# The Power of Interaction in Local Differential Privacy

Matthew Joseph[*]

January 28, 2020

**Abstract**

A communication protocol is locally differentially private if the communicated messages reveal little about the data of participants. We survey two papers that show interactive local privacy can be much more powerful than noninteractive local privacy. First, Kasiviswanathan, Lee, Nissim, Raskhodnikova, and Smith [19] prove a general equivalence between locally private learning and statistical query (SQ) learning. They use this equivalence to give a new concept class, masked parity, which is much harder to learn under noninteractive local privacy than interactive local privacy. However, a drawback of this result is that masked parity is a concept class constructed exclusively for separation. Daniely and Feldman [7] extend this line of work by connecting a concept class' noninteractive SQ learnability with its margin complexity. This is useful because past work has shown that the margin complexity of decision lists is exponential in the dimension of the example space. In conjunction with the local-SQ equivalence given by Kasiviswanathan et al. [19], this yields another separation: noninteractive locally private decision list learning is much harder than interactive locally private decision list learning.

[*]University of Pennsylvania CIS Department. This document submitted to fulfill the WPE II Exam requirement.

## 1 Introduction

Differential privacy [11] is a formal property of algorithms that process data. A randomized algorithm that maps databases to output distributions is differentially private if similar inputs yield similar output distributions. This clean definition, along with robust mathematical guarantees, has driven a now-large body of academic work [12, 24] as well as real-world industrial [1, 3, 8, 15] and governmental [14] deployments.

Nearly all work on differential privacy studies one of two models. In *central differential privacy* [11], the algorithm enjoys "central" access to the raw, unnoised database and may perform arbitrary computations on this data. The only constraint is that the algorithm's eventual public output must satisfy differential privacy.

We focus on the stricter *local* model of differential privacy. Local differential privacy [11, 2, 19] places additional restrictions on the data processing pipeline. Locally differentially private algorithms no longer receive trusted central access to raw data. Data instead remains distributed on users' devices, and learning takes place through a public communication with users. Users "locally" add randomness to their communications to introduce uncertainty about their inputs, and the transcript of communication is now the differentially private object. Unlike the central model, the local model folds privacy into the learning process itself.

Relative to conventional (central) differential privacy, local differential privacy has both strengths and weaknesses. Its foremost strength is the privacy it guarantees to users. Because users add their own randomness to their communications, they do not require a trusted central analyst or even secure communication channels. Unfortunately, strong privacy cuts both ways: more noise typically means more error. For example, an $\varepsilon$-centrally private algorithm can compute a sum of $n$ bits to additive error $O\left(\frac{1}{\varepsilon}\right)$, but for constant $\epsilon$, no $\varepsilon$-locally private algorithm can achieve $o(\sqrt{n})$ error [6].

Nonetheless, strong privacy guarantees have made local differential privacy a promising model of private computation. In fact, most of the aforementioned industrial applications employ the local model. It is therefore important to understand what the local model can and cannot do. In this survey, we focus on the role of *interactivity* in local differential privacy.

Interactivity is an important protocol property for two reasons. First, as demonstrated by communication problems like pointer-chasing [21], it may be possible to trade more rounds of communication for much less communication overall. This suggests that interactive locally private protocols may obtain similar savings, for example in sample complexity.

However, interaction also incurs many forms of overhead. Because users must now receive as well as send messages, interaction introduces scalability constraints; because interaction requires multiple rounds, running time must grow as well; and because users now coordinate interaction with one another, network latency and liveness constraints become nontrivial problems [23]. Motivated by these complicating factors, we would like to allocate the resources for interaction precisely to the problems where they are necessary. The focus of this survey is to cover work by Kasiviswanathan et al. [19] and Daniely and Feldman [7] identifying such problems.

First, we present work by Kasiviswanathan et al. [19] showing that locally private learning is — in a formal sense that we will define later — equivalent to statistical query (SQ) learning in an interaction-preserving way (Section 3). Using this result, it only remains to find problems that are hard (in terms of necessary and sufficient sample complexity) for noninteractive SQ learners but easy for interactive SQ learners. In Section 4, we show this is the case for the concept class of masked parity.

Next, we turn to more general results by Daniely and Feldman [7]. Their connection between margin complexity and SQ learnability appears in Section 5. In conjunction with past work on (interactive) SQ learning decision lists and the margin complexity of decision lists, this leads to our second separation: under local privacy, decision list learning is much easier for interactive protocols than noninteractive ones (Section 6). Related work and preliminaries appear in Sections 1.1 and 2, respectively.

## 1.1 Related Work

Local differential privacy's history in part predates that of differential privacy. Although it did not define local privacy in general, Warner's randomized response [26] is an early example of a locally differentially private protocol. Dwork, McSherry, Nissim, and Smith [11], Beimel, Nissim, and Omra [2], and Kasiviswanathan et al. [19] all helped to formalize the idea behind local differential privacy into the definition given in Section 2.

We briefly describe other work studying interaction in local privacy. Duchi et al. [10] first distinguished between sequential and full interactivity. They also give information-theoretic lower bounds for sequentially interactive protocols. However, these results have only shown that many noninteractive protocols are actually optimal even with regard to the class of sequentially interactive protocols. In particular, Duchi et al. [10] do not give any separations between noninteractive and (sequentially) interactive local privacy.

We now turn to work on fully interactive local privacy. Duchi and Rogers [9] showed how to extend the lower bounds of Duchi et al. [10] to fully interactive protocols but, again, obtained no separations between different interactive models. Joseph, Mao, Neel, and Roth [16] proved the first separation between approximate fully interactive local privacy and central privacy for the problem of simple hypothesis testing They also showed how to convert fully interactive locally private protocols to sequential ones and gave a polynomial sample complexity separation for sequential and full interaction. Follow-up work [17] extended this to an exponential separation and also gave a general exponential separation between $k$ and $k+1$-round sequential protocols for any $k \in \mathbb{N}$, albeit for a highly specific pointer-chasing problem.

In the context of the preceding work delineating noninteraction, sequential interaction, and full interaction, we emphasize that this paper focuses on the relative power of noninteraction and sequential interaction. Accordingly, we shorthand these as noninteraction and interaction.

The work of Smith et al. [23] is close to, but nonetheless different from, the focus of this survey. Smith et al. [23] show that certain protocols based on neighborhood oracles must use interaction to solve certain optimization problems in a number of samples polynomial in the problem parameters. In particular, this separation extends to neighborhood oracle-based locally private protocols as well. However, this result is qualitatively different from the results we study. This is because the neighborhood oracle, not local privacy, is the main constraint driving the result. The separation holds because each user only has information about an infinitesimal region around their point. This need not be true in general.

Finally, we briefly discuss related work on statistical query (SQ) learning. As noted by Kasiviswanathan et al. [19], Blum, Furst, Jackson, Kearns, Mansour, and Rudich [4] showed that noninteractive and interactive distribution-dependent weak SQ learners are equivalent. This explains why the separations we cover do not follow from the work of Blum et al. [4]: Kasiviswanathan et al. [19] study distribution-dependent *strong* SQ learners, while Daniely and Feldman [7] study distribution-*independent* weak SQ learners.

## 2 Preliminaries for Local Privacy, SQ Learning, and PAC Learning

We start with basic definitions for local differential privacy and its different models of interaction before defining the SQ learning and PAC learning frameworks. We will define both the local and SQ models in terms of transcripts and protocols, but the exact model in question at any given time should be clear from context (or explicitly specified). In both models, there is a (possibly unknown) distribution $\mathcal{D}$ over unlabelled examples, and a labeling function denoted here by $f$. Both local and SQ learners will have restricted forms of access to the behavior of $f$ on examples drawn from $\mathcal{D}$.

## 2.1 Differential Privacy

A randomized algorithm $\mathcal{A}$ satisfies central differential privacy if it maps raw databases to outcomes such that the distribution over outcomes is relatively insensitive to small changes in the database. This insensitivity, which hides the presence or absence of any one user, provides the privacy guarantee.

**Definition 1** (Central differential privacy [11])**.** *Given data domain $X$ and two databases $D, D' \in X^n$ for some $n \in \mathbb{N}$, $D$ and $D'$ are* neighbors *if they differ in $\leq 1$ element. Given algorithm $\mathcal{A} \colon X^n \to Y$, $\mathcal{A}$ is* $(\varepsilon, \delta)$-differentially private *if for all subsets $S \subset Y$,*

$$\mathbb{P}_{\mathcal{A}}\left[\mathcal{A}(D) \in S\right] \leq e^{\varepsilon}\mathbb{P}_{\mathcal{A}}\left[\mathcal{A}(D') \in S\right] + \delta.$$

For this work, it is important to note that centrally private algorithms enjoy trusted (central) access to the entire raw database. In particular, they may perform arbitrary computations on raw data before releasing a private output.

