis class and accuracy parameters as $\mathbb{D}$ would. Then, pair them with the challenge, and run the distinguisher. Clearly, any distinguishind advantage of $\mathbb{D}$ is inherited. We can thus conclude that

$$\left\{\mathsf{real}_{\mathsf{CLP}}^{\mathcal{O}_{D_k}}\right\} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

□

Proof of Theorem 2.15. Theorem 2.15 follows immediately by combining Proposition 2.16 and Proposition 2.17 to satisfy the necessary guarantees.

□

Remark 2.18. *The algorithm is efficient. The queries are constructed in time polynomial in n, and the same is true for the decoding process. From there, it's easy to see that since we run $O(log(n/\delta))$ iterations, the entire algorithm runs in time polynomial in n and $log(1/\delta)$.*

2.4 Covert Learning of Low-degree Fourier Coefficients

In this section, we will extend our techniques from the warm-up to present a Covert Learning algorithm for "heavy" $O(log\ n)$-degree Fourier coefficients. We refer the reader to Appendix D for a very brief introduction to Fourier analysis if necessary. This learning problem will no longer live in the distributional learning case, as in the warm-up.

The learning problem is nontrivial for Covert Learning: the problem of efficiently identifying heavy, even $O(log\ n)$-degree, Fourier coefficients from random examples is a fundamental problem that has so far evaded intense research effort from the learning theory community. In particular, such an algorithm would imply the PAC learnability of DNFs via a "weak learning" parity function and boosting results [Jac97]. Moreover, such an algorithm would be considered a massive breakthrough in computational learning theory [Blu03, OS07].

2.4.1 Our Task

We consider the following natural class of concepts.

Definition 2.19. *Let $\mathcal{X}_n = \{0,1\}^n$, and $\mathcal{F}_n$ be a class of functions $f : \mathcal{X}_n \to \{-1,1\}$. We define $\mathcal{D}_{\mathcal{F}_n}$. to be the concept class containing all distributions over $\mathcal{X}_n \times \{-1,1\}$ that have the following properties,*

- *The input (marginal) distribution over $\mathcal{X}_n$ of any $D_f \in \mathcal{D}_{\mathcal{F}_n}$ is uniform.*
- *For any $D_f \in \mathcal{D}_{\mathcal{F}_n}$, there exists a polynomial time computable target function $f : \mathcal{X}_n \to \{-1,1\}$ such that $f \in \mathcal{F}_n$ and*

$$\Pr_{(x,y)\sim D_f}[f(x)y = 1] = 1$$

The learner will be allowed to interact with a membership query oracle to any concept in $\mathcal{D}_{\mathcal{F}_n}$.

Definition 2.20. *Let $\mathcal{O}_{\mathcal{F}_n}$ be a class of membership oracles indexed by $D_f \in \mathcal{D}_{\mathcal{F}_n}$, such that $\mathcal{O}_f$ implements membership query access to f. To simplify notation, we may write $\mathcal{O}_f$ instead of $\mathcal{O}_{D_f}$.*

Hence, the learner will have access to $\mathcal{O}_f$ when tasked with learning the "heavy" Fourier coefficients of the target function of D_f.

In plain English, the task is as follows. The learner chooses a hypothesis class characterized by a subset T of $[n]$ and a bound $b < n$, where the hypothesis class consists of a subset of all n-bit parity functions. A parity function is in the hypothesis class if it is of degree less than b and if all it's relevant

variables are included in T. The learner must then find all parity functions in the hypothesis class which have Fourier coefficients larger than some given threshold τ.

More formally, the goal is to learn the following hypothesis class with respect to the following loss function:

Definition 2.21. *Let $k \in \{0,1\}^n$. Define the parity function $\chi_k : \{0,1\}^n \to \{-1,1\}$ as*

$$\chi_k(x) = (-1)^{\langle k,x \rangle}$$

We will call $|k|$ the degree of χ_k.

Definition 2.22. *Let $T \subseteq [n]$. Define the hypothesis class $\mathsf{FOURIER}_{T,b,n} = \mathcal{P}\{\chi_k \mid k_i \notin T \implies k_i = 0, \wedge |k| \leq b\}$. In other words, $\mathsf{FOURIER}_{T,b,n}$ is the powerset of the set all parity functions on subsets of $[n]$ contained in T, with degree at most b. Let the collection of hypothesis classes $\mathcal{C}_{\mathsf{FOURIER},b,n} = \{\mathsf{FOURIER}_{T,b,n} \mid T \subseteq [n]\}$. For any hypothesis class $\mathsf{FOURIER}_{T,b,n} \in \mathcal{C}_{\mathsf{FOURIER},b,n}$, we will call T the relevant set.*

Definition 2.23. *Let $f : \{0,1\}^n \to \{0,1\}$ be a function. Let $P = \mathcal{P}\{\chi_k \mid k \in \{0,1\}^n\}$. Also, let $\hat{f}_b^{\geq \tau} = \{\chi_k \mid \hat{f}(k) \geq \tau, |k| \leq b\}$. $\mathcal{L}^{\tau,b} : P \to [0,1]$ is a loss function given by*

$$\mathcal{L}^{\tau,b}(T) = \begin{cases} \Pr\limits_{\chi_k \sim \hat{f}_b^{\geq \tau}} \left[\chi_k \in T\right] & \text{when } |T| \leq poly(n) \\ 1 & otherwise \end{cases}$$

where $\chi_k \sim \hat{f}_b^{\geq \tau}$ is a uniformly random sample $\chi_k \in \hat{f}_b^{\geq \tau}$.

The learning algorithm, given a hypothesis class $\mathsf{FOURIER}_{T,b,n} \in \mathcal{C}_{\mathsf{FOURIER},b,n}$, should hide any information about the relevant set T, as well as any information about f, as formalized by the privacy guarantee of Covert Verifiable Learning. Finally, the protocol should achieve computational soundness and be efficient (i.e. run in time $poly(n, 1/\tau, log(1/\delta)$ for a soundness parameter δ).

2.4.2 The Construction

Overview. We construct Covert Verifiable Learning for $\mathcal{C}_{\mathsf{FOURIER},b,n}$ with respect to $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, and $\mathcal{L}^{\tau,b}$. The overall flow of the algorithm will be similar to that of our Covert Learning algorithm for noisy parities. Instead of using the masking technique to encode unit vectors, we instead use masking to run Goldreich-Levin queries.

Theorem 2.24. ***Goldreich-Levin learning algorithm.*** *Given query access to a function $f : \{0,1\}^n \to \{-1,1\}$ and given parameters τ, δ, there exists a $poly(n, \frac{1}{\tau}, \frac{1}{\delta})$ time algorithm that outputs a list $L = \{S_1, ..., S_\ell\}$ such that the following hold,*

1. *if $|\hat{f}(S)| \geq \tau$, then $S \in L$.*
2. *if $S \in L$, $|\hat{f}(S)| \geq \frac{\tau}{2}$.*

with probability $1 - \delta$.

The Goldreich-Levin queries are those that are selected by the above algorithm. Using the Goldreich-Levin algorithm, all the Fourier coefficients satisfying $|\hat{f}(S)| \geq 1/poly(n)$ can be found in polynomial time (with high probability).

For any subset $T \subseteq [n]$, the Goldreich-Levin algorithm can be executed in a way that it only outputs subsets S such that $S \subseteq T$. In this case, the algorithm skips doing majority voting on the indices not in T, and uses less queries. In the event that the algorthim is executed in the described restricted manner, we will refer to the queries as the Goldreich-Levin queries on T.

For our construction, we will need the following "chopped tail" binomial distribution.

Definition 2.25. *Define the distribution, $\tilde{\beta}^n_\mu$, as the output of the following process. Draw $y \sim \beta^n_\mu$, and if $\mu n/2 \leq |y| \leq 3\mu n/2$ output y, else output $\perp$.*

For $\mu = \log(n)/n$, $\tilde{\beta}^n_\mu$ can be seen to have min-entropy $\Theta(log^2\ n)$ [YZ16].

Fixing a $D_f \in \mathcal{D}_{\mathcal{F}_n}$, the Covert Learning algorithm begins by requesting "masked queries" from $\mathcal{O}_f$. Let $\ell = \Theta(log^2 n)$. For $x_1, \cdots, x_n \xleftarrow{\$} \beta^\ell_{1/2}$ and $y_1, \cdots, y_\ell \xleftarrow{\$} \beta^n_{1/2}$, let

$$\mathbf{U}_1 = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \end{bmatrix}, \mathbf{U}_2 = \begin{bmatrix} y_1 & y_2 & y_3 & \cdots & x_\ell \end{bmatrix}$$

Note that, each x_i or y_i is a column of $\mathbf{U}_1, \mathbf{U}_2$. Now, let $\mathbf{X} = \mathbf{U}_2\mathbf{U}_1$ Furthermore, let $s \xleftarrow{\$} \tilde{\beta}^n_\mu$ for $\mu = \log(n)/n$. A masked query $\hat{q} \in \{0,1\}^n$ for query $q \in \{0,1\}^n$ is generated by taking

$$\hat{q} = \mathbf{X}s \oplus e \oplus q$$

In the algorithm, each query q requested by the learner is a Goldreich-Levin query under the above masking. Indeed, the Goldreich-Levin queries are only masked and requested if they are one of the Goldreich-Levin queries on the relevant set T given by the target hypothesis class (as discussed above).

Upon receiving the responses to the masked Goldreich-Levin queries, denoted by $\mathcal{O}_f(\hat{q})$, the secret s for the masked query $\hat{q}$ is utilized to post-process $\mathcal{O}_f(\hat{q})$. The post-processed responses correlate with $f'(q)$, where f' is a function whose $O(log\ n)$-degree Fourier coefficients are greater than $\Omega(\tau n^{-2})$ wherever f has a Fourier coefficient greater than τ. By following the technique of Goldreich and Levin, we recover all the $O(log\ n)$-degree Fourier coefficients of f' greater than $\Omega(\tau n^{-2})$—and therefore all the $O(log\ n)$-degree Fourier coefficients of f greater than τ.

To prove privacy of the algorithm, the idea is to produce a simulator that first emulates the learner's queries, and then can interact with an AI by also passing as the oracle to the concept. It turns out that, assuming subexponential hardness of LPN, the masking procedure described above maps each query to a pseudorandom distribution. Thus, we will construct a simulator that requests truly random queries. Intuitively, it can then be shown that if there exists and AI such that an adversary can distinguish between the simulated interaction and the real interaction (where the requested queries are pseudorandom), then the adversary can distinguish the pseudorandom masked queries from truly random queries.

We now give the construction.

1 **Algorithm:** $\mathsf{CLF}(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \tau)$

2 *Initialize* $\mathbb{L} = \emptyset$, $\theta = \tau n^{-2}$, $t = 2log(\frac{1}{\theta})$, $k = |T|$, $m = O(log\ n) \cdot k2^t$

3 **for** $i \in [O(log(\delta/\tau))]$ **do**

4 $\quad \mathbf{L} = \emptyset$

5 $\quad$ Select $x_1, \cdots, x_t \in \{0,1\}^n$ uniformly at random.

6 $\quad$ Define $x^S = \bigoplus_{\sigma \in S} x_\sigma$, and let u_j be the j^{th} unit vector of length n.

7 $\quad$ *Initialize the Goldreich-Levin queries on T as follows:*

8 $\quad$ **for** $\ell \in [O(log\ n)]$ **do**

9 $\quad\quad z \xleftarrow{\$} \{0,1\}^n$

10 $\quad\quad$ **for** $S \subseteq [t] \setminus \emptyset, j \in T$ **do**

11 $\quad\quad\quad$ Define $\mathrm{gl}_{S,j,z} = x^S \oplus u_j \oplus z$

12 $\quad\quad\quad$ Define $\mathrm{gl}_{S,-,z} = x^S \oplus z$

13 $\quad\quad$ **end**

14 $\quad$ **end**

15 $\quad$ *Initialize and apply the masking of the GL queries as follows:*

16 $\quad$ **for** $i \in [m]$ **do**

17 $\quad\quad \mathbf{R}_i = \mathbf{U}_{n\times\lambda}\mathbf{U}_{\lambda\times n}$ where $\mathbf{U}_{q\times r}$ is a uniformly random $q \times r$ binary matrix and $\lambda = \Theta(log^2 n)$.

18 $\quad\quad s_i \xleftarrow{\$} \tilde{\beta}^n_{\mu^*}$ for $\mu^* = log(n)/n$ $\qquad$ (s_i is the i^{th} *secret.*)

19 $\quad\quad$ If $\tilde{\beta}^n_{\mu^*}$ outputs $\perp$, then stop and output **reject**.

20 $\quad\quad e_i \xleftarrow{\$} \beta^n_\mu$ for $\mu = 1/8$.

21 $\quad$ **end**

22 $\quad$ For each GL query, apply a masking as follows: Without loss of the indexing defined above, we define the i^{th} masked GL query as $\hat{\mathrm{gl}}^i_{S,j,z} = \mathbf{R}_i s_i \oplus e_i \oplus \mathrm{gl}_{S,j,z}$ and $\hat{\mathrm{gl}}^i_{S,-,z} = \mathbf{R}_i s_i \oplus e_i \oplus \mathrm{gl}_{S,-,z}$

23 $\quad$ *Request the masked Goldreich-Levin queries from the oracle*

24 $\quad$ Letting $\tilde{f}(x)$ be the claimed result by the AI of query x, define an i^{th} "unmasking function" $\phi_i(x) = \tilde{f}(x)\chi_{s_i}(r)$ where $r \xleftarrow{\$} \beta^n_{1/2}$.

25 $\quad$ **for** every $(b_1, \cdots, b_t) \in \{-1,1\}^t$ **do**

26 $\quad\quad$ Initialize $a \in \{0,1\}^n$ by $a_j = 0\ \forall j \in [n]$.

27 $\quad\quad$ Define $b^S = \prod_{\sigma \in S} b_\sigma$

28 $\quad\quad$ **for** $j \in T$ **do**

29 $\quad\quad\quad$ Compute $a_{jz} = \mathsf{majority}_{S\subseteq[t]\setminus\emptyset}(\phi_i(\hat{\mathrm{gl}}^i_{S,-,z}) \cdot b^S) \cdot \mathsf{majority}_{S\subseteq[t]\setminus\emptyset}(\phi_i(\hat{\mathrm{gl}}^i_{S,j,z}) \cdot b^S)$. Here, we apply ϕ_i to the i^{th} masked GL query, which is indexed by $S, j/-, z$.

30 $\quad\quad\quad$ Compute $a_j = \mathsf{majority}_z(a_{jz})$

31 $\quad\quad$ **end**

32 $\quad\quad a = (a_1 \cdots a_n)$

33 $\quad\quad \mathbf{L} = \mathbf{L} \cup \chi_a$

34 $\quad$ **end**

35 $\quad \mathbb{L} = \mathbb{L} \cup \mathbf{L}$

36 **end**

37 Repeat lines 3-36, while instead defining $\phi_i(x) = (-1)\tilde{f}(x)\chi_{s_i}(r)$ (in line 24)

38 **output** $\mathbb{L}$

Algorithm 4: Covert Learning of large, low-degree Fourier coefficients

Henceforth, we will let N denote the number of queries requested by CLF. Note that, $N = O(log(\delta/\tau) \cdot k(log\ n)/\theta^2)$.

Theorem 2.26. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption and for sufficiently large* n, CLF *is a proper* $(poly(n), 1)$*-covert learning algorithm for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $b \leq O(log\ n), \tau \geq 1/poly(n)$, $\delta \geq exp(-n)$.

We claim that CLF satisfies completeness, privacy and computational soundness, and now prove each. The following completeness proof (Proposition 2.30) will use standard Fourier-analytic notation. We refer the reader to Appendix D for an overview (if necessary).

We need the following sequence of lemmas to prove completeness.

Lemma 2.27. *For any integer* $n \geq 1$ *and* $a \sim \beta^n_{\frac{1}{2}}, b \sim \beta^n_{\mu^*}$, $\mu^* = log(n)/n$, $\Pr[\langle a, b\rangle = 0] \geq \frac{1}{2} + \frac{1}{4n}$.

Proof. Note that, for $n \geq 1$, $1 - \frac{1}{2n} \geq 1 - (1 - \frac{2log\ n}{n})^n$. By the Piling-up lemma, $\Pr[\langle a, b\rangle = 1] = \frac{1}{2}(1 - (1 - 2\mu^*)^n)$, and therefore $\Pr[\langle a, b\rangle = 1] \leq \frac{1}{2}(1 - \frac{1}{2n})$, and the statement follows. □

Lemma 2.28. *For* $\mu^* = log(n)/n, a \sim \beta^n_\mu, b \sim \tilde{\beta}^n_{\mu^*}$, *and sufficiently large* n, $\Pr[\langle a, b\rangle = 0] \geq \frac{1}{2} + \frac{1}{6n}$.

Proof. Let E be the event that $log(n)/2 \leq |a| \leq 3log(n)/2$. By Chernoff bound, $\Pr[\neg\mathsf{E}] \leq 2exp(-\frac{log(n)}{10})$. By Lemma 2.27, we have that $\frac{1}{2} + \frac{1}{4n} \leq \Pr[\langle a, b\rangle = 0]$, and therefore

$$\begin{aligned}\frac{1}{2} + \frac{1}{4n} &\leq \Pr[\langle a, b\rangle = 0 \mid \neg\mathsf{E}] \Pr[\neg\mathsf{E}] + \Pr[\langle a, b\rangle = 0 \mid \mathsf{E}] \Pr[\mathsf{E}] \\ &\leq \Pr[\neg\mathsf{E}] + \Pr[\langle a, b\rangle = 0] \\ \frac{1}{2} + \frac{1}{4n} - \Pr[\neg\mathsf{E}] &\leq \Pr[\langle a, b\rangle = 0]\end{aligned}$$

and the statement follows. □

Lemma 2.29. ***LPN on squared-log entropy*** *(Simplified from [YZ16]). Let* n *be a security parameter and let* $\mu \leq 1/2$ *be a constant. Assume that the* $\mathsf{SLPN}_{\mu,n}$ *problem is* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})$*-hard, then for every* $\lambda = \Theta(log^2\ n)$, $q = poly(n)$, *and every polynomial time sampleable* $x \in \{0,1\}^n$ *with* $\boldsymbol{H}_\infty(x) \geq 2\lambda$,

$$(\boldsymbol{A}, \boldsymbol{A}x + e) \overset{c}{\approx} (\boldsymbol{A}, u)$$

where $\boldsymbol{A} = U_{q\times\lambda}U_{\lambda\times n}$ *and* $U_{m\times n}$ *is a uniformly random* $m \times n$ *binary matrix, and* $e \sim \beta^q_\mu$, $u \sim \{0,1\}^q$. *We will call the task of distinguishing the above distributions the decisional squared-log entropy LPN problem.*

Proposition 2.30. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CLF *satisfies the completeness guarantee of Covert Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $b \leq O(log\ n), \tau \geq 1/poly(n)$, $\delta \geq exp(-n)$.

