An Overview of the Equivalence between Differentially-Private Classification and Online Prediction.

Kelsey Ball*
University of Texas at Austin

Sabee Grewal†
University of Texas at Austin

Abstract

Recently, Bun, Livni, and Moran [BLM20] and Alon, Livni, Malliaris, and Moran [ALMM19] showed that a concept class $\mathcal{H} \subseteq \{\pm 1\}^{|\mathcal{X}|}$ is online learnable if and only if it is differentially-privately PAC learnable. This paper surveys the main technical ingredients involved in proving this equivalence and describes the relevant background. We also highlight some follow-up work as well as remaining open questions.

Contents

*kelseyball@utexas.edu
†sgrewal@utexas.edu

1 Introduction

In this paper, we survey a line of work characterizing the close relationship between differential privacy and online learning for binary classification. In particular, we focus on a recent milestone comprising two major results: (1) Alon, Livni, Malliaris, and Moran [ALMM19] showed that private PAC learnability implies a finite Littlestone dimension; subsequently, (2) Bun, Livni, and Moran [BLM20] proved that every concept class with finite Littlestone dimension can be learned by an approximate differentially-private learner. Together, these two results imply an equivalence between online learnability and differentially-private PAC learnability.[1]

The works of Bun et al. and Alon et al. mark significant progress towards characterizing the relationship between online and private learning. Below, we cover the background for understanding this equivalence and sketch the main proofs. We start by defining the private and online learning models and associated preliminaries. Then, we cover the main results of Bun et al. and Alon et al. Finally, we highlight some recent work as well as remaining open questions.

1.1 PAC Learning

In the PAC model [Val84], a learner is provided with a training sequence $S = ((x_1, y_1), \ldots, (x_m, y_m))$. Each x_i is drawn i.i.d. from a domain set $\mathcal{X}$ according to an unknown distribution $\mathcal{D}$ and labeled by y_i from a label set $\mathcal{Y}$ according to an unknown concept $c : \mathcal{X} \to \mathcal{Y}$ from a concept class $\mathcal{H}$. The goal of a PAC learning algorithm is to output a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ which has good generalization error with respect to the unknown distribution $\mathcal{D}$. In this work, we only consider binary classification, so we fix $\mathcal{Y} := \{\pm 1\}$.

Definition 1.1 (Generalization error)**.** Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$. The generalization error of a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ is defined by

$$\mathcal{L}_{\mathcal{D}}(h) = \mathop{\mathbf{Pr}}_{(x,y)\sim\mathcal{D}}[h(x) \neq y].$$

Given this setup, we now define when a concept class is PAC learnable.

Definition 1.2 (PAC learning)**.** A concept class $\mathcal{H} \subseteq \mathcal{Y}^{|\mathcal{X}|}$ is (ε, δ)-PAC learnable if there exists a learning algorithm $\mathcal{A}$ (called the learner) and a function $m : (\varepsilon, \delta) \to \mathbb{N}$ (called the sample complexity), such that given any sample S of size at least $m(\varepsilon, \delta)$, $\mathcal{A}$ will output a hypothesis h such that the following holds:

$$\mathop{\mathbf{Pr}}_{S\sim\mathcal{D}^m}[\mathcal{L}_{\mathcal{D}}(h) > \varepsilon] < \delta$$

for every distribution $\mathcal{D}^m$ on $(\mathcal{X} \times \mathcal{Y})^m$ and every $\varepsilon, \delta \in (0, 1)$.

If the learner outputs a hypothesis $h \in \mathcal{H}$, then we call $\mathcal{A}$ a *proper learner* and say that $\mathcal{H}$ is *properly PAC-learnable*. Otherwise, $\mathcal{A}$ can output any arbitrary function to minimize the generalization error, and we call $\mathcal{A}$ an *improper learner*.

Since the learner does not have access to $\mathcal{D}$, it cannot compute the generalization error. It is useful to define a notion of error that depends only on the training sequence.

Definition 1.3 (Empirical error)**.** Let $S = ((x_1, y_1), \ldots, (x_m, y_m))$ be a training sequence. The empirical error of a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ with respect to S is defined by

$$\mathcal{L}_S(h) = \frac{1}{m}|i \in [m] : h(x_i) \neq y_i|.$$

[1] A concept class has finite Littlestone dimension if and only if it is online-learnable [Lit87].

We can also define a learner which has low empirical error with high probability

Definition 1.4 (Empirical learner)**.** An algorithm $\mathcal{A}$ is an (α, β)-accurate empirical learner for a concept class $\mathcal{H}$ with sample complexity m if, for every $h \in \mathcal{H}$ and for every sample $S = ((x_1, h(x_1)), \ldots, (x_m, h(x_m))$, the algorithm outputs a function f satisfying:

$$\Pr_{f \sim A(S)}[\mathcal{L}_S(f) \leq \alpha] \geq 1 - \beta$$

Due to the work of Vapnik and Chervonenkis [VC71] and Blumer et al. [BEHW89], it is known that the sample complexity m to PAC-learn a concept class $\mathcal{H}$ is tightly characterized by a combinatorial parameter of the concept class called the VC dimension:

$$m = \Theta(\text{VCdim}(\mathcal{H})).$$

The VC dimension of a concept class bounded above by the log of its size: $\text{VCdim}(\mathcal{H}) \leq \log|\mathcal{H}|$, but in general it can be much smaller, as we see in the following example.

1.1.1 Example: PAC-learning Thresholds

We consider an example of PAC-learning the concept class of threshold functions. A *threshold function*over an ordered domain $f_t : \mathcal{X} \to \{\pm 1\}$, parameterized by a threshold $t \in \mathcal{X}$, is one that outputs $+1$ for all inputs less than or equal to the threshold, and outputs -1 for all other inputs. Let $T \in \mathbb{N}$, and let $\mathcal{X} = [T] = \{1, \ldots, T\}$ be our input domain. The class of possible threshold functions is defined as $\text{THR}_T = \{f_t : \mathcal{X} \to \{\pm 1\},\ \forall t \in [T]\}$, where $f_t(x) = \{+1 \text{ if } x \leq t \text{ else } -1\}$.

A natural learning algorithm is to output a hypothesis $\hat{h} \in \text{THR}_T$ whose threshold is equal to the largest positively-labeled point from the training sequence. One can show that this learning algorithm is sufficient to PAC-learn THR_T. That is, given a sample of size $m = \Theta\big(\frac{\log(1/\delta)}{\varepsilon}\big)$,

$$\mathcal{L}_{\mathcal{D}}(\hat{h}) \leq \varepsilon$$

with probability at least $1 - \delta$.

From the perspective of VC dimension, one can show that $\text{VCdim}(\text{THR}_T) = 1$, notably independent of T. Consequently, we can PAC-learn the class of thresholds over an infinite domain with sample complexity $\Theta(\text{VCdim}(\text{THR}_\infty)) = \Theta(1)$. We will revisit the concept class of thresholds throughout this paper.

1.2 Differential Privacy

The statistical analysis of large datasets underpins many computer programs which aid in complex decision-making. These datasets often contain sensitive information (e.g., health and financial records). Private learning seeks algorithms which can extract statistical insights from data without revealing any information at the level of the individual.

Suppose that a hospital has a dataset of patient records which we use to produce a statistical model. Consider an adversary who queries our model with the intent of learning information about a particular patient using the responses they receive. Differential privacy requires that the adversary should not be able to tell the difference between *the responses they get* and *the responses they would have gotten if that patient's data was removed from the dataset and replaced with arbitrary other values*. In other words, differential privacy guarantees that the effect of each individual on the learned model is "hidden" from outside observers.

Differential privacy enables the statistical analysis of datasets while providing mathematical guarantees that individual-level information does not leak to any adversary. This notion of privacy is accepted as the "gold standard" for protection. The notion of differential privacy developed over a series of works [DN04, BDMN05], culminating in the work of Dwork, McSherry, Nissim, and Smith [DMNS16]. See [DMNS16] or [Vad17] for additional information.