A *locally differentially private* algorithm satisfies a still more restrictive privacy guarantee[1]. A locally private algorithm is a public interaction between users, each of whom privately holds a single data element. To ensure privacy, users only contribute to this interaction through *randomizers.*

**Definition 2.** *An* $(\varepsilon, \delta)$-randomizer *$R$ is an $(\varepsilon, \delta)$-differentially private function taking a single data point as input. Here, a user's input data will typically consist of a labeled point $(x, f(x)) \in X \times \{-1, 1\}$.*

Because communication occurs only through randomizers, the overall record of public interaction is private. We more formally study this interaction in terms of its *transcript.*

**Definition 3.** *A* transcript *$\pi$ is a vector of 4-tuples $(R_t, \varepsilon_t, \delta_t, y_t)$ respectively indicating the randomizer, privacy parameters, and output produced at each time $t$.*

We can then view a locally private protocol as a coordinating mechanism that takes a transcript and selects a randomizer for the next user.

**Definition 4.** *Let $S_\pi$ denote the collection of transcripts and $S_R$ the collection of randomizers. Then a* protocol *$\mathcal{A}$ is a function $\mathcal{A} \colon S_\pi \to S_R \cup \{\perp\}$ mapping transcripts to randomizers or halting ($\perp$).*

A locally private protocol generally includes some post-processing of the transcript to generate some final output. Since this post-processing is still a function of the transcript, we abstract it away and focus only on the transcript. Next, we distinguish between different notions of interactivity for locally private protocols.

**Definition 5.** *If locally private protocol $\mathcal{A}$ makes all randomizer assignments before receiving any outputs, then $\mathcal{A}$ is* noninteractive*. If $\mathcal{A}$ makes these assignments adaptively, but each user is queried at most once, then $\mathcal{A}$ is* sequentially interactive *[10]. Note that in all cases the randomizer assignment is independent of the data of the user who will apply the randomizer to their data.*

The most general model of local privacy allows *full interactivity*: users may produce arbitrarily many outputs in arbitrary sequences. In particular, the protocol may solicit multiple outputs from a user across time. We focus on the relative power of noninteractive and sequentially interactive local privacy, so we simply shorthand these terms noninteractive and interactive. Finally, we define local differential privacy.

**Definition 6.** *A protocol $\mathcal{A}$ is* $(\varepsilon, \delta)$-locally differentially private *if its transcript is an $(\varepsilon, \delta)$-differentially private function of the user data. If $\delta = 0$, we say $\mathcal{A}$ is* $\varepsilon$-locally differentially private.

In particular, a sequentially interactive protocol is $(\varepsilon, \delta)$-locally differentially private if and only if every output is produced by an $(\varepsilon, \delta)$-randomizer.

[1]Our presentation of local differential privacy imitates that given by Joseph et al. [16].

## 2.2 Statistical Query Learning

Kearns [20] first introduced statistical query (SQ) learning, but we adopt the definitions and notation used by Kasiviswanathan et al. [19] and Daniely and Feldman [7]. In SQ learning, a learner receives access to an SQ oracle. The learner passes the oracle a function $\phi$ and tolerance $\tau$, and the SQ oracle returns a $\tau$-accurate approximation of the expected value of $\phi$ over the randomness of examples drawn from $\mathcal{D}$ and labeled by $f$. This formalizes the notion of a coarse population-level learning[2]. We start by defining an SQ oracle.

**Definition 7** ([20, 19])**.** *Let $\mathcal{D}$ be a distribution over data domain $X$. An* SQ oracle $\mathsf{SQ}_{\mathcal{D},f}$ *takes as input a function $\phi\colon X \times \{-1,1\} \to [-1,1]$ on labeled examples and a tolerance parameter $\tau \in (0,1)$ and computes an output $\mathsf{SQ}_{\mathcal{D},f}(\phi,\tau) = v$ such that*

$$|v - \mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x,f(x))\right]| \leq \tau.$$

*We also place restrictions on $\phi$ and $\tau$: it must be possible to evaluate $\phi$ in time polynomial in the learning problem parameters, and $\tau$ must be no smaller than an inverse polynomial in the problem parameters.*

An SQ oracle thus approximates the expected output of $f$ when examples are drawn from $\mathcal{D}$ and labeled by $f$, up to an additive error $\tau$. Where a locally private algorithm learns by communicating with individual users, an *SQ algorithm* learns through calls to an SQ oracle.

**Definition 8.** *An* SQ transcript *$\lambda$ is a vector of 3-tuples $(\phi_t, \tau_t, v_t)$ of the function, tolerance, and output produced at each time $t$. Let $S_\lambda$ denote the collection of SQ transcripts and $S_\phi$ the collection of input functions. Then an* SQ algorithm *$\mathcal{A}$ is a function $\mathcal{A}\colon S_\lambda \to (S_\phi \times (0,1)) \cup \{\perp\}$ mapping SQ transcripts to SQ oracle calls or halting ($\perp$).*

We also define notions of interactivity that directly parallel those of the local model.

**Definition 9.** *If SQ algorithm $\mathcal{A}$ decides all SQ oracle calls before receiving any answers, then $\mathcal{A}$ is* noninteractive*. Otherwise, $\mathcal{A}$ is* interactive*.*

## 2.3 PAC Learning

In probably approximately correct (PAC) learning problems, a learner receives examples from a domain $X$ labeled by points in $Y$, and the learner's goal is to use these examples to learn a concept labeling the points of $X$. The concept must be probably (over the distribution of labeled points received during learning) approximately correct (labels future points with low error). Valiant [25] first introduced PAC learning, but we imitate the definitions given by Daniely and Feldman [7].

**Definition 10.** *Let $X$ be a domain and let $C$ be a class of boolean functions over $X$. Algorithm $\mathcal{A}$* PAC learns *$C$ with error $\alpha$ and failure probability $\beta$ if, for every distribution $\mathcal{D}$ over $X$ and $f \in C$, given access (via oracle or samples) to the input distribution over samples $(x, f(x))$ for $x \sim \mathcal{D}$, with probability $\geq 1-\beta$ over oracle calls or samples and $\mathcal{A}$, $\mathcal{A}$ outputs a function $h$ such that $\mathbb{P}_{x\sim\mathcal{D}}\left[f(x) \neq h(x)\right] \leq \alpha$. If $\mathcal{A}$ has running time $\mathsf{poly}(\log(|X|), \log(|C|), 1/\varepsilon)$ then $\mathcal{A}$ is* efficient*.*

We will show that certain PAC learning problems are much harder for noninteractive algorithms than interactive ones. Along the way, we distinguish between PAC learners that are *distribution-independent* (as given in Definition 10) and *distribution-dependent* (require examples drawn from a specific distribution). We also distinguish between *weak* learners (which may have error arbitrarily close to $\frac{1}{2}$) and *strong* learners (which obtain error $\leq \alpha$ for any $\alpha > 0$ using $\mathsf{poly}(1/\alpha)$ samples or oracle calls). The separation from Kasiviswanathan et al. [19] is for distribution-dependent strong learning, while the separation from Daniely and Feldman [7] is for distribution-independent weak learning.

[2]By the same token, a key result of Kearns [20] is that SQ learning implies *noise-tolerant* learning, where some fraction of the labels are corrupted

## 3 Local Privacy and SQ Learning

Our first step in separating noninteractive and interactive local privacy is a connection between local privacy and SQ learning. As the eventual goal is to use SQ lower bounds to prove local privacy lower bounds, it is enough to show that SQ learners can simulate locally private learners in a way that preserves (non)interactivity. Instead, Kasiviswanathan et al. [19] prove a stronger result: up to polynomial factors in sample complexity, locally private and SQ learning are *equivalent*. We recap this result in this section.

Throughout, we assume that there is a common distribution $\mathcal{D}$ over examples and labeling concept $f$ mapping examples to $\{-1, 1\}$ for SQ and locally private learners. Therefore the locally private learner interacts with users taking samples $(x, f(x))$, and the SQ learner interacts with an SQ oracle $\mathsf{SQ}_{\mathcal{D},f}$.

### 3.1 SQ Learning to Locally Private Learning

We start with the easier direction of the equivalence: transforming an SQ learner into a locally private learner. First, we focus on simulating a single SQ oracle call under local privacy. An oracle call to $\mathsf{SQ}_{\mathcal{D},f}(\phi, \tau)$ returns a $\tau$-close approximation of $\mathbb{E}_{x\sim\mathcal{D}}[\phi(x, f(x))]$. $\phi$ has range $[-1, 1]$, so absent any privacy restrictions, by a standard concentration argument we can mimic this by taking $n = O\left(\frac{\log(1/\beta)}{\tau^2}\right)$ samples $x_1, \ldots, x_n \sim \mathcal{D}$ and returning the empirical mean $\frac{1}{n}\sum_{i=1}^n \phi(x_i, f(x_i))$.

$\varepsilon$-local privacy only slightly complicates this argument. Now, instead of averaging the raw values $\phi(x_1, f(x_1)), \ldots, \phi(x_n, f(x_n))$, we average the randomizer outputs $\frac{1}{n}\sum_{i=1}^n R(\phi(x_i, f(x_i)))$. Here $R$ is an $\varepsilon$-randomizer that simply adds Laplace noise: $R(y) = y + \eta$ for $\eta \sim \mathsf{Lap}\left(\frac{2}{\varepsilon}\right)$. A similar concentration argument implies that the aggregate Laplace noise also concentrates, and the cost is a $O(\frac{1}{\varepsilon^2})$ blowup in the number of samples. Pseudocode for this algorithm appears below.

