Learning Privately with Labeled and Unlabeled Examples

Amos Beimel\textsuperscript{1} · Kobbi Nissim\textsuperscript{2} · Uri Stemmer\textsuperscript{1}*

Received: 14 August 2018 / Accepted: 16 July 2020 / Published online: 3 August 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

A private learner is an algorithm that given a sample of labeled individual examples outputs a generalizing hypothesis while preserving the privacy of each individual. In 2008, Kasiviswanathan et al. (FOCS 2008) gave a generic construction of private learners, in which the sample complexity is (generally) higher than what is needed for non-private learners. This gap in the sample complexity was then further studied in several followup papers, showing that (at least in some cases) this gap is unavoidable. Moreover, those papers considered ways to overcome the gap, by relaxing either the privacy or the learning guarantees of the learner. We suggest an alternative approach, inspired by the (non-private) models of semi-supervised learning and active-learning, where the focus is on the sample complexity of labeled examples whereas unlabeled examples are of a significantly lower cost. We consider private semi-supervised learners that operate on a random sample, where only a (hopefully small) portion of this sample is labeled. The learners have no control over which of the sample elements are labeled. Our main result is that the labeled sample complexity of private learners is characterized by the VC dimension. We present two generic constructions of private semi-supervised learners. The first construction is of learners where the labeled sample complexity is proportional to the VC dimension of the concept class, however, the unlabeled sample complexity of the algorithm is as big as the representation length of domain elements. Our second construction presents a new technique for decreasing the labeled sample complexity of a given private learner, while roughly maintaining its unlabeled sample complexity. In addition, we show that in some settings the labeled sample complexity does not depend on the privacy parameters of the learner.

Keywords Differential privacy · PAC learning · Semi-supervised learning · Active learning

A preliminary version of this paper appeared in SODA’15.

* Uri Stemmer
u@uri.co.il

Extended author information available on the last page of the article
1 Introduction

A private learner is an algorithm that given a sample of labeled examples, where each example represents an individual, outputs a generalizing hypothesis while preserving the privacy of each individual. This formal notion, combining the requirements of PAC learning [40] and Differential Privacy [23], was presented in 2008 by Kasiviswanathan et al. [32], who also gave a generic construction of private learners. However, the sample complexity of the learner of [32] is (generally) higher than what is needed for non-private learners. Namely, their construction requires $O(\log |C|)$ samples for learning a concept class $C$, as opposed to the non-private sample complexity of $\Theta(\text{VC}(C))$.

This gap in the sample complexity was studied in several followup papers. For pure differential privacy, it was shown that in some cases this gap can be closed with the price of giving up proper learning—where the output hypothesis should be from the learned concept class—for improper learning. Indeed, it was shown that for the class of point functions over domain of size $2^d$, the sample complexity of every proper private learner is $\Omega(d)$ (matching the upper bound of [32]), whereas there exist improper private learners with sample complexity $O(1)$ that use pseudorandom or pairwise independent functions as their output hypotheses [8, 9]. A complete characterization for the sample complexity of pure-private improper-learners was given in [9] in terms of a new dimension—the Representation Dimension. They showed that $\Theta(\text{RepDim}(C))$ examples are both necessary and sufficient for a pure-private improper-learner for a class $C$. Following that, Feldman and Xiao [27] separated the sample complexity of pure-private learners from that of non-private ones, and showed that the representation dimension can sometimes be significantly bigger than the VC dimension. For example, they showed that every pure-private learner (proper or improper) for the class of thresholds over $\{0, 1\}^d$ requires $\Omega(d)$ samples [27] (while there exists a non-private proper-learner with sample complexity $O(1)$).

Another approach for reducing the sample complexity of private learners is to relax the privacy requirement to approximate differential privacy. This relaxation was shown to be significant as it allows privately and properly learning point functions with $O(1)$ sample complexity, and threshold functions with sample complexity $O\left(\left(\log^* d\right)^{1.5}\right)$ [10, 16, 31]. This dependency in $\log^* d$ was shown to be necessary by Bun et al. [16] and Alon et al. [4]. Specifically, [4] showed that every approximate-private learner for the class of thresholds over $\{0, 1\}^d$ requires $\Omega(\log^* d)$ samples. This separates the sample complexity of approximate-private learners from that of non-private learners.

Alon et al. [4] and Bun et al. [14] related the sample complexity of approximate-private improper-learners to the Littlestone dimension of the target concept class $C$ (denoted $\text{LDim}(C)$). Specifically, they showed that $\Omega\left(\log^* (\text{LDim}(C))\right)$ samples are

---

1 To simplify the exposition, we omit in this section dependency on all variables except for $d$, corresponding to the representation length of domain elements.
necessary, and that $2^{O(L\text{Dim}(C))}$ samples suffices to learn $C$ (improperly) with approximate differential privacy.

Tables 1 and 2 summarize the currently known bounds on the sample complexity of private learners. Table 1 specifies general upper bounds, and Table 2 specifies known upper and lower bounds on the sample complexity of privately learning thresholds over $\{0, 1\}^d$.

### 1.1 This Work

In this work we examine an alternative approach for reducing the costs of private learning, inspired by the (non-private) models of semi-supervised learning [41] and active learning [34]. In both models, the focus is on reducing the sample complexity of labeled examples whereas it is assumed that unlabeled examples can be obtained with a significantly lower cost. In this vein, Balcan and Feldman [5] suggested a generic conversion of active learners in the model of statistical queries [33] into learners that also provide differential privacy. For example, Balcan and Feldman showed an active pure-private proper-learner for the class of thresholds over $\{0, 1\}^d$ that uses $O(1)$ labeled examples and $O(d)$ unlabeled examples.

