2008 49th Annual IEEE Symposium on Foundations of Computer Science

What Can We Learn Privately?*

Shiva Prasad Kasiviswanathan† Homin K. Lee‡ Kobbi Nissim§ Sofya Raskhodnikova†
Adam Smith†

Abstract

Learning problems form an important category of computational tasks that generalizes many of the computations researchers apply to large real-life data sets. We ask: what concept classes can be learned privately, namely, by an algorithm whose output does not depend too heavily on any one input or specific training example? More precisely, we investigate learning algorithms that satisfy differential privacy*, a notion that provides strong confidentiality guarantees in the contexts where aggregate information is released about a database containing sensitive information about individuals. We present several basic results that demonstrate general feasibility of private learning and relate several models previously studied separately in the contexts of privacy and standard learning.*

1. Introduction

The data privacy problem in modern databases is similar to that faced by statistical agencies and medical researchers: to learn and publish global analyses of a population while maintaining confidentiality of the participants in a survey. There is a vast body of work on this problem in statistics and computer science. However, until recently, most schemes proposed in the literature lacked rigorous analysis of privacy and utility.

A recent line of work, initiated by Dinur and Nissim [15] and called *private data analysis*, seeks to place data privacy on firmer theoretical foundations and has been successful at formulating a strong, yet attainable privacy definition. The notion of *differential privacy* [17] that emerged from this line of work provides rigorous guarantees even in the presence of a malicious adversary with access to arbitrary auxiliary information. It requires that whether an individual supplies her actual or fake information has almost no effect on the outcome of the analysis.

Given this definition, it is natural to ask: what computational tasks can be performed while maintaining privacy? Research on data privacy, to the extent that it formalizes precise goals, has mostly focused on *function evaluation* ("what is the value of $f(\mathrm{z})$?"), namely, how much privacy is possible if one wishes to release (an approximation to) a particular function f, evaluated on the database z. (A notable exception is the recent work of McSherry and Talwar, using differential privacy in the design of auction mechanisms [34]). Our goal is to expand the utility of private protocols by examining which other computational tasks can be performed in a privacy-preserving manner.

Private Learning. In this work, we ask what can be learned *privately*, namely, by an algorithm whose output does not depend too heavily on any one input or specific training example? Our goal is a broad understanding of the resources required for private learning in terms of samples, computation time, and interaction. We examine two basic notions from learning: Valiant's probabilistically approximately correct (PAC) learning [39] model and Kearns' statistical query (SQ) model [29].

Informally, a concept is a function from examples to labels, and a class of concepts is learnable if for any distribution $\mathcal{D}$ on examples, one can, given limited access to examples sampled from $\mathcal{D}$ labeled according to some target concept c, find a small circuit (hypothesis) which predicts c's labels with high probability over future examples taken from the same distribution. In the PAC model, a learning algorithm can access a polynomial number of labeled examples. In the SQ model, instead of accessing examples directly, the learner can specify some properties (i.e., predicates) on the examples, for which he is given an estimate, up to an additive polynomially small error, of the probability that a random example chosen from $\mathcal{D}$ satisfies the prop-

*All omitted proofs and details appear in the full version [27].

†Department of Computer Science and Engineering, Pennsylvania State University, {kasivisw,sofya,asmith}@cse.psu.edu. S.P.K. is supported in part by NSF award CCF-072891. S.R. and A.S. are supported in part by NSF award CCF-0729171.

‡Department of Computer Science, Columbia University, hkl7@columbia.edu

§Department of Computer Science, Ben-Gurion University, kobbi@cs.bgu.ac.il. Supported in part by the Israel Science Foundation (grant 860/06), and by the Frankel Center for Computer Science.

0272-5428/08 $25.00 © 2008 IEEE
DOI 10.1109/FOCS.2008.27

erty. PAC learning is strictly stronger than the SQ learning [29].

We model a statistical database as a vector $z = (z_1, \cdots, z_n)$, where each entry has been contributed by an individual. When analyzing how well a private algorithm learns a concept class, we assume that entries z_i of the database are random examples generated i.i.d. from the underlying distribution $\mathcal{D}$ and labeled by a target concept c. This is exactly how (not necessarily private) learners are analyzed. For instance, an example might consist of an individual's gender, age, and blood pressure history, and the label, whether this individual has had a heart attack. The algorithm has to learn to predict whether an individual has had a heart attack, based on gender, age, and blood pressure history, generated according to $\mathcal{D}$.

We require a private algorithm to keep entire examples (not only the labels) confidential. In the scenario above, it translates to not revealing each participant's gender, age, blood pressure history, and heart attack incidence. More precisely, the output of a private learner should not be significantly affected if a particular example z_i is replaced with arbitrary z_i', for all z_i and z_i'. In contrast to correctness or utility, which is analyzed with respect to distribution $\mathcal{D}$, differential privacy is a worst-case notion. Hence, when we analyze the privacy of our learners we do not make any assumptions on the underlying distribution. Such assumptions are fragile and, in particular, would fall apart in the presence of auxiliary knowledge that the adversary might have: conditioned on the adversary's auxiliary knowledge, the distribution over examples might look very different from $\mathcal{D}$.

1.1. Our Contributions

We introduce and formulate private learning problems, as discussed above, and develop novel algorithmic tools and bounds on the sample size required by private learning algorithms. Our results paint a picture of the classes of learning problems that are solvable subject to privacy constraints. Specifically, we provide:

(1) A Private Version of Occam's Razor. We present a generic private learning algorithm. For any concept class $\mathcal{C}$, we give a distribution-free differentially-private agnostic PAC learner for $\mathcal{C}$ that uses a number of samples proportional to $\log |\mathcal{C}|$. This is a private analogue of *Occam's razor*, a basic sample complexity bound from the non-private learning setting. The sample complexity of our version is similar to that of the original, although the private algorithm is very different. As in Occam's razor, the learning algorithm is not necessarily computationally efficient.

(2) An Efficient Private Learner for Parity. We give a computationally efficient, distribution-free differentially private PAC learner for the class of *parity* functions[1] over $\{0,1\}^d$. The sample and time complexity are comparable to that of the best non-private learner.

(3) Equivalence of Local ("Randomized Response") and SQ Learning. We precisely characterize the power of *local*, or *randomized response*, private learning algorithms. Local algorithms are a special (practical) class of private algorithms and are popular in the data mining and statistics literature. They add randomness to each individual's data independently before processing the input. We show that a concept class is learnable by a local differentially private algorithm if and only if it is learnable in the *statistical query* (SQ) model. This equivalence relates notions that were conceived in very different contexts.