Proof Idea. We first analyze the "unmasking" procedure ϕ defined in line 24. Essentially, the unmasking ϕ_i, which is applied to the response for the i^{th} masked query, reintroduces a dependency on the secret used to construct the masking for the i^{th} query. In this way, we may cancel some noisy terms in an expanded analysis of the oracle responses. We then leverage the pseudorandomness of the masked queries to show that, roughly, the responses to the unmasked queries can be written as noisy inner products with any $O(log\ n)$-degree parity function which the target function of the concept has a "heavy" Fourier coefficient attached. Using this fact, we prove that running a "local decoding" of each bit of any $O(log\ n)$-degree parity function where this is true suffices to recover the parity functions we are interested in. We prove that this is the case by using techniques inspired by the original analysis of the Goldreich-Levin algorithm [GL89].

Before beginning the proof of Proposition 2.30 prove a lemma that uncovers the relationship between a masked query and the underlying query. Recall that the "unmasking" operation (line 24) is defined by $\phi_i(x) = \tilde{f}(x)\chi_{s_i}(r)$, where s_i is the i^{th} secret and $r \sim \beta^n_{1/2}$. Furthermore, we refer the reader to Algorithm 4 to see the constructions of the Goldreich-Levin queries and their masks.

Lemma 2.31. *Let $k \in \{0,1\}^n$ with $|k| = O(log\ n)$. Also, for $i \in [poly(n)]$, let $x_i \in \{0,1\}^n$ be a query and $\hat{x}_i$ be the masked version, as in line 22 of* CLF. *If $|\hat{f}(k)| \geq \tau$, then $|\mathbb{E}[\phi_i(\hat{x}_i)\chi_k(x_i)]| \geq \frac{1}{6}\tau n^{-2}$ where the expectation is over the randomness of ϕ_i and the masking of x_i, unless the $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ assumption does not hold.*

Proof. By Lemma 2.29 and using closure under polynomial composition,

$$(\hat{x}_i)_{i\in[m]} \overset{c}{\approx} (u_i)_{i\in[m]}$$

is immediate (where $m = poly(n)$ and $u_i \xleftarrow{\$} \{0,1\}^n$).

Since f is computable in polynomial time and $\hat{x}_i \overset{c}{\approx} u_i$, we have that

$$\left| \mathop{\mathbb{E}}_{u_i,e,s,r}\left[\chi_k(x_i)\chi_k(e_i)\mathcal{E}_k(u_i)\chi_{s_i}(r)\right] - \mathop{\mathbb{E}}_{\phi_i}\left[\phi_i(\hat{x}_i)\right]\right| \leq negl(n)$$

and also that

$$\left| \mathop{\mathbb{E}}_{u_i,e,s,r}\left[\chi_k(e_i)\mathcal{E}_k(u_i)\chi_{s_i}(r)\right] - \mathop{\mathbb{E}}_{\phi_i}\left[\phi_i(\hat{x}_i)\chi_k(x_i)\right]\right| \leq negl(n)$$

for all $i \in [m]$. Using the Piling-up lemma (Lemma C.5), $\mathbb{E}_{e_i}[\chi_k(e_i)] \geq 1/n^{log(4/3)}$. Since $|\hat{f}(k)| \geq \tau$, $|\mathbb{E}_{u_i}[\mathcal{E}_k(u_i)]| \geq \tau$. By Lemma 2.28, $\mathbb{E}[\chi_{s_i}(r)] \geq 1/6n$. As a result,

$$\left|\mathbb{E}\left[\mathcal{E}_k(u_i)\chi_k(e_i)\chi_{s_i}(r)\right]\right| \geq \frac{1}{6}\tau n^{-(log(4/3)+1)}$$

since each term is independent. And therefore

$$\left|\mathop{\mathbb{E}}_{\phi_i}\left[\phi_i(\hat{x}_i)\chi_k(x_i)\right]\right| \geq \frac{1}{6}\tau n^{-2}$$

for all $i \in [m]$, as desired.

□

Proof of Proposition 2.30. For completeness, we assume that the AI is honest, in that it never tampers with any of the oracle responses to the learner's queries. Now, let us consider one iteration of CLF, assuming that line 20 of CLF does not reject (we remove this assumption later). First, let us examine the "unmasking" function ϕ_i that is applied to each of the masked queries in the non-negated case (see line 37), without loss of generality—the negated case is analogous. In the following, let $\mathcal{E}_k(x) = f(x)\chi_k(x)$. We observe that,

$$\begin{aligned}
\phi_i(\hat{\mathrm{gl}}^i) &= f(\hat{\mathrm{gl}}^i)\chi_{s_i}(r) && \text{(where } r \text{ is a uniformly random } n\text{-bit vector)}\\
&= f(\mathbf{R}_i s_i \oplus e_i \oplus \mathrm{gl}^i)\chi_{s_i}(r)\\
&= \chi_k(\mathbf{R}_i s_i)\chi_k(e_i)\chi_k(\mathrm{gl}^i)\mathcal{E}_k(\hat{\mathrm{gl}}^i)\chi_{s_i}(r)\\
&= \chi_k(\mathrm{gl}^i)\chi_k(e_i)\mathcal{E}_k(\hat{\mathrm{gl}}^i)\chi_{s_i}(\hat{r}) && \text{(where } \hat{r} \text{ is a uniformly random } n\text{-bit vector)}
\end{aligned}$$

The last step is justified by using linearity to rearrange the first term to $\chi_{s_i}(\mathbf{R}_i^T k)$, and then "cancelling" by using linearity to combine it with the last term.

Now, we turn our attention to line 29 of CLF. Let $E_i = \frac{1}{2}(1 + \phi_i(\hat{\mathrm{gl}}^i_{S,-,z})\chi_k(\mathrm{gl}^i_{S,-,z}))$. Observe that, for any k such that $\hat{f}(k) \geq \tau$ and $k = O(log\ n)$, Lemma 2.31 implies that, $\mathbb{E}[E_i] \geq \frac{1}{2} + \frac{1}{12}\tau n^{-2}$. Thus, when $b^S = \chi_k(x^S)$ (x^S is as defined in line 7 of CLF) the following is true,

$$\begin{aligned}
\Pr\left[\underset{S\subseteq[t]\setminus\emptyset}{\mathsf{majority}}(\phi_i(\hat{\mathrm{gl}}^i_{S,-,z}) \cdot b^S) \neq \chi_k(z)\right] &= \Pr\left[\sum_i E_i < 2^t/2\right]\\
&\leq \Pr\left[\left|\sum_i E_i - \mathbb{E}\left[\sum_i E_i\right]\right| < \frac{1}{12}\tau n^{-2}2^t\right]\\
&\leq \left(\frac{1}{12}\tau n^{-2}\cdot 2^t\right)^{-2} \cdot \sum_i Var\left[E_i\right]\\
&\leq \left(\frac{1}{12}\tau n^{-2}\cdot 2^t\right)^{-2} \cdot \frac{2^t}{4}\\
&\leq \frac{1}{576\tau^2 n^{-4} 2^t}
\end{aligned}$$

where the probability is taken over ϕ_i, z and the randomness of $\hat{\mathrm{gl}}^i_{S,-,z} \forall S \subseteq [t]\setminus\emptyset$. The second inequality follows by the pairwise independence of the masked Goldreich-Levin queries and Chebyshev's inequality. The third implication follows from the fact that the variance of a 0/1 variable E_i is at most 1/4, and all E_i are pairwise independent. Likewise, the above probability bound is true for every $j \in [n]$ when replacing $\hat{\mathrm{gl}}^i_{S,-,z}$ with $\hat{\mathrm{gl}}^i_{S,j,z}$, $\chi_k(z)$ with $\chi_k(z \oplus u_j)$, and appropriately modifying the definition of E_i (recall, u_j is the j^{th} unit vector).

Since in CLF we have taken $t = -2log(\tau n^{-2})$, we may conclude by Markov's inequality that

$$\Pr\left[\Pr_z\left[\underset{S\subseteq[t]\setminus\emptyset}{\mathsf{majority}}(\phi_i(\hat{\mathrm{gl}}^i_{S,-,z}) \cdot b^S) \neq \chi_k(z)\right] \leq \frac{1}{8}\right] \geq \frac{3}{4}$$

and likewise

$$\Pr\left[\Pr_z\left[\underset{S\subseteq[t]\setminus\emptyset}{\mathsf{majority}}(\phi_i(\hat{\mathrm{gl}}^i_{S,j,z})\cdot b^S)\neq\chi_k(z\oplus u_j)\right]\leq\frac{1}{8}\right]\geq\frac{3}{4}$$

where the outer probabilities are taken over ϕ_i and the randomness of $\hat{\mathrm{gl}}^i_{S,-,z}, \hat{\mathrm{gl}}^i_{S,j,z}\forall S\subseteq[t]\setminus\emptyset$.

We will now show that the "local decoding" of lines 30-34 of CLF correctly recovers (bit-by-bit) any single $O(log\ n)$-degree Fourier coefficient of size at least τ with constant probability. Let C_1 be the event that

$$\mathsf{majority}_{S\subseteq[t]\setminus\emptyset}(\phi_i(\hat{\mathrm{gl}}^i_{S,j,z})\cdot b^S)=\chi_k(z\oplus u_j)$$

and let C_2 be the event that

$$\mathsf{majority}_{S\subseteq[t]\setminus\emptyset}(\phi_i(\hat{\mathrm{gl}}^i_{S,-,z})\cdot b^S)=\chi_k(z)$$

We thus have that given $b^S=\chi_k(x^S)$,

$$\begin{aligned}\Pr\left[\Pr_z\left[a_{jz}=k_j\right]\geq\frac{3}{4}\right]&\geq\Pr\left[\Pr_z\left[C_1\wedge C_2\right]\geq\frac{3}{4}\right]\\&\geq\frac{1}{2}\end{aligned}$$

where the outer probability is taken over ϕ_i and the randomness of $\hat{\mathrm{gl}}^i_{S,-,z}, \hat{\mathrm{gl}}^i_{S,j,z}\forall S\subseteq[t]\setminus\emptyset$.

In CLF, we iterate over all possible assignments of $(b_1,\cdots b_t)$. Therefore,

$$\begin{aligned}\Pr\left[a=k\right]&\geq 1-\sum_j\Pr\left[a_j\neq k_j\right]\\&\geq 1-\sum_j\Pr\left[\mathsf{majority}_z(a_{jz})\neq k_j\right]\\&\geq 1-\sum_j exp\big(-O(log\ n)\big)\\&\geq 1-\sum_j O(1/n)\\&\geq O(1)\end{aligned}$$

The first inequality follows from the union bound, while the second is justified since $a_j=\mathsf{majority}_z(a_{jz})$. The number of a_{jz} for a fixed j is $O(log\ n)$, and therefore a standard application of a Hoeffding bound gives the third inequality. Thus, with probability at least some constant, we have recovered any single parity function with Fourier coefficient at least τ (in one inner iteration). We may remove the initial condition on the event that line 19 does not reject; the event of rejection happens with probability negligible in n, so the probability remains larger than a constant.

Hence, the set of $O(log(\delta/\tau))$ independent iterations of CLF obtains every Fourier coefficient in $\hat{f}^{\geq\tau}_{O(log\ n)}$, with constant probability (for an appropriate constant, and because Parseval's theorem entails that there are at most $1/\tau^2$ Fourier coefficients larger than τ, then applying the union bound). We note that, $|\mathbb{L}|$ must be bounded by a polynomial in n. To see that this is the case, we can observe that at most $O(log(\delta/\tau))\cdot 2^t$ elements are ever added to $\mathbb{L}$. We have stipulated that $\delta\geq exp(-n)$ and $\tau\geq 1/poly(n)$,

so this quantity is polynomial in n as well. Therefore, $\mathcal{L}^{\tau,b}(\mathbb{L}) = \mathcal{L}^{\tau,b}(\mathsf{FOURIER}_{T,b,n})$ with probability $1-\delta$.

□

Proposition 2.32. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CLF *satisfies the privacy guarantee of Covert Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *where* $b \leq O(log\ n), \tau \geq 1/poly(n)$, $\delta \geq exp(-n)$.

Proof. We will construct a simulator Sim, which, for the algorithm CLF, it holds that

$$\left\{\mathsf{real}_{\mathsf{CLF}}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}\right\} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

We point out that, since the chosen queries are not inherently adaptive, it is possible to run all the iterations of CLF in parallel. We may compress all the messages into one large message that fits into a $poly(n) \times n$ matrix, and decode the results from there. We prove security in this setting. Consider the following p.p.t. simulation algorithm,

Algorithm: $Sim(\mathcal{S}, N)$

Inputs:

N *is the total number of queries requested by the real learner in* CLF*, and* $\mathcal{S}$ *is a set of* N *uniformly random examples from the concept.*

output $\mathcal{S}$

Algorithm 5: Simulator for Covert Learning of low-degree heavy Fourier coefficients

We will show that an adversary $\mathbb{A}$ who can distinguish the above two distributions can solve the decisional squared-log entropy LPN problem (from there, the proof concludes by Lemma 2.29).

Recall that

$$\left\{\mathsf{real}_{\mathsf{CLF}}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathsf{transcript}_{\mathsf{CLF}^{\mathcal{O}_{D_n}}(\mathsf{FOURIER}_{T,b,n},\epsilon,\delta)}\right)\right\}$$

where $\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathcal{D}_f$ are chosen ahead of time by the adversary. In a similar fashion,

$$\left\{\mathsf{ideal}_{Sim}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, Sim(\mathcal{S}, \ell)\right)\right\}$$

Thus, suppose there is an adversary $\mathbb{A}$ who can distinguish between the distributions

$$\left\{\mathsf{real}_{\mathsf{CLF}}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}\right\} \overset{c}{\approx} \left\{\mathsf{ideal}_{Sim}\right\}$$

with some non negligible distinguishing advantage. We claim that there exists an algorithm $\mathbb{B}$ that solves the decisional squared-log entropy LPN problem problem as follows.

1. $\mathbb{B}$ obtains challenge samples $(C_i, c_i) \in \mathbb{Z}_2^{n\times n} \times \mathbb{Z}_2^n\ \forall i \in [N]$ from either the LPN distribution or uniformly random. Note that, under the stated parameter regime, $N = poly(n)$.
2. $\mathbb{B}$ computes the set of queries $Q = \{c_i \oplus \mathrm{gl}^i \mid \forall i \in [N]\}$.

3. $\mathbb{B}$ obtains $\mathcal{O}_f(Q)$. $\mathbb{B}$ runs $\mathbb{A}$ on

$$\Big(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, (Q, \mathcal{O}_f(Q))\Big)$$

where $\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, D_f$ are those that would be chosen by $\mathbb{A}$ in the distinguishing experiment.

4. $\mathbb{B}$ outputs "uniform" if $\mathbb{A}$ outputs "ideal" and "LPN" otherwise.

We analyze the behavior of the above algorithm. First, let E_j denote the event that CLF did not output reject prior to making any membership queries in the j^{th} iteration of CLF (in line 20). Second, let L denote the event that the ℓ samples (C_i, c_i) came from the LPN oracle and $\neg L$ be the event that the ℓ samples (C_i, c_i) are uniformly random. Finally, we denote by X the distribution that $\mathbb{A}$ is executed on above.

We observe that,

$$\left\{\mathsf{real}_{\mathsf{CLF}}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}} \;\middle|\; \bigwedge_j E_j\right\} = \Big\{\mathsf{X} \mid L\Big\}$$

and

$$\Big\{\mathsf{ideal}_{Sim}^{\mathcal{M}}\Big\} = \Big\{\mathsf{X} \mid \neg L\Big\}$$

At iteration j of CLF, a standard application of a Chernoff bound shows that $\Pr[E_j] \leq negl(n)$. Since we have that $\delta \geq exp(-n)$ and $\tau \geq 1/poly(n)$, the total number of iterations is bounded by a polynomial of n. Invoking the union bound, it follows that

$$\Pr\left[\bigwedge_j E_j\right] \geq 1 - negl(n)$$

and this gives that

$$\left\{\mathsf{real}_{\mathsf{CLF}}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}\right\} \overset{c}{\approx} \Big\{\mathsf{X} \mid L\Big\}$$

Therefore, the non-negligible distinguishing advantage of $\mathbb{A}$ is inhertied by $\mathbb{B}$.

□

2.5 Covert Learning of Polynomial Size Decision Trees

In this section, we supply a natural application of Covert Learning for low-degree Fourier coefficients. Specifically, we will show that the collection of hypothesis classes given by taking all subsets of polynomial size decision trees is covertly learnable. Recall that we are focused on collections that contain hypothesis classes which are not (or not known to be) efficiently agnostically PAC learnable from uniformly random examples. The problem of learning decision trees under the uniform distribution has long been considered, and yet no polynomial time (in the size of the smallest decision tree) algorithms exist for arbitrary functions, and some distributions over functions [BFKL93, IKOS19] (even in the realizable case). In fact, any such algorithm would be considered a massive breakthrough in computational learning theory [Blu03, OS07][3].

[3]Not much formal work has been done on identifying "hard distributions" over DNF formulas (or other function classes) [BFKL93, IKOS19], as it is not relevant in the usual learning models. However, even some relatively simple distributions appear to defy all known techniques. For example, consider the distribution over polynomial size DNFs (also, decision trees), constructed as follows. Select at random two disjoint subsets of $[n]$ of size $log\ n$ each. Let the first subset be denoted S and the second T. The distribution over DNFs induced by defining $f(x) = \chi_S(x) \oplus \mathsf{majority}_T(x)$ seems hard to even weakly predict over the uniform distribution [BFKL93]. Indeed, such a distribution over DNF formulas could be used to instantiate our Covert Learning algorithms of this section.