We show that while the unlabeled sample complexity of private learners is subject to the lower bounds mentioned in Tables 1 and 2, the labeled sample complexity

---

2 A semi-supervised learner uses a small batch of labeled examples and a large batch of unlabeled examples, whereas an active-learner operates on a large batch of unlabeled example and chooses (maybe adaptively) a subset of the examples and gets their labels.
is characterized by the VC dimension of the target concept class. We present two
generic constructions of private semi-supervised learners via an approach that devi-
ates from most of the research in semi-supervised and active learning: (1) Semi-
supervised learning algorithms and heuristics often rely on strong assumptions
about the data, e.g., that close points are likely to be labeled similarly, that the data
is clustered, or that the data lies on a low dimensional subspace of the input space.
In contrast, we work in the standard PAC learning model, and need not make any
further assumptions. (2) Active learners examine their pool of unlabeled data and
then choose (maybe adaptively) which data examples to label. Our learners have no
control over which of the sample elements are labeled.

Our main result is that the labeled sample complexity of such learners is char-
acterized by the VC dimension. Our first generic construction is of learners where
the labeled sample complexity is proportional to the VC dimension of the concept
class. However, the unlabeled sample complexity of the algorithm is as big as the
representation length of domain elements. The learner for a class $C$ starts with an
unlabeled database and uses private sanitization to create a synthetic database, with
roughly $\text{VC}(C)$ points, that can answer queries in a class related to $C$. It then uses
this database to choose a subset of the hypotheses of size $2^{O(\text{VC}(C))}$ and then uses
the exponential mechanism \cite{35} to choose from these hypotheses using $O(\text{VC}(C))$
labeled examples.

As an example, applying this technique with the private sanitizer for threshold
functions from \cite{10, 16, 31} we get a (semi-supervised) approximate-private proper-
learner for thresholds over $\{0, 1\}^d$ with optimal $O(1)$ labeled sample complexity and
near optimal $O\left((\log^* d)^{1.5}\right)$ unlabeled sample complexity. This matches the labeled
sample complexity of Balcan and Feldman \cite{5} (ignoring the dependency in all
parameters except for $d$), and improves on the unlabeled sample complexity.\(^3\)

Our second construction presents a new technique for decreasing the labeled sam-
ple complexity of a given private learner $A$. At the heart of this construction is a
technique for choosing (non-privately) a hypothesis using a small labeled database;
this hypothesis is used to relabel the small database and label the bigger database,
which is given to the private learner $A$.

Consider, for example, the concept class $\text{RECTANGLE}^\ell_d$ containing all axis-aligned,
$\ell$-dimensional rectangles, where each dimension consists of $2^d$ points. Applying
our techniques on the learner from \cite{10} results in a non-active semi-supervised pri-
vate learner with optimal $O(\ell)$ labeled sample complexity and with $\widetilde{O}(\ell^3 \cdot 8^{\log^* d})$
unlabeled sample complexity.\(^4\) This matches the labeled sample complexity of Bal-
can and Feldman \cite{5}, and improves the unlabeled sample complexity whenever the
dimension $\ell$ is not too big (roughly, $\ell \leq \sqrt{d}$).

\(^3\) We remark that—unlike this work—the focus in \cite{5} is on the dependency of the labeled sample com-
plexity in the approximation parameter. As our learners are non-active, their labeled sample complexity
is lower bounded by $\Omega(\frac{1}{\ell^2})$ (where $a$ is the approximation parameter).

\(^4\) Combining the technique of \cite{10} and the recent result of \cite{31}, the unlabeled sample complexity can be
reduced to $\widetilde{O}(\ell^3 \cdot (\log^* d)^{1.5})$. 

\(\copyright\) Springer
Private Agnostic Learners We show that the technique of our second algorithm (the relabeling algorithm) is also very useful for the standard model of private learning, i.e., not only for semi-supervised private learning. We prove that if a class of hypotheses $C$ is privately learnable in the realizable case (when the examples are labeled by some hypothesis in the class) by a proper learner then the class $C$ is privately learnable in the agnostic case (when the examples are arbitrarily labeled and the learner should find an hypothesis that is (almost) as good as the best hypothesis in the class); the resulting agnostic learner is proper and its sample complexity is roughly the same as the sample complexity of the learner in the realizable case (ignoring the accuracy and privacy parameters). Furthermore, if the original learner provided pure differential privacy, then the resulting agnostic learner also provides pure differential privacy. This relabeling algorithm was also used in a follow-up work of Alon et al. [3] to show closure properties of private learning, e.g., if a class $C$ is privately learnable then the class $C^{⊕} = \{c_1 \oplus c_2 : c_1, c_2 \in C\}$ is privately learnable with roughly the same same sample complexity. Alon et al. [3] also extended the reduction from private agnostic learning to private learning in the realizable case to improper learners.

Private Active Learners We study the labeled sample complexity of private active learners, i.e., learners that operate on a pool of unlabeled examples (individuals’ data) and adaptively query the labels of specific examples. As those queries depend on individuals’ data, they may breach privacy if exposed. We, therefore, introduce a stronger definition for private active learners that remedies this potential risk, and show that (most of) our learners satisfy this stronger definition, while the learners of [5] do not. This strong definition has its downside, as we show that (at least in some cases) it introduces a $\frac{1}{\alpha}$ blowup to the labeled sample complexity (where $\alpha$ is the approximation parameter). On the other hand, when considering private active learners that only satisfy the definition of [5] (which is still a reasonable definition), we show that the labeled sample complexity has no dependency on the privacy parameters.