**Algorithm 1** Locally Private Simulation of Query to $\mathsf{SQ}_{\mathcal{D}}$

```
Require: Distribution $\mathcal{D}$, privacy parameter $\varepsilon$, query function $\phi$, tolerance $\tau$
  $n \leftarrow \max\left(\frac{8\ln(4/\beta)}{\tau^2}, \frac{64\ln(2/\beta)}{\varepsilon^2\tau^2}\right)$
  $\hat{v} \leftarrow 0$
  for $i = 1, 2, \ldots, n$ do
    $\eta_i \sim \mathsf{Lap}\left(\frac{2}{\varepsilon}\right)$
    $\hat{v} \leftarrow \hat{v} + (\phi(x_i, f(x_i)) + \eta_i)$
    Output $\frac{\hat{v}}{n}$
  end for
```

**Lemma 1** (Lemma 5.6 [19]). *Given SQ oracle call $\mathsf{SQ}_{\mathcal{D},f}(\phi, \tau)$, there exists $\varepsilon$-locally private $\mathcal{A}'$ requiring $n = O\left(\frac{\log(1/\beta)}{\varepsilon^2\tau^2}\right)$ samples that with probability at least $1 - \beta$ returns $\hat{v}$ with*

$$|\hat{v} - \mathbb{E}_{x\sim\mathcal{D}}[\phi(x, f(x)]| \leq \tau.$$

*Proof.* Privacy: Each randomizer output is of the form $\phi(x_i, f(x_i)) + \eta_i$. Since $\phi$ has range $[-1, 1]$, it has $\ell_1$ sensitivity 2. A standard result in differential privacy says that, for any function with $\ell_1$ sensitivity $\Delta$, adding $\mathsf{Lap}\left(\frac{\Delta}{\varepsilon}\right)$ noise to its output ensures $\varepsilon$-differential privacy (see e.g. Theorem 3.4 in the survey of Dwork and Roth [12]). Thus $\phi(x_i, f(x_i)) + \eta_i$ is an $\varepsilon$-randomizer, so Algorithm 1 is $\varepsilon$-locally private.

Accuracy: Let $v = \mathbb{E}_{x\sim\mathcal{D}}[\phi(x, f(x))]$ denote the true mean. It suffices to show that with probability $\geq 1 - \beta$, $|\hat{v} - v| \leq \tau$. First, we decompose $\hat{v} = \hat{v}_1 + \hat{v}_2$ into noiseless and noise components

$$\hat{v}_1 = \tfrac{1}{n}\sum_{i=1}^{n} \phi(x_i, f(x_i)) \text{ and } \hat{v}_2 = \tfrac{1}{n}\sum_{i=1}^{n} \eta_i.$$

By an additive Chernoff bound,

$$\mathbb{P}_{\mathcal{D}}\left[|\hat{v}_1 - v| \geq \frac{\tau}{2}\right] \leq 2\exp\left(-\frac{\tau^2 n}{8}\right).$$

Thus with $n \geq \frac{8\ln(4/\beta)}{\tau^2}$ samples, with probability $\geq 1 - \frac{\beta}{2}$, $|\hat{v}_1 - v| \leq \frac{\tau}{2}$.

It remains to derive a lower bound on $n$ to ensure $|\hat{v}_2| \leq \frac{\tau}{2}$. We use the following concentration result for sums of Laplace random variables.

**Lemma 2** (Lemma A.3 in Kasiviswanathan et al. [19])**.** *Let $\eta_1, \ldots, \eta_n \sim \mathsf{Lap}(\zeta)$. Then for every $\alpha > 0$,*

$$\mathbb{P}\left[\left|\frac{\sum_{i=1}^{n}\eta_i}{n}\right| \geq \alpha\right] = \exp\left(-\frac{\alpha^2 n}{4\zeta^2}\right).$$

Substituting $\zeta = \frac{2}{\varepsilon}$ in Lemma 2, with $n \geq \frac{64\ln(2/\beta)}{\varepsilon^2\tau^2}$ samples, with probability $\geq 1 - \frac{\beta}{2}$, $|\hat{v}_2| \leq \frac{\tau}{2}$. Combining these results and taking $n = \max\left(\frac{8\ln(4/\beta)}{\tau^2}, \frac{64\ln(2/\beta)}{\varepsilon^2\tau^2}\right)$, with probability at least $1 - \beta$

$$|\hat{v} - v| = |(\hat{v}_1 - v) + \hat{v}_2| \leq |\hat{v}_1 - v| + |\hat{v}_2| \leq \tau.$$

□

With Lemma 1, it is straightforward to simulate an entire SQ algorithm with minimum tolerance $\tau$: for each SQ call, simulate it using Algorithm 1. The cost of simulating $t$ calls is a $t$ blowup in sample complexity (for simulating each call separately) and a $\ln(t/\beta)$ rather than a $\ln(1/\beta)$ term due to union bounding across the $t$ calls. $\varepsilon$-local privacy follows since we query new users for each simulation. We call this algorithm $\mathcal{A}'$ and get our main theorem for this section.

**Theorem 1** (Theorem 5.7 in Kasiviswanathan et al. [19])**.** *Let $\mathcal{A}$ be an SQ algorithm that makes (possibly interactive) SQ oracle calls $\mathsf{SQ}_{\mathcal{D},f}(\phi_1, \tau_1), \ldots, \mathsf{SQ}_{\mathcal{D},f}(\phi_t, \tau_t)$ where $\min\{\tau_1, \ldots, \tau_t\} \geq \tau$. The simulation $\mathcal{A}'$ above is $\varepsilon$-locally private, has sample complexity $n = O\left(\frac{t\log(t/\beta)}{\varepsilon^2\tau^2}\right)$, and with probability at least $1 - \beta$ returns $\hat{v}_1, \ldots, \hat{v}_t$ such that for each $i \in [t]$,*

$$|\hat{v}_i - \mathbb{E}_{x\sim\mathcal{D}}[\phi(x, f(x))]| \leq \tau_i.$$

*Moreover, $\mathcal{A}'$ is noninteractive if and only if $\mathcal{A}$ is noninteractive.*

## 3.2 Locally Private Learning to SQ Learning

We now turn to the more involved direction of the equivalence: transforming a locally private learner to an SQ learner. Kasiviswanathan et al. [19] show that this transformation is possible even for fully interactive protocols, albeit with a number of SQ oracle calls scaling with the number of randomizer calls. However, we only want to separate noninteractive and interactive protocols. As a result, we will prove a slightly weaker result that holds only for sequentially interactive protocols. The benefit is that our result does not need to account for re-queried users, and we get a simpler and shorter proof.

We briefly sketch this proof. A reasonable smaller goal is to simulate a single randomizer call, i.e. a single step in a locally private protocol, using an SQ oracle: given randomizer $R$ and SQ oracle $\mathsf{SQ}_{\mathcal{D},f}$, we want to use $\mathsf{SQ}_{\mathcal{D},f}$ to simulate the output of $R(x, f(x))$ when $x \sim \mathcal{D}$. For a fixed randomizer output $y$, let $p_y = \mathbb{P}_{x\sim\mathcal{D},R}[R(x, f(x)) = y]$, and let $q_y = \mathbb{P}_R[R(\perp) = y]$ (the exact identity of $\perp \in X \times \{-1, 1\}$ is irrelevant). For an $\varepsilon$-locally private protocol, $R$ is an $\varepsilon$-local randomizer, so $q_y$ approximates $p_y$ within a multiplicative $e^\varepsilon$ factor. To get a better approximation, we will use $q_y$ to rejection sample from $p_y$.

The main obstacle to rejection sampling with $q_y$ and $p_y$ is that the learner does not know $\mathcal{D}$, and thus does not know $p_y$. Our response is to use $\mathsf{SQ}_{\mathcal{D},f}$ to learn about $p_y$. Once we learn a sufficiently good approximation $\tilde{p}_y$, we can use it to rejection sample. Pseudocode for his process appears in Algorithm 2.