(4) Separation of Interactive and Noninteractive Local Learning. Local algorithms can be *noninteractive*, that is, using one round of interaction with individuals holding the data, or *interactive*, that is, using more than one round (and in each receiving randomized responses from individuals). We construct a concept class, called *masked-parity*, that is efficiently learnable by *interactive* local algorithms, but requires an exponential (in the dimension) number of samples to be learned by a *noninteractive* local algorithm. The equivalence **(3)** of local and SQ learning shows that interaction in local algorithms corresponds to *adaptivity* in SQ algorithms. The *masked-parity* class thus also separates adaptive and nonadaptive SQ learning.

1.1.1 Implications

"Anything" learnable is privately learnable using few samples. The generic agnostic learner **(1)** has an important consequence: if some concept class $\mathcal{C}$ is learnable by any algorithm, not necessarily a private one, whose output length in bits is polynomially bounded, then $\mathcal{C}$ is learnable privately using a polynomial number of samples (possibly in exponential time). This result establishes the basic feasibility of private learning: it was not clear *a priori* how severely privacy affects sample complexity, even ignoring computation time.

Learning with noise is different from private learning. There is an intuitively appealing similarity between learning from noisy examples and private learning: algorithms for both problems must be robust to small variations in the data. This apparent similarity is strengthened by a result of Blum *et al.* [9] showing that any algorithm in Kearns'

[1] While the Occam's razor result **(1)** extends easily to "agnostic" learning (defined below), the learner for parity does not. The limitation is not surprising, since even non-private agnostic learning of parity is at least as hard as learning parity with random noise.

statistical query (SQ) model [29] can be implemented in a differentially private manner. SQ was introduced to capture a class of noise-resistant learning algorithms. These algorithms access their input only through a sequence of approximate averaging queries. One can *privately* approximate the average of a function with values in $[0, 1]$ over the data set of n individuals to within additive error $O(1/n)$ (Dwork and Nissim [18]). Thus, one can simulate the behavior of an SQ algorithm privately, query by query.

Our efficient private learner for parity **(2)** dispels the similarity between learning with noise and private learning. First, SQ algorithms provably require exponentially many (in the dimension) queries to learn parity [29]. More compellingly, learning parity with noise is thought to be computationally hard, and has been used as the basis of several cryptographic primitives (e.g., [11, 25, 4, 38]).

Limitations of local ("randomized response") algorithms. *Local* algorithms (also referred to as *randomized response*, *input perturbation*, *Post Randomization Method (PRAM)*, and *FRAPP*) have been studied extensively in the context of privacy-preserving data mining, both in statistics and computer science (e.g., [41, 2, 1, 3, 40, 20, 35, 26]). Roughly, a local algorithm accesses each individual's data via independent randomization operators. Local algorithms were introduced to encourage truthfulness in surveys: respondents who know that their data will be randomized are more likely to answer honestly (Warner [41]). The accepted privacy requirement for local algorithms is equivalent to imposing differential privacy on each randomization operator [20]. Local algorithms are popular because they are easy to understand and implement. In the extreme case, users can retain their data and apply the randomization operator themselves, using a physical device [41, 36] or a cryptographic protocol [5].

The equivalence between local and SQ algorithms **(3)** is a powerful tool that allows us to apply results from learning theory. In particular, since parity is not learnable with a small number of SQ queries [29] but is PAC learnable privately **(2)**, we get that local algorithms require exponentially more data for some learning tasks than do general private algorithms. Our results also imply that local algorithms are strictly *less* powerful than (non-private) algorithms for learning with classification noise because subexponential (non-private) algorithms can learn parity with noise [11].

Adaptivity in SQ algorithms is important. Just as local algorithms can be interactive, SQ algorithms can be *adaptive*, that is, the averaging queries they make may depend on answers to previous queries. The equivalence of SQ and local algorithms **(3)** preserves interaction/adaptivity: a concept class is nonadaptively SQ learnable if and only if it is noninteractively locally learnable. The masked parity class **(4)** shows that interaction (resp., adaptivity) adds considerable power to local (resp., SQ) algorithms.

Most of the reasons that local algorithms are so attractive in practice, and have received such attention, apply only to noninteractive algorithms. This suggests that further investigating the power of nonadaptive SQ learners is an important problem. For example, the SQ algorithm for learning conjunctions [32] is nonadaptive, but SQ formulations of the perceptron and k-means algorithms [9] seem to rely on adaptivity.

Understanding the "price" of privacy for learning problems. The SQ result of Blum *et al.* [9] and our learner for parity **(2)** provide efficient (i.e., polynomial time) private learners for essentially all the concept classes known (by us) to have efficient non-private distribution-free learners. Finding a concept class that can be learned efficiently, but not privately and efficiently, remains an interesting and important question.

Our results also lead to questions of optimal sample complexity for learning problems of practical importance. The private simulation of SQ algorithms in [9] uses a factor of approximately $\sqrt{t}/\epsilon$ more data points than the naïve non-private implementation, where t is the number of SQ queries and ϵ is the parameter of differential privacy (typically a small constant). In contrast, the generic agnostic learner **(1)** uses a factor of at most $1/\epsilon$ more samples than the corresponding non-private learner. For parity, our private learner uses a factor of roughly $1/\epsilon$ more samples than, and about the same computation time as, the non-private learner. What, then, is the additional cost of privacy when learning practical concept classes (half-planes, low-dimensional curves, etc)? Can the theoretical sample bounds of **(1)** be matched by (more) efficient learners?

1.1.2 Techniques

Our generic private learner **(1)** adapts the exponential sampling technique of [34], developed in the context of auction design. Our use of the exponential mechanism inspired an elegant *subsequent* result of Blum, Liggett, and Roth [12] (BLR) on simultaneously approximating many different functions. Their result can, in turn, be used to derive a version of our statement for hypotheses of bounded VC-dimension (rather than bounded cardinality), when the space of examples has bounded size. The generic private learner from this paper and that implied by the BLR result are incomparable: roughly, our original result requires discretizing (quantizing) the set of hypotheses, whereas the BLR result requires discretizing the space of examples. Neither achieves the generality of the original Vapnik-Chernovenkis bound (see [32]), which requires only bounded VC-dimension and makes no assumptions on

the cardinality of either the hypothesis or example space.

The efficient private learner for parity **(2)** uses a very different technique, based on sampling, running a non-private learner, and occasionally refusing to answer based on delicately calibrated probabilities. Running a non-private learner on a random subset of examples is a very intuitive approach to building private algorithms, but it is not private in general. The private learner for parity illustrates both why this technique can leak private information and how it can sometimes be repaired based on special (in this case, algebraic) structure.