A learner is differentially-private if perturbing a single example in the training sequence does not appreciably change the output hypothesis. Below, we define the differentially-private PAC model introduced in [KLN+11]. We start by defining (ε, δ)-indistinguishable distributions.

Definition 1.5 ((ε, δ)-indistinguishable distributions)**.** For $a, b \in \mathbb{R}$, let $a =_{\varepsilon,\delta} b$ denote the statement

$$a \leq e^{\varepsilon} b + \delta \quad \text{and} \quad b \leq e^{\varepsilon} a + \delta.$$

Two distributions p, q are (ε, δ)-indistinguishable if $p(E) =_{\varepsilon,\delta} q(E)$ for every event E.

The definition of indistinguishable distributions has nice composition properties, such as the following:

Lemma 1.6 (Basic Composition Lemma [Vad17])**.** *If p, q are (ε, δ)-indistinguishable then for all $k \in \mathbb{N}$, p^k and q^k are $(k\varepsilon, k\delta)$-indistinguishable, where p^k, q^k are k-fold products of p, q.*

We now define the differentially-private PAC model.

Definition 1.7 (Differentially-private PAC learning)**.** A randomized learning algorithm

$$\mathcal{A} : (\mathcal{X} \times \{\pm 1\})^m \to \{\pm 1\}^{|\mathcal{X}|}$$

is (ε, δ)-differentially private if, for every two samples $S, S' \in (\mathcal{X} \times \{\pm 1\})^m$ that disagree on a single example, the output distributions $\mathcal{A}(S)$ and $\mathcal{A}(S')$ are (ε, δ)-indistinguishable. A concept class $\mathcal{H} = \{h : \mathcal{X} \to \{\pm 1\}\}$ is differentially-private PAC learnable if there exists an algorithm $\mathcal{A}$ that is (ε, δ)-differentially private.

The case of $\delta = 0$ is referred to as *pure differential privacy*. In the *approximate differential privacy* setting, a class $\mathcal{H}$ is privately learnable if it is PAC-learnable by an algorithm $\mathcal{A}$ that is $(\varepsilon(m), \delta(m))$-differentially private with $\varepsilon(m) \leq o(1)$, and $\delta(m) \leq m^{-\omega(1)}$.

We will need the following lemma, which states that for any approximate differentially-private learner, there is a private empirical learner with the same privacy and accuracy parameters and with slightly larger sample size.

Lemma 1.8 (Lemma 5.9 [BNSV15])**.** *Suppose $\varepsilon < 1$ and $\mathcal{A}$ is an (ε, δ)-differentially private, (α, β)-accurate learning algorithm for a concept class $\mathcal{H}$ with sample complexity m. Then there exists an (ε, δ)-differentially private, (α, β)-accurate empirical learner for $\mathcal{H}$ with sample complexity $9m$.*

1.2.1 Example: Privately Learning Thresholds

Consider the learning algorithm which we proposed for PAC-learning thresholds in Section 1.1.1. This algorithm is notably *not* differentially-private, because it reveals the value of the training example selected for the hypothesis threshold $\hat{h}$.

1.3 Online Learning

Online learning models the scenario in which the learner must make predictions on sequentially arriving data. Rather than being given a training sequence (as in the PAC model), online learning is performed in rounds. A single round works as follows: the learner is challenged with an instance $x_t \in \mathcal{X}$ and is required to answer with a label $\hat{y}_t \in \{\pm 1\}$. Once the learner makes a prediction, the true label $y_t \in \{\pm 1\}$ is revealed to the learner. The example (x_t, y_t) can be used to improve prediction in future rounds. The goal of the learner is to make as few prediction mistakes as possible.

Definition 1.9. Let $\mathcal{H} = \{h : \mathcal{X} \to \{\pm 1\}\}$ be a concept class over the domain $\mathcal{X}$ and $\mathcal{A}$ be an online learning algorithm. Given a sequence of examples S, let $M_{\mathcal{A}}(S)$ be the number of mistakes $\mathcal{A}$ makes on the sequence S. Denote the supremum of $M_{\mathcal{A}}(S)$ over all possible sequences S by $M_{\mathcal{A}}(\mathcal{H})$. A bound of the form $M_{\mathcal{A}}(\mathcal{H}) \leq B < \infty$ is called a *mistake bound*. A hypothesis class $\mathcal{H}$ is online learnable if there exists an algorithm $\mathcal{A}$ for which $M_{\mathcal{A}}(\mathcal{H}) \leq B < \infty$.

This notion of online learning is called the *online mistake-bound model*. It was first introduced by Littlestone [Lit87] and then extended to the agnostic case by Ben-David, Pál, and Shalev-Shwartz [BDPSS09].

1.3.1 Littlestone Dimension

The Littlestone dimension is a combinatorial parameter that captures mistake and regret bounds in online learning [Lit87]. Littlestone showed that the mininum mistake bound achievable by any online learner for a given concept class is exactly the Littlestone dimension of the class. More precisely, in the context of the online mistake-bound model, the Littlestone dimension of a concept class $\mathcal{H}$ is given by $\text{Ldim}(\mathcal{H}) = \min\{B \mid \exists \text{ a learner for } \mathcal{H} \text{ with mistake bound } B\}$. Furthermore, he described an explicit algorithm, called the *Standard Optimal Algorithm* (SOA), which achieves this optimal mistake bound. In round t of the algorithm, we receive an instance x_t. Then, we partition our hypothesis class $\mathcal{H}_t$ into two subclasses based on x_t: those which label the instance positively and those which label the instance negatively. Our prediction $\hat{y}_t$ is the label which corresponds to the subclass with the larger Littlestone dimension. Finally, after receiving the correct response y_t, we prune our hypothesis class to be the subclass corresponding to y_t.

Algorithm 1: Standard Optimal Algorithm

$\mathcal{H}_1 \leftarrow \mathcal{H}$;
for round $t \leftarrow 1, 2, 3, \ldots$ **do**
  For each $b \in \{\pm 1\}$ and $x \in \mathcal{X}$, let $\mathcal{H}_t^b(x) = \{h \in \mathcal{H}_t : h(x) = b\}$. Define $h_t : \mathcal{X} \to \{\pm 1\}$ by $h_t(x) = \text{argmax}_b \text{Ldim}(\mathcal{H}_t^b(x))$;
  Receive instance x_t;
  Predict $\hat{y}_t = h_t(x)$;
  Receive correct response y_t;
  Update $\mathcal{H}_{t+1} \leftarrow \mathcal{H}_t^{y_t}(x_t)$;
end

A training sequence $\{(x_t, y_t)\}_{t=1}^T$ is said to be *realizable* by the concept class $\mathcal{H}$ if there exists some target concept $c \in \mathcal{H}$ such that $y_t = c(x_t)$, $\forall t = 1, \ldots, T$. In this case, the sequence is said to be *consistent* with c.

1.3.2 Example: Online-learning Thresholds

Let us revisit the example of learning the class of threshold functions in the context of online learning. Let the space of input examples be $[T] = \{1, \ldots, T\}$, and let our concept class be $\mathrm{THR}_T = \{f_t : [T] \to \{\pm 1\}\}$. In the first round, not having seen any examples, we guess the threshold to lie in the middle of the domain at $T/2$. Then, the adversary gives us a point x_1; if $x_1 \leq T/2$ we predict $+1$, otherwise we predict -1. The adversary then reveals the correct response y_1. If our prediction is correct, we leave our hypothesis as is. If we guessed incorrectly, we adjust our hypothesis threshold to be the midpoint between our current known bound and this newly mislabeled point. We proceed in this binary-search fashion, which implies a mistake bound of $\log T$. Therefore, $\mathrm{Ldim}(\mathrm{THR}_T) = \log T$.