**Algorithm 2** SQ Simulation of $\varepsilon$-randomizer $R$

```
Require: Privacy parameter $\varepsilon$, randomizer $R$, SQ oracle $\mathsf{SQ}_{\mathcal{D},f}$, number of randomizer calls $t$
    Sample $y \sim R(\perp)$
    Set $q_y \leftarrow \mathbb{P}_R[R(\perp) = y]$
    Define function $\phi(x, f(x)) = \frac{\mathbb{P}_R[R(x,f(x))=y]-q_y}{q_y(e^\varepsilon - e^{-\varepsilon})}$
    Set $\tau \leftarrow \frac{\beta}{3e^{2\varepsilon}t}$
    Set $v \leftarrow \mathsf{SQ}_{\mathcal{D},f}(\phi, \tau)$
    Set $\tilde{p}_y \leftarrow vq_y(e^\varepsilon - e^{-\varepsilon}) + q_y$
    With probability $\frac{\tilde{p}_y}{q_y(1+\frac{\beta}{3t})e^\varepsilon}$ output $y$, otherwise repeat
```

**Theorem 2** (Claim 5.9 [19]). *For every interactive (noninteractive) $\varepsilon$-locally private algorithm $\mathcal{A}$ on $t$ samples, there exists an interactive (noninteractive) SQ algorithm $\mathcal{A}'$ that in expectation makes $te^\varepsilon$ queries to $\mathsf{SQ}_{\mathcal{D},f}$ with tolerance $\tau = \Theta\left(\frac{\beta}{e^{2\varepsilon}t}\right)$ such that the statistical distance between the output distributions of $\mathcal{A}$ and $\mathcal{A}'$ is $\leq \beta$.*

*Proof.* First, we verify that the defined function $\phi$ maps to $[-1, 1]$. $R$ is an $\varepsilon$-local randomizer, so

$$\begin{aligned}\phi(x, f(x)) &= \frac{\mathbb{P}_R[R(x, f(x)) = y] - q_y}{q_y(e^\varepsilon - e^{-\varepsilon})} \\ &\leq \frac{q_y(e^\varepsilon - 1)}{q_y(e^\varepsilon - e^{-\varepsilon})} \leq 1\end{aligned}$$

and

$$\begin{aligned}\phi(x, f(x)) &= \frac{\mathbb{P}_R[R(x, f(x)) = y] - q_y}{q_y(e^\varepsilon - e^{-\varepsilon})} \\ &\geq \frac{q_y(e^{-\varepsilon} - 1)}{q_y(e^\varepsilon - e^{-\varepsilon})} \geq -1\end{aligned}$$

since $\varepsilon \geq 0$. Thus $\phi$ is a valid function for an SQ call. Next, we generate $v$ by an SQ call $\mathsf{SQ}_{\mathcal{D},f}(\phi, \tau)$. By our definition of $\phi$, the expected value of such an SQ call is

$$\begin{aligned}\mathbb{E}\left[\mathsf{SQ}_{\mathcal{D},f}(\phi, \tau)\right] &= \frac{\mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{P}_R[R(x, f(x)) = y]\right] - q_y}{q_y(e^\varepsilon - e^{-\varepsilon})} \\ &= \frac{p_y - q_y}{q_y(e^\varepsilon - e^{-\varepsilon})}\end{aligned}$$

so $\left|v - \frac{p_y y - q_y}{q_y(e^\varepsilon - e^{-\varepsilon})}\right| \leq \tau$. Multiplying both sides of this inequality by $q_y(e^\varepsilon - e^{-\varepsilon})$ and using $\tilde{p}_y = vq_y(e^\varepsilon - e^{-\varepsilon}) + q_y$ gives

$$\begin{aligned}|\tilde{p}_y - p_y| &\leq \tau \cdot q_y(e^\varepsilon - e^{-\varepsilon}) \\ &\leq \tau \cdot p_y e^\varepsilon(e^\varepsilon - e^{-\varepsilon}) \\ &\leq \tau p_y e^{2\varepsilon} = p_y \cdot \frac{\beta}{3t}\end{aligned}$$

where the second inequality uses the fact that $q_y$ comes from an $\varepsilon$-randomizer. Thus

$$\tilde{p}_y \in [(1-\alpha)p_y, (1+\alpha)p_y] \tag{1}$$

where $\alpha = \frac{\beta}{3t}$. We have therefore shown that $\tilde{p}_y$ approximates $p_y$, the expected probability of randomizer output $y$ over the input draw and randomizer.

We can now use this knowledge of $\tilde{p}_y$ to show that our rejection sampling based on $\tilde{p}_y$ is a good approximation. First, for a given iteration, any particular element $y$ gets output with probability

$$\mathbb{P}\left[\text{output } y\right] = q_y \cdot \frac{\tilde{p}_y}{q_y(1+\alpha)e^\varepsilon} = \frac{\tilde{p}_y}{(1+\alpha)e^\varepsilon} \in \left[\frac{(1-\alpha)}{(1+\alpha)e^\varepsilon} \cdot p_y, \frac{1}{e^\varepsilon} \cdot p_y\right]$$

where the last relationship follows from Equation 1. By similar logic, the probability that Algorithm 2 terminates is

$$\mathbb{P}\left[\text{terminate}\right] = \sum_y q_y \cdot \frac{\tilde{p}_y}{q_y(1+\alpha)e^\varepsilon} \in \left[\frac{1-\alpha}{(1+\alpha)e^\varepsilon}, \frac{1}{e^\varepsilon}\right]. \tag{2}$$

Finally, we combine the three previous expressions to get

$$\begin{aligned}\mathbb{P}\left[\text{output } y \mid \text{terminate}\right] &= \frac{\mathbb{P}\left[\text{output } y, \text{ terminate}\right]}{\mathbb{P}\left[\text{terminate}\right]} \\ &= \frac{\mathbb{P}\left[\text{output } y\right]}{\mathbb{P}\left[\text{terminate}\right]} \\ &\in \left[\frac{1-\alpha}{1+\alpha}p_y, \frac{1+\alpha}{1-\alpha}p_y\right].\end{aligned}$$

We have therefore shown that, upon terminating, the output distribution for Algorithm 2 approximates that of the local randomizer. It remains to show that this approximation, parameterized by $\alpha$, is not too bad.

Recall that $\alpha = \frac{\beta}{3t} \leq \frac{1}{3}$. Thus $\alpha \geq 3\alpha^2$, so $1+\alpha \leq 1 + 2\alpha - 3\alpha^2$, and $\frac{1+\alpha}{1-\alpha} \leq 1 + 3\alpha$. Similarly, $\frac{1-\alpha}{1+\alpha} \geq 1 - 3\alpha$, so $\mathbb{P}\left[\text{output } y \mid \text{terminate}\right] \in [(1-3\alpha)p_y, (1+3\alpha)p_y]$. In particular,

$$|\mathbb{P}\left[\text{output } y \mid \text{terminate}\right] - p_y| \leq 3\alpha = \frac{\beta}{t}$$

so union-bounding over the $t$ randomizer calls gives the statistical distance portion of the claim.

We conclude by analyzing sample complexity. By Equation 2 in every iteration

$$\mathbb{P}\left[\text{terminate}\right] \geq \frac{1-\alpha}{1+\alpha} \cdot e^{-\varepsilon},$$

so the expected number of iterations to terminate is $\frac{1+\alpha}{1-\alpha}e^\varepsilon \leq 2e^\varepsilon$ by the preceding paragraph. Thus the total expected number of SQ oracle calls across the $t$ simulated randomizer calls is $O(t \cdot e^\varepsilon)$. $\square$

We now pause to summarize our results so far. Theorem 1 showed how to simulate an SQ learner with a locally private learner. Conversely, Theorem 2 showed how to simulate a (sequentially interactive) locally private learner with an SQ learner. Overall, we have shown an equivalence (up to polynomial factors) between locally private learning and SQ learning.

We proved Theorem 1 for completeness, but our most useful tool for the rest of this paper will be Theorem 2. We will prove that certain problems are hard for noninteractive SQ learners and easy for interactive SQ learners. By Theorem 2, this yields an analogous separation for noninteractive and interactive locally private learners.

## 4 Separation for Masked Parity

The first concept class separating noninteractive and interactive locally private learning is *masked parity* [19]. A given $d$-dimensional masked parity concept $c^d_{m,p}$ splits the input into halves based on the last bit $x_{-1}$ of each example $x$. If $x_{-1} = 1$ then $c^d_{m,p}(x)$ is a function of $p_{x_{d+1}}$, the $x^{th}_{d+1}$ bit of $p$. If $x_{-1} = 0$, then the mask $m$ becomes relevant. If $m = 0$, then $c^d_{m,p}(x)$ is a function of the parity of $p$ and $x_{\le d}$, while if $m = 1$ then $c^d_{m,p}(x)$ is the negation of this quantity. Intuitively, a learner must recover $m$ and $p$ to get better than 3/4 accuracy, but it is much easier to do this with two rounds of interaction than one. The formal statement and proof of this result appear in Theorem 3 near the end of this section.

**Definition 11.** *For parameters $d \in \mathbb{N}$, $m \in \{0,1\}$, and $p \in \{0,1\}^d$, a* masked parity *concept is a function $c^d_{m,p} \colon \{0,1\}^d \times \{0,1\}^{\log(d)} \times \{0,1\} \to \{-1,1\}$ that labels $x \in \{0,1\}^d \times [d] \times \{0,1\}$*[3] *by*

$$c^d_{m,p}(x) = \begin{cases} (-1)^{m+\langle p, x_{\le d}\rangle \ mod\ 2} & x_{-1} = 0 \\ (-1)^{p_{x_{d+1}}} & x_{-1} = 1 \end{cases}$$

*where $x_{\le d}$ denotes the first $d$ bits of $x$, $p_{x_{d+1}}$ denotes the bit of $p$ indexed by $x_{d+1} \in [d]$, and $x_{-1}$ denotes the last bit of $x$. The class of all such concepts is denoted $C^d_{MP}$.*

We first show that an SQ learner can easily learn $C^d_{MP}$ in two rounds of interaction in the distribution-specific setting where $\mathcal{D}$ is uniform. At a high level, the learner spends the first round learning each bit $j$ of $p$ using one statistical query per bit about examples where $x_{d+1} = j$, $x_{-1} = 1$, and the label is $-1$. If $p_j = 1$, such examples make up a $\frac{1}{2d}$ fraction of the example distribution. If instead $p_j = 0$, then there are no examples. Thus it is enough to approximate the frequency of such examples by $\tau_1 \le \frac{1}{5d}$. Doing so for each of the $d$ bits takes $d$ statistical queries and recovers $\hat{p} = p$ exactly.