Definition 2.33. *Let* $\mathsf{DT}_{n,s}$ *be the hypothesis class of all* $f : \{0,1\}^n \to \{-1,1\}$ *computable by a size* s *decision tree. Let* $\mathcal{C}_{\mathsf{DT}_{n,s}} = \{\mathcal{H}_n | \mathcal{H}_n \subseteq \mathsf{DT}_{n,s}\}$.

This collection of hypothesis classes is motivated for the following simple reason. If an adversary has no information about which subset of decision trees has been learned, then the adversary has no information about the learned decision tree. This claim is easily seen to be true from the contrapositive.

The covert learning algorithm for $\mathcal{C}_{\mathsf{DT}_{n,s}}$ is as follows. We apply standard techniques from computational learning theory. Note that, the returned hypothesis is not a decision tree, but rather the sign of a multilinear polynomial.

Algorithm: $\mathsf{CLDT}(\mathcal{H}_n, \epsilon, \delta)$
$\gamma = \sqrt{\epsilon/2}$
$\tau = \gamma\epsilon/32s$
$\mathbf{L} = \mathsf{CLF}(\mathsf{FOURIER}_{[n],log(32s/\epsilon)}, 0, \delta/2, \tau, \mathcal{S})$
if $\boldsymbol{L} =$ reject **then**
  output reject
end
$\delta' = \frac{\delta}{2|\mathbf{L}|}$
for *each* $\chi_k \in \boldsymbol{L}$ **do**
  Using random examples from $\mathcal{S}$, estimate $\tilde{f}(k) = \mathbb{E}_x[f(x)\chi_k(x)]$ within an additive interval of $\sqrt{\epsilon/|\mathbf{L}|}$ with probability $1 - \delta'$.
end
$g(x) = \sum_{\chi_k \in \mathbf{L}} \tilde{f}(k)\chi_k(x)$
output $h = \mathrm{sign}(g)$

Algorithm 6: Covert Verifiable Learning of polynomial size decision trees

We state the theorem and then dedicate the rest of the section to assembling the proof.

Theorem 2.34. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption, the collection* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s = poly(n)$ *is* $(poly(n), 4)$*- covertly learnable, with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}_D$, *and where* $\epsilon \geq 1/poly(n)$, *and* $\delta \geq exp(-n)$.

Proposition 2.35. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CLDT *satisfies the completeness guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s = poly(n)$, *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}_D$, *and where* $\epsilon \geq 1/poly(n)$, *and* $\delta \geq exp(-n)$.

Proof. We will show that CLDT satisfies completeness when the input hypothesis class is fixed to be $\mathsf{DT}_{n,s}$. This suffices to prove completeness for all other $\mathcal{H}_n \in \mathcal{C}_{\mathsf{DT}_{n,s}}$, as every $\mathcal{H}_n \in \mathcal{C}_{\mathsf{DT}_{n,s}}$ is a subset of $\mathsf{DT}_{n,s}$. First, we will introduce the following three facts which are well known. In this work, we omit the proofs and refer the interested reader to lemmas 14 and 15 of [BF02], as well as [KM93] for more information.

Fact 2.36. *Let f be any boolean function and let g be any real valued function. Then*

$$\Pr_x[f(x) \neq \text{sign}(g)(x)] \leq \mathop{\mathbb{E}}_x[(f(x) - g(x))^2] = \sum_{S \subseteq [n]} (\hat{f}(S) - \hat{g}(S))^2$$

Fact 2.37. *Let T be a decision tree of d-depth. All the Fourier coefficients of T of degree larger than d are 0 and all the non-zero Fourier coefficients of T are larger than 2^{-d}.*

Fact 2.38. *Let T be a decision tree of size s. There exists a function h such that all the Fourier coefficients of h with degree larger than $log(4s/\epsilon)$ are 0, and all non-zero Fourier coefficients of h are larger than $\epsilon/4s$. Lastly, $\mathbb{E}_x[(T(x) - h(x))^2] \leq \epsilon$. In particular, if T is boolean then h is boolean.*

Fix any concept $D_f \in \mathcal{D}_{\mathcal{F}_n}$. We will separate our proof into two parts:

1. There exists a boolean function g such that $\mathcal{L}_{D_f}(g)$ is close to the loss of the optimal decision tree T for f, and all the zonzero Fourier coefficients of g are contained in $\mathbf{L}$.
2. Knowing 1, the output h of CVLDT has a loss which is at most $O(\epsilon)$ greater than $4\mathcal{L}_{D_f}(T)$.

For the first statement, let $\mathcal{L}_{D_f}(\mathsf{DT}_{n,s}) \leq \ell$. Then, there exists a $T \in \mathsf{DT}_{n,s}$ such that $\mathcal{L}_D(T) \leq \ell$. Now, by Fact 2.38, there exists a boolean function h' such that $\mathbb{E}_x[(T(x) - h'(x))^2] \leq \epsilon/8$, and where $\hat{h'}(S) = 0$ for $|S| > log(32s/\epsilon)$, and all nonzero $\hat{h'}(S) > \epsilon/32s$. Now let $V = \hat{h'}^{>0} \cap \hat{f}_b^{>\gamma\epsilon/32s}$ for $\gamma = \sqrt{\epsilon/2}$ and $b = log(32s/\epsilon)$, and consider the function $g(x) = \sum_{S \in V} \hat{f}(S)\chi_S(x)$. We have,

$$\begin{aligned}
\mathcal{L}_{D_f}(g) &= \frac{1}{4} \mathop{\mathbb{E}}_x \left[(f(x) - g(x))^2 \right] \\
&= \frac{1}{4} \sum_{S \subseteq [n]} \left(\hat{f}(S) - \hat{g}(S) \right)^2 \\
&= \frac{1}{4} \sum_{S \notin V} \left(\hat{f}(S) - \hat{g}(S) \right)^2 \\
&= \frac{1}{4} \left(\sum_{S \notin \hat{h'}^{>0}} \hat{f}(S)^2 + \sum_{S \in \hat{h'}^{>0} \setminus \hat{f}_b^{>\gamma\epsilon/32s}} \hat{f}(S)^2 \right)
\end{aligned}$$

We can bound each term individually. The second term is less than or equal to $|\hat{h'}^{>0}| \cdot (\gamma\epsilon/32s)^2 = \gamma^2 = \epsilon/2$ by Parseval's theorem. For the first term, consider

$$\begin{aligned}
\sum_{S \notin \hat{h'}^{>0}} \hat{f}(S)^2 &= \sum_{S \notin \hat{h'}^{>0}} \left(\hat{f}(S) - \hat{h'}(S) \right)^2 \\
&\leq \sum_{S \subseteq [n]} \left(\hat{f}(S) - \hat{h'}(S) \right)^2
\end{aligned}$$

and we may write this as an expectation to obtain

$$\begin{aligned}\sum_{S\subseteq[n]}\Big(\hat{f}(S)-\hat{h'}(S)\Big)^2 &\leq \mathop{\mathbb{E}}_{x}\Big[\Big(f(x)-h'(x)\Big)^2\Big]\\ &\leq 4\Pr_{x}\Big[f(x)\neq h'(x)\Big]\\ &\leq 4\Big((1-\epsilon/32)\ell+\epsilon/32(1-\ell)\Big)\end{aligned}$$

and thus

$$\mathcal{L}_{D_f}(g)\leq \ell+\epsilon$$

For the second statement, observe that by Fact 2.36,

$$\mathcal{L}_{D_f}(h)\leq \mathop{\mathbb{E}}_{x}[(f(x)-g(x))^2]=\sum_{S\in\mathbf{L}}(\hat{f}(S)-\hat{g}(s))^2+\sum_{S\notin\mathbf{L}}f(\hat{S})^2$$

where the first term is less than $|\mathbf{L}|\cdot(\sqrt{\epsilon/|\mathbf{L}|})^2=\epsilon$. For the second term, note that we know that there exists a boolean function g such that

$$\mathcal{L}_{D_f}(g)\leq \mathcal{L}_{D_f}(\mathsf{DT}_{n,s})+\epsilon$$

and all the nonzero Fourier coefficients of g are in $\mathbf{L}$. Thus consider,

$$\begin{aligned}\sum_{S\notin\mathbf{L}}\hat{f}(S)^2 &\leq \sum_{S\notin\mathbf{L}}\Big(\hat{f}(x)-\hat{g}(x)\Big)^2\\ &\leq \sum_{S\subseteq[n]}\Big(\hat{f}(S)-\hat{g}(S)\Big)^2\\ &\leq \mathop{\mathbb{E}}_{x}\Big[\big(f(x)-g(x)\big)^2\Big]\\ &\leq 4\big(\mathcal{L}_{D_f}(\mathsf{DT}_{n,s})+\epsilon\big)\end{aligned}$$

and therefore $\mathcal{L}_{D_f}(h)\leq 4\mathcal{L}_{D_f}(\mathsf{DT}_{n,s})+5\epsilon$.

Of course, the preceding arguments are only true when both the CLF algorithm outputs correctly and all the estimates from CLDT are indeed within the correct interval. Each estimate is inside the interval with probability $1-\delta/2|\mathbf{L}|$ and thus they will all be correct probability $1-\delta/2$ by applying the union bound. Observing that the failure probability of CLF is also $\delta/2$, we conclude that the probability that all the estimates are correct and the CLF algorithm returns correctly is at least $1-\delta$ as desired (again applying a union bound). Thus CLDT returns an hypothesis as desired. We note that, in order to maintain efficiency, $|\mathbf{L}|$ must be bounded by a polynomial in n. Recall that at most $O(log(\delta/\tau))\cdot\tau^{-2}n^4$ elements are ever added to $\mathbf{L}$. This quantity is polynomial in n under the stipulated parameter regime. □

Next, we consider the privacy of CLDT.

Proposition 2.39. *Under the* $(2^{\omega(n^{\frac{1}{2}})},2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CLDT *satisfies the privacy guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s=poly(n)$*, with respect to* $\mathcal{D}_{\mathcal{F}_n}$*,* $\mathcal{O}_{\mathcal{F}_n}$*, and* $\mathcal{L}_D$*, and where* $\epsilon\geq 1/poly(n)$*, and* $\delta\geq exp(-n)$*.*

Proof. Because the interactive aspect of CLDT is fully contained inside CLF, the proof of privacy of CLDT is identical to Proposition 2.32.

□

3 Covert Verifiable Learning

In this section we define and construct the notion of Covert *Verifiable* Learning. The Covert Verifiable Learning setting can be viewed as an interactive protocol between a learner and an *adversarial intermediary* (AI). Here, the adversarial intermediary acts monitors access to the membership oracle. Figure 2 depicts this perspective. In this context, the learner must request queries from the oracle, but the responses are intercepted by the AI who then either truthfully reports the oracle's responses, or lies.

3.1 Definition of Covert Verifiable Learning

For Covert Verifiable Learning, we augment the desired properties of Covert Learning by allowing the learner to abort, and requiring: If the AI corrupts any queries or results, the learner will not output an incorrect hypothesis except with small probability. In addition, we will extend the privacy requirements of Covert Learning to capture the active nature of the adversarial intermediary. Let us informally describe the Covert Verifiable Learning setting in more detail.

The learner's inputs: Similarly to the Covert Learning setting, the learner will receive as input a specific target hypothesis class $\mathcal{H}_n$ (within a fixed collection $\mathcal{C}_n$), in addition to accuracy parameters ϵ, δ. The learner will *also* receive a set of auxiliary random examples from a concept D_n within a concept class $\mathcal{D}_n$ which are private—the AI has no information on the identity of these random examples.

The interaction: The learner will interact with an oracle $\mathcal{O}_{D_n}$ that implements query access to the concept. However, the responses have the potential to be corrupted by an AI who lives between the learner and the oracle. The learner tries to learn $\mathcal{H}_n$ with respect to the concept D_n.

The security experiment: We define a real and ideal experiment.

Definition 3.1. *Let $\mathcal{D}_n$ be a concept class, and let $\mathcal{C}_n$ be a collection of hypothesis classes. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, and a set of queries and the oracle's responses on those queries. We define $\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}$ is drawn from D_n. $\mathcal{S}$ is given to the learner, along with $\mathcal{H}_n, \epsilon, \delta$, while the adversarial intermediary $\mathcal{I}$ is given ϵ, δ.*
3. *The learner begins to interact with the concept oracle $\mathcal{O}_{D_n}$ by requesting membership queries in order to agnostically learn $\mathcal{H}_n$. $\mathcal{I}$ sees the learner's queries and responses and is given the chance to modify the responses. At the end of the interaction, $\mathcal{I}$ outputs a string denoted by $\mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}$.*
4. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\right)$*

Definition 3.2. *Let Sim be a p.p.t. algorithm, which takes as input two sets of random examples from the concept and a length parameter ℓ which signifies the number of queries requested by the learner in the real interaction. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, and a set of queries and oracle's responses. We define $\{\mathsf{Videal}_{Sim,\mathcal{I}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}'$ is drawn from D_n.*
3. *The simulator is given $\epsilon, \delta, \mathcal{S}, \mathcal{S}'$ (where $\mathcal{S}$ is the set of examples given to the learner in the real interaction), while an adversarial intermediary $\mathcal{I}$ is given ϵ, δ.*
4. *Sim begins to "interact" with the $\mathcal{O}_{D_n}$ by "requesting" membership queries. $\mathcal{I}$ "views" the queries and responses, and is given the chance to change the responses. The simulator outputs a string, which is denoted by $\mathsf{ideal}_{\mathcal{I}}^{Sim}$.*
5. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{ideal}_{\mathcal{I}}^{Sim}\right)$*

Now the formal definition of Covert Verifiable Learning:

Definition 3.3. ***Covert Verifiable Learning.*** *Let $\mathcal{C}_n$ be a collection of hypothesis classes, let $\mathcal{D}_n$ be a class of concepts, let $\mathcal{O}_{\mathcal{D}_n}$ be a class of oracles indexed by $D_n \in \mathcal{D}_n$, and let $\mathcal{L}$ be a loss function. An algorithm $\mathcal{A}$ is an $(m(n), \alpha)$-covert verifiable learning algorithm for $\mathcal{C}_n$, with respect to $\mathcal{D}_n$, $\mathcal{O}_{\mathcal{D}_n}$ and $\mathcal{L}$, if for every $\epsilon, \delta > 0$, the following are true.*

- *Completeness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, the randomized output of $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*
$$\Pr_h\left[\mathcal{L}(h) \leq \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon\right] \geq 1 - \delta$$
- *Soundness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, then for any adversarial intermediary $\mathcal{I}$ that corrupts queries or responses from $\mathcal{A}$ to $\mathcal{O}_{D_n}$, the random variable $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*
$$\Pr_h\left[\mathcal{L}(h) > \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon \;\middle|\; h \neq \text{reject}\right] < \delta$$
We say that soundness is computational if $\mathcal{I}$ is p.p.t..
- *Privacy. For any adversarial intermediary $\mathcal{I}$, there exists a p.p.t. simulation algorithm Sim that satisfies:*
$$\left\{\mathsf{Vreal}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\right\} \overset{c}{\approx} \left\{\mathsf{Videal}_{Sim,\mathcal{I}}\right\}$$

We stipulate that each of the sets of random examples given to the simulator are of size $m(n)$.

In keeping with the terminology from the computational learning theory literature, we will often say that a collection of hypothesis classes $\mathcal{C}$ is verifiably $(m(n), \alpha)$-*verifiably covertly learnable* if there exists a $(m(n), \alpha)$-covert verifiable learning algorithm for $\mathcal{C}$.

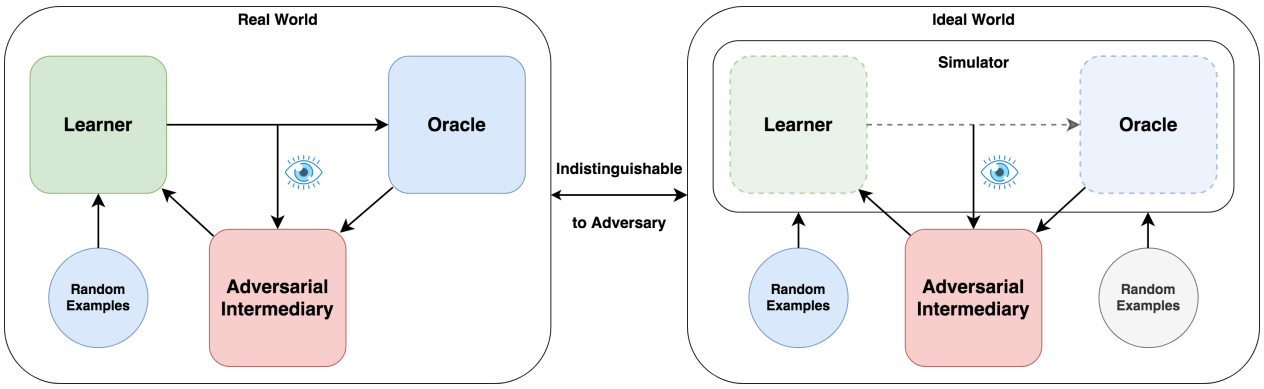

Figure 4: Privacy of Covert Verifiable Learning. The "real world," where the AI interacts with the learner and oracle, should be indistinguishable from the "ideal world," where the AI interacts with a simulator that plays both roles of learner and oracle. Importantly, the simulator works without knowledge of the underlying hypothesis classes or the actual oracle, though it does have access to random examples from the concept.