1.2 Related Work

Differential privacy was defined in [23] and the relaxation to approximate differential privacy is from [22]. Most related to our work is the work on private learning and its sample complexity [4, 8–11, 15, 16, 18, 24, 27, 32] and the early work on sanitization [12]. Blum et al. [11] showed that computationally efficient private learners exist for all concept classes that can be efficiently learned in the statistical queries model of [33]. Kasiviswanathan et al. [32] showed an example of a concept class—the class of parity functions—that is not learnable in the statistical queries model but can be learned privately and efficiently. These positive results show that many “natural” learning tasks that are efficiently learned non-privately can be learned privately and efficiently.

Chaudhuri and Hsu [18] presented upper and lower bounds on the sample complexity of label-private learners, a relaxation of private learning where the learner is required to only protect the privacy of the labels in the sample. Following that,
Beimel et al. [10] showed that the VC dimension completely characterizes the sample complexity of such learners.

Dwork et al. [24] showed how to boost the accuracy of private learning algorithms. That is, given a private learning algorithm that has a big classification error, they produced a private learning algorithm with small error. Other tools for private learning include, e.g., private SVM [37], private logistic regression [19], private empirical risk minimization [20], private stochastic convex optimization [6, 26], and deep learning [1, 36].

2 Preliminaries

Notation We use \( O_{\gamma}(g(n)) \) as a shorthand for \( O(h(\gamma) \cdot g(n)) \) for some non-negative function \( h \). In informal discussions, we sometimes write \( O(g(n)) \) to indicate that \( g(n) \) is missing lower order terms. We use \( X \) to denote an arbitrary domain, and \( X_d \) for the domain \( \{0, 1\}^d \).

2.1 Differential Privacy

Consider a database where each entry contains information pertaining to an individual. An algorithm operating on such databases is said to preserve differential privacy if its outcome is insensitive to any modification in a single entry. Formally:

**Definition 2.1 (Differential privacy [22, 23])** Databases \( S_1 \in X^n \) and \( S_2 \in X^n \) over a domain \( X \) are called neighboring if they differ in exactly one entry. A randomized algorithm \( A \) is \((\epsilon, \delta)\)-differentially private if for all neighboring databases \( S_1, S_2 \in X^n \), and for all sets \( F \) of outputs,

\[
\Pr[A(S_1) \in F] \leq \exp(\epsilon) \cdot \Pr[A(S_2) \in F] + \delta.
\]

The probability is taken over the random coins of \( A \). When \( \delta = 0 \) we omit it and say that \( A \) preserves pure differential privacy, otherwise (when \( \delta > 0 \)) we say that \( A \) preserves approximate differential privacy.

2.2 Semi-supervised PAC Learning

The standard PAC model (and similarly private PAC) focuses on learning a class of concepts from a sample of labeled examples. In a situation where labeled examples are significantly more costly than unlabeled ones, it is natural to attempt to use a combination of labeled and unlabeled data to reduce the number of labeled examples needed. Such learners may have no control over which of the examples are labeled, as in semi-supervised learning, or may specifically choose which examples to label, as in active learning. In this section we focus on semi-supervised learning. Active learning will be discussed in Sect. 7.
A concept $c : X \to \{0, 1\}$ is a predicate that labels examples taken from the domain $X$ by either 0 or 1. A concept class $C$ over $X$ is a set of concepts (predicates) mapping $X$ to $\{0, 1\}$. A semi-supervised learner is given $n$ examples sampled according to an unknown probability distribution $\mu$ over $X$, where $m \leq n$ of these examples are labeled according to an unknown target concept $c \in C$. The learner succeeds if it outputs a hypothesis $h$ that is a good approximation of the target concept according to the distribution $\mu$. Formally:

**Definition 2.2** Let $c$ and $\mu$ be a concept and a distribution over a domain $X$. The *generalization error* of a hypothesis $h : X \to \{0, 1\}$ w.r.t. $c$ and $\mu$ is defined as $\text{error}_\mu(c, h) = \Pr_{x \sim \mu}[h(x) \neq c(x)]$. Let $\mu$ be a distribution over $X \times \{0, 1\}$. The *generalization error* of a hypothesis $h$ w.r.t. $\mu$ is defined as $\text{error}_\mu(h) = \Pr_{(x, y) \sim \mu}[h(x) \neq y]$. When a hypothesis $h$ has generalization error at most $\alpha$, we say that it is $\alpha$-good.

**Definition 2.3** (Semi-supervised learning [40, 41]) Let $C$ be a concept class over a domain $X$, and let $\mathcal{A}$ be an algorithm operating on (partially) labeled databases. Algorithm $\mathcal{A}$ is an $(\alpha, \beta, n, m)$-SSL (semi-supervised learner) for $C$ if for all concepts $c \in C$ and all distributions $\mu$ on $X$ the following holds.

Let $D = (x_i, y_i)_{i=1}^n \in (X \times \{0, 1, \perp\})^n$ be a database s.t. (1) each $x_i$ is drawn i.i.d. from $\mu$; (2) in the first $m$ entries $y_i = c(x_i)$; (3) in the last $(n - m)$ entries $y_i = \perp$. Then, $\Pr[\mathcal{A}(D) = h \text{ s.t. } \text{error}_\mu(c, h) > \alpha] \leq \beta$. The probability is taken over the choice of the samples from $\mu$ and the coin tosses of $\mathcal{A}$.

If $n = m$ (i.e., all examples are labeled), then an $(\alpha, \beta, n, n)$-SSL is called an $(\alpha, \beta)$-PAC learner with sample complexity $n$.