In the second round, the learner makes a statistical query about examples where $x_{-1} = 0$ and the label is not $(-1)^{\langle \hat{p}, x\rangle \text{ mod } 2}$. If the mask $m = 0$, then no such examples exist. If $m = 1$, then half of the examples fall into this category. Thus a single SQ oracle call of tolerance $\tau_2 \le \frac{1}{5}$ tells the learner $m$. Having recovered $p$ and $m$, the learner exactly learns $c^d_{m,p}$ in $d+1$ SQ oracle calls in two rounds with probability 1. Pseudocode for this interactive SQ learner appears in Algorithm 3.

**Algorithm 3** Interactive SQ Learner for $C^d_{MP}$

```
Require: SQ oracle $\mathsf{SQ}_{\mathcal{D},c^d_{m,p}}$ over uniform distribution $\mathcal{D}$
    Set tolerance $\tau_1 \leftarrow \frac{1}{5d}$
    for $j = 1, \ldots, d$ (in parallel) do
        Define function $\phi_j(x, c^d_{m,p}(x)) = (x_{d+1} = j \wedge x_{-1} = 1 \wedge c^d_{m,p}(x) = -1)$
        Set $v_j \leftarrow \mathsf{SQ}_{\mathcal{D},c^d_{m,p}}(\phi_j, \tau_1)$
        Set $\hat{p}_j \leftarrow \left(v_j \ge \frac{3}{10d}\right)$
    end for
    Set $\hat{p} \leftarrow \hat{p}_1, \circ \cdots \circ \hat{p}_d \in \{0,1\}^d$
    Define function $\phi_{d+1}(x, c^d_{m,p}(x)) = (x_{-1} = 0 \wedge c^d_{m,p}(x) \neq (-1)^{\langle \hat{p}, x_{\le d}\rangle \text{ mod } 2})$
    Set tolerance $\tau_2 \leftarrow \frac{1}{5}$
    Set $w \leftarrow \mathsf{SQ}_{\mathcal{D},c^d_{m,p}}(\phi_{d+1}, \tau_2)$
    Set $\hat{m} \leftarrow (w \ge \frac{3}{10})$
    Output $c^d_{\hat{m},\hat{p}}$
```

[3]This extends to purely binary examples by replacing the element from $[d]$ by an element from $\{0,1\}^{\log(d)}$. We avoid this for the sake of neatness.

Note that the interactive learner always outputs an exactly correct hypothesis. This makes it a (perfectly) strong learner. However, the distribution-dependence is crucial. For example, if $\mathcal{D}$ is a distribution with no mass on examples where $x_{-1} = 1$, then the first round of Algorithm 3 no longer works. Instead, masked parity becomes as hard as parity, which requires a number of SQ oracle calls exponential in $d$ even for interactive SQ learners [20].

In contrast, Kasiviswanathan et al. [19] show that no noninteractive SQ learner for the uniform example distribution can get error $< \frac{1}{4}$ with a number of SQ oracle subexponential in $d$. This means that strong learning of masked parity over the uniform distribution is much harder for noninteractive SQ learners than it is for interactive SQ learners.

Before the formal statement and proof, we offer some intuition. The main idea is that before the SQ learner sees the results of its SQ oracle calls, it has no idea what the parity $p$ and mask $m$ are. In contrast, the interactive SQ learner could first learn $p$ and then learn $m$ using the knowledge of $p$. Defining $\phi_{d+1}$ in terms of $\hat{p}$ made this knowledge of $p$ necessary. Without this knowledge, the noninteractive learner cannot define $\phi_{d+1}$, and must instead effectively learn parity, which is known to be hard [20].

**Lemma 3** (Theorem 5.16 [19])**.** *Fix the example distribution to be uniform. Then the interactive SQ learner given in Algorithm 3 learns $C_{MP}^d$ to error 0 with probability 1 in 2 rounds using* $\mathsf{poly}(d)$ *SQs with minimum tolerance $\tau = \frac{1}{5d}$. However, there exists an SQ oracle $\mathcal{O}$ such that for any noninteractive SQ learner making $t$ queries to $\mathcal{O}$, if the concept $c_{\bar{m},\bar{p}}^d$ is drawn uniformly at random from $C_{MP}^d$, then with probability at least $\frac{1}{2} - \frac{t}{2^{d/3+2}}$ over the draw of $c_{\bar{m},\bar{p}}^d$ the learner outputs a hypothesis with error $\geq \frac{1}{4}$.*

*Proof.* Interactive upper bound: First consider the $j^{th}$ query in round one, which aims to learn $p_j$. If $p_j = 1$, then

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\phi_j(x, c_{m,p}^d(x))\right] = \mathbb{P}_{x\sim\mathcal{D}}\left[x_{d+1} = j \wedge x_{-1} = 1 \wedge c_{m,p}^d(x) = -1\right] = \frac{1}{2d}$$

where the last equality uses the fact that $\mathcal{D}$ is uniform. Thus for $\tau_1 = \frac{1}{5d}$ and $v_j = \mathsf{SQ}_{\mathcal{D},c_{m,p}^d}(\phi_j, \tau_1)$, if $p_j = 1$ then $v_j \geq \frac{1}{2d} - \frac{1}{5d} = \frac{3}{10d}$, whereas if $p_j = 0$ then $v_j \leq \frac{1}{5d}$. It follows that each of the $d$ queries for $j \in [d]$ reveals $p_j$ exactly, and after the first round $\hat{p} = p$.

Since $\hat{p} = p$, if $m = 0$ then

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\phi_{d+1}(x, c_{m,p}^d(x))\right] = \mathbb{P}_{x\sim\mathcal{D}}\left[x_{-1} = 0 \wedge c_{m,p}^d(x) \neq (-1)^{\langle \hat{p}, x_{\leq d}\rangle \bmod 2}\right] = 0$$

and if $m = 1$ then $\mathbb{E}_{x\sim\mathcal{D}}\left[\phi_{d+1}(x, c_{m,p}^d(x))\right] = \frac{1}{2}$. Thus for $\tau_2 = \frac{1}{5}$ and $w = \mathsf{SQ}_{\mathcal{D},c_{m,p}^d}(\phi_{d+1}, \tau_2)$, if $m = 0$ then $w \leq \frac{1}{5}$ and if $m = 1$ then $w \geq \frac{3}{10}$. Therefore the learner recovers $m$, $c_{\hat{m},\hat{p}}^d = c_{m,p}^d$, and the learned hypothesis has 0 error.

Noninteractive lower bound: Suppose the SQ learner makes queries using functions $\phi_1, \ldots, \phi_t$. Since these queries are made in a single nonadaptive batch, the functions are all independent of $\bar{m}$ and $\bar{p}$. Our argument rests on proving a similar claim for the answers to these queries. We will decompose each such answer into three parts: one depending only on $\phi_i$ (but not $\bar{m}$ and $\bar{p}$), one depending on $\phi_i$ and $\bar{p}$ (but not $\bar{m}$), and one depending on all of $\phi_i$, $\bar{p}$, and $\bar{m}$. The crux of the argument is showing that we can (typically) approximate the $t$ answers while ignoring this last component. It follows that the answers (typically) do not reveal $\bar{m}$. This in turn prevents strong learning.

Consider one of the query functions $\phi$ (we drop the subscript from $\phi_i$ for neatness) and define

$$g_\phi = \frac{\mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x,1) + \phi(x,-1)\right]}{2} \text{ and } h_\phi(x) = \frac{\phi(x,1) - \phi(x,-1)}{2}.$$

We now define the notation $\langle\cdot,\cdot\rangle_\mathcal{D}$ for function inputs to denote correlation over a distribution $\mathcal{D}$. For functions $f_1$ and $f_2$, we define

$$\langle f_1, f_1\rangle_\mathcal{D} = \mathbb{E}_{x\sim\mathcal{D}}\left[f_1(x)f_2(x)\right].$$

Thus we can rewrite

$$\begin{aligned}\mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x, c^d_{\bar{m},\bar{p}}(x))\right] &= \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)+\phi(x,-1)}{2} + \frac{(\phi(x,1)-\phi(x,-1))c^d_{\bar{m},\bar{p}}(x)}{2}\right]\\ &= g_\phi + \langle h_\phi, c^d_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}.\end{aligned}$$

This is the first step in our decomposition, as $g_\phi$ depends only on $\phi$, but not on $\bar{m}$ and $\bar{p}$. We now further decompose $\langle h_\phi, c^d_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}$. We start by defining slightly different versions of $h_\phi$ and $c^d_{\bar{m},\bar{p}}$ parameterized by a bit $s \in \{0,1\}$:

$$h^s_\phi(x) = \begin{cases} h_\phi(x) & x_{-1} = s \\ 0 & x_{-1} \neq s \end{cases}$$

and

$$c^{d,s}_{\bar{m},\bar{p}} = \begin{cases} c^d_{\bar{m},\bar{p}}(x) & x_{-1} = s \\ 0 & x_{-1} \neq s \end{cases}.$$

Thus $h^s_\phi$ and $c^{d,s}_{\bar{m},\bar{p}}$ respectively behave like $h_\phi$ and $c^d_{\bar{m},\bar{p}}$ on points $x$ where $x_{-1} = s$, and are constant 0 otherwise. We therefore get

$$\langle h_\phi, c^d_{\bar{m},\bar{p}}\rangle_{\mathcal{D}} = \langle h^1_\phi, c^{d,1}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}} + \langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}$$

and we have our desired decomposition

$$\mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x, c^d_{\bar{m},\bar{p}}(x))\right] = g_\phi + \langle h^1_\phi, c^{d,1}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}} + \langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}$$

since $g_\phi$ depends only on $\phi$, $\langle h^1_\phi, c^{d,1}_{\bar{m},p}\rangle_{\mathcal{D}}$ depends only on $\phi$ and $\bar{p}$ (because $x_{-1} = 1$), and only $\langle h^0_\phi, c^{d,0}_{\bar{m},p}\rangle_{\mathcal{D}}$ depends on $\phi$, $\bar{p}$, and $\bar{m}$.