3.1.1 Discussion

Variants. We would like to highlight some salient variants of the model that we have presented above. The variants are on the nature of the random examples that are present in the interaction. For example, we could also consider the case that the learner's random examples are publicly known. We call this setting the *public* Covert Verifiable learning variant. In this public variant, achieving soundness and privacy is much more difficult, as the learner has no private examples to leverage against the AI. However, this variant greatly increases the practicality of the model because it may be infeasible for the learner to acquire private examples. In Section 3.4, we focus on this case. Another variant of the formally stated model involves weakening the privacy requirement to require indistinguishability of only the membership queries, and not for the joint distribution of the private random examples and membership queries. This model (called the *fully private examples* variant), may be justified, as we already consider private examples in order to achieve soundness. In Section 3.5, we show that this model is quite powerful, even if we require *perfect* privacy and *statistical* soundness. We opt to focus (in Section 2.4 and Section 2.5) on the case where privacy is with respect to the joint distribution since it seems to be the "right" level of difficulty. Additionally, this model provides strong "zero-knowledge-style" guarantees in a forward focused manner. That is, even if private examples used for verification (a one time event) become known in the future, then the privacy guarantees remain intact.

3.2 Making CLF Verifiable

In this section, we show how to add the soundness guarantee of Covert Verifiable Learning to CLF. More specifically, we want to provide the guarantee that if for any concept $D_f \in \mathcal{D}_{\mathcal{F}_n}$, any hypothesis class $\mathsf{FOURIER}_{T,b,n} \in \mathcal{C}_{\mathsf{FOURIER},b,n}$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_f, then for any adversarial intermediary $\mathcal{I}$ that corrupts oracle responses from the interaction between CLF and $\mathcal{O}_f$, the random variable $h = \mathsf{CLF}^{\mathcal{O}_f}(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathcal{S})$ satisfies

$$\Pr_h \left[\mathcal{L}^{\tau,b}(h) > \alpha \cdot \mathcal{L}^{\tau,b}(\mathsf{FOURIER}_{T,b,n}) + \epsilon \;\middle|\; h \neq \text{reject} \right] < \delta$$

Our basic idea to achieve verifiability is to wrap the CLF algorithm with an outer loop, which attempts to catch the adversarial intermediary cheating by randomly deciding to either execute CLF

(the "learning" case) or send queries which are part of the learner's private example set $\mathcal{S}$ (the "test" case). The crucial point is: the queries of the learning case can be shown to be computationally indistinguishable from the test queries (which are simply uniformly random). This system gives an easy proof idea for soundness: The (p.p.t.) adversarial intermediary must lie a similar amount on the learning case and the test case, else it would contradict the pseudorandomness of the queries made by CLF. Therefore, since the AI can always be detected if it lies in the test case, it cannot reliably lie on the learning case, without being detected. The following is the described outer loop for CLF.

Algorithm: CVLF($\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \tau, \mathcal{S}$)
Initialize:
$\mathbb{L} = \emptyset$, $\theta = \tau n^{-2}$
for $i \in [O(log(\frac{1}{\delta}))]$ **do**
  $v \xleftarrow{\$} \{0,1\}$
  In this case, the algorithm sends meaningful queries via the subroutine CLF.
  if $v = 0$ **then**
    $\mathbf{L} = \mathsf{CLF}(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta/2, \theta)$
    $\mathbb{L} = \mathbb{L} \cup \mathbf{L}$
  end
  In this case, the algorithm sends "test" queries for verification against the AI.
  if $v = 1$ **then**
    Let $\mathcal{S}_i$ be the i^{th} block of N queries in $\mathcal{S}$. Recall that N is the number of queries requested by CLF.
    Request $\mathcal{S}_i$ from the oracle, and verify that the response is consistent.
  end
end
output $\mathbb{L}$

Algorithm 7: Covert Verifiable Learning of large low-degree Fourier coefficients

Theorem 3.4. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLF *is a* $(poly(n), 1)$*-covert verifiable learning algorithm for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where the degree bound* $b \leq O(log\ n)$ *and* $\tau \geq 1/poly(n)$.

We claim that CVLF satisfies the computational soundness guarantee of Covert Verifiable Learning.

Proposition 3.5. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLF *satisfies the computational soundness guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where the degree bound* $b \leq O(log\ n)$ *and* $\tau \geq 1/poly(n)$.

Proof. We will show that on a single iteration of CVLF, the probability the AI can corrupt even a single oracle response and not be caught in CVLF is $p < 0.3$. From here, the statement follows using the completeness guarantee (Proposition 2.30).

For purposes of contradiction, suppose there exists $D_f \in \mathcal{D}_{\mathcal{F}_n}$ such that the AI has a p.p.t. algorithm $\tilde{\mathcal{B}}^f$ that, on the input of a set of queries $Q = \{q_1 \cdots q_m\}$ from a single iteration of CVLF, outputs a list of purported results $\tilde{f}(Q)$ such that $f(q) \neq \tilde{f}(q)$ for at least some $q \in Q$ that is not in the learner's private random examples, with probability at least 0.3. We will call such queries that are not in the learner's private example set "valuable," and call such a malicious strategy "successful" if the output is as described above. Recall that in CVLF, each iteration contains *only* valuable queries in the "learning" case, and otherwise no valuable queries in the "test" case. Thus, we have that when given queries from the "learning" case, the malicious strategy is successful with probability at least 0.6, and when given queries from the "test" case, is never successful.

We claim that if such a malicious strategy exists, then there also exists an efficient algorithm $\mathbb{B}^f$ that solves the decisional squared-log entropy LPN problem as follows:

1. $\mathbb{B}^f$ obtains N challenge samples $(C_i, c_i) \in \mathbb{Z}_2^{n \times n} \times \mathbb{Z}_2^n \ \forall i \in [N]$ from either the LPN distribution or uniformly random. Recall that N is the number of queries requested by one execution of CLF.
2. $\mathbb{B}^f$ constructs the set $Q = \{c_i \oplus \mathrm{gl}^i \mid i \in [N]\}$.
3. $\mathbb{B}^f$ executes $R = \tilde{\mathcal{B}}^f(Q)$, and then checks to see if the purported results R differ from $f(Q)$ (i.e. $\tilde{\mathcal{B}}^f$ was successful).
4. Repeat step 3 $O(1)$ times in order to estimate the success probability within an additive error of 0.01 with probability 0.9.
5. If the estimated success probability is at least 0.5, output "LPN," else output "uniform."

Let us analyze the behavior this algorithm. Observe that when the challenge samples come from the uniform distribution, then $\tilde{\mathcal{B}}^f$ operates on a distribution of queries which is identical to the "test" case of CVLF. Thus, the estimated success probability in step 4 is at most 0.01, with probability 0.9. On the other hand, when the challenge samples come from the LPN distribution, then $\tilde{\mathcal{B}}^f$ operates on a distribution of queries which is computationally indistinguishable from the "learning" case of CVLF. Hence, the success probability estimated in step 4 is at least 0.5, with probability at least 0.9 (up to negligible additive factors). As a result, $\mathbb{B}$ distinguishes correctly between the LPN distribution and the uniform distribution with high probability.

Therefore, with probability at least 0.3, a single iteration of the protocol contains all correct labels (and this is true independently for each independent round). By Proposition 2.30 (completeness), we conclude that CVLF satisfies the computational soundness guarantee, since only one fully truthful iteration is needed.

□

Now, we turn our attention to proving privacy of CVLF in the verifiable setting. We need to adapt the proof of privacy for CLF by tweaking the simulator as well as the reduction slightly to account for the new "test" query case, and the modified security experiments of the CVL definition.

Proposition 3.6. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLF *satisfies the privacy guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where the degree bound* $b \leq O(log\ n)$ *and* $\tau \geq 1/poly(n)$, *and* $\delta \geq exp(-n)$.

Proof. For any adversarial intermediary $\mathcal{I}$, we will construct a simulator Sim, which, for the algorithm CVLF, it holds that

$$\left\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}}\right\} \overset{c}{\approx} \left\{\mathsf{Videal}_{Sim,\mathcal{I}}\right\}$$

Our simulator is going to be given access to the total number of queries that is requested by the learner in the real interaction. This quantity is necessary and is therefore leaked. Also, recall that the simulator gets access to the random examples (denoted by $\mathcal{S}$) that are known to the real-world learner. We point out that, since the chosen queries are not inherently adaptive, it is possible to run all the iterations of CLF in parallel. We may compress all the messages into one large message that fits into a $poly(n) \times n$ matrix, and decode the results from there. We prove security in this setting.

The following is the p.p.t. simulation algorithm with respect to an adversarial intermediary $\mathcal{I}$:

Algorithm: $Sim(\epsilon, \delta, N, \mathcal{S}, \mathcal{S}')$

Inputs:
N *is the total number of queries requested by the real learner in one iteration of* CVLF.
$\mathcal{S}$ *are the set of examples known to the real world learner, while* $\mathcal{S}'$ *are an independently sampled set of examples to the concept.*

$W = \emptyset$
for $i \in [O(log(\frac{1}{\delta}))]$ **do**
  $v \xleftarrow{\$} \{0,1\}$
  if $v = 0$ **then**
    Sim adds the i^{th} block of N examples from $\mathcal{S}$ to W.
  else
    Sim adds the i^{th} block of N examples from $\mathcal{S}'$ to W.
  end
end
Sim executes the code of $\mathcal{I}$ on inputs (ϵ, δ, W). Let the output of $\mathcal{I}$ be z.
output z

Algorithm 8: Simulator for Covert Verifiable Learning of low-degree heavy Fourier coefficients

We will show that an adversary $\mathbb{A}$ who can distinguish $\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}}\}$ and $\{\mathsf{Videal}_{Sim,\mathcal{I}}\}$ can solve the decisional squared-log entropy LPN problem (from there, the proof concludes by Lemma 2.29). Thus, suppose there is an AI $\mathcal{I}^*$ which allows $\mathbb{A}$ to distinguish between the distributions

$$\left\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}^*}\right\} \overset{c}{\approx} \left\{\mathsf{Videal}_{Sim,\mathcal{I}^*}\right\}$$

with some non negligible distinguishing advantage.

Recall that

$$\left\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}^*}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathcal{S}, \mathsf{real}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}^*}\right)\right\}$$

where $\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta$ and D_f are chosen ahead of time by the adversary and $\mathcal{S} \sim D_f$ is the set of examples given to CVLF. On the other hand,

$$\left\{\mathsf{Videal}_{Sim,\mathcal{I}^*}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathcal{S}, \mathsf{ideal}^{Sim}_{\mathcal{I}^*}\right)\right\}$$

where $\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta$ and $\mathcal{D}_f$ are chosen ahead of time by the adversary and $\mathcal{S} \sim D_f$ is the first set of examples given to the simulator. Then, there exists an algorithm $\mathbb{B}$ that solves the decisional squared-log entropy LPN problem problem as follows.

1. Let r be the total number of rounds in CVLF. Let m be the size of the set of random examples given privately to CVLF. Note that, under the stated parameter regime, $m = poly(n)$.
2. $\mathbb{B}$ computes a set of queries and responses W as follows. Repeat r times:
   (a) $\mathbb{B}$ obtains N challenge samples $(C_i, c_i) \in \mathbb{Z}_2^{n \times n} \times \mathbb{Z}_2^n \; \forall i \in [N]$ from either the LPN distribution or uniformly random.
   (b) Let D_f be the concept that would be chosen by the distinguisher $\mathbb{A}$, and let $\mathcal{S}$ be the set of (uniformly random) examples to D_f that were obtained by the learner. $\mathbb{B}$ computes the set of inputs $Q = \{c_i \oplus \mathrm{gl}^i \mid \forall i \in [N]\}$.
   (c) Choose a bit b at random. If $b = 0$, add a block of N examples from $\mathcal{S}$ to W. If $b = 1$, add $(Q, \mathcal{O}_f(Q))$.
3. $\mathbb{B}$ executes $z = \mathcal{I}^*(\epsilon, \delta, W)$. $\mathbb{B}$ runs $\mathbb{A}$ on
$$\Big(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \mathcal{S}, z\Big)$$
   where $\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta$ are those that would be chosen by $\mathbb{A}$ in the distinguishing experiment.
4. $\mathbb{B}$ outputs "uniform" if $\mathbb{A}$ outputs "ideal" and "LPN" otherwise.

We note that the challenges obtained in step (a) are of the same type over all iterations. Now, let us analyze the behavior of the above algorithm. First, let E_j denote the event that CVLF did not output reject prior to making any membership queries in the j^{th} iteration of CLF (in line 20). Second, let L denote the event that the $N \cdot r$ samples (C_i, c_i) came from the LPN oracle and $\neg L$ be the event that the $N \cdot r$ samples (C_i, c_i) are uniformly random. Finally, we denote by X the distribution that $\mathbb{A}$ is executed on above.

We observe that,
$$\Big\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}^*} \;\Big|\; \bigwedge_j E_j\Big\} = \Big\{\mathsf{X} \mid L\Big\}$$
and
$$\Big\{\mathsf{Videal}_{Sim,\mathcal{I}^*}\Big\} = \Big\{\mathsf{X} \mid \neg L\Big\}$$

At iteration j of CLF, a standard application of a Chernoff bound shows that $\Pr[\neg E_j] \leq negl(n)$. Since we have that $\delta \geq exp(-n)$ and $\tau \geq 1/poly(n)$, the total number of iterations is bounded by a polynomial of n. Invoking the union bound, it follows that
$$\Pr\Big[\bigwedge_j E_j\Big] \geq 1 - negl(n)$$
and this gives that
$$\Big\{\mathsf{Vreal}^{\mathcal{O}_{\mathcal{D}_{\mathcal{F}_n}}}_{\mathsf{CVLF},\mathcal{I}^*}\Big\} \overset{c}{\approx} \Big\{\mathsf{X} \mid L\Big\}$$

Therefore, any noticeable distinguishing advantage of $\mathbb{A}$ is inherited by $\mathbb{B}$.

□

Finally, we consider completeness. Intuitively, since the AI is assumed to be honest for completeness, we can nearly immediately conclude that CVLF satisfies completeness: all the learning is done by the CLF subroutine, and the "test" case of CVLF will never accidentally reject when interacting with an honest AI.

Proposition 3.7. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLF *satisfies the completeness guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $b \leq O(log\ n), \tau \geq 1/poly(n)$, $\delta \geq exp(-n)$.

Proof. Let E be the event that at least one "learning case" is executed by CVLF. We need to prove that $\Pr[E] \geq 1 - \delta/2$ (then the statement follows immediately by Proposition 2.30). Observe that at each iteration the probability of a learning case occurring is $1/2$. Thus $\Pr[E] \geq 1 - \delta^2$, when taking an appropriate constant on the number of iterations $O(log(1/\delta))$. Thus, for $\delta < 1/2$, the statement holds. □

Finally, the proof of Theorem 3.4 is done:

Proof of Theorem 3.4. Immediate, following from Proposition 3.5, Proposition 3.6, and Proposition 3.7. □

3.3 Making CLDT Verifiable

To make CLDT verifiable, almost all of the work has already been done by constructing CVLF. We may modify CLDT by replacing the execution of CLF in line 4 with CVLF, and this alone suffices. For clarity:

1 **Algorithm:** $\mathsf{CVLDT}(\mathcal{H}_n, \epsilon, \delta)$
2 $\gamma = \sqrt{\epsilon/2}$
3 $\tau = \gamma\epsilon/32s$
4 $\mathbf{L} = \mathsf{CVLF}(\mathsf{FOURIER}_{[n],log(32s/\epsilon)}, \mathcal{C}_{\mathsf{FOURIER},b,n}, 0, \delta/2, \tau, \mathcal{S})$
5 **if** $\boldsymbol{L}$ = reject **then**
6 | **output reject**
7 **end**
8 $\delta' = \frac{\delta}{2|\mathbf{L}|}$
9 **for** *each* $\chi_k \in \boldsymbol{L}$ **do**
10 | Using random examples from $\mathcal{S}$, estimate $\tilde{f}(k) = \mathbb{E}_x[f(x)\chi_k(x)]$ within an additive interval of $\sqrt{\epsilon/|\mathbf{L}|}$ with probability $1 - \delta'$.
11 **end**
12 $g(x) = \sum_{\chi_k \in \mathbf{L}} \tilde{f}(k)\chi_k(x)$
13 **output** $h = \text{sign}(g)$

Algorithm 9: Covert Verifiable Learning of polynomial size decision trees

All three guarantees of Covert Verifiable Learning intuitively hold for CLDT, as all the communication of CVLDT is contained in CVLDT. We include all three statements and their proofs in Appendix E.

Theorem 3.8. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption, the collection* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s \leq poly(n)$ *is* $(poly(n), 4)$*- covertly verifiably learnable, with respect to* $\mathcal{D}_{\mathcal{F}_n}$*,* $\mathcal{O}_{\mathcal{F}_n}$*, and* $\mathcal{L}_D$*, and where* $\epsilon \geq 1/poly(n)$*, and* $\delta \geq exp(-n)$*.*

Proof. Immediate, from Proposition E.1, Proposition E.2, and Proposition E.3. Note that, CVLDT will be efficient for $s \leq poly(n)$ and $\epsilon \geq 1/poly(n)$, and $\delta \geq exp(-n)$.

□

3.4 Verifiability Without Secret Examples

In this section, we pose the question: *can we achieve Covert Verifiable Learning in a setting where the learner has no private examples to leverage against the adversarial intermediary?* Indeed, we are considering the *public* Covert Verifiable Learning model briefly discussed in Section 3.1.1.

We will demonstrate that our CVL protocol for low-degree Fourier coefficients of Section 2.4 can be adapted to fit the Public Covert Verifiable Learning (PCVL) model (formally defined as Definition B.7). From there, we can conclude that an application to decision trees is suitable, similar to that of Section 2.5.

Our algorithm CVLF (and soundness proof) falls short of the PCVL model—it makes crucial use of secret examples. Specifically, the AI will always know when the learner is executing a "test" case, because it has access to the test examples before hand, and as a result can distinguish them from the learning case. Our idea to adapt is as follows. Instead of threatening to send private random examples at each iteration (with probability 1/2), we threaten to send the public examples under the same masking that we use on the Goldreich-Levin queries. In this way, we can show that the computationally bounded AI will be caught lying with high probability; the AI will not be able to link the masked public examples with the real public examples. We will require that the concept is computed by a polynomial size DNF formula[4], and this will be essential to letting the learner detect an AI. Why this is the case will become clear shortly, but intuitively, we must assume some structure on the concept; otherwise the learner has no hope in obtaining any correlation on the public examples save querying for them. Clearly, if the learner cannot get any correlation on the public examples without querying them, then the AI will always be able to deceive the learner.