If a semi-supervised learner is restricted to only output hypotheses from the target concept class $C$, then it is called a proper learner. Otherwise, it is called an improper learner. We sometimes refer to the input of a semi-supervised learner as two databases $S \in (X \times \{0, 1\})^m$ and $D \in (X \times \{\perp\})^{n-m}$, where $m$ and $n$ are the labeled and unlabeled sample complexities of the learner.

**Definition 2.4** Given a labeled sample $S = (x_i, y_i)_{i=1}^m$, the *empirical error* of a hypothesis $h$ on $S$ is $\text{error}_S(h) = \frac{1}{m} |\{i : h(x_i) \neq y_i\}|$. Given an unlabeled sample $D = (x_i)_{i=1}^n$ and a target concept $c$, the *empirical error* of $h$ w.r.t. $D$ and $c$ is $\text{error}_D(h, c) = \frac{1}{n} |\{i : h(x_i) \neq c(x_i)\}|$.

Semi-supervised learning algorithms operate on a (partially) labeled sample with the goal of choosing a hypothesis with a small generalization error. Standard arguments in learning theory state that the generalization of a hypothesis $h$ and its empirical error (observed on a large enough sample) are similar. Hence, in order to output a hypothesis with small generalization error it suffices to output a hypothesis with small empirical error. The difference between the empirical error and the generalization error can be analyzed using the VC dimension of the hypothesis class (see Sect. 2.3). Furthermore, a recent line of work [7, 21, 30]...
shows that in the context of private learning, this difference is guaranteed to be small, even when the VC dimension of the hypothesis class is infinite.

Agnostic Learners Consider a semi-supervised learner for an unknown class $C$ that uses a (known) hypotheses class $H$. If $H \neq C$, then a hypothesis with small empirical error might not exist in $H$. Such learners are referred to in the literature as agnostic-learners, and are only required to produce a hypothesis $f \in H$ (approximately) minimizing $\text{error}_\mu(c, f)$, where $c$ is the (unknown) target concept.

**Definition 2.5 (Agnostic semi-supervised learning)** Let $H$ be a concept class over a domain $X$, and let $\mathcal{A}$ be an algorithm operating on (partially) labeled databases. Algorithm $\mathcal{A}$ is an $(\alpha, \beta, n, m)$-agnostic-SSL using $H$ if for all distributions $\mu$ on $X \times \{0, 1\}$ (i.e., a distribution over examples and labels) the following holds.

Let $D = (x_i, y_i)_{i=1}^m \in (X \times \{0, 1, \perp\})^m$ be a database s.t. (1) for every $1 \leq i \leq m$ it holds that $(x_i, y_i)$ is sampled independently from $\mu$; (2) for every $m < i \leq n$ it holds that $x_i$ is sampled independently from the marginal distribution of $\mu$ on $X$, and $y_i = \perp$. Then, $\mathcal{A}(D)$ outputs a hypothesis $h \in H$ satisfying $\Pr[\text{error}_\mu(c, h) \leq \min_{f \in H}\{\text{error}_\mu(c, f)\} + \alpha] \geq 1 - \beta$. The probability is taken over the choice of the samples from $\mu$ and the coin tosses of $\mathcal{A}$.

**Private Semi-Supervised PAC learning.** Similarly to [32] we define private semi-supervised learning as the combination of Definitions 2.1 and 2.3.

**Definition 2.6 (Private semi-supervised learning)** Let $\mathcal{A}$ be an algorithm that gets an input $S \in (X \times \{0, 1, \perp\})^n$. Algorithm $\mathcal{A}$ is an $(\alpha, \beta, \epsilon, \delta, n, m)$-PSSL (private SSL) for a concept class $C$ over $X$ if $\mathcal{A}$ is an $(\alpha, \beta, n, m)$-SSL for $C$ and $\mathcal{A}$ is $(\epsilon, \delta)$-differentially private.

Note that the utility requirement in the above definition is an average-case requirement, as the learner is only required to do well on typical samples (i.e., samples drawn i.i.d. from some underlying distribution, where the first $m$ entries are correctly labeled by a target concept $c \in C$, and the last $(n - m)$ entries are labeled by $\perp$). In contrast, the privacy requirement is a worst-case requirement, and Inequality (1) must hold for every pair of neighboring databases of size $n$ (no matter how they are generated and labeled).

**Active Learning** Semi-supervised learners are a subset of the larger family of active learners. Active learners can adaptively request to reveal the labels of specific examples. See formal definition and discussion in Sect. 7.

### 2.3 Preliminaries from Learning Theory: VC Bounds

The Vapnik-Chervonenkis (VC) Dimension is a combinatorial measure of concept classes that characterizes the sample size of PAC learners. Let $C$ be a
concept class over a domain \( X \), and let \( B = \{ b_1, \ldots, b_\ell \} \subseteq X \). The set of all dichotomies on \( B \) that are realized by \( C \) is \( \Pi_C(B) = \{ (c(b_1), \ldots, c(b_\ell)) : c \in C \} \). A set \( B \subseteq X \) is shattered by \( C \) if \( C \) realizes all possible dichotomies over \( B \), i.e., \( \Pi_C(B) = \{0, 1\}^{|B|} \).

**Definition 2.7 (VC-dimension [42])** The VC-dimension \( \text{VC}(C) \) of \( C \) is the cardinality of the largest set \( B \subseteq X \) shattered by \( C \). If arbitrarily large finite sets can be shattered by \( C \), then \( \text{VC}(C) = \infty \).

Sauer’s lemma bounds the cardinality of \( \Pi_C(B) \) in terms of \( \text{VC}(C) \) and \(|B|\).