We now define the SQ oracle $\mathcal{O}_{\mathcal{D}, c^d_{\bar{m},\bar{p}}}$. Our definition relies on the decomposition above by selectively omitting the last term in the decomposition from its answers. For a given query $\phi$ and tolerance $\tau$,

$$\mathcal{O}_{\mathcal{D}, c^d_{\bar{m},\bar{p}}}(\phi, \tau) = \begin{cases} g_\phi + \langle h^1_\phi, c^{d,1}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}} & |\langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}| < \tau \\ \mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x, c^d_{\bar{m},\bar{p}}(x))\right] & \text{otherwise} \end{cases}.$$

$\mathcal{O}$ thus omits the last term of the decomposition when $|\langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}|$ is small and returns the true answer otherwise. By our decomposition, if every query $\phi$ has $|\langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}| < \tau$, then $\mathcal{O}$ never gives an answer that depends on $\bar{m}$. We now show that this is "typically" (over the draw of the target concept) the case.

**Lemma 4.** *With probability at least $1 - \frac{t}{2^{d/3+2}}$ over the draw of $c^d_{\bar{m},\bar{p}}$, for $\tau \geq \frac{1}{2^{d/3}}$ and any nonadaptively chosen queries $\phi_1, \ldots, \phi_t$ with minimum tolerance at least $\tau$, for each query $\phi_i$, $\mathcal{O}$ returns $g_{\phi_i} + \langle h^1_{\phi_i}, c^{d,1}_{\bar{m},\bar{p}}\rangle_{\mathcal{D}}$ and is a valid SQ oracle.*

*Proof.* Consider $p, p' \in \{0,1\}^d$ and $m \in \{0,1\}$. Then if $p = p'$,

$$\begin{aligned}\langle c^{d,0}_{m,p}, c^{d,0}_{m,p'}\rangle_{\mathcal{D}} &= \mathbb{E}_{x\sim\mathcal{D}}\left[c^d_{m,p}(x)\cdot c^d_{m,p}(x)\cdot \mathbb{1}\left[x_{-1}=0\right]\right]\\ &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{1}\left[x_{-1}=0\right]\right] = \frac{1}{2}\end{aligned}$$

since $\mathcal{D}$ is uniform. If $p \neq p'$ then

$$\begin{aligned}\langle c^{d,0}_{m,p}, c^{d,0}_{m,p'}\rangle_{\mathcal{D}} &= \mathbb{E}_{x\sim\mathcal{D}}\left[c^d_{m,p}(x)\cdot c^d_{m,p'}(x)\cdot \mathbb{1}\left[x_{-1}=0\right]\right]\\ &= \mathbb{E}_{x\sim\mathcal{D}}\left[(-1)^{m+\langle p, x_{\leq d}\rangle \bmod 2}\cdot(-1)^{m+\langle p', x_{\leq d}\rangle \bmod 2}\cdot \mathbb{1}\left[x_{-1}=0\right]\right]\\ &= \mathbb{E}_{x\sim\mathcal{D}}\left[(-1)^{\langle p, x_{\leq d}\rangle + \langle p', x_{\leq d}\rangle \bmod 2}\cdot \mathbb{1}\left[x_{-1}=0\right]\right]\\ &= \mathbb{E}_{x\sim\mathcal{D}}\left[(-1)^{\langle p\oplus p', x_{\leq d}\rangle \bmod 2}\cdot \mathbb{1}\left[x_{-1}=0\right]\right]\\ &= 0\end{aligned}$$

where $\oplus$ denotes XOR. Fix $m \in \{0,1\}$. By this second $p \neq p'$ case, $\{c^{d,0}_{m,p}\}_{p \in \{0,1\}^d}$ is a set of orthogonal functions under $\langle \cdot, \cdot \rangle_{\mathcal{D}}$. Using the first $p = p'$ case, we normalize by $\sqrt{2}$ and get that $\{\sqrt{2} \cdot c^{d,0}_{m,p,}\}_{p \in \{0,1\}^d}$ is orthonormal.

Now we return to analyzing $h^0_\phi$, which we recall behaves like $h_\phi$ when $x_{-1} = 0$ and is constant 0 otherwise. We get

$$\begin{aligned}\sum_{p \in \{0,1\}^d} \langle h^0_\phi, \sqrt{2} \cdot c^{d,0}_{0,p} \rangle^2_{\mathcal{D}} &\leq ||h^0_\phi||_2^2 \\ &= \langle h^0_\phi, h^0_\phi \rangle_{\mathcal{D}} \\ &\leq \frac{1}{2}\end{aligned}$$

where the first inequality uses Parseval's identity and the fact that $h^0_\phi$ is a $d' > d$-dimensional object and $\{\sqrt{2} \cdot c^{d,0}_{0,p}\}_{p \in \{0,1\}^d}$ is a $d$-dimensional orthonormal set by above, and the last inequality holds because $h_\phi \leq 1$ in general and is 0 on the half of the examples where $x_{-1} = 1$. The same logic gives $\sum_{p \in \{0,1\}^d} \langle h^0_\phi, \sqrt{2} \cdot c^{d,0}_{1,p} \rangle^2_{\mathcal{D}} \leq \frac{1}{2}$ so summing the previous two equations gives

$$\sum_{(m,p) \in \{0,1\} \times \{0,1\}^d} 2 \cdot \langle h^0_\phi, c^{d,0}_{m,p} \rangle^2_{\mathcal{D}} \leq 1.$$

Thus at most $2^{(2d/3)-1}$ of the $2^{d+1}$ possible functions $c^d_{m,p}$ have $|\langle h^0_\phi, c^{d,0}_{m,p} \rangle_{\mathcal{D}}| \geq \frac{1}{2^{d/3}}$. Since our original function $c^d_{\bar{m},\bar{p}}$ was a uniform random draw from the set of all $2^{d+1}$ possible functions $c^d_{m,p}$, for any $\phi$

$$\mathbb{P}_{(m,p) \sim U(\{0,1\} \times \{0,1\}^d)} \left[ |\langle h^0_\phi, c^{d,0}_{m,p} \rangle_{\mathcal{D}}| \geq \frac{1}{2^{d/3}} \right] \leq \frac{2^{2d/3-1}}{2^{d+1}} = \frac{1}{2^{(d/3)+2}}.$$

Moreover, $|\langle h^0_\phi, c^{d,0}_{0,p} \rangle_{\mathcal{D}}| = |\langle h^0_\phi, c^{d,0}_{1,p} \rangle_{\mathcal{D}}|$ so the same statement holds irrespective of $m$. Thus

$$\mathbb{P}_{p \sim U\{0,1\}^d} \left[ |\langle h^0_\phi, c^{d,0}_{\bar{m},p} \rangle_{\mathcal{D}}| \geq \frac{1}{2^{d/3}} \right] \leq \frac{1}{2^{d/3+2}}.$$

Since the learner makes $t$ queries $\phi_1, \ldots, \phi_t$, by a union bound

$$\mathbb{P}_{\bar{p} \sim U\{0,1\}^d} \left[ |\langle h^0_{\phi_1}, c^{d,0}_{\bar{m},\bar{p}} \rangle_{\mathcal{D}}| < \frac{1}{2^{d/3}}, \ldots, |\langle h^0_{\phi_t}, c^{d,0}_{\bar{m},\bar{p}} \rangle_{\mathcal{D}}| < \frac{1}{2^{d/3}} \right] \geq 1 - \frac{t}{2^{d/3+2}}$$

and in particular $\mathcal{O}$ answers all queries $\phi_i$ for $i \in [t]$ using $v_i = g_{\phi_i} + \langle h^1_{\phi_i}, c^{d,1}_{\bar{m},\bar{p}} \rangle_{\mathcal{D}}$. Since for each query $|\langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}} \rangle_{\mathcal{D}}| < \frac{1}{2^{d/3}}$, by our decomposition each such $v_i$ has

$$|v_i - \mathbb{E}_{x \sim \mathcal{D}} \left[ \phi(x, c^d_{\bar{m},\bar{p}}(x)) \right]| = |\langle h^0_\phi, c^{d,0}_{\bar{m},\bar{p}} \rangle_{\mathcal{D}}| < \frac{1}{2^{d/3}}$$

and $\mathcal{O}$ can answer such queries to any tolerance $\tau \geq \frac{1}{2^{d/3}}$. □

By Lemma 4, with probability at least $1 - \frac{t}{2^{(d/3)+2}}$ the learner only receives answers that are independent of $\bar{m}$. Thus with probability at least $\frac{1}{2} - \frac{t}{2^{(d/3)+2}}$ the learner recovers the wrong value of $\bar{m}$. The resulting hypothesis therefore mislabels at least half of the examples with $x_{-1} = 0$ and gets error $\geq \frac{1}{4}$ over the uniform distribution. □

By Lemma 3, a noninteractive SQ learner requires $\Omega\left(2^{(d/3)+2}\right)$ SQ oracle calls to strong learn $C^d_{MP}$ over the uniform example distribution, but an interactive SQ learner only needs $O(d)$. Using the SQ-to-local transformation from Theorem 1, we almost immediately get the analogous separation for local privacy.

