Covert Learning: How to Learn with an Untrusted Intermediary*

Ran Canetti and Ari Karchmer

Boston University, Boston MA 02215, USA
canetti@bu.edu, arika@bu.edu

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 secure outsourcing of automated scientific discovery in drug design and molecular biology. It also uncovers limitations of current techniques for defending against model extraction attacks.

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.

* Supported by the DARPA SIEVE program, Agreement Nos. HR00112020020 and HR00112020021. A full version of this work appears in [CK21].

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 learns nothing about the molecular relationship.

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 in Section 1.3 and in the full version [CK21] of this paper.

1.1 Our Contributions

We define and construct learning algorithms that satisfy the above requirements. We first present our definitions, then state our results, and finally overview our techniques.

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.

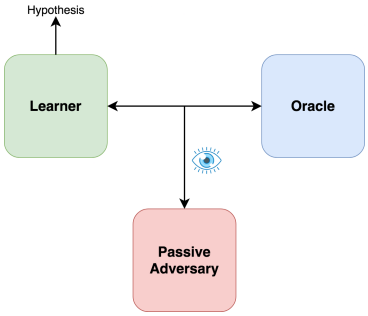

Fig. 1: A Covert Learning scenario. The learner interacts with the concept by making queries to an oracle that implements access to the concept at arbitrary points. Meanwhile, an adversary attempts to deduce information about the learner's hypothesis or about the concept itself, given a view: the set 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*[1]. Let $\mathcal{H}$ denote a *hypothesis class*, which is a subset

[1] Alternatively, one may think of a concept as a tuple consisting of a distribution $D_{\mathcal{X}}$ over the input domain $\mathcal{X}$ and a target function $f : \mathcal{X} \to \{0,1\}$ which labels inputs. However, the notion described above (and used in the rest of this paper) is more general, as a joint distribution allows concepts which are *probabilistic*.

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* (with respect 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. ***Covert Learning*** *(informal version of Definition 9). 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 *tolerance testing*[2] is to decide if a class of hypotheses contains a good approximator

[2] In tolerance testing [PRR06], 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.

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).

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 ground truth examples 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.*

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 (with respect to 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.*

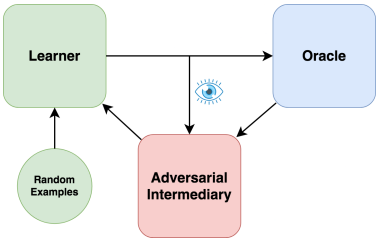

Fig. 2: The "intermediate model" Covert Verifiable Learning scenario. A learner, given a set of random examples of a concept, accesses supplementary data using an oracle in the presence of an adversarial intermediary. While attempting to deduce information about the concept or the learner's hypothesis,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

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 2. ***Covert Verifiable Learning*** *(informal version of Definition 25) 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.

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 $D_{\mathsf{LPN}} = \{D^k_{\mathsf{LPN}} \,|k \in \{0,1\}^n\}$. 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. *(Informal version of Theorem 7) Assuming hardness of the low-noise LPN assumption, there is a covert learning algorithm for the collection* $\mathcal{C} = \{\mathcal{H}_T \mid T \subseteq [n]\}$*, w.r.t. the concept class* D_{LPN} *and loss function* $\mathcal{L}_D$*.*

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 $\tau \geq 1/\text{poly}(n)$ is a given parameter and $\chi_k(x) = (-1)^{\langle k,x \rangle}$. We denote by $\hat{f}_b^{\geq \tau}$ 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 $\mathcal{H}_T^b = \{\chi_k \mid k_i = 0 \implies i \notin T, |k| \leq b\}$, where $T \subseteq [n]$, and a loss function $\mathcal{L}^{\tau,b} : \mathcal{P}([n]) \to [0,1]$ given by

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

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

Theorem 2. *(Informal version of Theorem 9) 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* $\mathcal{C} = \{\mathcal{H}_T^b \mid T \subseteq [n]\}$*, with respect to the concept* $D_{\mathcal{F}}$ *and the loss function* $\mathcal{L}^{\tau,b}$ *and for* $b \leq O(\log\, n)$*,* $\tau \geq 1/\text{poly}(n)$*.*

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 3. *(Informal version of Theorem 10) 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* $\text{poly}(n)$ *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 4. *(Informal version of Theorem 11) 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* $b \leq O(\log n)$*,* $\tau \geq 1/\text{poly}(n)$*.*

Theorem 5. *(Informal version of Theorem 12) 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* $\text{poly}(n)$ *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 6. *(Informal version of Theorem 13) 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* $s \leq \text{poly}(n)$*,* $b \leq O(\log n)$*, and* $\tau \geq 1/\text{poly}(n)$*.*

In particular, the result of Theorem 13 gives the first verifiable PAC learning protocol without any private examples, even in the model of [GRSY20] which does not consider privacy.