2 Main Ideas

In this section, we summarize the main ideas involved in proving the following equivalence.

Theorem 2.1 (Private PAC Learning $\equiv$ Online Prediction)**.** *The following statements are equivalent for a concept class $\mathcal{H} \subseteq \{\pm 1\}^{|\mathcal{X}|}$:*

1. *$\mathcal{H}$ is online learnable.*
2. *$\mathcal{H}$ is approximate differentially-privately PAC learnable.*

2.1 Private PAC Learning $\implies$ Online Learning

To show that private PAC learnability implies online learnability, Alon et al. prove the contrapositive: if a concept class $\mathcal{H}$ has infinite Littlestone dimension, then $\mathcal{H}$ cannot be privately learned. The proof can be broken into two parts:

Step 1: If a concept class $\mathcal{H}$ has infinite Littlestone dimension, then it contains the class of thresholds over an infinite domain.

Step 2: Any class that contains thresholds over an infinite domain is not privately learnable.

Steps 1 and 2 together imply, by contraposition, that private PAC learnability implies online learnability.

Step 1: Littlestone classes contain finite thresholds. Step 1 is implied by previous results of Shelah (1978) and Hodges (1997), which provide a simple connection between Littlestone dimension and thresholds.

Theorem 2.2 (Littlestone dimension and thresholds [She78, Hod97])**.**

1. *If the $\mathrm{Ldim}(\mathcal{H}) \geq d$ then $\mathcal{H}$ contains $\lfloor \log d \rfloor$ thresholds.*
2. *If $\mathcal{H}$ contains d thresholds then its $\mathrm{Ldim}(\mathcal{H}) \geq \lfloor \log d \rfloor$.*

We say that a concept class $\mathcal{H}$ contains k thresholds if there are $x_1, \ldots, x_k \in \mathcal{X}$ and $h_1, \ldots, h_k \in \mathcal{H}$ such that $h_i(x_j) = +1$ if and only if $i \leq j$ for all $i, j \leq k$.

Step 2: Thresholds cannot be privately learned. The first major progress towards showing Step 2 came in 2015. Bun, Nissim, Stemmer, and Vadhan [BNSV15] showed that thresholds over an infinite domain are not properly, privately learnable. However, to show Step 2, one must extend this result to improper learners. Why is improper learning relevant here? In improper learning, being non-learnable is an inherited property: if a class $\mathcal{C}$ contains a non-learnable class $\mathcal{D}$, then the class $\mathcal{C}$ is also not learnable. This is because if we could learn $\mathcal{C}$, then we have improperly learned anything it contains. Hence, if you can show that a class $\mathcal{C}$ is not improperly learnable, then any class that contains $\mathcal{C}$ is also not learnable. The main result of Alon et al. is to extend the work of Bun et al. to improper private learners.

Theorem 2.3 (Informal version of Theorem 3.8)**.** *The class of thresholds over an infinite domain cannot be learned privately, even by an improper learner.*

Finally, we briefly remark on the difficulty of showing lower bounds for improper learners. The main obstacle comes from the fact that an improper learner can output any function. So, it is necessary to find some structure in the output of the algorithm that is always present. The authors accomplish this via Ramsey Theory. Specifically, for a sufficiently large input domain $\mathcal{X}$, one can make a Ramseyean argument that for every possible learner $\mathcal{A}$ there must be a subset $\mathcal{X}'$ with a certain structure with respect to $\mathcal{A}$. Then, $\mathcal{X}'$ is used to construct hard distributions which imply lower bounds on the sample complexity for any algorithm that improperly privately learns thresholds.

2.2 Private PAC Learning $\Longleftarrow$ Online Learning

The proof of this implication hinges on the notion of *global stability*. A PAC learner is globally-stable with respect to a distribution if you can specify a probability η and a number of samples n such that the learner is guaranteed to output some particular hypothesis with probability η, given any training sequence of at least n points. The main result consists of two steps:

Step 1: Any concept class with finite Littlestone dimension d has a globally-stable PAC-learner.

Step 2: We can convert any globally-stable PAC learner into an DP PAC-learner with finite sample complexity.

Step 1: Online learning implies globally-stable learning. To prove the existence of a globally-stable PAC learner, we start with a concept class $\mathcal{H}$ that is online learnable with mistake bound d. Our high-level strategy is to force an online learner to make d mistakes with respect to a special training sequence that we construct. With (exponentially small) positive probability, our constructed training sequence will be consistent with the target concept (i.e., all constructed labels agree with those produced by the target concept). In the case that the constructed sequence is consistent with the target concept, our learning algorithm will output the target concept, having already made d mistakes. This construction is enough to show that the target concept will always be output with some small positive probability, which is enough to show global stability. Moreover, we show that the hypotheses output by this procedure are guaranteed to have good generalization error.

Step 2: Globally-stable learning implies private learning. Given a globally-stable PAC learner G, we use standard techniques from differential privacy to convert it into a DP PAC learner. Specifically, we subsample from the original distribution $\mathcal{D}$ and run each sample through G to produce a set of hypotheses $\mathcal{H}$. We then select the most frequently occurring of these hypotheses using

DP Heavy Hitter Identification, a differentially-private subroutine, to produce a subset of hypotheses $\mathcal{H}'$. Given the guarantee of both likelihood and good generalization error on the hypotheses output by our globally-stable learner, we know that a good hypothesis will end up in $\mathcal{H}'$. Finally, we can apply a generic private learner (defined in Section 4.2.3) to a fresh sample from $\mathcal{D}$ using $\mathcal{H}'$ as our concept class in order to produce our final hypothesis.

3 Private Learning Implies Finite Littlestone Dimension

In this section, we discuss and summarize the main technical components involved in showing that differentially-private PAC learnability implies finite Littlestone dimension. We begin by introducing some definitions and notation.

A sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$ is increasing, balanced, and realizable when $x_1 < x_2 < \ldots < x_m$, m is even, and when the first half of the y_i's are -1 and the second half are $+1$. $S_{\mathcal{X}} = \{x_1, \ldots, x_m\}$ is the set of unlabelled examples corresponding to S. $\text{ord}_S(x) = |\{i | x_i < x\}|$. $\mathcal{A}(S)$ is a distribution over hypotheses output by the randomized learning algorithm $\mathcal{A}$, given the sample S. Then,

$$\mathcal{A}_S(x) = \mathop{\mathbf{Pr}}_{h \sim \mathcal{A}(S)}[h(x) = +1].$$

The *tower function* is defined as

$$\text{twr}_i(x) = \begin{cases} x & \text{if } i = 1. \\ 2^{\text{twr}_{i-1}(x)} & \text{if } i > 1. \end{cases}$$

The *iterated logarithm* is defined as

$$\log^{(i)} x = \begin{cases} \log x & \text{if } i = 0. \\ 1 + \log^{(i-1)} \log x & \text{if } i > 0. \end{cases}$$

Finally, the function $\log^* x$ is the number of times the iterated logarithm needs to be applied before the result is less than or equal to 1. Formally,

$$\log^* x = \begin{cases} 0 & \text{if } x \leq 1. \\ 1 + \log^* \log x & \text{if } i > 0. \end{cases}$$

3.1 Every Learning Algorithm Has Homogeneous Sets

We cover the Ramseyean argument used to show that every learning algorithm must have some structure. Specifically, for every learning algorithm, there is a subset of the domain $\mathcal{X}$ that is *homogeneous with respect to the learning algorithm*, which we explain below.

A *set system* or *family* $\mathcal{F}$ on the set X is a collection of sets where each set is a subset of X. A family is k-uniform when all its members are k-element sets. A *graph*, typically denoted by a pair $G = (V, E)$, can be described as a family: E is a 2-uniform family on the set V. Because of this relation, families can be viewed as a generalization of graphs and are often called *hypergraphs*. The members of a family $\mathcal{F}$ are often called *hyperedges*. A q-coloring of hyperedges $\chi : \mathcal{F} \to [q]$ of a family $\mathcal{F}$ assigns one of the colors $1, \ldots, q$ to each member of $\mathcal{F}$.