The interesting direction of the equivalence between SQ and local learners **(3)** is proved via a simulation of any local algorithm by a corresponding SQ algorithm. We found this simulation surprising since local protocols can, in general, have very complex structure (see, e.g., [20]). The SQ algorithm proceeds by a direct simulation of the output of the randomization operators. For a given input distribution $\mathcal{D}$ and any operator R, one can sample from the corresponding output distribution $R(\mathcal{D})$ via rejection sampling. We show that if R is differentially private, the rejection probabilities can be approximated via low-accuracy SQ queries to $\mathcal{D}$.

Finally, the separation between adaptive and nonadaptive SQ **(4)** uses a Fourier analytic argument inspired by Kearns' SQ lower bound for parity [29].

1.2. Related Work

The literature on differential privacy has focused on function approximation tasks, with the exception of [34], devoted to mechanism design. Blum *et al.* [9] considered a specific class of learning algorithms (SQ), and showed that algorithms in the class could be simulated using function evaluations. In an independent unpublished work, Chaudhuri, Dwork, and Talwar considered a version of private learning in which privacy is afforded only to input labels, but not to examples.

Dwork *et al.* [17] separated interactive and noninteractive private protocols in the *centralized* model, where the user accesses the data via a server that runs differentially private algorithms on the database and sends back the answers. Any example of a computation that cannot be performed noninteractively in the centralized model must rely on the fact that the computational task is not defined until after the first answer from the server is received. (Otherwise, the user can send an algorithm for that task to the server holding the data, thus obviating the need for interaction.) Our separation of interactive and noninteractive local protocols **(3)** is of a different nature: the computational task that is hard for noninteractive local algorithms – learning *masked parity* – is defined in advance.

In the machine learning literature, several notions similar to differential privacy have been explored under the rubric of "algorithmic stability" [14, 30, 13, 33, 19, 7]. The most closely related notion is *change-one error stability*, which measures how much the generalization error changes when an input are changed (see the survey [33]). In contrast, differential privacy measures how the distribution over the entire output changes—a more complex measure of stability (in particular, differential privacy implies change-one error stability). A different notion, stability under resampling of the data from a given distribution [8, 7], is connected to the sample-aggregate method of [37] but is not directly relevant to the techniques considered here.

2. Preliminaries

We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. Logarithms base 2 and base e are denoted by $\log$ and $\ln$, respectively. $\mathcal{A}(x)$ is the probability distribution over outputs of a randomized algorithm $\mathcal{A}$ on input x. The *statistical difference* between distributions $\mathbb{P}$ and $\mathbb{Q}$ on a discrete space D is defined as $\max_{S \subset D} |\mathbb{P}(S) - \mathbb{Q}(S)|$.

2.1. Databases and Privacy

A statistical database is a vector $\mathrm{z} = (z_1, \ldots, z_n)$ over a domain D, where each entry $z_i \in D$ represents information contributed by one individual. Databases z and z' are *neighbors* if $z_i \neq z'_i$ for exactly one $i \in [n]$ (i.e., the Hamming distance between z and z' is 1).

A (randomized) algorithm (in our context, this will usually be a learning algorithm) is private if neighboring databases induce nearby distributions on its outcomes:

Definition 2.1 (ϵ-differential privacy [17])**.** *A randomized algorithm $\mathcal{A}$ is ϵ-differentially private if for all neighboring databases z, z', and for all sets $\mathcal{S}$ of outputs, $\Pr[\mathcal{A}(\mathrm{z}) \in \mathcal{S}] \leq \exp(\epsilon) \cdot \Pr[\mathcal{A}(\mathrm{z}') \in \mathcal{S}]$. The probability is taken over the random coins of $\mathcal{A}$.*

Claim 2.2 (Composition and Post-processing [16, 37, 34, 28])**.** *If a randomized algorithm $\mathcal{A}$ runs k algorithms $\mathcal{A}_1, ..., \mathcal{A}_k$, where each $\mathcal{A}_i$ is ϵ_i-differentially private, and outputs a function of the results (that is, $\mathcal{A}(x) = g(\mathcal{A}_1(x), \mathcal{A}_2(x), ..., \mathcal{A}_k(x))$ for some probabilistic algorithm g), then $\mathcal{A}$ is $(\sum_{i=1}^{k} \epsilon_i)$-differentially private.*

2.2. Preliminaries from Learning Theory

A concept is a function that labels *examples* taken from the domain X by the elements of the range Y. We focus on binary classification problems, where the range Y is $\{0, 1\}$ (or, equivalently, $\{+1, -1\}$). A *concept class* $\mathcal{C}$ is a set of concepts. It comes implicitly with a way to represent

concepts; $size(c)$ is the size of the (smallest) representation of c under the given representation scheme. The domain of the concepts in $\mathcal{C}$ is understood to be an ensemble $X = \{X_d\}_{d\in\mathbb{N}}$ where the representation of elements in X_d is of size at most d. (We use the parameter d to formulate asymptotic complexity notions.) The concept classes are ensembles $\mathcal{C} = \{\mathcal{C}_d\}_{d\in\mathbb{N}}$ where $\mathcal{C}_d$ is the class of concepts from X_d to $\{0,1\}$. When the size parameter is clear from the context or not important, we omit the subscript in $X_d, \mathcal{C}_d$.

Let $\mathcal{D}$ be a distribution over labeled examples in $X_d \times \{0,1\}$. A learning algorithm is given access to $\mathcal{D}$ (the method for accessing $\mathcal{D}$ depends on the type of learning algorithm). It outputs a hypothesis $h : X_d \to \{0,1\}$ from a hypothesis class $\mathcal{H} = \{\mathcal{H}_d\}_{d\in\mathbb{N}}$. The goal is to minimize the *misclassification error* of h on $\mathcal{D}$, defined as $error(h) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$. The success of a learning algorithm is quantified by parameters α and β, where α is the desired error and β bounds the probability of failure to output a hypothesis with this error.

A learning algorithm is usually given access to an oracle that produces i.i.d. samples from $\mathcal{D}$. Equivalently, one can view the learning algorithm's input as a list of n labeled examples, i.e., $\mathrm{z} \in D^n$ where $D = X_d \times \{0,1\}$.

PAC learning algorithms are frequently designed assuming a promise that the examples are labeled consistently with some *target* concept c from a class $\mathcal{C}$: namely, $c \in \mathcal{C}_d$ and $y = c(x)$ for all (x, y) in the support of $\mathcal{D}$. In that case, we can think of $\mathcal{D}$ as a distribution only over examples X_d. To avoid ambiguity, we use $\mathcal{X}$ to denote a distribution over X_d. In the PAC setting, $error(h) = \Pr_{x\sim\mathcal{X}}[h(x) \neq c(x)]$.