Definition 3.9. *Let* s*-*$\mathcal{DNF}_n$ *be the class of all* $f : \{0,1\}^n \to \{-1,1\}$ *such that* f *is computable by a size* s *DNF formula. A DNF formula is said to have size* s *if it has* s *clauses.*

We only modify CVLF as follows:

[4]Note that, this is still an agnostic setting, despite not being *fully* agnostic, as before.

Algorithm: $\mathsf{PCVLF}(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta, \tau, \mathcal{S})$

$\mathbb{L} = \emptyset$

$\theta = \tau n^{-2}$

for $i \in [O(log(\frac{1}{\delta}))]$ **do**

  $v \xleftarrow{\$} \{0,1\}$

  In this case, the algorithm sends meaningful queries via the subroutine CLF*.*

  if $v = 0$ **then**

    $\mathbf{L} = \mathsf{CLF}(\mathsf{FOURIER}_{T,b,n}, \epsilon, \delta/2, \theta)$

    $\mathbb{L} = \mathbb{L} \cup \mathbf{L}$

  end

  In this case, the algorithm sends "test" queries for verification against the AI.

  if $v = 1$ **then**

    N is the size of the set of Goldreich-Levin queries.

    Let $\mathcal{S}_i$ be the i^{th} block of N queries in $\mathcal{S}$.

    for $j \in [N]$ **do**

      $\mathbf{R}_j = \mathbf{U}_{n\times\lambda}\mathbf{U}_{\lambda\times n}$ where $\mathbf{U}_{q\times r}$ is a uniformly random $q \times r$ binary matrix and $\lambda = \Theta(log^2 n)$.

      $s_j \xleftarrow{\$} \tilde{\beta}^n_{\mu^*}$ for $\mu^* = log(n)/n$

      If $\tilde{\beta}^n_{\mu^*}$ outputs $\perp$, then stop and output **reject**.

      $e_j \xleftarrow{\$} \beta^n_\mu$ for $\mu = 1/8$.

    end

    Select an example $(x_j, y_j) \in \mathcal{S}$ for $j \in [N]$ uniformly at random.

    $\hat{x}_j = \mathbf{R}_j s_j \oplus e_j \oplus x_j$

    $Q = \{\hat{x}_j \mid \forall j \in [N]\}$

    Request Q from the oracle. Let $\tilde{f}(x)$ denote the purported result on query x.

    $\phi_j(\hat{x}_j) = \tilde{f}(\hat{x}_j)\chi_{s_j}(r)$ where $r \sim \beta^n_{1/2}$

    Checking the correlation with public examples. Larger Z means larger (positive) correlation. Note that Z is a real number.

    Compute $Z = \frac{1}{N}\sum_j \phi_j(\hat{x}_j)y_j$

    if $Z \leq \frac{2}{9}\tau n^{-2}$ **then**

      output reject

    end

  end

end

output $\mathbb{L}$

Algorithm 10: Public Covert Verifiable Learning of large low-degree Fourier coefficients

We state our theorem first, and spend the rest of the section assembling the proof.

Theorem 3.10. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* PCVLF *is a proper* $(poly(n), 1)$*-Public*

covert verifiable learning algorithm for $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{s\text{-}\mathcal{DNF}_n}$, $\mathcal{O}_{s\text{-}\mathcal{DNF}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $\delta \geq exp(-n), b \leq O(log\ n)$, $1/poly(n) \leq \tau \leq 1/2(2s+1)^2$, *and the DNF size* $s \leq poly(n)$.

Proof Idea. We begin with a lemma that establishes a correlation between the "test case" queries of the learner and the publicly available examples. Using this lemma, we can prove soundness by showing that if the AI lies on a "significant" amount of queries then the learner will be able to detect this using the correlation with the public examples. On the other hand, we observe that if the AI lies on a "less than significant" amount of queries, completeness still holds from the properties of CVLF—thus we conclude PCVLF is sound. For completeness, we need to prove that, essentially, the learner will not accidentally abort the interaction too often. This is done by bounding the probability that an honest AI is unlucky using standard probabilistic techniques. Finally, the proof of privacy is done by adapting the simulator and reduction of Proposition 3.6 to appropriately reflect the changes we made in the test case of the algorithm.

We start with the following lemma. Recall that f is the target function of the concept and it is computable by a DNF formula of size s, and ϕ_i is the i^{th} unmasking operation defined in line 26 of PCVLF.

Lemma 3.11. *For all* $i \in [N]$,

$$\mathbb{E}[\phi_i(\hat{x}_i)f(x_i)] \geq \frac{1}{3(2s+1)^2n^2} - negl(n)$$

when the AI responds honestly to the query $\hat{x}_i$, *and*

$$\mathbb{E}[\phi_i(\hat{x}_i)f(x_i)] \leq -\frac{1}{3(2s+1)^2n^2} + negl(n)$$

when the AI responds dishonestly to the query $\hat{x}_i$.

Proof. Here, as in lines 19-27 of PCVLF, x_i is a uniformly random example and $\hat{x}_i$ is x_i with the masking applied. Additionally, since there exists a size s DNF formula that computes f, it follows by a lemma of Bshouty and Feldman (lemma 18 of [BF02]) that there must be a $k \in \{0,1\}^n$ such that $\hat{f}(k) \geq 1/(2s+1)$ where χ_k is of degree $O(log\ s)$.

Therefore, by invoking Lemma 2.31, we have that for all i,

$$\mathbb{E}\Big[\phi_i(\hat{x}_i)\chi_k(x_i)\Big] \geq \frac{1}{6(2s+1)n^2}$$

when the AI is honest on $\hat{x}_i$ and

$$\mathbb{E}\Big[\phi_i(\hat{x}_i)\chi_k(x_i)\Big] \leq -\frac{1}{6(2s+1)n^2}$$

when the AI is dishonest on $\hat{x}_i$. Additionally, by Lemma 2.29 and using closure under polynomial composition (note that $N = poly(n)$ for our parameter regime),

$$(\hat{x}_i)_{i\in[N]} \overset{c}{\approx} (u_i)_{i\in[N]}$$

where $u_i \xleftarrow{\$} \{0,1\}^n$. Because f is polynomial time computable, this entails that

$$\mathbb{E}\left[\phi_i(u_i)\chi_k(x_i)\right] \geq \frac{1}{6(2s+1)n^2} - negl(n)$$

when the AI is honest on u_i and

$$\mathbb{E}\left[\phi_i(u_i)\chi_k(x_i)\right] \leq -\frac{1}{6(2s+1)n^2} + negl(n)$$

when the AI is dishonest on u_i.

By assumption, $\mathbb{E}[f(x_i)\chi_k(x_i)] \geq 1/(2s+1)$ for each x_i (recall, each x_i are independent and uniformly random queries, while f is a $poly(n)$ size DNF formula). Therefore we have that for all i,

$$\begin{aligned}
\mathbb{E}\left[\phi_i(\hat{x}_i)f(x_i)\right] &= \Pr\left[\phi_i(\hat{x}_i)f(x_i) = 1\right] - \Pr\left[\phi_i(\hat{x}_i)f(x_i) = -1\right] \\
&\geq \Pr\left[\phi_i(u_i)f(x_i) = 1\right] - \Pr\left[\phi_i(u_i)f(x_i) = -1\right] - negl(n) \\
&\geq \Pr\left[\phi_i(u_i)f(x_i) = 1 \middle| \chi_k(x_i) = 1\right] + \Pr\left[\phi_i(u_i)f(x_i) = 1 \middle| \chi_k(x_i) = -1\right] \\
&\quad - \Pr\left[\phi_i(u_i)f(x_i) = -1\right] - negl(n) \\
&\geq 1 + \frac{1}{6(2s+1)^2n^2} - \Pr\left[\phi_i(u_i)f(x_i) = -1 \middle| \chi_k(x_i) = 1\right] \\
&\quad - \Pr\left[\phi_i(u_i)f(x_i) = -1 \middle| \chi_k(x_i) = -1\right] - negl(n) \\
&\geq \frac{1}{3(2s+1)^2n^2} - negl(n)
\end{aligned}$$

in the honest case. Likewise, the dishonest case gives

$$\mathbb{E}\left[\phi_i(\hat{x}_i)f(x_i)\right] \leq -\frac{1}{3(2s+1)^2n^2} + negl(n)$$

□

Proposition 3.12. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* PCVLF *satisfies the completeness guarantee of Public Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{s\text{-}\mathcal{DNF}_n}$, $\mathcal{O}_{s\text{-}\mathcal{DNF}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $\delta \geq exp(-n), b \leq O(log\ n)$, $1/poly(n) \leq \tau \leq 1/2(2s+1)^2$, *and DNF size* $s \leq poly(n)$.

Proof. It suffices to show that the $v = 1$ "test" case will not cause PCVLF to reject if the AI is honest, except with probability $o(\delta)$ (note that, in PCVLF we take the failure probability of CLF subroutine as $\delta/2$). From there, completeness follows from Proposition 2.30, because in the "learning" case PCVLF is identical to CVLF. To prove this, we invoke Lemma 3.11, which shows that $\mathbb{E}[Z] \geq 1/3(2s+1)^2n^2 - negl(n)$. Let R be the event that PCVLF outputs reject in line 19 in a single iteration. Then,

$$\begin{aligned}
\Pr[R] &\leq \Pr\left[Z \leq \frac{2}{3}\mathbb{E}[Z]\right] \\
&\leq exp(-\Omega(N\tau n^{-2}))
\end{aligned}$$

by a Chernoff bound, and where the probability is taken over all the randomness of the protocol. Since $N = \Omega(n^4/\tau^2)$, $\Pr[R] \leq exp(-\Omega(n^2))$. Therefore, over the $O(log(1/\delta))$ iterations, the probability of rejection when the AI is honest is at most $O(log(1/\delta) \cdot exp(-\Omega(n^2)))$ by the union bound. Since $\delta \geq exp(-n)$, this quantity is $o(\delta)$.

□

We observe a stronger property of PCVLF which derives from CVLF.

Observation 3.13. *If the AI is sufficiently honest in the "learning" case, in the sense that for every z, for every $j \in [n] \cup \{-\}$, the AI returns query results denoted by $\tilde{f}(x)$ such that*

$$\Pr_S\left[\tilde{f}(\hat{\text{gl}}_{S,j,z}) \neq f(\hat{\text{gl}}_{S,j,z})\right] \leq \frac{1}{4}$$

then PCVLF *either outputs an ϵ-good hypothesis or rejects the interaction.*

Proof sketch. Let E be the event that for every z, for every $j \in [n] \cup \{-\}$

$$\Pr_S\left[\tilde{f}(\hat{\text{gl}}_{S,j,z}) \neq f(\hat{\text{gl}}_{S,j,z})\right] \leq \frac{1}{4}$$

It can be seen from the proof of completeness of CLF (Proposition 2.30) that given E, the completeness guarantee still holds. However, it remains to consider that the changes of the test case of PCVLF may cause the learner to unnecessarily reject the interaction. Indeed, with some probability, this occurs in line 19, but this does not affect the statement. □

We will use this observation to prove computational soundness.

Proposition 3.14. *Under the $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ assumption,* PCVLF *satisfies the computational soundness guarantee of Public Covert Verifiable Learning for $\mathcal{C}_{\mathsf{FOURIER},b,n}$ with respect to $\mathcal{D}_{s\text{-}\mathcal{DNF}_n}$, $\mathcal{O}_{s\text{-}\mathcal{DNF}_n}$, and $\mathcal{L}^{\tau,b}$, and where $b \leq O(log\ n)$, $1/poly(n) \leq \tau \leq 1/2(2s+1)^2$, and $s \leq poly(n)$.*

Proof. Let E_i be the event that E occurs on iteration i of PCVLF. Note that proving that for every $i \in [O(log(1/\delta))]$,

$$\neg E_i \implies \mathsf{PCVLF} = \text{reject}$$

(with probability 1/2, independently) suffices to prove the claim. Thus, (without loss of generality) pick $i \in [O(log 1/\delta)]$ and assume $\neg E_i$.

Fix any $D_f \in \mathcal{D}_{s\text{-}\mathcal{DNF}_n}$. Using the fact that for all $k, \ell \in [N]$ $\hat{\text{gl}}^k \overset{c}{\approx} \hat{\text{gl}}^\ell$, we have that for any z, the existence of a $j \in [n] \cup \{-\}$, such that the AI returns query results $\tilde{f}(x)$ with

$$\Pr_S\left[\tilde{f}(\hat{\text{gl}}_{S,j,z}) \neq f(\hat{\text{gl}}_{S,j,z})\right] > \frac{1}{4}$$

implies the same is true *for every* z and *every* $j \in [n] \cup \{-\}$, and replacing 1/4 with 1/5. If not true, then the workings of the AI can be fashioned into a distinguishing algorithm for $\hat{\text{gl}}_{S,i,z}$ and $\hat{\text{gl}}_{S,j,z}$ (for a fixed S, z and $i \neq j$), giving a contradiction. It follows, then, that the total fraction of dishonest query results is at least 1/5.

Considering this as the case, let us analyze the behavior of the $v = 1$ case of PCVLF. Since the distribution over the set of queries requested by the learner is computationally indistinguishable from the "learning" case, we can conclude, using a similar argument to above, that the total fraction of dishonest query results in the "test" case is at least $1/6$.

Now, since the fraction of dishonest queries can be taken to be at least $1/6$, we see that (in line 28, PCVLF) $\mathbb{E}[Z] \leq 2/3 \cdot 2/6(2s+1)^2 n^2$ when $\tau \leq 1/2(2s+1)^2$, and therefore PCVLF rejects in line 29 with probability $1/2$ in a single iteration (by the properties of the binomial distribution).

□

Proposition 3.15. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* PCVLF *satisfies the privacy guarantee of Public Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{FOURIER},b,n}$ *with respect to* $\mathcal{D}_{s\text{-}\mathcal{DNF}_n}$, $\mathcal{O}_{s\text{-}\mathcal{DNF}_n}$, *and* $\mathcal{L}^{\tau,b}$, *and where* $b \leq O(log\ n)$, $1/poly(n) \leq \tau \leq 1/2(2s+1)^2$, *and DNF size* $s \leq poly(n)$.

Proof. For any adversarial intermediary $\mathcal{I}$, we will construct a simulator Sim, which, for the algorithm PCVLF, it holds that

$$\left\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}}\right\} \overset{c}{\approx} \left\{\mathsf{PVideal}_{Sim,\mathcal{I}}\right\}$$

Our simulator is going to be given access to the total number of queries that is requested by the learner in the real interaction. This quantity is necessary and is therefore leaked. Also, recall that the simulator gets access to the random examples (denoted by $\mathcal{S}$) that are known to the real-world learner. We point out that, since the chosen queries are not inherently adaptive, it is possible to run all the iterations of CLF in parallel. We may compress all the messages into one large message that fits into a $poly(n) \times n$ matrix, and decode the results from there. We prove security in this setting.

The following is the p.p.t. simulation algorithm with respect to an adversarial intermediary $\mathcal{I}$:

Algorithm: $Sim(\epsilon, \delta, N, \mathcal{S}, \mathcal{S}')$

Inputs:
N *is the total number of queries requested by the real learner in one iteration of* CVLF.
$\mathcal{S}$ *are the set of examples known to the real world learner, while* $\mathcal{S}'$ *are an independently sampled set of examples to the concept.*

$W = \emptyset$
for $i \in [O(log(\frac{1}{\delta}))]$ **do**
  $v \xleftarrow{\$} \{0,1\}$
  if $v = 0$ **then**
    Sim adds the i^{th} block of N examples from $\mathcal{S}$ to W.
  else
    Sim adds the i^{th} block of N examples from $\mathcal{S}'$ to W.
  end
end
Sim executes the code of $\mathcal{I}$ on inputs $(\epsilon, \delta, W, \mathcal{S})$. Let the output of $\mathcal{I}$ be z.
output z

Algorithm 11: Simulator for Public Covert Verifiable Learning of low-degree heavy Fourier coefficients

We will show that an adversary $\mathbb{A}$ who can distinguish $\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}}\}$ and $\{\mathsf{PVideal}_{Sim,\mathcal{I}}\}$ can solve the decisional squared-log entropy LPN problem (from there, the proof concludes by Lemma 2.29). Thus, suppose there is an AI $\mathcal{I}^*$ which allows $\mathbb{A}$ to distinguish between the distributions

$$\left\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}^*}\right\} \overset{c}{\approx} \left\{\mathsf{PVideal}_{Sim,\mathcal{I}^*}\right\}$$

with some non negligible distinguishing advantage.

Recall that

$$\left\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}^*}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n},\epsilon,\delta,\mathcal{S},\mathsf{real}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{CVLF},\mathcal{I}^*}\right)\right\}$$

where $\mathsf{FOURIER}_{T,b,n},\epsilon,\delta$ and D_f are chosen ahead of time by the adversary and $\mathcal{S} \sim D_f$ is the set of examples given to PCVLF. On the other hand,

$$\left\{\mathsf{PVideal}_{Sim,\mathcal{I}^*}\right\} = \left\{\left(\mathsf{FOURIER}_{T,b,n},\epsilon,\delta,\mathcal{S},\mathsf{ideal}^{Sim}_{\mathcal{I}^*}\right)\right\}$$

where $\mathsf{FOURIER}_{T,b,n},\epsilon,\delta$ and $\mathcal{D}_f$ are chosen ahead of time by the adversary. Then, there exists an algorithm $\mathbb{B}$ that solves the decisional squared-log entropy LPN problem problem as follows.