The complete k-hypergraph on an n-element set, denoted by $K_n^{(k)}$, is equivalent to the family of all k-element subsets of $[n]$. Let $\chi : K_n^{(k)} \to [q]$ be a q-coloring of the complete k-hypergraph on n elements. The set $X \subseteq [n]$ is a *homogeneous set* when all the k-element subsets of X are assigned the same color by χ.

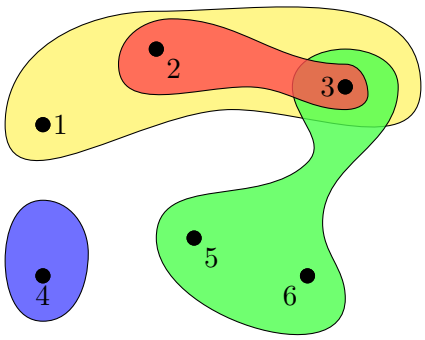

Figure 1: A hypergraph with 6 vertices and 4 hyperedges. This hypergraph can also be written as a family $\mathcal{F}$ on the set $\{1,\ldots,6\}$: $\mathcal{F} = \{\{1,2,3\},\{2,3\},\{4\},\{3,5,6\}\}$.

Theorem 3.1 (Erdős and Rado, [ER52])**.** *Let $s > t \geq 2$ and q be integers, and let*

$$N \geq \mathrm{twr}_t(3sq\log q).$$

Then, for every q-coloring of $K_N^{(t)}$ there is a homogeneous set of size s.

Another way to state Theorem 3.1 is as follows. For any q-coloring of the complete t-hypergraph on N vertices, there is a complete t-hypergraph on s vertices that is monochromatic.

We now introduce the notion of homogeneous sets with respect to a learning algorithm.

Definition 3.2 (m-homogeneous set)**.** A set $\mathcal{X}' \subseteq \mathcal{X}$ is m-homogeneous with respect to a learning algorithm $\mathcal{A}$ if there are numbers $p_i \in [0,1]$, for $0 \leq i \leq m$ such that for every increasing balanced realizable sample $S \in (\mathcal{X}' \times \{\pm 1\})^m$ and for every $x \in \mathcal{X}' \setminus S_{\mathcal{X}}$:

$$|\mathcal{A}_S(x) - p_i| \leq \frac{1}{10^2 m},$$

where $i = \mathrm{ord}_S(x)$. The list $(p_i)_{i=0}^m$ is called the probabilities-list of $\mathcal{X}'$ with respect to $\mathcal{A}$.

Using the result of Erdős and Rado, one can show that there is always a large subset of the domain that is homogeneous with respect to the learning algorithm.

Lemma 3.3 (Every learning algorithm has large homogeneous sets; Lemma 9 [ALMM19])**.** *Let $\mathcal{A}$ be a learning algorithm that is defined over input samples of size m over a domain $\mathcal{X} \subseteq \mathbb{R}$ with $|\mathcal{X}| = n$. Then, there is a set $\mathcal{X}' \subseteq \mathcal{X}$ that is m-homogeneous with respect to $\mathcal{A}$ of size*

$$|\mathcal{X}'| \geq \frac{\log^{(m)}(n)}{2^{O(m\log m)}}.$$

Proof. We will describe a coloring on $(m+1)$-element subsets of the domain $\mathcal{X}$. In other words, we will define an edge coloring for $K_n^{(m+1)}$ (recall, $|\mathcal{X}| = n$). Let $D = \{x_1 < x_2 < \ldots < x_{m+1}\}$ be an $(m+1)$-element subset of $\mathcal{X}$. For each $i \in [m+1]$, define $D^{-i} := D \setminus \{x_i\}$, and let S^{-i} denote the increasing balanced realizable sample on D^{-i}. Set p_i to be the fraction of the form $\frac{t}{10^2 m}$ that is closest to $\mathcal{A}_{S^{-i}}(x_i)$. Note that there are $10^2 m + 1$ total fractions of this form, and that, by construction, $|\mathcal{A}_{S^{-i}}(x_i) - p_i| \leq \frac{1}{10^2 m}$. The coloring assigned to D is then the list $(p_1,\ldots,p_{m+1})$. Since p_i can take on $10^2 m+1$ different values and there are $m+1$ elements in the list, our coloring requires at most $(10^2 m+1)^{m+1}$ colors. Set $t := m+1$, $q := (10^2 m+1)^{m+1}$, and $N := n$. Then, we can apply Theorem 3.1 to deduce a lower bound on the size of the homogeneous set $\mathcal{X}'$. □

3.2 Homogeneous Sets Imply Lower Bounds

We have shown that, given a learning algorithm $\mathcal{A}$, there exists a homogeneous set $\mathcal{X}'$. Now we will use $\mathcal{X}'$ to construct a family of hard distributions, which will imply lower bounds for privately learning thresholds.

Throughout the remainder of this section, let $\mathcal{A}$ be an $(0.1, \delta)$-differentially private algorithm with sample complexity m and $\delta \leq \frac{1}{10^3 m^2 \log m}$, and let $\mathcal{X}' = \{1, \ldots, k\}$ be a m-homogeneous set with respect to $\mathcal{A}$. Note that we lower bounded the size of $\mathcal{X}'$ in the previous section, and that $\mathcal{X}' \subseteq \mathcal{X}$, where we have relabeled the elements 1 through k for convenience. Finally, let $\mathcal{A}$ be a $(1/16, 1/16)$-empirical learner of thresholds over $\mathcal{X}'$. We begin by showing that the probabilities-list of $\mathcal{X}'$ with respect to $\mathcal{A}$ always has two consecutive values that are far apart.

Fact 3.4 (Claim 14 [ALMM19])**.** *Let $(p_i)_{i=0}^m$ denote the probabilities-list of $\mathcal{X}'$ with respect to $\mathcal{A}$. Then, for some $0 < i \leq m$,*

$$p_i - p_{i-1} \geq \frac{1}{4m}.$$

Proof sketch. Let S be a balanced increasing realizable sample such that $S_{\mathcal{X}} \subseteq \mathcal{X}'$. Use the fact that $\mathcal{A}$ is a $(1/16, 1/16)$-empirical learner to show

$$\frac{7}{8} \leq \mathop{\mathbf{E}}_{h \sim \mathcal{A}(S)} (1 - \mathcal{L}_S(h)) = \frac{1}{m} \sum_{i=1}^{m/2} [1 - \mathcal{A}_S(x_i)] + \frac{1}{m} \sum_{i=m/2+1}^{m} [\mathcal{A}_S(x_i)],$$

where the equality follows from the fact that S is balanced, increasing, and realizable. This implies that there exists an $m/2 \leq m_1 \leq m$ such that $\mathcal{A}_S(x_{m_1}) \geq 3/4$.

Now, let S' be the training sequence where x_{m_1} is replaced by $x_{m_1}+1$ (but the label y_{m_1} remains the same). Then, since $\mathcal{A}$ is $(0.1, \delta)$-differentially private, we know that

$$\mathcal{A}_{S'}(x_{m_1}) \geq \left(\frac{3}{4} - \delta\right) e^{-0.1} \geq \frac{2}{3}.$$

By the definition of m-homogeneity with respect to $\mathcal{A}$, we can further say that $p_{m_1-1} \geq 2/3 - 1/10^2 m$. Similarly, we can find an $1 \leq m_2 \leq m/2$ such that and $p_{m_2-1} \leq 1/3 + 1/10^2 m$. Together, this implies that there exists some i such that $m_2 - 1 \leq i \leq m_1 - 1$ and

$$p_i - p_{i-1} \geq \frac{1/3}{m} - \frac{1}{50m^2} \geq \frac{1}{4m}.$$

□

We will use this fact to construct a family of distributions that are hard for the private learning algorithm $\mathcal{A}$.