**Theorem 2.8 ([38])** Let \( C \) be a concept class over a domain \( X \), and let \( B \subseteq X \) such that \(|B| > \text{VC}(C)\). It holds that

\[
\Pi_C(B) \leq \left( \frac{e^{|B|}}{\text{VC}(C)} \right)^{|C|}.
\]

Classical results in computational learning theory state that a sample of size \( \Theta(\text{VC}(C)) \) is both necessary and sufficient for the PAC learning of a concept class \( C \). The following two theorems give upper and lower bounds on the sample complexity.

**Theorem 2.9 ([25])** For any \( (\alpha, \beta < \frac{1}{2}, n, m) \)-SSL for a class \( C \) it holds that

\[
m \geq \frac{\text{VC}(C) - 1}{16\alpha}.
\]

**Theorem 2.10 (Uniform convergence [13, 42])** Let \( C \) and \( \mu \) be a concept class and a distribution over a domain \( X \). Let \( \alpha, \beta > 0 \), and \( m \geq \frac{n}{\alpha} \left( \text{VC}(C) \ln \left( \frac{10 \text{VC}(C)}{\alpha} \right) + \ln \left( \frac{2}{\beta} \right) \right) \). Fix a concept \( c \in C \), and suppose that we draw a sample \( S = (x_i, y_i)_{i=1}^m \), where each \( x_i \) is drawn i.i.d. from \( \mu \) and \( y_i = c(x_i) \). Then,

\[
\Pr \left[ \exists h \in C \text{ s.t. } \text{error}_\mu(h, c) > \alpha \wedge \text{error}_S(h) = 0 \right] \leq \beta.
\]

Hence, an algorithm that takes a sample of \( m = \Omega_{\alpha, \beta}(\text{VC}(C)) \) labeled examples and outputs a concept \( h \in C \) that agrees with the sample is a PAC learner for \( C \). The following is a simple generalization of Theorem 2.10.

**Theorem 2.11 (Uniform convergence [13, 42])** Let \( C \) and \( \mu \) be a concept class and a distribution over a domain \( X \). Let \( \alpha, \beta > 0 \), and \( m \geq \frac{48}{\alpha} \left( 10 \text{VC}(C) \log \left( \frac{48e}{\alpha} \right) + \log \left( \frac{2}{\beta} \right) \right) \).

Suppose that we draw a sample \( S = (x_i)_{i=1}^m \), where each \( x_i \) is drawn i.i.d. from \( \mu \). Then,

\[
\Pr \left[ \exists c, h \in C \text{ s.t. } \text{error}_\mu(c, h) \geq \alpha \text{ and } \text{error}_S(c, h) \leq \alpha/10 \right] \leq \beta.
\]

The above theorem generalizes Theorem 2.10 in two aspects. First, it holds simultaneously for every pair \( c, h \in C \), whereas in Theorem 2.10 the target concept \( c \) is fixed before generating the sample. Second, Theorem 2.10 only ensures that a
hypothesis $h$ has small generalization error if $\operatorname{error}_S(h) = 0$. In Theorem 2.11 on the other hand, this is guaranteed even if $\operatorname{error}_S(h)$ is small (but non-zero).

The next theorem handles (in particular) the agnostic case, in which the concept class $C$ is unknown and the learner uses a hypotheses class $H$. In particular, given a labeled sample $S$ there may be no $h \in H$ for which $\operatorname{error}_S(h)$ is small.

**Theorem 2.12** (VC-dimension agnostic generalization bound) There exists a constant $\gamma$ such that for every domain $X$, every concept class $C$ over the domain $X$, and every distribution $\mu$ over the domain $X \times \{0, 1\}$: For a sample $S = (x_i, y_i)_{i=1}^m$ where

$$m \geq \gamma \frac{\operatorname{VC}(C) + \ln(\frac{1}{\beta})}{a^2}$$

and $\{(x_i, y_i)\}$ are drawn i.i.d. from $\mu$, it holds that

$$\Pr \left[ \exists h \in C \text{ s.t. } |\operatorname{error}_{\mu}(h) - \operatorname{error}_S(h)| \geq a \right] \leq \beta.$$

Notice that the sample size in Theorem 2.11 is smaller than the sample size in Theorem 2.12, where, basically, the former is proportional to $\frac{1}{a}$ and the latter is proportional to $\frac{1}{\gamma^2}$.

### 2.4 Some Differentially Private Tools

#### 2.4.1 The Exponential Mechanism

We next describe the exponential mechanism of McSherry and Talwar [35]. We present its private learning variant; however, it can be used in more general scenarios. The goal here is to choose a hypothesis $h \in H$ approximately minimizing the empirical error. The choice is probabilistic, where the probability mass that is assigned to each hypothesis decreases exponentially with its empirical error.

**Proposition 2.13** (The exponential mechanism) (i) The exponential mechanism is $\varepsilon$-differentially private. (ii) Let $\hat{\varepsilon} \triangleq \min_{f \in H} \{\operatorname{error}_S(f)\}$. For every $\Delta > 0$, the probability that the exponential mechanism outputs a hypothesis $h$ such that $\operatorname{error}_S(h) > \hat{\varepsilon} + \Delta$ is at most $|H| \cdot \exp(-\varepsilon \Delta m / 2)$. 
2.4.2 Data Sanitization

Given a database $S = (x_1, \ldots, x_m)$ containing elements from some domain $X$, the goal of data sanitization is to output (while preserving differential privacy) another database $\hat{S}$ that is in some sense similar to $S$. This returned database $\hat{S}$ is called a sanitized database, and the algorithm computing $\hat{S}$ is called a sanitizer.