Definition 2.3 (PAC Learning)**.** *A concept class $\mathcal{C}$ over X is* PAC learnable using hypothesis class $\mathcal{H}$ *if there exist an algorithm $\mathcal{A}$ and a polynomial $poly(\cdot,\cdot,\cdot)$ such that for all $d \in \mathbb{N}$, all concepts $c \in \mathcal{C}_d$, all distributions $\mathcal{X}$ on X_d, and all $\alpha, \beta \in (0, 1/2)$, given inputs α, β and $\mathrm{z} = (z_1, \cdots, z_n)$, where $n = poly(d, 1/\alpha, \log(1/\beta))$, $z_i = (x_i, c(x_i))$ and x_i are drawn i.i.d. from $\mathcal{X}$ for $i \in [n]$, algorithm $\mathcal{A}$ outputs a hypothesis $h \in \mathcal{H}$ satisfying*

$$\Pr[error(h) \leq \alpha] \geq 1 - \beta. \tag{1}$$

The probability is taken over the random choice of the examples z *and the coin tosses of $\mathcal{A}$.*

Class $\mathcal{C}$ is PAC learnable if there exists some hypothesis class $\mathcal{H}$ and a PAC learner $\mathcal{A}$ such that $\mathcal{A}$ PAC learns $\mathcal{C}$ using $\mathcal{H}$. Class $\mathcal{C}$ is efficiently *PAC learnable if $\mathcal{A}$ runs it time polynomial in $d, 1/\alpha$, and $\log(1/\beta)$.*

Remark: Our definition deviates slightly from the standard one (see, e.g., [32]) in that we do not take into consideration the size of the concept c. This choice allows us to treat PAC learners and agnostic learners identically.

Agnostic learning [23, 31] is an extension of PAC learning that removes assumptions about the target concept. In this setting, $error(h) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$.

Definition 2.4 (Agnostic Learning)**.** (Efficiently) agnostically learnable *is defined identically to (efficiently) PAC learnable with two exceptions:* (i) *the data are drawn from an arbitrary distribution $\mathcal{D}$ on $X_d \times \{0,1\}$;* (ii) *instead of Equation 1, the output of $\mathcal{A}$ has to satisfy:*

$$\Pr[error(h) \leq OPT + \alpha] \geq 1 - \beta,$$

where $OPT = \min_{f\in\mathcal{C}_d}\{error(f)\}$.

Definitions 2.3 and 2.4 capture *distribution-free* learning, in that they do not assume a particular form for the distributions $\mathcal{X}$ or $\mathcal{D}$.

3. Private PAC and Agnostic Learning

We define private PAC learners as algorithms that satisfy definitions of both differential privacy and PAC learning. We emphasize that these are qualitatively different requirements. Learning must succeed on average over a set of examples drawn i.i.d. from $\mathcal{D}$ (often under the additional promise that $\mathcal{D}$ is consistent with a concept from a target class). Differential privacy, in contrast, must hold in the worst case, with no assumptions on consistency.

Definition 3.1 (Private PAC Learning)**.** *Let d, α, β be as in Definition 2.3 and $\epsilon > 0$. Concept class $\mathcal{C}$ is privately PAC learnable using hypothesis class $\mathcal{H}$ if there exists an algorithm $\mathcal{A}$ that takes inputs $\epsilon, \alpha, \beta,$ z, where n, the number of labeled examples in* z*, is polynomial in $1/\epsilon, d, 1/\alpha, \log(1/\beta)$, and satisfies*

a. [Privacy] *For all $\epsilon > 0$, algorithm $\mathcal{A}(\epsilon,\cdot,\cdot,\cdot)$ is ϵ-differentially private (Definition 2.1);*

b. [Utility] *$\mathcal{A}$ PAC learns $\mathcal{C}$ using $\mathcal{H}$ (Definition 2.3).*

$\mathcal{C}$ is efficiently *privately PAC learnable if $\mathcal{A}$ runs in time polynomial in $d, 1/\epsilon, 1/\alpha$, and $\log(1/\beta)$.*

Definition 3.2 (Private Agnostic Learning)**.** *(Efficient)* private agnostic learning *is defined analogously to (efficient) private PAC learning with Definition 2.4 replacing Definition 2.3 in the utility condition.*

Evaluating the quality of a particular hypothesis is easy: one can privately compute the fraction of the data it classifies correctly (enabling cross-validation) using the sum query framework of [9]. The difficulty of constructing private learners lies in finding a good hypothesis in what is typically an exponentially large space.

3.1. A Generic Private Agnostic Learner

In this section, we present a private analogue of the cardinality version of Occam's razor, a classical construction of PAC learners that weeds out all *bad* hypotheses given a number of labeled examples that is logarithmic in the size of the hypothesis class (see [32] for details). Our generic private learner is based on the exponential mechanism of McSherry and Talwar [34].

Let $q : D^n \times \mathcal{H}_d \to \mathbb{R}$ take a database z and a candidate hypothesis h, and assign it a score $q(\mathrm{z}, h) = -|\{i : x_i \text{ is misclassified by } h, \textit{ i.e., } y_i \neq h(x_i)\}|$. That is, the score is minus the number of points in z misclassified by h. The classic Occam's razor argument assumes a learner that selects the hypothesis with maximum score (that is, minimum empirical error). Instead, our private learner $\mathcal{A}_q^\epsilon$ is defined to sample a random hypothesis with probability dependent on its score:

$$\mathcal{A}_q^\epsilon(\mathrm{z}): \quad \text{Output hypothesis } h \in \mathcal{H}_d \text{ with probability proportional to } \exp\left(\tfrac{1}{2}\epsilon q(\mathrm{z}, h)\right).$$

Since the score ranges from $-n$ to 0, hypotheses with low empirical error are exponentially more likely to be selected than ones with high error.

Algorithm $\mathcal{A}_q^\epsilon$ fits the framework of McSherry and Talwar, and so is ϵ-differentially private. This follows from the fact that changing one entry z_i in the database z can change the score by at most 1.

Lemma 3.3 (following [34])**.** *The algorithm $\mathcal{A}_q^\epsilon$ is ϵ-differentially private.*

A similar exponential weighting algorithm was considered by [22] for constructing binary classifiers with good generalization error bounds. We are not aware of any direct connection between the two results. Also note that, except for the case where $|\mathcal{H}_d|$ is polynomial, the exponential mechanism $\mathcal{A}_q^\epsilon(\mathrm{z})$ does not necessarily yield a polynomial time algorithm.