Lemma 3.5 (Lemma 12 [ALMM19])**.** *Set $n \coloneqq k - m$. There exists a family of distributions $\mathcal{P} = \{P_i\}_{i=1}^n$ over $\{\pm 1\}^n$ with the following properties.*

1. *Every P_i, $P_j \in \mathcal{P}$ are $(0.1, \delta)$-indistinguishable.*
2. *There exists an $r \in [0, 1]$ such that for all $i, j \leq n$*

$$\Pr_{v \sim P_i}[v(j) = +1] = \begin{cases} \leq r - \frac{1}{10m} & \text{if } j < i. \\ \geq r + \frac{1}{10m} & \text{if } j > i. \end{cases}$$

Proof. Let i be the index such that $p_i - p_{i-1} \geq 1/4m$, which must exist due to Fact 3.4. Let $S \in (\mathcal{X} \times \{\pm 1\})^m$ be an increasing realizable sample such that the following holds: for $x_{i-1}, x_{i+1} \in S$, let

$$J \subseteq \mathcal{X}' = \{x \in \mathcal{X}' : x \in (x_{i-1}, x_{i+1})\}$$

such that $|J| = k - m$. Let S_x be the sample S where x_i is replaced with x. This yields a family of samples $\{S_x : x \in J\}$ that differ in only one example. Therefore, the following properties hold:

1. The output distributions $\mathcal{A}(S_{x'})$ and $\mathcal{A}(S_{x''})$ are $(0.1, \delta)$-indistinguishable. (This follows from the fact that $\mathcal{A}$ is $(0.1, \delta)$-differentially private.)
2. Set $r = \frac{p_{i+1}+p_i}{2}$. Then for all $x, x' \in J$:

$$\Pr_{h \sim \mathcal{A}(S_x)}[h(x') = +1] = \begin{cases} \leq r - \frac{1}{10m} & \text{if } x' < x. \\ \geq r + \frac{1}{10m} & \text{if } x' > x. \end{cases}$$

Define $n \coloneqq k - m = |J|$ and restrict the hypothesis output by $\mathcal{A}$ to only label points in J. Then, the output distribution $\mathcal{A}(S_x)$ is a distribution over hypotheses h which are restricted to J, so they are isomorphic to a distribution over $\{\pm 1\}^n$. Similarly, since $|J| = n$, J is isomorphic to $[n]$. This concludes the proof. □

Thus far, we have shown that $k - m = |\mathcal{P}|$ where $\mathcal{P}$ is a family of distributions with a certain set of properties. We now show an upper bound on $|\mathcal{P}|$.

Lemma 3.6 (Lemma 13 [ALMM19])**.** *Let $\mathcal{P}$ be the family of distributions defined in Lemma 3.5. Then, $|\mathcal{P}| \leq 2^{10^3 m^2 \log^2 m}$.*

Proof sketch. Set $T \coloneqq 10^3 m^2 \log^2 m - 1$ and $D \coloneqq 10^2 m^2 \log T$. We want to show that $|\mathcal{P}| = n \leq 2^{T+1}$. For the sake of contradiction, assume that $n > 2^{T+1}$. Let $\mathcal{Q} = \{P_i^D : P_i \in \mathcal{P}\}$ be a family of distributions, which contain the D-fold product of the distributions in $\mathcal{P}$. By the Basic Composition Lemma of differential privacy (Lemma 1.6), we know that each distinct pair $Q_i, Q_j \in \mathcal{Q}$ is $(0.1D, \delta D)$-indistinguishable.

Since the distributions in the family $\mathcal{P}$ have support over $\{\pm 1\}^n$, the distributions in the family $\mathcal{Q}$ have support over $\mathbf{v} = (v_1, \ldots, v_D)$, where each $v_i \in \mathbf{v}$ is a vector in $\{\pm 1\}^n$ (i.e., $\mathbf{v}$ is a sequence of D vectors in $\{\pm 1\}^n$). We define $\bar{\mathbf{v}} \in \{\pm 1\}^n$ as

$$\bar{\mathbf{v}}(j) = \begin{cases} -1 & \text{if } \frac{1}{D}\sum_{i=1}^{D} v_i(j) \leq r. \\ +1 & \text{if } \frac{1}{D}\sum_{i=1}^{D} v_i(j) > r. \end{cases}$$

That is, to find the jth value of $\bar{\mathbf{v}}$, we average the jth values of each $v_i \in \mathbf{v}$ and pick ± 1 depending on if the average is above or below some threshold r.

We now define a mapping B according to the outcome of T steps of a binary search on $\bar{\mathbf{v}}$. The binary search works as follows. Initialize an index $j = n/2$. Query the jth entry of $\bar{\mathbf{v}}$. If it is $+1$, then recursively search the first half of the list. Else, recursively search the right half. Define the mapping $B(\bar{\mathbf{v}})$ to be the entry that was queried at the Tth step of the binary search. Let E_j be the probability of drawing a sample $\mathbf{v}$ from $Q \in \mathcal{Q}$ such that $B(\bar{\mathbf{v}}) = j$. Since we have assumed that $n > 2^{T+1}$, the events E_j corresponding to each outcome of B are mutually disjoint.

Recall that for each $P_i \in \mathcal{P}$ there exists an $r \in [0, 1]$ such that for all distinct $i, j \leq n$

$$\Pr_{v \sim P_i}[v(j) = +1] = \begin{cases} \leq r - \frac{1}{10m} & \text{if } j < i. \\ \geq r + \frac{1}{10m} & \text{if } j > i. \end{cases}$$

Thus, by taking a large i.i.d. sample from P_i, the probability will concentrate on the event that B outputs entry i. Specifically, given the D-fold sample $\mathbf{v}$ from Q_i, one can show that the event E_i (i.e., the event that $B(\mathbf{v}) = i$) has probability at least

$$1 - Te^{-2\frac{1}{10^2 m^2}D} = 1 - Te^{-2\log T} \geq \frac{2}{3},$$

by using a Chernoff Bound and the Union Bound. Then, we can use the fact that $Q_i, Q_j \in \mathcal{Q}$ are $(0.1D, \delta D)$-indistinguishable to show that for all $j \leq n$ and i in the image of B that

$$Q_j(E_i) \geq \frac{1}{2}e^{-0.1D}.$$

Then, since all 2^T of the E_i's are mutually disjoint, we have

$$1 \geq Q_j(\cup_i E_i) = \sum_i Q_j(E_i) \geq 2^{T-1}e^{-0.1D},$$

but $2^{T-1}e^{-0.1D} > 1$, which is a contradiction. Therefore we can conclude that $|\mathcal{P}| = n \leq 2^{T+1}$. □

Combining Lemmas 3.5 and 3.6 yields a lower bound for privately learning thresholds.

Lemma 3.7 (Large homogeneous sets imply lower bounds for private learning; Lemma 10 [ALMM19])**.** *Let $\mathcal{A}$ be an $(0.1, \delta)$-differentially private algorithm with sample complexity m and $\delta \leq \frac{1}{10^3 m^2 \log m}$. Let $\mathcal{X}' = \{1, \ldots, k\}$ be m-homogeneous with respect to $\mathcal{A}$. Then, if $\mathcal{A}$ empirically learns the class of thresholds over $\mathcal{X}'$ with $(1/16, 1/16)$-accuracy, then*

$$k = 2^{O(m^2 \log^2 m)}.$$

In other words, $m = \Omega\left(\frac{\sqrt{\log k}}{\log\log k}\right)$.