**Theorem 3** (Corollary 5.17 [19])**.** *Fix the example distribution to be uniform. Then there exists an interactive $\varepsilon$-locally private protocol $\mathcal{A}$ that learns $C_{MP}^d$ to error 0 with probability $\frac{3}{4}$ using $O\left(\frac{d^3 \log(d)}{\varepsilon^2}\right)$ samples, but any noninteractive $\varepsilon$-locally private protocol that learns $C_{MP}^d$ to error $< \frac{1}{4}$ with probability $\geq 3/4$ must use $n = \omega(\mathsf{poly}(d))$ samples.*

*Proof.* Interactive upper bound: Immediate from using Theorem 1 to convert the interactive SQ learner from Lemma 3 into an interactive locally private learner.

Noninteractive lower bound: If there exists a noninteractive $\varepsilon$-locally private algorithm that with probability $\geq 3/4$ learns $C_{MP}^d$ over the uniform distribution to error $< 1/4$ using $O(\mathsf{poly}(d))$ samples, then Theorem 2 and Markov's inequality gives a noninteractive SQ learner that with probability $\geq 2/3$ learns $C_{MP}^d$ to error $< 1/4$ in $O(\mathsf{poly}(d))$ SQ oracle calls of tolerance $\tau = \Theta\left(\frac{1}{e^{\varepsilon}\mathsf{poly}(d)}\right)$. However, Lemma 3 says that such any such noninteractive SQ learner for $C_{MP}^d$ requires $\Omega\left(2^{(d/3)+2}\right)$ SQ oracle calls, a contradiction. $\square$

The rest of the survey will focus on similar separations for weak distribution-independent learning.

## 5 PAC Learning and Margin Complexity

We now turn to the second work in this survey. At a high level, Daniely and Feldman [7] give a general connection between the number of noninteractive SQ oracle calls sufficient to learn a concept class and the margin complexity of that concept class. This connection enables them to import noninteractive SQ lower bounds as straightforward corollaries of margin complexity lower bounds. We start by defining margin complexity.

**Definition 12.** *Let $X$ be a domain and let $C$ be a class of boolean functions over $X$. The* margin complexity $\mathsf{MC}(C)$ *of $C$ is the minimal nonnegative number $M$ such that there exists natural number $d$ and embedding $\Upsilon\colon X \to \mathcal{B}^d(1)$ of $X$ in the $d$-dimensional $\ell_2$ unit ball with the following property: for every concept $f \in C$, there exists $w \in \mathcal{B}^d(1)$ such that*

$$\min_{x \in X}\{f(x) \cdot \langle w, \Upsilon(x)\rangle\} \geq \frac{1}{M}.$$

We can also think of margin complexity in a geometric way. The margin complexity of $C$ is the smallest nonnegative number such that there exists an embedding of examples in $\mathbb{R}^d$ (for some $d$) where each concept in $C$ is associated with a hyperplane correctly separating labeled examples with margin $\geq \frac{1}{M}$. Thus a smaller margin complexity $M$ corresponds to a larger separating margin for the hyperplanes. We should therefore expect that concept classes with smaller margin complexity are easier to learn than concept classes with larger margin complexity. This is the main result of Daniely and Feldman [7].

Having defined margin complexity, we start with a lemma from previous work by Feldman [13] and Kallweit and Simon [18]. Lemma 5 connects the margin complexity of a class with the existence of an algorithm $\mathcal{A}$ that outputs a collection of concepts such that at least one output concept is correlated with the true concept.

**Lemma 5** ([13, 18])**.** *Let $X$ be a domain and let $C$ be a class of boolean functions over $X$. Suppose there exists an algorithm $\mathcal{A}$ that generates a set of functions $h_1, \ldots, h_m$ such that for every concept $f \in C$ and distribution $\mathcal{D}$ over $X$, with probability at least $\beta > 0$ over any randomness of $\mathcal{A}$, there exists $i \in [m]$ such that $|\mathbb{E}_{x\sim\mathcal{D}}[f(x)h_i(x)]| \geq \frac{1}{m}$. Then*

$$\mathsf{MC}(C) \leq \frac{2m^{3/2}}{\beta}.$$

Note that Lemma 5 is not a result about learning. Instead, it is a result about the concept class $C$ in question. $\mathcal{A}$ examines $C$ and outputs $m$ functions such that with nonzero probability, for any example distribution, every concept in $C$ is correlated with at least one of the $m$ functions. If $\mathcal{A}$ can accomplish this using few functions, it suggests that $C$ is a simple class for distribution-independent learning. Accordingly, Lemma 5 upper bounds the margin complexity of $C$ depending on the size of this "covering" set of functions.

We now have all of the tools needed to prove the main result from Daniely and Feldman [7].

**Theorem 4.** *Let $C$ be a class of boolean functions closed under negation. Assume that for some $m$ there exists a noninteractive distribution-independent SQ learner $\mathcal{A}$ that, with success probability at least $\frac{2}{3}$, PAC learns $C$ with error $< \frac{1}{2}$ using $\leq m$ queries to an SQ oracle with tolerance $\geq 1/m$. Then $\mathsf{MC}(C) \leq 6m^{3/2}$.*

*Proof.* By the same decomposition used to prove Theorem 3, we can rewrite any statistical query function $\phi$ by $\phi(x, f(x)) = g_\phi(x) + f(x)h_\phi(x)$ where

$$g_\phi(x) = \frac{\phi(x,1) + \phi(x,-1)}{2} \text{ and } h_\phi(x) = \frac{\phi(x,1) - \phi(x,-1)}{2}.$$

We now associate each distribution and target concept pair $(\mathcal{D}, f)$ with its own SQ oracle $\mathcal{O}_{\mathcal{D},f}$. Like the SQ oracle defined in the proof of Theorem 3, $\mathcal{O}_{\mathcal{D},f}$ will use the decomposition of $\phi$ to give good approximations when possible and exact answers otherwise. The approximations will depend only on $g_\phi$ and thus be independent of the labels. More formally, we define $\mathcal{O}_{\mathcal{D},f}$ by

$$\mathcal{O}_{\mathcal{D},f}(\phi, \tau) = \begin{cases} \mathbb{E}_{x\sim\mathcal{D}}\left[g_\phi(x)\right] & |\mathbb{E}_{x\sim\mathcal{D}}\left[f(x)h_\phi(x)\right]| < \frac{1}{m} \\ \mathbb{E}_{x\sim\mathcal{D}}\left[\phi(x, f(x))\right] & \text{otherwise} \end{cases}$$

for any tolerance $\tau \geq \frac{1}{m}$. By our decomposition, whenever $\mathcal{O}_{\mathcal{D},f}$ does not return an exact answer, it returns an additive approximation that is better than $\frac{1}{m}$-close. Thus $\mathcal{O}_{\mathcal{D},f}$ is a valid SQ oracle with tolerance $\frac{1}{m}$.

Recall that we have a noninteractive SQ algorithm $\mathcal{A}$ that makes at most $m$ SQ oracle calls. Let $b \sim B$ denote any draw of random bits for $\mathcal{A}$ from some distribution $B$, let $\phi_1^b, \ldots, \phi_{m'}^b$ be the resulting $m' \leq m$ noninteractive SQ oracle calls made, and let $\hat{f}^b_{\mathcal{O}_{\mathcal{D},f}}$ denote the resulting concept output by $\mathcal{A}$ when used with the SQ oracle we defined above. Let $E_1$ be the event that $\hat{f}^b_{\mathcal{O}_{\mathcal{D},f}}$ has error $< \frac{1}{2}$, and let $E_2$ be the event that $|\langle f, h_{\phi_i^b}\rangle_{\mathcal{D}}| \geq \frac{1}{m}$ for some $i \in [m']$. We want to lower bound the probability that $E_2$ occurs. Once we do so, we can use those SQ oracle functions $\phi_1^b, \ldots, \phi_{m'}^b$ as the "covering set" of functions in Lemma 5 to upper bound $\mathsf{MC}(C)$.