Theorem 3.4 (Private version of Occam's Razor)**.** *For all $d \in \mathbb{N}$, any concept class $\mathcal{C}_d$ whose cardinality is at most $\exp(\mathrm{poly}(d))$ is privately agnostically learnable using $\mathcal{H}_d = \mathcal{C}_d$. More precisely, the learner uses $n = O((\ln|\mathcal{H}_d| + \ln\frac{1}{\beta}) \cdot \max\{\frac{1}{\epsilon\alpha}, \frac{1}{\alpha^2}\})$ labeled examples from $\mathcal{D}$, where ϵ, α, and β are parameters of the private learner. (The learner might not be efficient.)*

Proof. Let $\mathcal{A}_q^\epsilon$ be as defined above. The privacy condition in Definition 3.1 is satisfied by Lemma 3.3.

We now show that the utility condition is also satisfied. Consider the event $E = \{\mathcal{A}_q^\epsilon(\mathrm{z}) = h \text{ with } error(h) > \alpha + OPT\}$. We want to prove that $\Pr[E] \leq \beta$. Define the training error of h as

$$error_T(h) = \left|\{i \in [n] \,|\, h(x_i) \neq y_i\}\right|/n = -q(\mathrm{z}, h)/n\,.$$

By Chernoff-Hoeffding bounds,

$$\Pr\left[|error(h) - error_T(h)| \geq \rho\right] \leq 2\exp(-2n\rho^2)$$

for all hypotheses $h \in \mathcal{H}_d$. Hence,

$$\Pr\left[|error(h) - error_T(h)| \geq \rho \text{ for some } h \in \mathcal{H}_d\right] \leq 2|\mathcal{H}_d|\exp(-2n\rho^2).$$

We now analyze $\mathcal{A}_q^\epsilon(\mathrm{z})$ conditioned on the event that for all $h \in \mathcal{H}_d$, $|error(h) - error_T(h)| < \rho$. For every $h \in \mathcal{H}_d$, the probability that $\mathcal{A}_q^\epsilon(\mathrm{z}) = h$ is

$$\begin{aligned}
& \frac{\exp(-\frac{\epsilon}{2}\cdot n\cdot error_T(h))}{\sum_{h'\in\mathcal{H}_d}\exp(-\frac{\epsilon}{2}\cdot n\cdot error_T(h'))} \\
\leq\ & \frac{\exp\left(-\frac{\epsilon}{2}\cdot n\cdot error_T(h)\right)}{\max_{h'\in\mathcal{H}_d}\exp(-\frac{\epsilon}{2}\cdot n\cdot error_T(h'))} \\
=\ & \exp\left(-\frac{\epsilon}{2}\cdot n\cdot (error_T(h) - \min_{h'\in\mathcal{H}_d} error_T(h'))\right) \\
\leq\ & \exp\left(-\frac{\epsilon}{2}\cdot n\cdot (error_T(h) - (OPT+\rho))\right).
\end{aligned}$$

Hence, the probability that $\mathcal{A}_q^\epsilon(\mathrm{z})$ outputs a hypothesis $h \in \mathcal{H}_d$ such that $error_T(h) \geq OPT + 2\rho$ is at most $|\mathcal{H}_d|\exp(-\epsilon n\rho/2)$.

Now set $\rho = \alpha/3$. If $error(h) \geq OPT + \alpha$ then $|error(h) - error_T(h)| \geq \alpha/3$ or $error_T(h) \geq OPT + 2\alpha/3$. Thus $\Pr[E] \leq |\mathcal{H}_d|(2\exp(-2n\alpha^2/9) + \exp(-\epsilon n\alpha/6)) \leq \beta$ where the last inequality holds for $n \geq 6\left((\ln|\mathcal{H}_d| + \ln\frac{1}{\beta})\cdot\max\{\frac{1}{\epsilon\alpha}, \frac{1}{\alpha^2}\}\right)$. □

In the non-private case one can also bound the sample size (of a PAC learner) in terms of the VC-dimension of the concept class. A result in [12], inspired by the initial version of our work, can be used to get a private analogue of this VC-dimension bound, assuming a small domain size for examples. As explained in Section 1.1.2, this statement and the generic private learner from Theorem 3.4 are incomparable. Details are deferred to the full version [27].

Proposition 3.5. *Every concept class $\mathcal{C}_d$ is privately agnostically learnable using hypothesis class $\mathcal{H}_d = \mathcal{C}_d$ with $O\left(\frac{VCDIM(\mathcal{C}_d)\log(1/\alpha)d}{\alpha^3\epsilon} + \frac{\log(1/\beta)}{\alpha\epsilon}\right)$ labeled examples from $\mathcal{D}$. Here, ϵ, α, and β are parameters of the private agnostic learner, $d = \log|\mathcal{H}_d|$ and $VCDIM(\mathcal{C}_d)$ is the VC-dimension of $\mathcal{C}_d$. (The learner is not necessarily efficient.)*

Remark: In the non-private agnostic case, the standard VC-dimension (see, e.g., [6]) bound states that $O\left(\frac{VCDIM(\mathcal{C}_d)\log(1/\alpha)}{\alpha^2} + \frac{\log(1/\beta)}{\alpha^2}\right)$ labeled examples suffice to agnostically learn a concept class $\mathcal{C}_d$. Therefore, the current VC-dimension-based upper bounds on the sample size of private and non-private agnostic learners differ by a factor of $O(d/(\alpha\epsilon))$ for moderate values of β.

4. An Efficient Private Learner for PARITY

Let PARITY be the class of parity functions $c_r : \{0,1\}^d \to \{0,1\}$ indexed by $r \in \{0,1\}^d$, where $c_r(x) = r \odot x$ denotes the inner product modulo 2. In this section, we present an efficient private PAC learning algorithm for PARITY. The standard (non-private) PAC learner for PARITY [24, 21] looks for the hidden vector r by solving a system of linear equations imposed by examples $(x_i, c_r(x_i))$ that the algorithm sees. It outputs an arbitrary vector consistent with the examples, i.e., in the solution space of the system of linear equations. One important point is that the private algorithm has to be defined on all databases z, even the ones which are not consistent with any parity function because the privacy guarantee has to hold for all neighboring databases z and z′. In particular, we have to guarantee that the probability that the algorithm fails (does not find a consistent hypothesis and, in our notation, outputs $\perp$) is similar for all neighbors z and z′. Observe that this is not true for the standard learning algorithm for PARITY.

We first present a private algorithm $\mathcal{A}$ for learning PARITY with failure probability $1/2+\beta'$. Later we amplify its success probability and get a private PAC learner $\mathcal{A}^*$ for PARITY. Intuitively, the reason PARITY can be learned privately is that when a new example (corresponding to a new linear constraint) is added, the space of consistent hypotheses shrinks by at most a factor of 2. This holds unless the new constraint is inconsistent with previous constraints. In the latter case, the size of the space of consistent hypotheses goes to 0. Thus, the solution space changes drastically on neighboring inputs only when the algorithm fails (outputs $\perp$). The fact that algorithm outputs $\perp$ on a database z and a valid (non $\perp$) hypothesis on a neighboring database z′ might lead to privacy violations. To avoid this, our algorithm outputs $\perp$ with probability $\geq 1/2$ on any input (Step 1).