Proof. In Lemma 3.5 we showed $k - m = |\mathcal{P}|$. In Lemma 3.6 we showed $|\mathcal{P}| \leq 2^{10^3 m^2 \log^2 m}$. Combining these two results gives $k - m \leq 2^{10^3 m^2 \log^2 m} \implies k = 2^{O(m^2 \log^2 m)}$. □

3.3 Putting It All Together

So far we have shown that, for a m-homogeneous set $\mathcal{X}' \subset \mathcal{X}$, with respect to learning algorithm $\mathcal{A}$,

$$\frac{\log^{(m)} n}{2^{O(m \log m)}} \leq |\mathcal{X}'| \leq 2^{O(m^2 \log^2 m)}.$$

The first inequality was shown in Lemma 3.3 and the second inequality in Lemma 3.7. This implies that

$$\frac{\log^{(m)} n}{2^{O(m \log m)}} \leq 2^{O(m^2 \log^2 m)} \implies \log^{(m)} n \leq 2^{c \cdot m^2 \log m},$$

for some positive constant c. Applying the iterated logarithm $\log^*(2^{c \cdot m^2 \log m}) = \log^* m + O(1)$ times yields that

$$\log^{(m + \log^* m + O(1))} n \leq 1.$$

This implies that $\log^* n \leq \log^* m + m + O(1)$ which implies $m \geq \Omega(\log^* n)$. This is summarized in the main result of Alon et al.:

Theorem 3.8 (Thresholds are not privately learnable)**.** *Let $\mathcal{X} \subseteq \mathbb{R}$ of size $|\mathcal{X}| = n$ and let $\mathcal{A}$ be a $(\frac{1}{16}, \frac{1}{16})$-accurate learning algorithm for the class of thresholds over $\mathcal{X}$ with sample complexity m which satisfies (ε, δ)-differential privacy with $\varepsilon = 0.1$ and $\delta = O(\frac{1}{m^2 \log m})$. Then,*

$$m \geq \Omega(\log^* n).$$

In particular, the class of thresholds over an infinite $\mathcal{X}$ cannot be learned privately.

Combining Theorem 3.8 with Theorem 2.2 yields the following corollary.

Corollary 3.9 (Private learning implies finite Littlestone dimension)**.** *Let $\mathcal{H}$ be a concept class with Littlestone dimension $d \in \mathbb{N}$, and let $\mathcal{A}$ be a $(\frac{1}{16}, \frac{1}{16})$-accurate learning algorithm for the class of thresholds over $\mathcal{X}$ with sample complexity m which satisfies (ε, δ)-differential privacy with $\varepsilon = 0.1$ and $\delta = O(\frac{1}{m^2 \log m})$. Then,*

$$m \geq \Omega(\log^* d).$$

.

Proof. We prove the contrapositive. That is, we show that if the Littlestone dimension of a class $\mathcal{H}$ is infinite, then you cannot privately learn $\mathcal{H}$. Let $\mathcal{H}$ be a concept class with Littlestone dimension d. By Theorem 2.2, $\mathcal{H}$ contains $\lfloor \log d \rfloor$ thresholds. Therefore, Theorem 3.8 implies a lower bound of $m \geq \Omega(\log^* \log d) = \Omega(\log^* d)$ for any $(0.1, O(1/m^2 \log m))$-differentially private learning algorithm that learns $\mathcal{H}$ to $(1/16, 1/16)$-accuracy. Taking $d \to \infty$ completes the proof. □

4 Finite Littlestone Dimension Implies Private Learning

We state the main result of [BLM20]:

Theorem 4.1 (Littlestone Classes are Privately Learnable)**.** *Let $\mathcal{H} \subseteq \{\pm 1\}^{|\mathcal{X}|}$ be a class with Littlestone dimension d, let $\varepsilon, \delta \in (0, 1)$ be privacy parameters, and let $\alpha, \beta \in (0, 1/2)$ be accuracy parameters. For*

$$n = O\left(\frac{16^d \cdot d^2 \cdot (d + \log(1/\beta\delta))}{\alpha\varepsilon}\right) = O_d\left(\frac{\log(1/\beta\delta)}{\alpha\varepsilon}\right)$$

there exists an (ε, δ)-DP learning algorithm such that for every realizable distribution $\mathcal{D}$, given an input sample $S \sim \mathcal{D}^n$, the output hypothesis $f = \mathcal{A}(S)$ satisifies $\mathcal{L}_{\mathcal{D}}(f) \leq \alpha$ with probabilitiy at least $1 - \beta$, where the probability is taken over $S \sim \mathcal{D}^n$ as well as the internal randomness of $\mathcal{A}$.

In this section, we will give a sketch of the proof of the above theorem by explaining its key ingredients. Lemmas 4.3 and 4.4 establish the existence and generalization of a globally-stable learner. Theorem 4.7 establishes the existence of a private learner, given a globally-stable learner. Combining these results leads to Theorem 4.1.

4.1 Online Learning Implies Globally-Stable Learning

First, we define the notion of global stability and argue that any class $\mathcal{H}$ with finite Littlestone dimension can be learned by a globally-stable algorithm. We also show that global stability is sufficient to guarantee generalization bounds for a realizable distribution[2].

[2] A distribution $\mathcal{D}$ is said to be *realizable by* $\mathcal{H}$ if there is an $h \in \mathcal{H}$ such that $\mathcal{L}_{\mathcal{D}}(h) = 0$.

4.1.1 Global Stability

The notion of stability in general describes robustness of the output distribution with respect to small perturbations in the input, intuitively similar to the notion of differential privacy. Bun et al. introduce a stronger notion of algorithmic stability called *global stability* which provides the link between finite Littlestone dimension and private learnability. Whereas typical notions of stability describe robustness when changing a single sample in the training sequence, global stability is a robustness guarantee when changing the *entire* training sequence.

Definition 4.2 (Global stability)**.** For $\eta > 0$ and $n \in \mathbb{N}$, a learning algorithm $\mathcal{A}$ is (n, η)-*globally stable* with respect to a distribution $\mathcal{D}$ over examples if there exists a hypothesis h whose frequency as an output is at least η. That is,

$$\Pr_{S \sim \mathcal{D}^n} [\mathcal{A}(S) = h] \geq \eta.$$

Our goal is to define a globally-stable algorithm that can PAC-learn any class with finite Littlestone dimension. Our strategy will be to define a learning algorithm that, with probability η, outputs the target concept by forcing d mistakes on a specially constructed training sequence of length n. For any arbitrarily constructed training sequence, making d mistakes does not guarantee that we will output the target concept. However, if our training sequence is *consistent with the target concept*, then making d mistakes will indeed identify the target concept (given that the learner's mistake bound is d). The key to proving global stability for online learners thus lies in lower-bounding the probability with which we can construct a sequence that both (1) forces d mistakes and (2) is consistent with the target concept.

Below, we provide an algorithm that outputs such a training sequence[3]. For a pair of samples S, T, we denote their concatenation by $S \circ T$. In each iteration i of this algorithm, we add a mistake-inducing example to our training sequence by constructing a "contest" between two hypotheses h_1, h_2. First, we create a "contest example" (x, y) which is guaranteed to disagree with at least one the hypotheses. Then, we select the hypothesis which disagrees and append the training sample which produced it, as well as the contest example, to our aggregate training sequence.

Algorithm 2: Training sequence construction

Result: A training sequence $S_{n;d}$ which forces d mistakes
$S \leftarrow \{\}$;
for $i \leftarrow 1, \ldots, d$ **do**
  Run SOA on fresh samples $S_1, S_2 \sim \mathcal{D}^n$ until they produce distinct hypotheses h_1, h_2;
  Take any point x on which they disagree: $h_1(x) \neq h_2(x)$;
  Sample arbitrary label y uniformly from $\{\pm 1\}$;
  if $h_1(x) \neq y$ **then**
    $S \leftarrow S \circ S_1 \circ (x, y)$;
  else
    $S \leftarrow S \circ S_2 \circ (x, y)$;
  end
end
return S;

[3]We will construct a training sequence for an online learner, but we can trivially convert it to a standard, "batch" training sequence for PAC learning by taking the union of examples over all rounds. This is a technique commonly referred to as "online-to-batch" conversion.