Due to space constraints, we have omitted all proofs and refer the reader to [CK21]. Furthermore, we have removed two results, namely, a key exchange algorithm that arises from Theorem 7 and a statistically sound and perfectly private covert learning algorithm for the "junta problem" in the fully private examples model. These results may also be found in the [CK21].

Algorithmic Ideas. We give high level descriptions of the algorithmic techniques. Formal overviews precede the constructions in the following sections.

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 solving the variant of the LPN problem to a constructing putative distinguisher between a real execution and a simulated execution.

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.

1.3 Related Work

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

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.

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. 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.

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

Definition 4. *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 5. *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 6. *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. We define two distributions, $\{\mathsf{real}_{\mathcal{A}}^{\mathcal{O}_{\mathcal{D}_n}}\}$ and $\{\mathsf{ideal}_{Sim}\}$ as follows.

Definition 7. *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 8. *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 9. ***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. 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 $\mathcal{C}$.

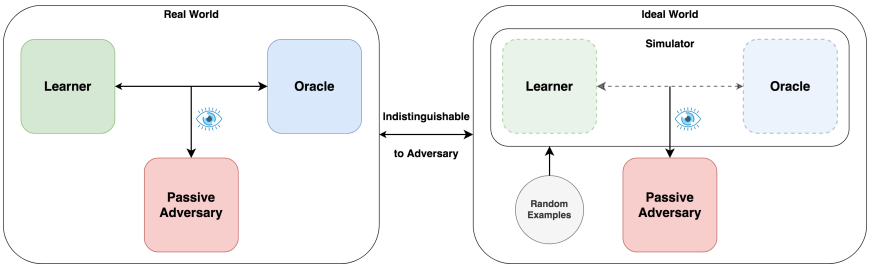

Fig. 3: Privacy of Covert Learning. The "real world," where the adversary views the learner access the oracle, should be indistinguishable from the "ideal world," where the adversary interacts with a simulator that simulates the learner accessing the oracle. The adversary gets to choose the concept which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

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 (given access to random examples). 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.

2.3 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 10. ***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 1. 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 $(O(n), \mathrm{poly}(n))$-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.

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 11. *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 12. *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 13. *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 14. *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 15. *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.

The Construction

Overview. We will refer to $\mathcal{D}_{\mathsf{LPN}}^{\frac{1}{\sqrt{n}},n}$ as $\mathcal{D}_{\mathsf{LPN}}^{\mathsf{Low}}$. We construct Covert Learning for $\mathcal{C}_{\mathsf{PARITY},n}$ with respect to $\mathcal{D}_{\mathsf{LPN}}^{\mathsf{Low}}$, $\mathcal{O}_{\mathcal{D}_{\mathsf{LPN}}^{\mathsf{Low}}}$, and $\mathcal{L}_D$, and where learning is considered when the concept is drawn from $\mathcal{M}_{\mathsf{LPN}}^n$. The Covert Learning algorithm begins by requesting "masked queries" from $\mathcal{O}_{D_k}$. For $x_1, \cdots, x_n \xleftarrow{\$} \beta_{1/2}^n$, 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 $e, s \xleftarrow{\$} \beta_{1/\sqrt{n}}^n$. 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 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,

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

Hence, 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 $r(x) = \langle (r_1, \cdots, r_n), x \rangle$.

Theorem 7. *Assuming* $\mathsf{DLPN}_{\mu,n}$ *is* $(O(n), \mathrm{poly}(n))$*-hard,* $\mathcal{C}_{\mathsf{PARITY},n}$ *is* $(\mathrm{poly}(n), 1)$*-covertly learnable with respect to* $\mathcal{D}_{\mathsf{LPN}}^{\mathsf{Low}}$, $\mathcal{O}_{\mathcal{D}_{\mathsf{LPN}}^{\mathsf{Low}}}$, *and* $\mathcal{L}_D$, *and where the concept is drawn according to* $\mathcal{M}_{\mathsf{LPN}}^n$.

The algorithm (called CLP and presented in [CK21]) 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.

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].

Our Task. We consider the following natural class of concepts.

Definition 16. *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 17. *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 18. *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 19. *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 20. *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*

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

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).

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 8. ***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 $|\hat{f}(S)| \geq 1/\text{poly}(n)$ 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 21. *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 $\Omega(\tau n^{-c})$ (for some small constant $c > 0$) 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 $\Omega(\tau n^{-c})$—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 interacts 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 distinguishes between the simulated interaction and the real interaction (where the requested queries are pseudorandom), then an adversary distinguishes the pseudorandom masked queries from random queries.