A PRIVATE LEARNER FOR PARITY, $\mathcal{A}(n, \mathrm{z}, \epsilon)$

1. With probability $1/2$, output $\perp$ and terminate.
2. Construct a set S by picking each element of $[n]$ independently with probability $p = \epsilon/4$.
3. Use Gaussian elimination to solve the system of equations imposed by examples, indexed by S: namely, $\{x_i \odot r = c_r(x_i) \,:\, i \in S\}$. Let V_S denote the resulting affine subspace.
4. Pick $r^* \in V_S$ uniformly at random and output c_{r^*}; if $V_S = \emptyset$, output $\perp$.

The (omitted) proof of privacy is based on showing that the inclusion of any single point in the sample set S increases the probability of a hypothesis being output by at most 2. The (omitted) proof of utility follows by considering all the possible situations in which the algorithms fails to satisfy the error bound, and by bounding the probabilities with which these situations occur.

Lemma 4.1 (Privacy, Utility of $\mathcal{A}$)**.** (a) *Algorithm $\mathcal{A}$ is ϵ-differentially private.* (b) *Let $\mathcal{X}$ be a distribution over $X = \{0,1\}^d$. Let $\mathrm{z} = (z_1, \ldots, z_n)$, where every $z_i = (x_i, c(x_i))$ with x_i drawn i.i.d. from $\mathcal{X}$ and $c \in$ PARITY. If $n \geq \frac{8}{\epsilon\alpha}(d \ln 2 + \ln(1/\beta'))$ then $\Pr[\mathcal{A}(n, \mathrm{z}, \epsilon) = h$ with $error(h) \leq \alpha] \geq \frac{1}{2} - \beta'$.*

It remains to amplify the success probability of $\mathcal{A}$. To do so, we repeat it $\Theta(\log \frac{1}{\beta})$ times, and output the answer returned by the first iteration that does not output $\perp$. See [27] for details. We obtain the following result.

Theorem 4.2. PARITY *is efficiently privately PAC learnable with $n = O\left(\frac{\log(1/\beta)}{\epsilon\alpha}(d + \log\frac{1}{\beta})\right)$ examples.*

Remark: It is possible to remove the quadratic dependency on $\log(1/\beta)$ in the previous theorem statement, by running $\mathcal{A}$ with a slightly smaller value of n (hence increasing the probability of outputting a bad hypothesis), and setting aside a small part of the data (a test set) to verify, using sum queries, how well the candidate hypotheses do. In this case, the upper bounds on the sample size and the running time of private and non-private PARITY learners only differ by a factor of $O(1/\epsilon)$.

5. Local Protocols and SQ learning

In this section, we relate private learning in the local model to SQ learning [29].

Local Model. We start by describing private computation in the local model. Informally, each individual holds her private information locally, and hands it to the learner after randomizing it. This is modeled by by letting the local algorithm access each entry z_i in the input database $\mathrm{z} = (z_1, \ldots, z_n) \in D^n$ only via *local randomizers*.

Definition 5.1 (Local Randomizer)**.** *An ϵ-*local randomizer *$R : D \to W$ is an ϵ-differentially private algorithm, i.e., $\Pr[R(u) = w] \leq e^\epsilon \Pr[R(u') = w]$ for all $u, u' \in D$ and all $w \in W$. The probability is taken over the coins of R (but* not *over the choice of the input).*

Note that a local randomizer works on a data set of size 1 and, therefore, u and u' are neighbors for all $u, u' \in D$. Let $LR_{\mathrm{z}}(\cdot,\cdot)$ denote an oracle that gets an index $i \in [n]$ and an ϵ-local randomizer R, and outputs a random value $w \in W$ chosen according to the distribution $R(z_i)$. The distribution $R(z_i)$ depends only on the entry z_i in z.

An ϵ-*local algorithm* accesses the database z via the oracle LR_{z} with the following restriction: for all $i \in [n]$, if $LR_{\mathrm{z}}(i, R_1), \ldots, LR_{\mathrm{z}}(i, R_k)$ are the invocations of LR_{z} on index i, where each R_j is an ϵ_j-local randomizer, then $\epsilon_1 + \cdots + \epsilon_k \leq \epsilon$. Since $\epsilon_1 + \cdots + \epsilon_k \leq \epsilon$, by Claim 2.2, ϵ-local algorithms are ϵ-differentially private.

Local algorithms that prepare all their queries to LR_{z} before receiving any reply are called *noninteractive*; otherwise, they are *interactive*.

SQ Model. In the statistical query (SQ) model, algorithms access statistical properties of a distribution rather than individual examples.

Definition 5.2 (SQ Oracle)**.** *Let $\mathcal{D}$ be a distribution over a domain D. An* SQ oracle $SQ_{\mathcal{D}}$ *takes as input a function $g : D \to \{+1, -1\}$ and a tolerance parameter $\tau \in (0, 1)$; it outputs v such that: $|v - \mathbb{E}_{u \sim \mathcal{D}}[g(u)]| \leq \tau$.*

An *SQ algorithm* accesses the distribution $\mathcal{D}$ via the SQ oracle $SQ_{\mathcal{D}}$. SQ algorithms that prepare all their queries to $SQ_{\mathcal{D}}$ before receiving any reply are called *nonadaptive*; otherwise, they are called *adaptive*.

5.1. Equivalence of Local and SQ Models

Both the SQ and local models restrict algorithms to access inputs in a particular manner. There is a significant difference though: an SQ oracle sees a distribution $\mathcal{D}$, whereas a local algorithm takes as input a fixed (arbitrary) database z. Nevertheless, we show that if the entries of z are chosen i.i.d. according to $\mathcal{D}$, then the models are equivalent. Specifically, an algorithm in one model can *simulate* an algorithm in the other model. Moreover, the expected query complexity is preserved up to polynomial factors. In the full version of this paper [27], we also investigate the efficiency of these simulations.

5.1.1 Local Simulation of SQ Algorithms

Blum *et al.* [9] used the fact that sum queries can be answered privately with little noise to show that any efficient SQ algorithm can be simulated privately and efficiently. We show that it can be simulated efficiently even by a local algorithm, albeit with slightly worse parameters. Let $g : D \to \{+1, -1\}$ be the SQ query we want to simulate. Let $\mathrm{Lap}(\lambda)$ denote the Laplace probability distribution with mean 0, standard deviation $\sqrt{2}\lambda$, and p.d.f. $f(x) = \frac{1}{2\lambda} e^{-|x|/\lambda}$. If $\eta \sim \mathrm{Lap}(2b/\epsilon)$, the algorithm $R(u) = g(u) + \eta$ is an ϵ-local randomizer [17].