4.1.2 Existence of Frequently Occurring Hypotheses

Let $S_{n;d}$ be the result of running Algorithm 2. Given this constructed training sequence, the following lemma establishes the existence of a frequently occurring hypothesis, which is the key to global stability:

Lemma 4.3. *Let $S = S_{n;d}$ and let $T \sim \mathcal{D}^n$. There exists a hypothesis f such that*

$$\mathbf{Pr}[\mathrm{SOA}(S \circ T) = f] \geq 2^{-d}$$

Proof. We show that this must hold when f is the target concept c. This follows from the fact that each contest label y_i is drawn independently of x_i and the sample so far. Therefore $y_i = c(x_i)$ with probability $1/2$ independently for each contest example, and the probability that all d contest examples are consistent with c is 2^{-d}. (Note that if this holds, the entire sequence $S \circ T$ is consistent with c, because all other examples drawn from $\mathcal{D}$ are consistent with c.) Now, since each contest example forces a mistake on the SOA, and since the SOA does not make more than d mistakes on realizable training sequences, it follows that $\mathrm{SOA}(S \circ T) = c$. Therefore

$$\mathbf{Pr}[\mathrm{SOA}(S \circ T) = c] \geq 2^{-d},$$

which concludes the proof. □

4.1.3 Generalization

The next lemma shows that only hypotheses f that generalize well satisfy the conclusion of Lemma 4.3.

Lemma 4.4 (Generalization)**.** *Let $S = S_{n;d}$ and let $T \sim \mathcal{D}^n$. Every f such that*

$$\mathbf{Pr}[\mathrm{SOA}(S \circ T) = f] \geq 2^{-d}$$

satisfies $\mathcal{L}_{\mathcal{D}}(f) \leq \frac{d}{n}$.

Proof. Let f be a hypothesis such that $\mathbf{Pr}[\mathrm{SOA}(S \circ T) = f] \geq 2^{-d}$ and let $\alpha = \mathcal{L}_{\mathcal{D}}(f)$. We will argue that

$$2^{-d} \leq (1-\alpha)^n. \tag{1}$$

Define the events A, B as follows.

1. A is the event that $\mathrm{SOA}(S \circ T) = f$. By assumption, $\mathbf{Pr}[A] \geq 2^{-d}$.
2. B is the event that f is consistent with T. Since $|T| = n$, we have that $\mathbf{Pr}[B] = (1-\alpha)^n$.

Note that $A \subseteq B$; whenever $\mathrm{SOA}(S \circ T) = f$, it must be the case that f is consistent with T, because the SOA always produces hypotheses which are consistent with the training sequence. Therefore $\mathbf{Pr}[A] \leq \mathbf{Pr}[B]$, which implies (1) and concludes the proof (using the fact that $(1-\alpha) < 2^{-\alpha}$ and taking the logarithm of both sides). □

So far we have shown that any class with finite Littlestone dimension can be learned by a globally-stable algorithm, and that the generalization error of the resulting hypothesis is bounded above by d/n. In the next section, we show how we use this globally-stable learner to produce a differentially-private one.

4.2 Globally-Stable Learning Implies Private Learning

Converting our globally-stable learner into a private learner leverages two tools from differential privacy. In this section, we first give an overview of the algorithm, then describe each of the tools used from differential privacy, then state the privacy guarantees and sample complexity of the overall algorithm.

4.2.1 Overview

A diagram of the algorithm is depicted in Figure 2. First, we draw k samples from $\mathcal{D}^m$. We run each sample through our globally-stable learner G to produce k different hypotheses. Given the global stability of G, we know that there exists a hypothesis which outputs with some frequency and generalizes well; we choose k to ensure that this likely hypothesis is among the $h_1, \ldots, h_k$ output by G with high probability. We then use an (ε, δ)-differentially private subroutine called Stable Histograms (labeled as "DP Heavy Hitter Identification" in the diagram) to select the most frequently occurring hypotheses, at least one of which will be our good and likely hypothesis. We use the result of this subroutine as our new hypothesis class and run a generic DP learning algorithm with a fresh sample to produce our final hypothesis.

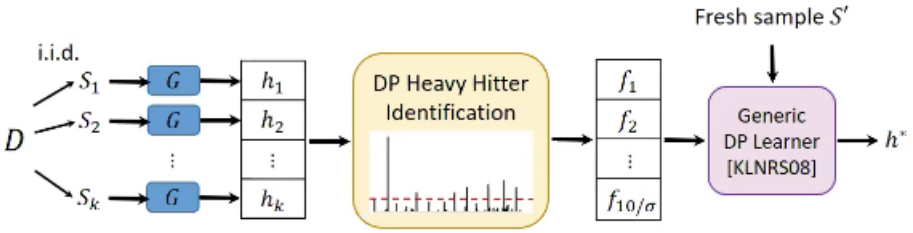

Figure 2: Converting a globally-stable learner G into a private learner. Source: slides from Mark Bun's TCS+ talk.

4.2.2 Stable Histograms

Here we describe the subroutine labeled in the diagram as "DP Heavy Hitter Identification", also known as Stable Histograms. Given a list of elements, this subroutine outputs all elements which occur with frequency at least η while guaranteeing (ε, δ)-differential privacy over the input list. Below we give a more formal definition.

Let $\mathcal{X}$ be the input domain, and let $S \in \mathcal{X}^n$. For an element $x \in \mathcal{X}$, define $\text{freq}_S(x) = \frac{1}{n} \cdot |\{i \in [n] : x_i = x\}|$, i.e., the fraction of the elements in S which are equal to x.

Lemma 4.5. (Stable Histograms [KKMN], [BNS16]). *Let $\mathcal{X}$ be any input domain. For*

$$n \geq O\left(\frac{\log(1/\eta\beta\delta)}{\eta\varepsilon}\right)$$

there exists an (ε, δ)-differentially private algorithm Hist *which, with probability at least $1 - \beta$, on input $S = (x_1, \ldots, x_n)$ outputs a list $L \subseteq \mathcal{X}$ and a sequence $a \in [0,1]^{|L|}$ such that*

1. *Every x with $\text{freq}_S(x) \geq \eta$ appears in L, and*
2. *For every $x \in L$, the estimate a_x satisfies $|a_x - \text{freq}_S(x)| \leq \eta$.*

4.2.3 Generic DP Learner

Using the Exponential Mechanism of McSherry and Talwar [MT07], Kasiviswanath et al. [KLN+11] described a generic differentially-private learner based on approximate empirical risk minimization.

Lemma 4.6. (Generic Private Learner [KLN+11]). *Let $\mathcal{H} \subseteq \{\pm 1\}^{|\mathcal{X}|}$ be a hypothesis class. For*

$$n = O\left(\frac{\log(\mathcal{H}) + \log(1/\beta)}{\alpha\varepsilon}\right)$$

there exists an ε-differentially private algorithm GenericLearner $: (X \times \{\pm 1\})^n \to \mathcal{H}$ *such that the following holds. Let $\mathcal{D}$ be a distribution over $(X \times \{\pm 1\})$ such that there exists $h^* \in \mathcal{H}$ with*

$$\mathcal{L}_{\mathcal{D}}(h^*) \leq \alpha.$$

Then on input $S \sim \mathcal{D}^n$, the algorithm GenericLearner *outputs, with probability at least $1 - \beta$, a hypothesis $\hat{h} \in \mathcal{H}$ such that*

$$\mathcal{L}_{\mathcal{D}}(\hat{h}) \leq 2\alpha.$$

4.2.4 Construction of a private learner

We now state in detail the algorithm which combines the Stable Histograms algorithm with the Generic DP Learner to convert any globally-stable learning algorithm into a differentially-private one. Theorem 4.7 gives the sample complexity of this algorithm. The algorithm is as follows:

Differentially-Private Learner M

1. Let $S_1, \ldots, S_k$ each consist of m i.i.d samples from $\mathcal{D}$. Run G on each batch of samples producing $h_1 = G(S_1), \ldots, h_k = G(S_k)$.
2. Run the Stable Histogram Hist algorithm on input $H = (h_1, \ldots, h_k)$ using privacy parameters $(\varepsilon/2, \delta)$ and accuracy parameters $(\eta/8, \beta/3)$, producing a list L of frequent hypotheses.
3. Let S' consist of n' i.i.d samples from $\mathcal{D}$. Run GenericLearner(S') using the collection of hypotheses L with privacy parameter $\varepsilon/2$ and accuracy parameters $(\alpha/2, \beta/3)$ to output a hypothesis $\hat{h}$.