1. Let r be the total number of rounds in PCVLF. Let m be the size of the set of random examples known publicly to PCVLF and $\mathcal{I}^*$. Note that, under the stated parameter regime, $m = poly(n)$.
2. $\mathbb{B}$ computes a set of queries and responses W as follows. Repeat r times:
   (a) $\mathbb{B}$ obtains N challenge samples $(C_i, c_i) \in \mathbb{Z}_2^{n\times n} \times \mathbb{Z}_2^n \; \forall i \in [N]$ from either the LPN distribution or uniformly random.
   (b) Let D_f be the concept that would be chosen by the distinguisher $\mathbb{A}$, and let $\mathcal{S}$ be the set of size m of (uniformly random) examples that was given to the learner. $\mathbb{B}$ computes the set of inputs $Q = \{c_i \oplus \mathrm{gl}^i \mid \forall i \in [N]\}$.
   (c) Choose a bit b at random. If $b = 0$, add a block of N examples from $\mathcal{S}$ to W. If $b = 1$, add $(Q, \mathcal{O}_f(Q))$.
3. $\mathbb{B}$ executes $z = \mathcal{I}^*(\epsilon,\delta,W,\mathcal{S})$. $\mathbb{B}$ runs $\mathbb{A}$ on
   $$\left(\mathsf{FOURIER}_{T,b,n},\epsilon,\delta,\mathcal{S},z\right)$$
   where $\mathsf{FOURIER}_{T,b,n},\epsilon,\delta$ are those that would be chosen by $\mathbb{A}$ in the distinguishing experiment.
4. $\mathbb{B}$ outputs "uniform" if $\mathbb{A}$ outputs "ideal" and "LPN" otherwise.

We note that the challenges obtained in step (a) are of the same type over all iterations. Now, let us analyze the behavior of the above algorithm. First, let E_j denote the event that PCVLF did not output reject prior to making any membership queries[5]. Second, let L denote the event that the $N \cdot r$ samples (C_i, c_i) came from the LPN oracle and $\neg L$ be the event that the $N \cdot r$ samples (C_i, c_i) are uniformly random. Finally, we denote by X the distribution that $\mathbb{A}$ is executed on above.

We observe that,

$$\left\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}^*} \;\middle|\; \bigwedge_j E_j\right\} = \left\{\mathsf{X} \mid L\right\}$$

[5]This event could occur either in line 20 of CLF (during the "learning" case of PCVLF) or in line 18 of the "test case" of PCVLF.

and

$$\left\{\mathsf{PVideal}_{Sim,\mathcal{I}^*}\right\} = \left\{\mathsf{X} \mid \neg L\right\}$$

A standard application of a Chernoff bound shows that $\Pr[\neg E_j] \leq negl(n)$. Since we have that $\delta \geq exp(-n)$ and $\tau \geq 1/poly(n)$, the total number of iterations is bounded by a polynomial of n. Invoking the union bound, it follows that

$$\Pr\left[\bigwedge_j E_j\right] \geq 1 - negl(n)$$

and this gives that

$$\left\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_{s\text{-}\mathcal{DNF}_n}}}_{\mathsf{PCVLF},\mathcal{I}^*}\right\} \overset{c}{\approx} \left\{\mathsf{X} \mid L\right\}$$

Therefore, any noticeable distinguishing advantage of $\mathbb{A}$ is inherited by $\mathbb{B}$.

□

The statement of Theorem 3.10 now easily follows from the above three propositions.

Proof of Theorem 3.10. Immediate, from Proposition 3.12, Proposition 3.14 and Proposition 3.15. □

Remark 3.16. *We omit statements and proofs, but it is not difficult to see that the above PCVL construction suffices to imply a PCVL construction for polynomial size decision trees under analogous conditions. Indeed, looking back to Section 2.5, the* CVLDT *algorithm uses* CVLF *as a black box, and similarly the proofs of the completeness, privacy and soundness are black box in that they just rely on the aforementioned properties of the* CVLF *algorithm. The proofs of Section 2.5 can be extended to capture PCVL for decision trees using* PCVLF*.*

3.5 Perfect Privacy and Statistical Soundness With Fully Private Examples

In this section, we operate in the fully private examples variant of Covert Verifiable Learning (CVLFP, Definition B.4). We focus on the problem of learning juntas. Informally, a k-junta is an n-bit boolean function that depends on at most k-out-of-n variables. Formally,

Definition 3.17. *The influence of a coordinate $i \in [n]$ of the function $f : \{0,1\}^n \to \{0,1\}$ is defined as*

$$Inf_i(f) = \Pr_{x \sim \{0,1\}^n}\left[f(x) \neq f(x^{\oplus i})\right]$$

where $x^{\oplus i}$ denotes the vector x with the i^{th} coordinate flipped. When $Inf_i(f) > 0$, we say that the i^{th} coordinate is relevant.

Definition 3.18. *We say that a function $f : \{0,1\}^n \to \{0,1\}$ is k-junta if it has at most k relevant variables.*

Definition 3.19. *Let k-$\mathcal{JUNTA}_n$ be the class of all $f : \{0,1\}^n \to \{0,1\}$ such that f is a k-junta.*

We will construct a CVLFP algorithm will output the *exact* truth table of a given concept in $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, so the hypothesis class we consider is also $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$ (we operate in the *realizable setting*).

Consider the following algorithm, which is assumed to know the value of k (the size of the junta). A very similar algorithm was described in [IKOS19], where it functions in the distributed variant of Cryptographic Sensing. We incorporate soundness and cast the problem in the new CVLFP framework. Note that the algorithm does not take any target hypothesis class because we focus only on concept-hiding.

1 **Algorithm:** $\mathsf{CVLJ}(\epsilon, \delta, \mathcal{S}, k, n)$
2 $\mathbb{L} = \emptyset$
3 Partition $\mathcal{S}$ into two blocks $\mathcal{S}_0$, $\mathcal{S}_1$ each of size $poly(n, 1/\delta, 2^k)$.
4 Let Q be a set of examples constructed by taking one random hamming neighbor of each of $\delta/2$ fraction of examples in $\mathcal{S}_0$, and leaving the rest unchanged (denote these unchanged as the set $O \subseteq Q$).
5 *Request Q from the concept oracle.*
6 Let $\tilde{f}(Q)$ be the response from the concept oracle.
7 **for** $q \notin O$ **do**
8 Find the hamming neighbor of q, $\hat{q} \in \mathcal{S}_0$.
9 If $f(q) \neq f(\hat{q})$, then add the index j for which $q_j \neq \hat{q}_j$ to $\mathbb{L}$.
10 **end**
11 Using the examples in $\mathcal{S}_1$, find a set of examples T (if it exists) that covers every assignment of the variables with indexes in $\mathbb{L}$.
12 **output** the truth table that is formed by T (if it exists). Otherwise **output** $\perp$.

Algorithm 12: Exact CVLFP protocol for k-juntas

We state the theorem and then compile the proof.

Theorem 3.20. CVLJ *is a $(poly(n), 1)$-covert verifiable learning algorithm with fully private examples and perfect privacy and statistical soundness, with respect to $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, $\mathcal{O}_{k\text{-}\mathcal{JUNTA}_n}$, and $\mathcal{L}_D$. We stipulate that $k = O(log\ n)$, and $\delta \geq 1/poly(n)$.*

Proposition 3.21. CVLJ *satisfies the completeness guarantee of CVLFP with respect to $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, $\mathcal{O}_{k\text{-}\mathcal{JUNTA}_n}$, and $\mathcal{L}_D$, when $k = O(log\ n)$ and $\delta \geq 1/poly(n)$ the algorithm runs in time $poly(n)$.*

Proof. For correctness we assume that the adversarial intermediary is honest, meaning that all responses to queries from the learner are correct. We will show that, with probability $1-\delta$, the exact truth table of any concept in $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$ can be recovered, which suffices to prove the statement. Fix any concept in $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, and let f be the k-junta which is the target function. Consider that, in line 12 of CVLJ, the index of a relevant variable of f is added to $\mathbb{L}$ with probability at least $n^{-1}2^{-k}$, by randomly selecting q to be sensitive at the random hamming neighbor $\hat{q}$. Thus, one relevant variable is found using one pair of neighbors with probability at least $n^{-1}2^{-k}$. Since finding a relevant variable of f is an independent event for each pair of neighbors, using $poly(n, 2^k)$ pairs suffices to find all k relevant variables with

constant probability. As a result, $\mathbb{L}$ contains all relevant variables with probability $1 - \delta/2$ (as the size of $\mathcal{S}_0$ is $poly(n, 2^k, 1/\delta)$).

Now, using $poly(n, 2^k, 1/\delta)$ examples from $\mathcal{S}_1$, it can be seen that the probability that all 2^k assignments of the k relevant variables chosen to $\mathbb{L}$ are found in T is $1 - \delta/2$ (note that an irrelevant variable is never added to $\mathbb{L}$, so $|\mathbb{L}| \leq k$). Therefore, the output of CVLJ is the exact truth table of the f, with probability $1 - \delta$ as desired.

Clearly, when $k = O(log\ n)$ and $\delta \geq 1/poly(n)$, the algorithm runs in polynomial time and uses polynomially many samples in n. □

Proposition 3.22. CVLJ *satisfies the perfect privacy guarantee of CVLFP with respect to* $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, $\mathcal{O}_{k\text{-}\mathcal{JUNTA}_n}$, *and* $\mathcal{L}_D$, *when* $k = O(log\ n)$. *Specifically, There exists a p.p.t. simulation algorithm Sim, such that for every adversarial intermediary* $\mathcal{I}$, *Sim satisfies*

$$\left\{\mathsf{FPVreal}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\right\} = \left\{\mathsf{FPVideal}_{Sim,\mathcal{I}}\right\}$$

where the above distributions are as in Definition B.4.

Proof. Consider the following p.p.t. simulator that interacts with any adversarial intermediary $\mathcal{I}$:

Algorithm: $Sim(\epsilon, \delta, \mathcal{S}, \ell)$

Sim takes ℓ examples from $\mathcal{S}$ as a set Q. Sim executes the code of $\mathcal{I}$ on inputs (ϵ, δ, Q), letting the result be z.

output z

Algorithm 13: Perfect simulator for CVLJ

Clearly, the set of queries Q that the simulator uses as input to the adversarial intermediary $\mathcal{I}$ is distributed identically to that of the real protocol. It follows immediately that

$$\left\{\mathsf{FPVreal}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\right\} \text{ and } \left\{\mathsf{FPVideal}_{Sim,\mathcal{I}}\right\}$$

are also distributed identically. □

Proposition 3.23. CVLJ *satisfies the statistical soundness guarantee of CVLFP with respect to* $\mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$, $\mathcal{O}_{k\text{-}\mathcal{JUNTA}_n}$, *and* $\mathcal{L}_D$, *when* $k = O(log\ n)$, $\delta \geq 1/poly(n)$.

Proof. We will show that the probability the AI can corrupt even a single oracle response and not be caught in CVLJ is $p < \delta$. From here, we can conclude computational soundness using the completeness guarantee of CVLJ.

Fix any $D_f \in \mathcal{D}_{k\text{-}\mathcal{JUNTA}_n}$. Suppose that $\tilde{\mathcal{B}}^f$ is a p.p.t. algorithm employed by the AI that takes as input of a set of queries $Q = \{q_1 \cdots q_m\}$ from CVLJ, and outputs a list of purported results $\tilde{f}(Q)$ such that $f(q) \neq \tilde{f}(q)$ for at least some $q \in Q$ that is not in the learner's private random examples, with probability at least δ. We will call such queries that are not in the learner's private example set "valuable," and call such an algorithm "successful" if the output is as described above.

We observe that the fraction of queries requested by the learner which are valuable is less than δ. Hence, since the valuable queries are distributed identically to the non-valuable queries, a successful p.p.t. algorithm $\tilde{\mathcal{B}}^f$ cannot exist.

□

Proof of Theorem 3.20. Putting together the above three claims, the statement follows. Note that, CVLJ runs in time $poly(n)$ when $k = O(log\ n)$ and $\delta \geq 1/poly(n)$. □

4 Key Exchange from Covert Learning

We switch gears and demonstrate how our Covert Learning algorithm for parity functions naturally lends itself to key exchange from the $(O(n), poly(n)) - \mathsf{DLPN}_{\frac{1}{\sqrt{n}},n}$ assumption. This result showcases the power of Covert Learning and suggests the possibility of investigating Covert Learning techniques for traditional cryptographic tasks like key exchange.

We will use the standard real/ideal paradigm for define a secure protocol. Thus, we first define the following ideal key exchange functionality.

Definition 4.1. ***Ideal Key Exchange.*** *Let T be a trusted third party. We consider the following process to be an Ideal Key Exchange for some keyspace κ.*

1. *Parties A and B interact for some finite number of rounds, sending each other a sequence of messages.*
2. *After A and B are done interacting, T draws a key from κ, and delivers it to A and B.*

Next, we define a Secure Key Exchange as a protocol that is indistinguishable from an Ideal Key Exchange, in that no computationally bounded adversary who views the interaction between parties A and B can decide if the interaction belonged to a real execution or ideal execution. More formally:

Definition 4.2. ***Secure Key Exchange.*** *A protocol is a secure key exchange if*

$$\left\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\right\} \overset{c}{\approx} \left\{\mathsf{transcript}^{\mathsf{ideal}}_{\mathsf{A,B}}\right\}$$

where $\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\}$ and $\{\mathsf{transcript}^{\mathsf{ideal}}_{\mathsf{A,B}}\}$ are the distributions over the strings containing all the messages sent between A and B, and the resulting agreed key from the real protocol and ideal key exchange protocol respectively.

Consider the following key exchange algorithm. Recall that:

Definition 4.3. *Let $\mathcal{X}_n = \{0,1\}^n$. We define the concept class $\mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ to be the family of distributions over $\mathcal{X}_n \times \{0,1\}$ indexed by a $k \in \{0,1\}^n$, that have the following properties,*

- *The input (marginal) distribution over $\mathcal{X}$ of any $D_k \in \mathcal{D}^{\mu,n}_{\mathsf{LPN}}$ is uniform.*
- *For any $D_k \in \mathcal{D}^{\mu,n}_{\mathsf{LPN}}$, the label $y \in \{0,1\}$ of the input x is generated by taking $\langle k, x\rangle$ and flipping the result with probability μ.*

Algorithm: $\mathsf{KeyExchange}(1^n)$

Bob selects a random key by drawing $k \xleftarrow{\$} \beta^n_{n^{-0.5}}$. Then, Bob prepares $\mathcal{O}_{D_k}$ corresponding to $D_k \in \mathcal{D}^{\mathsf{Low}}_{\mathsf{LPN}}$.

Alice begins running $\mathsf{CLP}^{O_{D_k}}(\mathsf{PARITY}_{[n]}, \frac{1}{8}, \delta, \mathcal{S})$, where all her calls to $\mathcal{O}_{D_k}$ are delegated to Bob, by sending Bob those queries.

Bob sends back all the query results.

Bob outputs k, and Alice outputs the result of CLP.

Algorithm 14: Key Exchange protocol from Covert Learning

Theorem 4.4. *If* $\mathsf{DLPN}_{\frac{1}{\sqrt{n}},n}$ *is* $(O(n), poly(n))$*-hard,* $\mathsf{KeyExchange}$ *is a secure key exchange protocol.*

Proof. We begin by arguing that the scheme is correct, in that Alice and Bob both output the same key with high probability. This follows directly from the completeness guarantee of CLP. With probability $1-\delta$, Alice gets a hypothesis $h \in \mathsf{PARITY}_{[n]}$ that is $\frac{1}{8}$-close to k. Now, since every linear function is $\frac{1}{2}$-far from every other linear function, it is easy to see that the output of CLP is k with probability $1-\delta$. Thus, Alice and Bob output the same k with probability $1-\delta$. For this cryptographic application, Alice should set δ to be some negligible function of n.

For security, we need to show that $\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\} \overset{c}{\approx} \{\mathsf{transcript}^{\mathsf{ideal}}_{\mathsf{A,B}}\}$ as in Definition 4.2. Define the ideal keyspace

$$\kappa = \{k \mid k \xleftarrow{\$} \beta^n_{n^{-0.5}}\}$$

We will prove something stronger:

$$\left\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\right\} \overset{c}{\approx} \left\{(R,k) | R \xleftarrow{\$} \mathbb{Z}_2^{m\times n+1}, k \xleftarrow{\$} \kappa\right\}$$

where $m = O(nlog(\frac{n}{\delta}))$ is the number of Alice's calls to $\mathcal{O}_{D_k}$ in $\mathsf{KeyExchange}$.

By Proposition 2.17, the distribution of messages between the parties is pseudorandom. Thus, suppose to the contrary that there exists an adversary $\mathbb{A}$ such that $\Pr[\mathbb{A}(\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\}) = 1] = \delta$ and $\Pr[\mathbb{A}(\{\mathsf{transcript}^{\mathsf{ideal}}_{\mathsf{A,B}}\}) = 1] = \gamma$ such that $|\delta - \gamma| = \epsilon \geq 1/poly(n)$. Then, there is an algorithm $\mathbb{B}$ that breaks the pseudorandomness of the messages as follows.

First, $\mathbb{B}$ obtains a challenge distribution *chal* (which is either real messages or uniformly random messages, each with probability 1/2). Let E be the event that *chal* is uniformly random messages. Second, draw $k \leftarrow \kappa$, and simulate an execution of $\mathsf{KeyExchange}$ by playing both the sender and receiver roles, using k. Let M be the messages sent in the simulated execution. Let $\mathbb{B}_{chal} = \mathbb{A}\big(\{(chal, k)\}\big)$ and $\mathbb{B}_M = \mathbb{A}\big(\{(M,k)\}\big)$. $\mathbb{B}$ outputs $\mathbb{B}_{chal}$ or $\mathbb{B}_M$ each with probability 1/2. Note that, in the event E,

$$\Big\{(chal, k)\Big\} = \Big\{(R,k) \mid R \xleftarrow{\$} \mathbb{Z}_2^{m\times n+1}, k \xleftarrow{\$} \kappa\Big\}$$

On the other hand, in the event $\neg E$,

$$\Big\{(chal, k)\Big\} = \Big\{(M,k)\Big\} = \Big\{\mathsf{transcript}^{\mathsf{real}}_{\mathsf{A,B}}\Big\}$$

Thus,

$$\begin{aligned}\Pr\Big[\mathbb{B}=1\Big|E\Big] &= \frac{1}{2}\Big(\Pr\Big[\mathbb{B}_{chal}=1\Big|E\Big]+\Pr\Big[\mathbb{B}_M=1\Big|E\Big]\Big)\\ &= \frac{1}{2}\Big(\delta+\gamma\Big)\end{aligned}$$

while

$$\begin{aligned}\Pr\Big[\mathbb{B}=1\Big|\neg E\Big] &= \frac{1}{2}\Big(\Pr\Big[\mathbb{B}_{chal}=1\Big|\neg E\Big]+\Pr\Big[\mathbb{B}_M=1\Big|E\Big]\Big)\\ &= \frac{1}{2}\Big(\delta+\delta\Big)=\delta\end{aligned}$$

and therefore

$$\begin{aligned}\Big|\Pr\Big[\mathbb{B}=1\Big|E\Big]-\Pr\Big[\mathbb{B}=1\Big|\neg E\Big]\Big| &= \frac{1}{2}\Big|\gamma-\delta\Big|\\ &\geq \frac{\epsilon}{2}\end{aligned}$$

Thus, the nonegligible distinguishing advantage of $\mathbb{A}$ is inherited by $\mathbb{B}$. We conclude that KeyExchange is a secure key exchange.