Let z be a database with $n = O(\log(1/\beta)\epsilon^{-2}\tau^{-2})$ entries sampled i.i.d. from a distribution $\mathcal{D}$ on D, and let $g : D \to \{+1, -1\}$. Construct a local algorithm $\mathcal{A}_g$ that for every $i \in [n]$ invokes LR_{z} with the randomizer R defined above and outputs the average of the responses.

Lemma 5.3. *$\mathcal{A}_g$ approximates $\mathbb{E}_{u \sim \mathcal{D}}[g(u)]$ within additive error $\pm\tau$ with probability at least $1 - \beta$.*

Simulation. Consider an SQ algorithm making at most t queries to $SQ_{\mathcal{D}}$. Our local algorithm simulates each query using Lemma 5.3 with parameters $\beta' = \beta/t$, τ, and ϵ, on a database z containing $O(t \log(1/\beta')\epsilon^{-2}\tau^{-2})$ entries sampled from $\mathcal{D}$. Each query is simulated with a fresh portion of z, and hence privacy is preserved as each entry is subjected to a single application of the ϵ-local randomizer R. By the union bound, the probability that any of the queries is not approximated within additive error τ is at most β.

5.1.2 SQ Simulation of Local Algorithms

Let z be a database containing n entries drawn i.i.d. from $\mathcal{D}$. Consider a local algorithm making t queries to LR_{z}. We show how to simulate any local randomizer invoked by this algorithm by using statistical queries to $SQ_{\mathcal{D}}$. Consider one such randomizer $R : D \to W$ applied to database entry z_i. To simulate R we need to sample $w \in W$ with probability $p(w) = \Pr_{z_i \sim \mathcal{D}}[R(z_i) = w]$ taken over choice of $z_i \sim \mathcal{D}$ and random coins of R. (For interactive algorithms, it is more complicated, as the outputs of different randomizers applied to the same entry z_i have to be correlated.)

The idea behind the (omitted) simulation is to sample from a distribution $\widetilde{p}(w)$ that is within statistical distance β/t from $p(w)$. This would ensure a statistical distance of at most β between the output distribution of the local algorithm and the distribution resulting from the simulation. We start by applying R to an arbitrary input (say, $\mathbf{0}$) and obtaining a sample $w \sim R(\mathbf{0})$. Let $q(w) = \Pr[R(\mathbf{0}) = w]$ (probability taken only over randomness in R). Since R is ϵ-differentially private, $q(w)$ approximates $p(w)$ within a multiplicative factor of e^{ϵ}. To sample w from $p(w)$ we use the following rejection sampling algorithm: (i) sample w according to $q(w)$; (ii) output w with probability $\frac{p(w)}{q(w)e^{\epsilon}}$; (iii) otherwise, repeat from (i). To complete the simulation we show how statistical queries can be used to estimate $p(w)$. We now summarize the main result.

Lemma 5.4. *Let* z *be a database with entries drawn i.i.d. from a distribution $\mathcal{D}$. For every noninteractive (resp. interactive) local algorithm $\mathcal{A}$ making t queries to LR_{z}, there exists a nonadaptive (resp. adaptive) statistical query algorithm $\mathcal{B}$ that makes $t \cdot e^{\epsilon}$ queries in expectation to $SQ_{\mathcal{D}}$ with accuracy $\tau = \Theta(\beta/(e^{2\epsilon}t))$, such that the statistical difference between $\mathcal{B}$'s and $\mathcal{A}$'s output distributions is at most β.*

5.2. Implications for Local Learning

In this section, we define learning in the local and SQ models, and state that they are equivalent. Then we use

results from learning theory to prove that local learners are less powerful than general private learners.

Definition 5.5 (Local Learning)**.** Locally learnable *is defined identically to privately PAC learnable (Definition 3.1), except for the additional requirement that for all $\epsilon > 0$, algorithm $\mathcal{A}(\epsilon, \cdot, \cdot, \cdot)$ is ϵ-local and invokes LR_z at most $poly(d, size(c), 1/\epsilon, 1/\alpha, \log(1/\beta))$ times.*

Let $\mathcal{X}$ be a distribution over an input domain X. Let $SQ_{c,\mathcal{X}}$ denote the statistical query oracle that takes as input a function $g : X \times \{+1, -1\} \to \{+1, -1\}$ and a tolerance parameter $\tau \in (0, 1)$ and outputs v such that: $|v - \mathbb{E}_{x\sim\mathcal{X}}[g(x, c(x))]| \leq \tau$.

Definition 5.6 (SQ Learning[2])**.** SQ learnable *is defined identically to PAC learnable (Definition 2.3), except that instead of having access to examples z, an SQ learner can make $poly(d, size(c), 1/\alpha, \log(1/\beta))$ queries to oracle $SQ_{c,\mathcal{X}}$ with tolerance $\tau \geq 1/poly(d, size(c), 1/\alpha, \log(1/\beta))$.*

From the simulations in Section 5.1.1 and 5.1.2 we obtain the equivalence between SQ and local learning:

Theorem 5.7. *A concept class is learnable by a noninteractive (resp. interactive) local learner* if and only if *it is learnable by a nonadaptive (resp. adaptive) SQ learner.*

Now we can use lower bounds for SQ learners for PARITY (see, e.g., [29, 10, 42]) to demonstrate limitations of local learners. The lower bound of [10] rules out SQ learners for PARITY (even under the uniform distribution) that have unlimited running time, as long as they use at most $2^{d/3}$ queries of tolerance at least $2^{-d/3}$. Thus, using Theorem 4.2 that states that PARITY is (efficiently) privately learnable and Theorem 5.7, we obtain:

Corollary 5.8. *Concept classes learnable by local learners are a strict subset of concept classes PAC learnable privately. This holds both with and without computational restrictions.*

5.3. Interaction in Local Protocols

To complete the picture, we examine whether interaction gives more power in the local (SQ) model. The question is motivated by the fact that sometimes interaction is costly, complicated, or even impossible (for instance, when statistical information is collected by an interviewer, or at a polling booth.) We show that the answer to this question is positive by specifying a concept class that an interactive algorithm can learn with a polynomial number of examples, whereas any noninteractive algorithm requires an exponential number of examples.

Let MASKED-PARITY be the class of functions $c_{r,a} : \{0,1\}^d \times \{0,1\}^{\log d} \times \{0,1\} \to \{+1, -1\}$ indexed by $r \in \{0,1\}^d$ and $a \in \{0,1\}$:

$$c_{r,a}(x, i, b) = \begin{cases} (-1)^{r\odot x + a} & \text{if } b = 0 \\ (-1)^{r_i} & \text{if } b = 1 \end{cases}$$

where $r \odot x$ denotes the inner product of r and x modulo 2, and r_i is the ith bit of r. For simplicity, assume d is an integral power of 2.