**Lemma 6.** *If $\mathbb{P}_{b\sim B}\left[E_1\right] \geq \frac{2}{3}$, then $\mathbb{P}_{b\sim B}\left[E_2\right] > \frac{1}{3}$.*

*Proof.* Conditioned on $\neg E_2$, the answer from $\mathcal{O}_{\mathcal{D},f}$ only depends on $|\mathbb{E}_{x\sim\mathcal{D}}\left[f(x)h_\phi(x)\right]|$ and $g_\phi$, so the answers given by $\mathcal{O}_{\mathcal{D},f}$ are identical to those given by $\mathcal{O}_{\mathcal{D},-f}$. Thus conditioned on $E_1$ and $\neg E_2$, if the target concept is actually $-f$ then $\mathcal{A}$ fails, i.e. $\mathbb{P}_{\mathcal{D}}\left[-f(x) \neq \hat{f}_{\mathcal{O}^b_{\mathcal{D},-f}}\right] > \frac{1}{2}$.

Assume for contradiction that $\mathbb{P}_{b\sim B}\left[E_2\right] \leq \frac{1}{3}$. Then $\mathbb{P}_{b\sim B}\left[E_1 \wedge \neg E_2\right] > \frac{1}{3}$. By the first paragraph, it follows that when the target function is $-f$ and the distribution is $\mathcal{D}$, with probability $> \frac{1}{3}$ over the draw of $b$, $\mathcal{A}$ fails. This contradicts our overall lemma supposition that $\mathbb{P}_{b\sim B}\left[E_1\right] \geq \frac{2}{3}$, so it must be that $\mathbb{P}_{b\sim B}\left[E_2\right] > \frac{1}{3}$. □

Since $f$ and $\mathcal{D}$ were arbitrary, slotting Lemma 6 into Lemma 5 gives

$$\mathsf{MC}(C) \leq \frac{2m^{3/2}}{1/3} = 6m^{3/2}.$$

□

We can use Theorem 4 to transform lower bounds on margin complexity into lower bounds on noninteractive statistical query complexity. This in turn yield noninteractive locally private lower bounds through the SQ-to-local transformation of Theorem 2. In the next section, we go through this process for the concept class of decision lists.

## 6 Separation for Decision Lists

We start by defining the concept class of decision lists [22]. A decision list is, informally, a sequence of conditionals. Each conditional has an associated value. At each conditional, if the conditional is met, then its associated value is the output label. If no conditional is met, a default label is output instead.

**Definition 13** ([22]). *A $d$-dimensional* decision list *is defined by an ordered sequence $f = (a_1, b_1), (a_2, b_2), \dots (a_j, b_j), b$ consisting of literal-label pairs $(a_i, b_i)$ and a final default label $b$. $f$ labels a point $x \in \{0,1\}^d$ by either 1. $b_i$, for the minimal $i$ such that $a_i(x)$ is true or 2. $b$ if no such $b_i$ exists. We denote the class of all $d$-dimensional decision lists by $C_{DL}^d$.*

A decision list therefore checks a list of literals in sequence, returns the label $b_i$ for the first literal $c_i$ that holds on $x$, or returns the default label $b$ if none of the literals hold. This corresponds to any decision process that simply checks rules in sequence.

Using the results of Section 3 and 5 with past work by Kearns [20], it is easy to separate noninteractive and interactive locally private learning of decision lists.

**Theorem 5.** *In the distribution-independent setting, there exists an interactive $\varepsilon$-locally private protocol $\mathcal{A}$ that learns $C_{DL}^d$ to error $< \frac{1}{2}$ with probability $\frac{2}{3}$ using $O\left(\frac{\mathsf{poly}(d)}{\varepsilon^2}\right)$ samples. However, any noninteractive $\varepsilon$-locally private protocol that learns $C_{DL}^d$ to error $< \frac{1}{2}$ with probability $\frac{2}{3}$ must use $\frac{2^{\Omega(d^{1/3})}}{e^\varepsilon}$ samples.*

*Proof.* Interactive upper bound: Kearns [20] gives an interactive weak SQ learner for $C_{DL}^d$ based on the original decision list learner given by Rivest [22]. It achieves success probability $\frac{2}{3}$ using $O(\mathsf{poly}(d))$ statistical queries with minimum tolerance $\tau = \Omega(\mathsf{poly}(1/d))$. Combining this SQ learner with Theorem 1 gives the result.

Noninteractive lower bound: First, we verify that $C_{DL}^d$ is closed under negation. This is because we can negate any decision list by negating all of its label bits $b_1, \dots, b_j, b$. Second, we use a lower bound on the margin complexity of decision lists from Buhrman et al. [5].

**Lemma 7** ([5]). $\mathsf{MC}\left(C_{DL}^d\right) = 2^{\Omega(d^{1/3})}$.

Next, we combine Lemma 7 with Theorem 4 to get that any noninteractive distribution-independent SQ algorithm that learns $C_{DL}^d$ to error $< \frac{1}{2}$ with probability $\geq \frac{2}{3}$ must have a worst-case number of SQ oracle calls

$$m \geq \left(\frac{\mathsf{MC}\left(C_{DL}^d\right)}{6}\right)^{2/3} = 2^{\Omega(d^{1/3})}. \tag{3}$$

We now turn to Theorem 2. If there exists a noninteractive $\varepsilon$-locally private algorithm that distribution-independently PAC learns $C_{DL}^d$ with probability at least $\frac{2}{3}$ and error $< 1/2$ using $n = \frac{2^{o(d^{1/3})}}{e^\varepsilon}$ samples, then Theorem 2 and Markov's inequality gives a noninteractive SQ learner that learns $C_{DL}^d$ to error $< \frac{1}{2}$ with probability $\geq \frac{2}{3}$ in $2^{o(d^{1/3})}$ SQ oracle calls. This contradicts Equation 3, so no such noninteractive $\varepsilon$-locally private protocol exists. □

## References

[1] Differential Privacy Team Apple. Learning with privacy at scale. Technical report, Apple, 2017.

[2] Amos Beimel, Kobbi Nissim, and Eran Omri. Distributed private data analysis: Simultaneously solving how and what. In *International Cryptology Conference (CRYPTO)*, 2008.

[3] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Symposium on Operating Systems Principles (SOSP)*, 2017.

[4] Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly learning dnf and characterizing statistical query learning using fourier analysis. In *Symposium on the Theory of Computing (STOC)*, 1994.

[5] Harry Buhrman, Nikolay Vereshchagin, and Ronald de Wolf. On computation and communication with small bias. In *Conference on Computational Complexity (CCC)*, 2007.

[6] TH Hubert Chan, Elaine Shi, and Dawn Song. Optimal lower bound for differentially private multi-party aggregation. In *European Symposium on Algorithms (ESA)*, 2012.

[7] Amit Daniely and Vitaly Feldman. Learning without interaction requires separation. In *Neural Information and Processing Systems (NeurIPS)*, 2019.

[8] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Neural Information Processing Systems (NIPS)*, pages 3574–3583, 2017.

[9] John Duchi and Ryan Rogers. Lower bounds for locally private estimation via communication complexity. In *Conference on Learning Theory (COLT)*, 2019.

[10] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy, data processing inequalities, and statistical minimax rates. *arXiv preprint arXiv:1302.3203*, 2013.

[11] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, 2006.

[12] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

[13] Vitaly Feldman. Evolvability from learning algorithms. In *Symposium on the Theory of Computing (STOC)*, 2008.

[14] Garfinkel, Simson L. Deploying differential privacy for the 2020 census of population and housing. `census.gov/content/dam/Census/newsroom/press-kits/2019/jsm/presentation-deploying-differential-privacy-for-the-2020-census-of-pop-and-housing.pdf`, 2019. Accessed: 09-12-2019.

[15] Miguel Guevara. Enabling developers and organizations to use differential privacy. `developers.googleblog.com/2019/09/enabling-developers-and-organizations.html`, 2019. Accessed: 09-12-2019.

[16] Matthew Joseph, Jieming Mao, Seth Neel, and Aaron Roth. The role of interactivity in local differential privacy. In *Foundations of Computer Science (FOCS)*, 2019.

[17] Matthew Joseph, Jieming Mao, and Aaron Roth. Exponential separations in local differential privacy. In *Symposium on Discrete Algorithms (SODA)*, 2020.

[18] Michael Kallweit and Hans Ulrich Simon. A close look to margin complexity and related parameters. In *Conference on Learning Theory (COLT)*, 2011.

[19] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *Foundations of Computer Science (FOCS)*, 2011.

[20] Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 1998.

[21] Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM Journal on Computing*, 1993.

[22] Ronald L Rivest. Learning decision lists. *Machine learning*, 1987.

[23] Adam Smith, Abhradeep Thakurta, and Jalaj Upadhyay. Is interaction necessary for distributed private learning? In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 58–77, 2017.

[24] Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017.

[25] Leslie G Valiant. A theory of the learnable. In *Symposium on the Theory of Computing (STOC)*, 1984.

[26] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.