For a concept $c : X \rightarrow \{0, 1\}$ define $Q_c : X^* \rightarrow [0, 1]$ as

$$Q_c(S) = \frac{1}{|S|} \cdot \left| \{ i : c(x_i) = 1 \} \right|.$$  

That is, $Q_c(S)$ is the fraction of the entries in $S$ that satisfy $c$. A sanitizer for a concept class $C$ is a differentially private algorithm that given a database $S$ outputs a database $\hat{S}$ s.t. $Q_c(S) \approx Q_c(\hat{S})$ for every $c \in C$.

**Definition 2.14 (Sanitization [12])** Let $C$ be a class of concepts mapping $X$ to $\{0, 1\}$. Let $\mathcal{A}$ be an algorithm that on an input database $S \in X^*$ outputs another database $\hat{S} \in X^*$. Algorithm $\mathcal{A}$ is an $(\alpha, \beta, \epsilon, \delta, m)$-sanitizer for predicates in the class $C$, if

1. $\mathcal{A}$ is $(\epsilon, \delta)$-differentially private;
2. For every input $S \in X^*$,

$$\Pr_{\mathcal{A}} \left[ \exists c \in C \text{ s.t. } |Q_c(S) - Q_c(\hat{S})| > \alpha \right] \leq \beta.$$  

The probability is over the coin tosses of algorithm $\mathcal{A}$. As before, when $\delta=0$ (pure privacy) we omit it from the set of parameters.

**Theorem 2.15 (Blum et al. [12])** For any class of predicates $C$ over a domain $X$, and any parameters $\alpha, \beta, \epsilon$, there exists an $(\alpha, \beta, \epsilon, m)$-sanitizer for $C$, where the size of the database $m$ satisfies:

$$m = O\left( \frac{\log |X| \cdot \text{VC}(C) \cdot \log(1/\alpha)}{\alpha^3 \epsilon} + \frac{\log(1/\beta)}{\epsilon \alpha} \right).$$

The returned sanitized database contains $O\left( \frac{\text{VC}(C)}{\alpha^2} \cdot \log(1/\alpha) \right)$ elements.
2.4.3 Sub-sampling Techniques

A known fact is that the privacy guarantees of a differentially private algorithm can be boosted using sub-sampling. Specifically, consider an algorithm that takes a large database as input, and runs an \((\varepsilon^*, \delta)\)-differentially private algorithm on a small random subset of the input database. The following claim shows that this improves the privacy guarantees. The intuition is simple: Fix two neighboring databases \(D, D'\) differing (only) on their \(i\)th entry. If the \(i\)th entry is ignored (which happens with high probability), then the executions on \(D\) and on \(D'\) are the same (i.e., perfect privacy). Otherwise, \((\varepsilon^*, \delta)\)-privacy is preserved.

Claim 2.16 ([8, 32]) Let \(A\) be an \((\varepsilon^*, \delta)\)-differentially private algorithm operating on databases of size \(n\). Fix \(\varepsilon \leq 1\), and denote \(t = \frac{4}{\varepsilon}(3 + \exp(\varepsilon^*))\). Construct an algorithm \(B\) that on input a database \(D = (z_i)_{i=1}^t\) uniformly at random selects a subset \(J \subseteq \{1, 2, \ldots, t\}\) of size \(n\), and runs \(A\) on the multiset \(D_J = (z_i)_{i \in J}\). Then, \(B\) is \((\varepsilon, \frac{4e}{3+\exp(\varepsilon^*)}\delta)\)-differentially private.

Remark 2.17 In Claim 2.16 we assume that \(A\) treats its input as a multiset. If this is not the case, then algorithm \(B\) should be modified to randomly shuffle the elements in \(D_J\) before applying \(A\) on \(D_J\).

For our constructions we will sometimes apply a differentially private algorithm on the outcome of \(i.i.d.\) samples from the input database. Consider two neighboring databases \(D, D'\) differing on their \(i\)th entry. Unlike in Claim 2.16, the risk is that this entry will appear several times in the database on which \(A\) is executed. As the next claim states, the effect on the privacy guarantees are small. The intuition is that the probability of the \(i\)th entry appearing “too many” times is negligible.

Claim 2.18 ([16]) Let \(\varepsilon \leq 1\) and \(A\) be an \((\varepsilon, \delta)\)-differentially private algorithm operating on databases of size \(n\). Construct an algorithm \(B\) that on input a database \(D = (z_i)_{i=1}^m\), where \(m \geq n\), applies \(A\) on a database \(D'\) containing \(n\) \(i.i.d.\) samples from \(D\). Then, \(B\) is \((\ln(244), 2467\delta)\)-differentially private.

3 A Generic Construction Achieving Low Labeled Sample Complexity

We next study the labeled sample complexity of private semi-supervised learners. We begin with a generic algorithm showing that for every concept class \(C\) there exist a pure-private proper-learner with labeled sample complexity (roughly) \(VC(C)\). This algorithm, called GenericLearner, is described in Algorithm 2. The algorithm operates on a labeled database \(S\) and on an unlabeled database \(D\). First, the algorithm produces a sanitization \(\tilde{D}\) of the unlabeled database \(D\) w.r.t. a related concept.
class $C^\oplus$ (to be defined). Afterwards, the algorithm uses $\tilde{D}$ to construct a small set of hypotheses $H$ (we will show that $H$ contains at least one good hypothesis). Finally, the algorithm uses the exponential mechanism to choose a hypothesis out of $H$.

Similar ideas have appeared in [10, 18] in the context of label-private learners, i.e., learners that are only required to protect the privacy of the labels in the sample (and not the privacy of the elements themselves). Like GenericLearner, the learners of [10, 18] construct a small set of hypotheses $H$ that “covers” the hypothesis space and then use the exponential mechanism in order to choose a hypothesis $h \in H$. However, GenericLearner differs in that it protects the privacy of the entire sample (both the labels and the elements themselves).