□

4.1 How Does Our Protocol Differ from Alekhnovich's?

In [Ale03], Alekhnovich introduces the low-noise LPN assumption and introduces a public key encryption scheme whose security hinges on the intractibility of the LPN assumption. As described in [BSBTD+16], the scheme roughly works as follows.

Let n be a security parameter, $m = 2n$, and $k = n^{1/2-\epsilon}$ for some small constant $\epsilon > 0$. The key generation works by selecting a random noise vector $e \in \mathbb{F}_2^m$ in which each component is set to 1 with probability k/m independently, selecting a uniformly random $G \in \mathbb{F}_2^{m\times n}$, and a uniformly random $w \in Image(G)$. The private key is the noise vector e and the public key is the $m \times (n+1)$ matrix $\tilde{G} = (G|b)$ obtained from G by appending the noisy codeword $b = w + e$ to the right of the matrix G. The encryption of $\sigma = 0$ is a random vector $c \in \mathbb{F}_2^m$ of the form $c = \tilde{w} + \tilde{e}$, where $\tilde{w}$ is a uniformly random vector in the kernel of $\tilde{G}^T$ and $\tilde{e} \in \mathbb{F}_2^m$ is a random noise vector distributed identically to but independently of the private key e. The encryption of $\sigma = 1$ is a uniformly random vector in $\mathbb{F}_2^m$. Decryption for a ciphertext c proceeds taking the inner product c, e. This inner product is a nearly uniform random bit when c is an encryption of 1, and is equal to the inner product $e, \tilde{e}$ when c is an encryption of 0 with high probability.

Our key exchange scheme can be transformed into a public key encryption scheme using the KEM (Key Encapsulation Mechanism) paradigm. Note that, in our protocol, the exchanged key k has slightly less than $\sqrt{n}$ bits of min-entropy. Thus, the sender can transmit the k using the key exchange scheme, and a message encrypted using a symmetric scheme with a uniformly random key of nearly $\sqrt{n}$ bits that is be obtained by applying a key derivation function to k. When the receiver obtains k, she can apply the derivation function and obtain the symmetric key used to encrypt the message, and then decyrpt it. The Alekhnovich scheme, though superficially similar to ours (it works over the binary field and uses low-noise LPN) possesses an important differences with ours. Namely, it directly encrypts only one bit a time. In contrast, in our scheme we encrypt almost $\sqrt{n}$ bits at a time.

Acknowledgements

We would like to thank Shafi Goldwasser and Ronitt Rubinfeld for very helpful discussions on the model and its motivation.

References

[ABK+04] Noga Alon, Richard Beigel, Simon Kasif, Steven Rudich, and Benny Sudakov. Learning a hidden matching. *SIAM Journal on Computing*, 33(2):487–501, 2004.

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180, 2010.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Annual International Cryptology Conference*, pages 595–618. Springer, 2009.

[AFK13] Pranjal Awasthi, Vitaly Feldman, and Varun Kanade. Learning using local membership queries. In *Conference on Learning Theory*, pages 398–431, 2013.

[Ale03] Michael Alekhnovich. More on average case vs approximation complexity. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 298–307. IEEE, 2003.

[Ang88] Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, 1988.

[BCK98] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 419–428, 1998.

[BDCVY17] Nader H Bshouty, Dana Drachsler-Cohen, Martin Vechev, and Eran Yahav. Learning disjunctions of predicates. In *Conference on Learning Theory*, pages 346–369. PMLR, 2017.

[BEHW87] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Occam's razor. *Information processing letters*, 24(6):377–380, 1987.

[BEK02] Nader H Bshouty, Nadav Eiron, and Eyal Kushilevitz. Pac learning with nasty noise. *Theoretical Computer Science*, 288(2):255–275, 2002.

[BF02] Nader H Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *Journal of Machine Learning Research*, 2(Feb):359–395, 2002.

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

[BHZ19] Nader H Bshouty and Catherine A Haddad-Zaknoon. Adaptive exact learning of decision trees from membership queries. In *Algorithmic Learning Theory*, pages 207–234. PMLR, 2019.

[Blu03] Avrim Blum. Open problems-learning a function of r relevant variables. *Lecture Notes in Computer Science*, 2777:731–733, 2003.

[BMOS05] Nader H Bshouty, Elchanan Mossel, Ryan O'Donnell, and Rocco A Servedio. Learning dnf from random walks. *Journal of Computer and System Sciences*, 71(3):250–265, 2005.

[BSBTD+16] Eli Ben-Sasson, Iddo Ben-Tov, Ivan Damgård, Yuval Ishai, and Noga Ron-Zewi. On public key encryption from noisy codewords. In *Public-Key Cryptography–PKC 2016*, pages 417–446. Springer, 2016.

[BWDSS20] Galit Bary-Weisberg, Amit Daniely, and Shai Shalev-Shwartz. Distribution free learning with local queries. In *Algorithmic Learning Theory*, pages 133–147. PMLR, 2020.

[CCG+20] Varun Chandrasekaran, Kamalika Chaudhuri, Irene Giacomelli, Somesh Jha, and Songbai Yan. Exploring connections between active learning and model extraction. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1309–1326, 2020.

[Cla11] David E Clark. Outsourcing lead optimization: the eye of the storm. *Drug discovery today*, 16(3-4):147–157, 2011.

[DAG06] Arkadiusz Z Dudek, Tomasz Arodz, and Jorge Gálvez. Computational methods in developing quantitative structure-activity relationships (qsar): a review. *Combinatorial chemistry & high throughput screening*, 9(3):213–228, 2006.

[DGRDR08] Kurt De Grave, Jan Ramon, and Luc De Raedt. Active learning for high throughput screening. In *International Conference on Discovery Science*, pages 185–196. Springer, 2008.

[DHH00] Dingzhu Du, Frank K Hwang, and Frank Hwang. *Combinatorial group testing and its applications*, volume 12. World Scientific, 2000.

[DKL09] Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 621–630, 2009.

[DKS18] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Learning geometric concepts with nasty noise. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1061–1073, 2018.

[Döt15] Nico Döttling. Low noise lpn: Kdm secure public key encryption and sample amplification. In *IACR International Workshop on Public Key Cryptography*, pages 604–626. Springer, 2015.

[Fel09] Vitaly Feldman. On the power of membership queries in agnostic learning. *The Journal of Machine Learning Research*, 10:163–182, 2009.

[FGKP06] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. New results for learning noisy parities and halfspaces. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 563–574. IEEE, 2006.

[FGKP09] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009.

[GL89] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32, 1989.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[Gol98] Oded Goldreich. *Modern cryptography, probabilistic proofs and pseudorandomness*, volume 17. Springer Science & Business Media, 1998.

[GRSY20] Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning. *Electronic Colloquium on Computational Complexity (ECCC)*, 27:58, 2020.

[GRV11] Elena Grigorescu, Lev Reyzin, and Santosh Vempala. On noise-tolerant learning of sparse parities and related problems. In *International Conference on Algorithmic Learning Theory*, pages 413–424. Springer, 2011.

[IKOS19] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptographic sensing. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 583–604, Cham, 2019. Springer International Publishing.

[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.

[JSMA19] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. Prada: protecting against dnn model stealing attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 512–527. IEEE, 2019.

[KKMS08] Adam Tauman Kalai, Adam R Klivans, Yishay Mansour, and Rocco A Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008.

[KLN+11] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[KM93] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

[KMAM18] Manish Kesarwani, Bhaskar Mukhoty, Vijay Arya, and Sameep Mehta. Model extraction warning in mlaas paradigm. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 371–380, 2018.

[KMV08] Adam Tauman Kalai, Yishay Mansour, and Elad Verbin. On agnostic boosting and parity learning. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 629–638, 2008.

[KS06] Jonathan Katz and Ji Sun Shin. Parallel and concurrent security of the hb and hb+ protocols. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 73–87. Springer, 2006.

[KSS94] Michael J Kearns, Robert E Schapire, and Linda M Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

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

[KWJ+04] Ross D King, Kenneth E Whelan, Ffion M Jones, Philip GK Reiser, Christopher H Bryant, Stephen H Muggleton, Douglas B Kell, and Stephen G Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427(6971):247–252, 2004.

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM (JACM)*, 40(3):607–620, 1993.

[LPP04] Gregory A Landrum, Julie E Penzotti, and Santosh Putta. Machine-learning models for combinatorial catalyst discovery. *Measurement Science and Technology*, 16(1):270, 2004.

[O'D14] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[OS07] Ryan O'Donnell and Rocco A Servedio. Learning monotone decision trees in polynomial time. *SIAM Journal on Computing*, 37(3):827–844, 2007.

[Pie12] Krzysztof Pietrzak. Cryptography from learning parity with noise. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 99–114. Springer, 2012.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

[Sch90] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[SKL17] Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3520–3532, Red Hook, NY, USA, 2017. Curran Associates Inc.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[TZJ+16] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618, 2016.

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

[YS16] Yu Yu and John Steinberger. Pseudorandom functions in almost constant depth from low-noise lpn. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, pages 154–183, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[YZ16] Yu Yu and Jiang Zhang. Cryptography with auxiliary input and trapdoor from constant-noise lpn. In *Annual International Cryptology Conference*, pages 214–243. Springer, 2016.

[ZFZS20] Jiaheng Zhang, Zhiyong Fang, Yupeng Zhang, and Dawn Song. Zero knowledge proofs for decision tree predictions and accuracy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 2039–2053, 2020.

Appendices

A More on Related Work

A.1 Cryptographic Sensing

In [IKOS19], the study of Cryptographic Sensing is initiated. Informally, Cryptographic Sensing explores the possibility of embedding cryptography into nature. This setting is modeled by an unknown object $x \in \mathcal{X}$, where $\mathcal{X} = \{0,1\}^n$ by default. Additionally, there is a probabilistic sensing algorithm Sen, which applies a certain sequence of measurement functions $f_i \in \mathcal{F}$, for some class of functions $\mathcal{F}$, to the unknown object x. Upon getting the results of the measurements $f_i(x)$, Sen attempts to decode x. Sen is required to work efficiently, and securely, in the sense that, without knowledge of the private randomness used by Sen, it should be computationally infeasible for any passive adversary who views the measurements to make sense of the measurements and decode x.

In [IKOS19], a duality between sensing and learning is noted. More specifically, the authors acknowledge that rather than dealing with an unknown object, one could formulate the problem in the context of computational learning theory by dealing with an unknown concept $c \in \mathcal{C}$ for some class of concepts $\mathcal{C}$. Here, there is a possibly active, randomized learning algorithm that is tasked with constructing a training set $\mathcal{S}$ such that given the labels $[\mathcal{S}, c(\mathcal{S})]$, it can efficiently learn some representation of c. This formulation is dual in the sense that before we were applying functions to an input, and now we are evaluating inputs to a function. Yet, the goal and setting remain largely the same.

An important initial motivation of the authors of [IKOS19] was to explore the possibility of public key cryptography "in the real world." Thus following a construction that required modular arithmetic, the authors set out to remove this hard-to-find-in-the-natural-world requirement. In this work, we focus on developing the concept of Covert Learning without considering the "real world" aspirations of Cryptographic Sensing. Instead, we explore the consequences of a dedicated Covert Learning definition.

A.2 PAC-verification

The work of [GRSY20] initiates the study of interactive proofs for machine learning, which directly influences our work. Goldwasser et al. outline the setting of PAC-verification as follows. There is a verifier and a prover who interact to run a verification protocol where the verifier outputs either "reject" or a hypothesis meant to model some class of concepts $\mathcal{D}$ over $\{0,1\}^n \times \{0,1\}$. Let $\mathcal{H}$ be a class of hypotheses and let $\mathcal{O}_V$ and $\mathcal{O}_P$ be oracles that the verifier and prover have access to, respectively. Generally, one can think of $\mathcal{D}$ as the class of distributions that implements a data source of uniformly random examples to some target function $f : \{0,1\}^n \to \{0,1\}$. The verifier's oracle, $\mathcal{O}_V$, will operate

as a realization of some $D \in \mathcal{D}$ and $\mathcal{O}_P$ will generally be an oracle that implements a data source acting as membership query access to the target function f of D. PAC-verification aims to construct interactive proofs with the property that, for any $D \in \mathcal{D}, \epsilon, \delta > 0$, the verifier either outputs "reject" or a hypothesis which is ϵ-close to the optimal hypothesis within $\mathcal{H}$ with respect to the loss function $\mathcal{L}_D(h) = \Pr_{(x,y)\sim D}[h(x) \neq y]$, with probability $1 - \delta$. We will refer to such a hypothesis as ϵ-good with respect to D. More formally, the interactive proof system will output a hypothesis h such that

$$\Pr\left[h \neq \text{reject} \wedge \mathcal{L}_D(h) \leq \inf_{h\in\mathcal{H}} \mathcal{L}_D(h) + \epsilon\right] \geq 1 - \delta$$

where the probability is over the random coins of the interactive protocol. In [GRSY20], the focus is on the agnostic formulation of PAC learning– PAC-verification under the realization assumption[6] is trivial in some cases. For example, the prover could simply send a claimed ϵ-good hypothesis h with respect to D, and the verifier would draw a sufficiently large set of random examples, and reject if the fraction of incorrectly predicted labels is larger than ϵ (more or less, depending on the desired failure probability δ, and applying the Chernoff Bound). The difficulty in the agnostic case stems from the fact that it is not trivial how to get the prover to convince the verifier that $\inf_{h\in\mathcal{H}} \mathcal{L}_D(h)$ is, say, 0.3. Therefore, a hypothesis h with $\mathcal{L}_D(h) = 0.31$ may be considered unacceptable in the realizable case, yet very good in the in the agnostic case. As noted in [GRSY20], distinguishing between these two cases may be extremely difficult if $\mathcal{H}$ is very large or complex.

B Variants of Definition 2.7 and Definition 3.3

B.1 Covert Learning Variants

The following definition implements the focus on concept-hiding. The concept-hiding version is specifically useful in cases where the concept is known to be from a fixed class, in which case the hypothesis-hiding guarantees maybe unmotivated (and awkward to define). These two definitions can also transformed to stricter forms of soundness, privacy (i.e. statistical, perfect).

Definition B.1. ***Concept-Hiding Learning.*** *We define Concept-Hiding learner identically to Covert Learning, except we modify* $\{\mathsf{ideal}_{Sim}\}$ *to the following:*

Let Sim be a p.p.t. algorithm, which takes as input ϵ, δ *and a set of random examples from the concept, and a length parameter which denotes the number of queries requested by the learner in the real interaction. We define* $\{\mathsf{ideal}_{Sim}\}$ *to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) selects* $\epsilon, \delta > 0$*, a hypothesis class* $\mathcal{H}_n \in \mathcal{C}_n$*, and a concept* $D_n \in \mathcal{D}_n$*.*
2. *A set of examples* $\mathcal{S}$ *is drawn from* D_n*.*
3. *A p.p.t. simulator Sim receives* $\mathcal{H}_n, \mathcal{S}, \ell$*, and "interacts" with the* $\mathcal{O}_{D_n}$ *and outputs the set queries and responses denoted as* $\mathsf{transcript}_{Sim(\mathcal{S},\ell)}$*. Here* ℓ *is the number of queries that the learner requests in the real interaction.*
4. *Output*

$$\left(\mathcal{H}_n, \epsilon, \delta, \mathsf{transcript}_{Sim(\mathcal{S},\ell)}\right)$$

[6] The realization assumption refers to the case that $\inf_{h\in\mathcal{H}} \mathcal{L}_D(h) = 0$.

We note that the focus on the concept-hiding guarantee can also be executed in the verifiable model.