The following theorem is realized by the learning algorithm M described above.

Theorem 4.7. *Let $\mathcal{H}$ be a concept class over input domain $\mathcal{X}$. Let $G : (\mathcal{X} \times \{\pm 1\})^m \to \{\pm 1\}^{|\mathcal{X}|}$ be a randomized algorithm such that, for $\mathcal{D}$ a realizable distribution and $S \sim \mathcal{D}^m$, there exists a hypothesis h such that $\mathbf{Pr}[G(S) = h] \leq \eta$ and $\mathcal{L}_{\mathcal{D}}(h) \leq \alpha/2$.*

Then for some

$$n = O\left(\frac{m \cdot \log(1/\eta\beta\delta)}{\eta\varepsilon} + \frac{\log(1/\eta\beta)}{\alpha\varepsilon}\right)$$

there exists an (ε, δ)-differentially private algorithm $M : (\mathcal{X} \times \{\pm 1\})^n \to \{\pm 1\}^X$ which, given n i.i.d samples from $\mathcal{D}$, produces a hypothesis $\hat{h}$ such that $\mathcal{L}_{\mathcal{D}}(\hat{h}) \leq \alpha$ with probability at least $1 - \beta$.

Here, the parameter

$$k = O\left(\frac{\log(1/\eta\beta\delta)}{\eta\varepsilon}\right)$$

is chosen so that Lemma 4.5 guarantees algorithm Hist succeeds with the stated accuracy parameters. The parameter

$$n' = O\left(\frac{\log(1/\eta\beta)}{\alpha\varepsilon}\right)$$

is chosen so that Lemma 4.5 guarantees that GenericLearner succeeds on a list L of size $|L| \leq 2/\eta$ with the given accuracy and confidence parameters.

5 Recent Work and Open Questions

We conclude by listing some follow-up work and open questions.

1. Jung, Kim, and Tewari [JKT20] study the relationship between private and online learning beyond the setting of binary classification. For multiclass classification, they show that the equivalence holds, using a combinatorial parameter analogous to Littlestone dimension. In the case of regression, they show that private learning implies online learning; however, the converse remains an open question.

2. A natural related question is: are there efficient transformations between these two types of learners? Gonen, Hazan, and Moran [GHM19] showed a restricted class of polynomial-time, differentially-private learners with low sample complexity which can be efficiently transformed into polynomial-time online learners. However, such a conversion does not exist in general; Bun [Bun20] shows that, assuming the existence of one-way functions, such an efficient conversion is impossible even for general pure-private learners with polynomial sample complexity. Whether the converse holds (does polynomial-time online learning imply polynomial-time private learning?) is still an open question.

3. The fundamental theorem of PAC learning relies on showing several results: (i) uniform convergence implies learnability, (ii) learnability implies finite VC dimension, (iii) and finite VC dimension implies uniform convergence. With the works surveyed in this paper, we are close to a fundamental theorem of private PAC learning. It has been shown that: proper private learning implies private uniform convergence [BLM19] and private learnability implies finite Littlestone dimension [ALMM19]. To complete a fundamental theorem of private PAC learning, all that remains to be shown is that finite Littlestone dimension implies private uniform convergence. See [BLM19] for definitions.

References

[ALMM19] N. Alon, R. Livni, M. Malliaris, and S. Moran. Private PAC Learning Implies Finite Littlestone Dimension. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 2019. doi:10.1145/3313276.3316312. (document), 1, 3.3, 3.4, 3.5, 3.6, 3.7, 3

[BDMN05] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138, 2005. 1.2

[BDPSS09] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic Online Learning. In *COLT*, 2009. 1.3

[BEHW89] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989. 1.1

[BLM19] Olivier Bousquet, Roi Livni, and Shay Moran. Passing tests without memorizing: Two models for fooling discriminators. *arXiv preprint arXiv:1902.03468*, 2019. 3

[BLM20] M. Bun, R. Livni, and S. Moran. An Equivalence Between Private Classification and Online Prediction. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 2020. doi:10.1109/FOCS46700.2020.00044. (document), 1, 4

[BNS16] Mark Bun, Kobbi Nissim, and Uri Stemmer. Simultaneous private learning of multiple concepts. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, ITCS '16, page 369–380, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2840728.2840747. 4.5

[BNSV15] Mark Bun, Kobbi Nissim, Uri Stemmer, and Salil Vadhan. Differentially private release and learning of threshold functions. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 634–649. IEEE, 2015. 1.8, 2.1

[Bun20] Mark Bun. A computational separation between private learning and online learning. *CoRR*, abs/2007.05665, 2020. 2

[DMNS16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. *Journal of Privacy and Confidentiality*, 7(3):17–51, 2016. 1.2

[DN04] Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In *Annual International Cryptology Conference*, pages 528–544. Springer, 2004. 1.2

[ER52] P. Erdós and R. Rado. Combinatorial theorems on classifications of subsets of a given set. *Proceedings of the London Mathematical Society*, s3-2(1):417–439, 1952. URL: https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s3-2.1.417, arXiv:https://londmathsoc.onlinelibrary.wiley.com/doi/pdf/10.1112/plms/s3-2.1.417, doi:https://doi.org/10.1112/plms/s3-2.1.417. 3.1

[GHM19] Alon Gonen, Elad Hazan, and Shay Moran. Private learning implies online learning: An efficient reduction. *CoRR*, abs/1905.11311, 2019. 2

[Hod97] Wilfrid Hodges. *A shorter model theory*. Cambridge university press, 1997. 2.2

[JKT20] Young Jung, Baekjin Kim, and Ambuj Tewari. On the equivalence between online and private learnability beyond binary classification. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16701–16710. Curran Associates, Inc., 2020. URL: https://proceedings.neurips.cc/paper/2020/file/c24fe9f765a44048868b5a620f05678e-Paper.pdf. 1

[KKMN] Aleksandra Korolova, Krishnaram Kenthapadi, Nina Mishra, and Ros Ntoulas. Www 2009 madrid! track: Data mining / session: Web mining releasing search queries and clicks privately. 4.5

[KLN+11] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What Can We Learn Privately? *SIAM Journal on Computing*, 40(3):793–826, 2011. doi:10.1137/090756090. 1.2, 4.2.3, 4.6

[Lit87] N. Littlestone. Learning Quickly When Irrelevant Attributes Abound: A New Linear-Threshold Algorithm. In *1987 IEEE 28th Annual Symposium on Foundations of Computer Science*, 1987. doi:10.1109/SFCS.1987.37. 1, 1.3, 1.3.1

[MT07] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, October 2007. URL: https://www.microsoft.com/en-us/research/publication/mechanism-design-via-differential-privacy/. 4.2.3

[She78] Saharon Shelah. *Classification theory: and the number of non-isomorphic models*. Elsevier, 1978. 2.2

[Vad17] Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017. 1.2, 1.6

[Val84] L. G. Valiant. A theory of the learnable, 1984. 1.1

[VC71] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 1971. 1.1