We consider the concept class MASKED-PARITY $= \{c_{r,a}\}$ when the underlying distribution $\mathcal{X}$ is uniform over binary strings of length $d + \log d + 1$. Our adaptive learner for MASKED-PARITY uses two rounds of communication with the SQ oracle: first, to learn r from the $b = 1$ half of the input, and second, to retrieve the bit a from the $b = 0$ half of the input via queries that depend on r. The impossibility result (Theorem 5.9) for nonadaptive learners uses ideas from statistical query lower bounds (see, e.g., [29, 10, 42]). The intuition is that as the queries are prepared nonadaptively, any information about r gained from the $b = 1$ half of the input cannot be used to prepare queries to the $b = 0$ half. Since information about a is contained only in the $b = 0$ half, in order to extract a, the SQ algorithm is forced to learn PARITY. Our separation in the SQ model directly translates to a separation in the local model (using Theorem 5.7).

Theorem 5.9. *There exists an efficient adaptive SQ learner for* MASKED-PARITY *over the uniform distribution. However, no nonadaptive SQ learner can learn* MASKED-PARITY *(with polynomial number of queries) even over the uniform distribution.*

Remark: The learning theory literature distinguishes between *strong* learning, in which the learning algorithm can be required to produce hypotheses with arbitrarily low error (as in Definition 2.3, where the parameter α can be arbitrarily small), and *weak* learning, in which the learner is only required to produce a hypothesis with error bounded below $1/2$. The separation of Theorem 5.9 applies only to *strong* learning: it is simple to design a noninteractive weak SQ learner for MASKED-PARITY. It is impossible to obtain an analogous separation for weak learning, since the characterization of SQ learnable classes in terms of "SQ dimension" by Blum *et al.* [10] implies that adaptive and nonadaptive SQ algorithms are equivalent for weak learning (this is not explicit in [10] but follows from the fact that the weak learner constructed for classes with low SQ dimension is non-adaptive).

Acknowledgments. We thank Enav Weinreb for many discussions related to the local model, Avrim Blum and

[2]Unlike in the standard SQ definition, we allow a failure probability β as our simulations can fail with tiny probability.

Rocco Servedio for discussions about related work in learning theory, and Katrina Ligett and Aaron Roth for discussions about [12].

References

[1] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS*, pages 247–255. ACM, 2001.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *SIGMOD*, volume 29(2), pages 439–450. ACM, 2000.

[3] S. Agrawal and J. R. Haritsa. A framework for high-accuracy privacy-preserving mining. In *ICDE*, pages 193–204. IEEE Computer Society, 2005.

[4] M. Alekhnovich. More on average case vs approximation complexity. In *FOCS*, pages 298–307. IEEE, 2003.

[5] A. Ambainis, M. Jakobsson, and H. Lipmaa. Cryptographic randomized response techniques. In *PKC*, volume 2947 of *LNCS*, pages 425–438. Springer, 2004.

[6] M. Anthony and P. L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[7] S. Ben-David, D. Pál, and H.-U. Simon. Stability of k-means clustering. In *COLT*, LNCS, pages 20–34, 2007.

[8] S. Ben-David, U. von Luxburg, and D. Pál. A sober look at clustering stability. In *COLT*, LNCS, pages 5–19. Springer, 2006.

[9] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: The SuLQ framework. In *PODS*, pages 128–138. ACM, 2005.

[10] A. Blum, M. L. Furst, J. Jackson, M. J. Kearns, Y. Mansour, and S. Rudich. Weakly learning DNF and characterizing statistical query learning using Fourier analysis. In *STOC*, pages 253–262. ACM, 1994.

[11] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003.

[12] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, pages 609–618. ACM, 2008.

[13] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499 – 526, 2002.

[14] L. Devroye and T. Wagner. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory*, 25(5):601–604, 1979.

[15] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210. ACM, 2003.

[16] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, LNCS, pages 486–503. Springer, 2006.

[17] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, LNCS, pages 265–284. Springer, 2006.

[18] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO*, LNCS, pages 528–544. Springer, 2004.

[19] A. Elisseeff, T. Evgeniou, and M. Pontil. Stability of randomized learning algorithms. *Journal of Machine Learning Research*, 6:55–79, 2005.

[20] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, pages 211–222. ACM, 2003.

[21] P. Fischer and H.-U. Simon. On learning ring-sum-expansions. *SIAM Journal on Computing*, 21(1):181–192, 1992.

[22] Y. Freund, Y. Mansour, and R. E. Schapire. Generalization bounds for averaged classifiers. *Annals of Statistics*, 32(4):1698–1722, 2004.

[23] D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

[24] D. Helmbold, R. Sloan, and M. K. Warmuth. Learning integer lattices. *SIAM Journal on Computing*, 21(2):240–266, Apr. 1992.

[25] N. J. Hopper and M. Blum. Secure human identification protocols. In *ASIACRYPT*, volume 2248 of *LNCS*, pages 52–66. Springer, 2001.

[26] W. Jank and G. Shmueli. *Statistical Methods in eCommerce Research*. Wiley & Sons, 2008.

[27] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *CoRR*, arXiv:0803.0924 [cs.LG], 2008.

[28] S. P. Kasiviswanathan and A. Smith. A note on differential privacy: Defining resistance to arbitrary side information. *CoRR*, arXiv:0803.39461 [cs.CR], 2008.

[29] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998. Preliminary version in *proceedings of STOC'93*.

[30] M. Kearns and D. Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Computation*, 11(6):1427 – 1453, 1999.

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

[32] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. MIT press, Cambridge, Massachusetts, 1994.

[33] S. Kutin and P. Niyogi. Almost-everywhere algorithmic stability and generalization error. In *UAI*, pages 275–282, 2002.

[34] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103. IEEE, 2007.

[35] N. Mishra and M. Sandler. Privacy via pseudorandom sketches. In *PODS*, pages 143–152. ACM, 2006.

[36] T. Moran and M. Naor. Polling with physical envelopes: A rigorous analysis of a human-centric protocol. In *EUROCRYPT*, LNCS, pages 88–108. Springer, 2006.

[37] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, pages 75–84. ACM, 2007.

[38] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.

[39] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.

[40] A. van den Hout and P. van der Heijden. Randomized response, statistical disclosure control and misclassification: A review. *International Statistical Review*, 70:269–288, 2002.

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

[42] K. Yang. New lower bounds for statistical query learning. *Journal of Computer and System Sciences*, 70(4):485–509, 2005.