B.2 Covert Verifiable Learning Variants

The next two definitions concern assumptions made on the nature of the random examples. In the first definition, we consider the case where all random examples are private always. This means that the learner has private examples available at execution time (to leverage for soundness), and also they are never revealed to the distiguisher. Thus, the privacy requirement is modified to require simulatability of *only* the membership queries. We begin by modifying the security experiments to the following (note that the main difference is in the output of each process, i.e. what is observed by the distinguisher):

Definition B.2. *Let $\mathcal{D}_n$ be a concept class, and let $\mathcal{C}_n$ be a collection of hypothesis classes. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, and a set of queries and the oracle's responses on those queries. We define $\{\mathsf{FPVreal}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}$ is drawn from D_n. $\mathcal{S}$ is given to the learner, along with $\mathcal{H}_n, \epsilon, \delta$, while the adversarial intermediary $\mathcal{I}$ is given ϵ, δ.*
3. *The learner begins to interact with the concept oracle $\mathcal{O}_{D_n}$ by requesting membership queries in order to agnostically learn $\mathcal{H}_n$. $\mathcal{I}$ sees the learner's queries and responses and is given the chance to modify the responses. At the end of the interaction, $\mathcal{I}$ outputs a string denoted by $\mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}$.*
4. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\right)$*

Definition B.3. *Let Sim be a p.p.t. algorithm, which takes as input two sets of random examples from the concept and a length parameter ℓ which signifies the number of queries requested by the learner in the real interaction. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, and a set of queries and oracle's responses. We define $\{\mathsf{FPVideal}_{Sim,\mathcal{I}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}$ is drawn from D_n.*
3. *The simulator is given $\epsilon, \delta, \mathcal{S}$, while an adversarial intermediary $\mathcal{I}$ is given ϵ, δ.*
4. *Sim begins to "interact" with the $\mathcal{O}_{D_n}$ by "requesting" membership queries. $\mathcal{I}$ "views" the queries and responses, and is given the chance to change the responses. The simulator outputs a string, which is denoted by $\mathsf{ideal}^{Sim}_{\mathcal{I}}$.*
5. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathsf{ideal}^{Sim}_{\mathcal{I}}\right)$*

Definition B.4. ***Covert Verifiable Learning with Fully Private Examples.*** *Let $\mathcal{C}_n$ be a collection of hypothesis classes, let $\mathcal{D}_n$ be a class of concepts, let $\mathcal{O}_{\mathcal{D}_n}$ be a class of oracles indexed by $D_n \in \mathcal{D}_n$, and let $\mathcal{L}$ be a loss function. An algorithm $\mathcal{A}$ is an $(m(n), \alpha)$-covert verifiable learning algorithm for $\mathcal{C}_n$, with respect to $\mathcal{D}_n$, $\mathcal{O}_{\mathcal{D}_n}$ and $\mathcal{L}$, if for every $\epsilon, \delta > 0$, the following are true.*

- *Completeness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, the randomized output of $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*
$$\Pr_h\left[\mathcal{L}(h) \leq \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon\right] \geq 1 - \delta$$
- *Soundness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, then for any adversarial intermediary $\mathcal{I}$ that corrupts queries or responses from $\mathcal{A}$ to $\mathcal{O}_{D_n}$, the random variable $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*
$$\Pr_h\left[\mathcal{L}(h) > \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon \;\middle|\; h \neq \text{reject}\right] < \delta$$
  We say that soundness is computational if $\mathcal{I}$ is p.p.t..
- *Privacy. For any adversarial intermediary $\mathcal{I}$, there exists a p.p.t. simulation algorithm Sim that satisfies:*
$$\left\{\mathsf{FPVreal}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\right\} \overset{c}{\approx} \left\{\mathsf{FPVideal}_{Sim,\mathcal{I}}\right\}$$

Finally, we give the definition for the Public Covert Verifiable Learning model (when the learner has no private queries to use to achieve soundness). We start by modifying the security experiments. The main difference is that here, the adversarial intermediary also gets a set of random examples on the concept—which it did not before.

Definition B.5. *Let $\mathcal{D}_n$ be a concept class, and let $\mathcal{C}_n$ be a collection of hypothesis classes. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, a set of random examples from the concept, and a set of queries and the oracle's responses on those queries. We define $\{\mathsf{PVreal}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\}$ to be the distribution generated by the following process.*

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}$ is drawn from D_n. $\mathcal{S}$ is given to the learner, along with $\mathcal{H}_n, \epsilon, \delta$, while the adversarial intermediary $\mathcal{I}$ is given $\epsilon, \delta, \mathcal{S}$.*
3. *The learner begins to interact with the concept oracle $\mathcal{O}_{D_n}$ by requesting membership queries in order to agnostically learn $\mathcal{H}_n$. $\mathcal{I}$ sees the learner's queries and responses and is given the chance to modify the responses. At the end of the interaction, $\mathcal{I}$ outputs a string denoted by $\mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}$.*
4. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{real}^{\mathcal{O}_{\mathcal{D}_n}}_{\mathcal{A},\mathcal{I}}\right)$*

Definition B.6. *Let Sim be a p.p.t. algorithm, which takes as input two sets of random examples from the concept and a length parameter ℓ which signifies the number of queries requested by the learner*

in the real interaction. Let $\mathcal{I}$ be a p.p.t adversarial intermediary algorithm, which takes as input ϵ, δ, a set of random examples from the concept, and a set of queries and oracle's responses. We define $\{\mathsf{PVideal}_{Sim,\mathcal{I}}\}$ to be the distribution generated by the following process.

1. *An adversary (a distinguisher) chooses a target hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, a concept $D_n \in \mathcal{D}_n$, and accuracy parameters $\epsilon, \delta > 0$.*
2. *A set of random examples $\mathcal{S}'$ is drawn from D_n.*
3. *The simulator is given $\epsilon, \delta, \mathcal{S}, \mathcal{S}'$, while an adversarial intermediary $\mathcal{I}$ is given $\epsilon, \delta, \mathcal{S}$. Here $\mathcal{S}$ is the same set of examples that was held be the learner in the real interaction.*
4. *Sim begins to "interact" with the $\mathcal{O}_{D_n}$ by "requesting" membership queries. $\mathcal{I}$ "views" the queries and responses, and is given the chance to change the responses. The simulator outputs a string, which is denoted by $\mathsf{ideal}_{\mathcal{I}}^{Sim}$.*
5. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{ideal}_{\mathcal{I}}^{Sim}\right)$*

Definition B.7. ***Public Covert Verifiable Learning.*** *Let $\mathcal{C}_n$ be a collection of hypothesis classes, let $\mathcal{D}_n$ be a class of concepts, let $\mathcal{O}_{\mathcal{D}_n}$ be a class of oracles indexed by $D_n \in \mathcal{D}_n$, and let $\mathcal{L}$ be a loss function. An algorithm $\mathcal{A}$ is an $(m(n), \alpha)$-covert verifiable learning algorithm for $\mathcal{C}_n$, with respect to $\mathcal{D}_n$, $\mathcal{O}_{\mathcal{D}_n}$ and $\mathcal{L}$, if for every $\epsilon, \delta > 0$, the following are true.*

- *Completeness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, the randomized output of $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*

$$\Pr_h\left[\mathcal{L}(h) \leq \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon\right] \geq 1 - \delta$$

- *Soundness. If for any distribution $D_n \in \mathcal{D}_n$, any hypothesis class $\mathcal{H}_n \in \mathcal{C}_n$, and where $\mathcal{S}$ is a set of size $m(n)$ of examples drawn i.i.d. from D_n, then for any adversarial intermediary $\mathcal{I}$—who has access to $\mathcal{S}$—that corrupts queries or responses from $\mathcal{A}$ to $\mathcal{O}_{D_n}$, the random variable $h = \mathcal{A}^{\mathcal{O}_{D_n}}(\mathcal{H}_n, \epsilon, \delta, \mathcal{S})$ satisfies*

$$\Pr_h\left[\mathcal{L}(h) > \alpha \cdot \mathcal{L}(\mathcal{H}_n) + \epsilon \;\middle|\; h \neq \text{reject}\right] < \delta$$

  We say that soundness is computational if $\mathcal{I}$ is p.p.t..

- *Privacy. For any adversarial intermediary $\mathcal{I}$, there exists a p.p.t. simulation algorithm Sim that satisfies:*

$$\left\{\mathsf{PVreal}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\right\} \overset{c}{\approx} \left\{\mathsf{PVideal}_{Sim,\mathcal{I}}\right\}$$

  We stipulate that each of the sets of random examples given to the simulator are of size $m(n)$.

C Frequently Used Concepts and Lemmas

Definition C.1. ***Computational Indistinguishability.*** *Let $\{X_n\}, \{Y_n\}$ be sequences of distributions with X_n, Y_n ranging over $\{0,1\}^{m(n)}$ for some $m(n) = n^{O(1)}$. $\{X_n\}$ and $\{Y_n\}$ are computationally indistinguishable if for every polynomial time algorithm A and sufficiently large n,*

$$|\Pr[A(1^n, X_n) = 1] - \Pr[A(1^n, Y_n) = 1]| \leq negl(n)$$

Often, n is clear from the context, so the subscript is omitted.

Definition C.2. ***Agnostic PAC Learning.*** *We say that a hypothesis class $\mathcal{H}$ is agnostically PAC-learnable with respect to a distribution class $\mathcal{D}$ over $\mathcal{X}, \mathcal{Y}$ if there exists and algorithm $\mathcal{A}$ and a function $m : [0,1]^2 \to \mathbb{N}$ such that for any $\epsilon, \delta > 0$, and any distribution $D \in \mathcal{D}$, if $\mathcal{A}$ receives as an input a list of $m(\epsilon, \delta)$ samples from D, then $\mathcal{A}$ outputs a function h such that*

$$\Pr[\mathcal{L}_D(h) \leq \mathcal{L}_D(\mathcal{H}) + \epsilon] \geq 1 - \delta$$

Definition C.3. ***α-PAC Verifiability.*** *[GRSY20] Let $\mathcal{H}$ be a class of hypotheses, let $\mathcal{D}$ be some family of distributions over $\mathcal{X} \times \{0,1\}$. We say that $\mathcal{H}$ is PAC verifiable with respect to $\mathcal{D}$ using oracles $\mathcal{O}_V, \mathcal{O}_P$ if there exists a pair of algorithms (P, V) that satisfy the following for every $\epsilon, \delta > 0$,*

- *Completeness: For any distribution $D \in \mathcal{D}$, the random variable $h = [V^{\mathcal{O}_\mathcal{V}}(\epsilon, \delta), P^{\mathcal{O}_\mathcal{P}}(\epsilon, \delta)]$ satisfies*
$$\Pr[h \neq \text{reject} \wedge \mathcal{L}_D(h) \leq \alpha \cdot \mathcal{L}_D(\mathcal{H}) + \epsilon] \geq 1 - \delta$$
- *Soundness: For any distribution $D \in \mathcal{D}$, and any possibly unbounded prover P', the random variable $h = [V^{\mathcal{O}_\mathcal{V}}(\epsilon, \delta), P'^{\mathcal{O}_\mathcal{P}}(\epsilon, \delta)]$ satisfies*
$$\Pr[h \neq \text{reject} \wedge \mathcal{L}_D(h) > \alpha \cdot \mathcal{L}_D(\mathcal{H}) + \epsilon] < \delta$$

Definition C.4. ***Unpredictable Distributions.*** *Let $\mathcal{F}_n$ be a class of n-bit boolean functions. Let $\mathcal{D}$ be a distribution over $\mathcal{F}$. Let $\mathcal{P}$ be a distribution over $\{0,1\}^n$. We say that $\mathcal{F}_n$ is ϵ-predictable with respect to $\mathcal{D}$ and $\mathcal{P}$ if there exists a polynomial time algorithm M taking example set $(\mathcal{S}, f(\mathcal{S}))$ and a test input x with $|\mathcal{S}| = poly(n)$, such that for infinitely many n,*

$$\Pr_{f \sim \mathcal{D}, \mathcal{S} \sim \mathcal{P}^n, x \sim \mathcal{P}} \Big[M(\mathcal{S}, f(\mathcal{S}), x) = f(x) \Big] \geq 1 - \epsilon$$

If there exists some fixed polynomial $q(n)$ such that $\mathcal{F}_n$ is $(\frac{1}{2} - \frac{1}{q(n)})$-predictable with respect to $\mathcal{D}, \mathcal{P}$, then we say that $\mathcal{F}_n$ is weakly predictable with respect to $\mathcal{D}, \mathcal{P}$. If for any polynomial $q(n)$, $\mathcal{F}_n$ is $(\frac{1}{q(n)})$-predictable with respect to $\mathcal{D}, \mathcal{P}$, then we say that $\mathcal{F}_n$ is strongly predictable with respect to $\mathcal{D}, \mathcal{P}$.

Lemma C.5. ***Piling-up lemma.*** *For $\mu \in [0, \frac{1}{2}]$, and random variables $E_1 \cdots E_m$ that are i.i.d. from β_μ,*

$$\Pr\left[\bigoplus_{i=1}^{m} E_i = 0 \right] = \frac{1}{2} + \frac{1}{2}(1 - 2\mu)^m$$

Lemma C.6. ***Chernoff Bound.*** *For any $n \in \mathbb{N}$, let $X_1 \cdots X_n$ be i.i.d. from β_μ, and let $\bar{X} = \sum_{i=1}^{n} X_i$. Then for any $\epsilon \geq 0$,*

$$\Pr[\bar{X} \geq (1 + \epsilon)\mu n] \leq \exp\left(- \frac{\epsilon^2 \mu n}{2 + \epsilon} \right)$$

$$\Pr[\bar{X} \leq (1 - \epsilon)\mu n] \leq \exp\left(- \frac{\epsilon^2 \mu n}{2} \right)$$

Lemma C.7. ***Hoeffding Bound.*** *Let X_i be independent random variables over the intervals $[a_i, b_i]$, and let $X = \sum_i^n X_i$. Then for $\delta \geq 0$,*

$$\Pr\left[\left|\mathbb{E}[X] - X\right| \geq \delta\right] \leq 2exp\left(\frac{-2\delta^2}{\sum_i^n (a_i - b_i)^2}\right)$$

Lemma C.8. ***Chebyshev Inequality.*** *Let X be a random variable and $\delta > 0$. Then*

$$\Pr\left[\left|X - \mathbb{E}[X]\right| \geq \delta\right] \leq \frac{Var(X)}{\delta^2}$$

Lemma C.9. ***Pairwise Independent Sampling using Chebyshev.*** *Let $X_1, X_2, \cdots X_n$ be pairwise independent random variables with same expectation μ and the same variance σ^2. The for every $\epsilon > 0$,*

$$\Pr\left[\left|\frac{\sum_{i=1}^n X_i}{n} - \mu\right| \geq \epsilon\right] \leq \frac{\sigma^2}{\epsilon^2 n}$$

Lemma C.10. ***Union Bound.*** *Let $E_1 \cdots E_n$ be any (not necessarily independent) events such that $\Pr[E_i] \geq 1 - \epsilon_i$ for every $i \in [n]$. Then,*

$$\Pr[E_1 \wedge E_2 \cdots \wedge E_n] \geq 1 - \left(\sum_{i=1}^n \epsilon_i\right)$$

Lemma C.11. ***Sampling Bound.*** *For any $n \in \mathbb{N}$, let $X_1 \cdots X_n$ be i.i.d. from β_μ, and let $\bar{X}_n = \frac{1}{n}\sum_{i=1}^n X_i$. Then for $n \geq \frac{2+\epsilon}{\epsilon^2} ln(\frac{2}{\delta})$,*

$$\Pr\left[\left|\bar{X}_n - \mu\right| \leq \epsilon\right] \geq 1 - \delta$$

Lemma C.12. ***Markov Inequality.*** *Let X be a random variable with expectation μ. Then,*

$$\Pr[X \geq a] \leq \mu/a$$

D Fourier Analysis

We work with $f : \mathbb{F}_2^n \to \mathbb{R}$. In particular, we will care about functions $f : \mathbb{F}_2^n \to \{-1, 1\}$.

Definition D.1. *For $S \subseteq [n]$, we define $\chi_S : \mathbb{F}_2^n \to \mathbb{R}$ by*

$$\chi_S(x) = (-1)^{\sum_{i \in S} x_i}$$

Fact D.2. $\chi_S(x + y) = \chi_S(x)\chi_S(y)$

Definition D.3. *The Fourier expansion of* $f : \mathbb{F}_2^n \to \mathbb{R}$ *is*

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S(x)$$

where $\hat{f}(S)$ *is the Fourier coefficient on* S*.*

Sometimes, it will make sense to correspond subsets $S \subseteq [n]$ with vectors in $\mathbb{F}_2^n$ (we do this in the proof of Theorem 2.26). The natural correspondence of a subset S to a vector x is that $x_i = 1$ if $i \in S$ and 0 otherwise. When this is the case, we have:

Definition D.4. *For* $k \subseteq \mathbb{F}_2^n$*, we define* $\chi_k : \mathbb{F}_2^n \to \mathbb{R}$ *by* $\chi_k(x) = (-1)^{\langle k,x \rangle}$

Fact D.5. $\chi_{x+y} = \chi_x \chi_y$

Definition D.6. *The Fourier expansion of* $f : \mathbb{F}_2^n \to \mathbb{R}$ *is*

$$f(x) = \sum_{k \in \mathbb{F}_2^n} \hat{f}(k)\chi_k(x)$$

Theorem D.7. ***Parseval's Theorem.*** *For any* $f : \{-1,1\}^n \to \mathbb{R}$*,*

$$\langle f, f \rangle = \mathop{\mathbb{E}}_x[f(x)^2] = \sum_{S \subseteq [n]} \hat{f}(S)^2$$

and in the case of $f : \{-1,1\}^n \to \{-1,1\}$*,*

$$\sum_{S \subseteq [n]} \hat{f}(S)^2 = 1$$

Theorem D.8. ***Plancherel's Theorem.*** *for any* $f, g : \{-1,1\}^n \to \mathbb{R}$*,*

$$\langle f, g \rangle = \mathop{\mathbb{E}}_x[f(x)g(x)] = \sum_{S \subseteq [n]} \hat{f}(S)\hat{g}(S)$$

E Proofs of CVLDT Guarantees

We prove completeness:

Proposition E.1. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})$*−*$\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLDT *satisfies the completeness guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s \leq poly(n)$*, with respect to* $\mathcal{D}_{\mathcal{F}_n}$*,* $\mathcal{O}_{\mathcal{F}_n}$*, and* $\mathcal{L}_D$*, and where* $\epsilon \geq 1/poly(n)$*, and* $\delta \geq exp(-n)$

Proof. The interactive aspect of CVLDT is fully contained inside CVLF. Completeness clearly follows from the completeness of CVLF and identical arguments to that of the proof of completeness for CLDT. □

Similarly, soundness:

Proposition E.2. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLDT *satisfies the computational soundness guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s \leq poly(n)$*, with respect to* $\mathcal{D}_{\mathcal{F}_n}$*,* $\mathcal{O}_{\mathcal{F}_n}$*, and* $\mathcal{L}_D$*.*

Proof. The interactive aspect of CVLDT is fully contained inside CVLF. Therefore, soundness follows from soundness of CVLF, and the completeness of CVLDT. Note that we choose the soundness parameter of the CVLF subroutine to be $\delta/2$, and select an appropriate constant on $O(log(1/\delta))$ for the number of iterations. This

□

Finally, privacy:

Proposition E.3. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption,* CVLDT *satisfies the privacy guarantee of Covert Verifiable Learning for* $\mathcal{C}_{\mathsf{DT}_{n,s}}$ *for* $s \leq poly(n)$*, with respect to* $\mathcal{D}_{\mathcal{F}_n}$*,* $\mathcal{O}_{\mathcal{F}_n}$*, and* $\mathcal{L}_D$*.*

Proof. As above, the interactive aspect of CVLDT is fully contained inside CVLF. Therefore, privacy follows from privacy of CVLF, and the completeness of CVLDT. Note that we only need to change the simulator and the reduction of Proposition 3.6 to handle the proper amount of queries. □