Theorem 9. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption and for sufficiently large* n*, there exists a proper* $(\mathrm{poly}(n), 1)$*-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* $b \leq O(\log\, n), \tau \geq 1/\mathrm{poly}(n)$*,* $\delta \geq \exp(-n)$*.*

The algorithm, called CLF, is presented in [CK21]. An important ingredient in the proof is the following hardness lemma due to [YZ16].

Lemma 1. ***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* $\lambda = \Theta(log^2\ n)$*,* $q = \mathrm{poly}(n)$*, 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* $e \sim \beta_\mu^q$*,* $u \sim \{0,1\}^q$*. We will call the task of distinguishing the above distributions the decisional squared-log entropy LPN problem.*

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].

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].

Definition 22. *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 algorithm is presented as CLDT in [CK21].

Theorem 10. *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* $s = \mathrm{poly}(n)$ *is* $(\mathrm{poly}(n), 4)$*- covertly learnable, with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}_D$, *and where* $\epsilon \geq 1/\mathrm{poly}(n)$, *and* $\delta \geq \exp(-n)$.

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 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.

[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.

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 23. *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 $\{\mathsf{Vreal}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\}$ 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 $\mathsf{real}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}$.*
4. *Output $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{real}_{\mathcal{A},\mathcal{I}}^{\mathcal{O}_{\mathcal{D}_n}}\right)$*

Definition 24. *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 $\left(\mathcal{H}_n, \epsilon, \delta, \mathcal{S}, \mathsf{ideal}_{\mathcal{I}}^{Sim}\right)$*

Definition 25. ***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:*

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

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

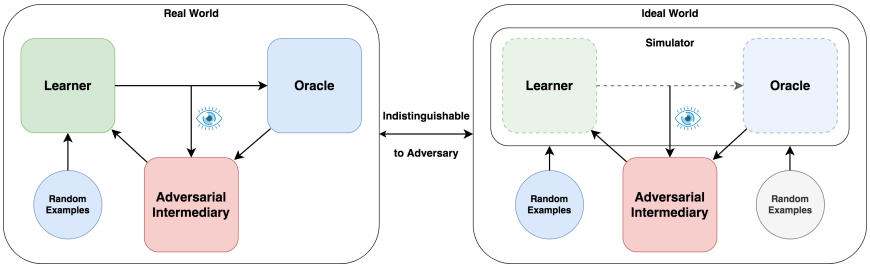

Fig. 4: Privacy of Covert Verifiable Learning. The "real world," where the AI interacts with the learner and oracle, should be indistinguishable from the "ideal world," where the AI interacts with a simulator that plays both roles of learner and oracle. Importantly, the simulator works without knowledge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

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 $\mathcal{C}$.

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 significantly 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 the full version [CK21] of this paper, 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.

Theorem 11. *Under the* $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ *assumption, there exists a* $(\mathrm{poly}(n), 1)$*-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* $\tau \geq 1/\mathrm{poly}(n)$.

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 subroutine of CLF in CVLF, and this alone suffices. The resulting algorithm, called CVLDT, is presented [CK21].

All three guarantees of Covert Verifiable Learning intuitively hold for CLDT, as all the communication of CVLDT is contained in CVLDT.

Theorem 12. *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* $s \leq \mathrm{poly}(n)$ *is* $(\mathrm{poly}(n), 4)$*- covertly verifiably learnable, with respect to* $\mathcal{D}_{\mathcal{F}_n}$, $\mathcal{O}_{\mathcal{F}_n}$, *and* $\mathcal{L}_D$, *and where* $\epsilon \geq 1/\mathrm{poly}(n)$, *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.

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 in [CK21]). 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.

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

Definition 26. *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.*

Theorem 13. *Under the $(2^{\omega(n^{\frac{1}{2}})}, 2^{\omega(n^{\frac{1}{2}})})-\mathsf{SLPN}_{\mu,n}$ assumption, there exists a proper $(\mathrm{poly}(n), 1)$-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 $\delta \geq \exp(-n), b \leq O(\log n)$, $1/\mathrm{poly}(n) \leq \tau \leq 1/2(2s+1)^2$, and the DNF size $s \leq \mathrm{poly}(n)$.*

The algorithm is presented as PCVLF in [CK21].

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 the proof of Theorem 11 to appropriately reflect the changes we made in the test case of the algorithm.

Acknowledgements

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

References

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.

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.

CK21. Ran Canetti and Ari Karchmer. Covert learning: How to learn with an untrusted intermediary. 2021. `https://ia.cr/2021/764`.

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.

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

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.

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.

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.

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.

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.

PRR06. Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.

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.