**Definition 3.1** Given two concepts $h, f \in C$, we denote $(h \oplus f) : X_d \rightarrow \{0, 1\}$, where $(h \oplus f)(x) = 1$ if and only if $h(x) \neq f(x)$. Let $C^\oplus = \{(h \oplus f) : h, f \in C\}$.

We next explain why we use a sanitizer for $C^\oplus$. To preserve the privacy of the examples in $D$, we first create a sanitized version of it that we call $\tilde{D}$. If the entries of $D$ are drawn i.i.d. according to the underlying distribution (and if $D$ is big enough), then a hypothesis with small empirical error on $D$ also has small generalization error (see Theorem 2.12). Our learner classifies the sanitized database $\tilde{D}$ with small error, thus we require that a small error on $\tilde{D}$ implies a small error on $D$. Specifically, if $c$ is the target concept, then we require that for every $f \in C$,

$$\text{error}_D(f, c) = \frac{1}{|D|} |\{x \in D : f(x) \neq c(x)\}|$$

is approximately the same as

$$\text{error}_{\tilde{D}}(f, c) = \frac{1}{|\tilde{D}|} |\{x \in \tilde{D} : f(x) \neq c(x)\}|.$$

Observe that this is exactly what we would get from a sanitization of $D$ w.r.t. the concept class $C^{\oplus c} = \{(f \oplus c) : f \in C\}$. As the target concept $c$ is unknown, we let $\tilde{D}$ be a sanitization of $D$ w.r.t. $C^\oplus$, which contains $C^{\oplus c}$.

To apply the sanitization of Blum et al. [12] to $D$ w.r.t. the class $C^\oplus$, we recall a known result on the VC dimension of $C^{\oplus}$; for sake of completeness we provide a proof of this result.

**Claim 3.2** For any concept class $C$ over $X_d$ it holds that $\text{VC}(C^\oplus) = O(\text{VC}(C))$.

**Proof** Recall that the projection of $C$ on a set of domain points $B = \{b_1, \ldots, b_\ell\} \subseteq X_d$ is $\Pi_c(B) = \{(c(b_1), \ldots, c(b_\ell)) : c \in C\}$. Now note that for every $B = \{b_1, \ldots, b_\ell\} \subseteq X_d$
Therefore, by Sauer’s Lemma 2.8, 
\[
\Pi_{C^\oplus}(B) = \{(h \oplus f)(b_1), \ldots, (h \oplus f)(b_\ell) : h, f \in C\}
\]
\[
= \{(h(b_1), \ldots, h(b_\ell)) \oplus (f(b_1), \ldots, f(b_\ell)) : h, f \in C\}
\]
\[
= \{(h(b_1), \ldots, h(b_\ell)) : h \in C\} \oplus \{(f(b_1), \ldots, f(b_\ell)) : f \in C\}
\]
\[
= \Pi_C(B) \oplus \Pi_C(B).
\]
Hence,
\[
\Pi_C^\oplus(B) \leq \Pi_C(B) \leq \left(\frac{\ell \cdot \vc}{\vc(C)}\right)^{2\vc(C)}
\]
\[
= \Pi_C(B) \leq \left(\frac{\ell \cdot \vc}{\vc(C)}\right)^{2\vc(C)}
\]
For \(\ell \geq 10\vc(C)\) this inequality does not hold, and we can conclude that 
\[
\vc(C^\oplus) \leq 10\vc(C).
\]

---

**Algorithm 2** GenericLearner

**Input:** parameter \(\epsilon\), an unlabeled database \(D = (x_i)_{i=1}^{n-m}\), and a labeled database \(S = (x_i, y_i)_{i=1}^{m}\).

1. Initialize \(H = \emptyset\).
2. Construct an \(\epsilon\)-private sanitization \(\tilde{D}\) of \(D\) w.r.t. \(C^\oplus\), where \(|\tilde{D}| = O\left(\frac{\vc(C^\oplus)}{\alpha^2} \log\left(\frac{1}{\alpha}\right)\right) = O\left(\frac{\vc(C)}{\alpha^2} \log\left(\frac{1}{\alpha}\right)\right)\) (e.g., using Theorem 2.15).
3. Let \(B = \{b_1, \ldots, b_\ell\}\) be the set of all (unlabeled) points appearing at least once in \(\tilde{D}\).
4. For every \((z_1, \ldots, z_\ell) \in \Pi_C(B) = \{(c(b_1), \ldots, c(b_\ell)) : c \in C\}\), add to \(H\) an arbitrary concept \(c \in C\) s.t. \(c(b_i) = z_i\) for every \(1 \leq i \leq \ell\).
5. Choose and return \(h \in H\) using the exponential mechanism with inputs \(\epsilon, H, S\).

---

**Theorem 3.3** Let \(C\) be a concept class over \(X_d\). For every \(\alpha, \beta, \epsilon\), there exists an \((\alpha, \beta, \epsilon, \delta=0, n, m)\)-private semi-supervised proper-learner for \(C\), where

\[
m = O\left(\frac{1}{\alpha \epsilon} \cdot \left(\vc(C) \log\left(\frac{1}{\alpha}\right) + \log\left(\frac{1}{\beta}\right)\right)\right),
\]

and

\[
n = O\left(\frac{d \cdot \vc(C)}{\alpha^3 \epsilon} \log\left(\frac{1}{\alpha}\right) + \frac{1}{\alpha \epsilon} \log\left(\frac{1}{\beta}\right)\right).
\]

The learner might not be efficient.
Proof Note that GenericLearner only accesses $D$ via a sanitizer, and only accesses $S$ using the exponential mechanism (on Step 5). As each of these two mechanisms is $\epsilon$-differentially private, and as $D$ and $S$ are two disjoint samples, GenericLearner is $\epsilon$-differentially private. We, thus, only need to prove that with high probability the learner returns a good hypothesis.

Fix a target concept $c \in C$ and a distribution $\mu$ over $X$, and define the following three “good” events:

$E_1$: There is a hypothesis $h \in H$ such that $\text{error}_S(h) \leq \frac{3\alpha}{5}$.

$E_2$: The exponential mechanism chooses an $h \in H$ such that $\text{error}_S(h) \leq \frac{\alpha}{5} + \min_{f \in H} \{ \text{error}_S(f) \}$.

$E_3$: For every $h \in H$ s.t. $\text{error}_S(h) \leq \frac{4\alpha}{5}$, it holds that $\text{error}_\mu(c, h) \leq \frac{\alpha}{10}$.

Observe that when these three events happen algorithm GenericLearner returns an $\alpha$-good hypothesis: Event $E_1 \cap E_2$ ensures that GenericLearner chooses (using the exponential mechanism) a hypothesis $h \in H$ s.t. $\text{error}_S(h) \leq \frac{4\alpha}{5}$. Event $E_3$ ensures, therefore, that this $h$ satisfies $\text{error}_\mu(c, h) \leq \alpha$. We will now show $E_1 \cap E_2 \cap E_3$ happens with high probability.

First note that, by Theorem 2.11, Event $E_3$ happens with probability at least $(1 - \frac{\beta}{5})$, assuming that $m \geq O\left(\frac{1}{\alpha} \left( VC(C) \log(\frac{1}{\alpha}) + \log(\frac{1}{\beta}) \right) \right)$.

Next, recall that the exponential mechanism ensures that the probability of event $E_2$ is at least $1 - |H| \cdot \exp(-\epsilon am/10)$ (see Proposition 2.13). By Sauer’s lemma

$$\log |H| \leq \log |\Pi_C(B)| \leq VC(C) \cdot \log \left( \frac{e \cdot |B|}{VC(C)} \right) \leq VC(C) \cdot \log \left( \frac{e \cdot \tilde{D}}{VC(C)} \right) \leq O\left( VC(C) \cdot \log \left( \frac{1}{\alpha} \right) \right).$$

Therefore, for

$$m \geq O\left( \frac{1}{\alpha \epsilon} \cdot \left( VC(C) \log(\frac{1}{\alpha}) + \log(\frac{1}{\beta}) \right) \right),$$

Event $E_2$ occurs with probability at least $(1 - \frac{\beta}{5})$.

It remains to show that Event $E_1$ occurs with high probability. By setting the size of the unlabeled database $(n - m)$ to be at least $O\left(\frac{1}{a^2} (VC(C) + \ln(\frac{1}{\beta})) \right)$, Theorem 2.12 ensures that with probability at least $(1 - \frac{\beta}{5})$ the following inequality holds for every $h \in C$. 
\begin{equation}
|\text{error}_D(h,c) - \text{error}_\mu(h,c)| \leq \frac{\alpha}{5}
\end{equation}

Moreover, by setting the size of the unlabeled database \((n - m)\) to be at least

\[
(n - m) \geq O \left( \frac{d \cdot \text{VC}(C^\oplus) \log(\frac{1}{\epsilon})}{\alpha^3 \epsilon} + \frac{\log(\frac{1}{\beta})}{\epsilon \alpha} \right)
\]

Theorem 2.15 ensures that with probability at least \((1 - \frac{\beta}{5})\) for every \((h \oplus f) \in C^\oplus\) (i.e., for every \(h, f \in C\)) it holds that

\[
\frac{\alpha}{5} \geq |Q_{(h \oplus f)}(D) - Q_{(h \oplus f)}(\hat{D})|
\]

\[
= \left| \frac{|\{x \in D : (h \oplus f)(x) = 1\}|}{|D|} - \frac{|\{x \in \hat{D} : (h \oplus f)(x) = 1\}|}{|\hat{D}|} \right|
\]

\[
= \left| \frac{|\{x \in D : h(x) \neq f(x)\}|}{|D|} - \frac{|\{x \in \hat{D} : h(x) \neq f(x)\}|}{|\hat{D}|} \right|
\]

\[
= |\text{error}_D(h,f) - \text{error}_\hat{D}(h,f)|.
\]

In particular, for every \(h \in C\) it holds that

\begin{equation}
|\text{error}_D(h,c) - \text{error}_\hat{D}(h,c)| \leq \frac{\alpha}{5}.
\end{equation}

Now recall that for every \((z_1, \ldots, z_\ell) \in \Pi_c(B)\), algorithm GenericLearner adds to \(H\) a hypothesis \(f\) s.t. \(\forall 1 \leq i \leq \ell, f(b_i) = z_i\). In particular, \(H\) contains a hypothesis \(h^*\) s.t. \(h^*(x) = c(x)\) for every \(x \in B\), that is, a hypothesis \(h^*\) s.t. \(\text{error}_\hat{D}(h^*, c) = 0\). Using Inequalities (2) and (3), with probability at least \((1 - \frac{\beta}{5})\) we have that

\[
\text{error}_\mu(h^*, c) \leq \text{error}_D(h^*, c) + \frac{\alpha}{5} \leq \text{error}_\hat{D}(h^*, c) + \frac{2\alpha}{5} = \frac{2\alpha}{5}.
\]

We continue with the analysis assuming that this is the case. Since \(h^*\) is independent of the labeled database \(S\) (it is constructed only based on the unlabeled database \(D\)), and since \(\text{error}_\mu(h^*, c) \leq \frac{2\alpha}{5}\), by the Chernoff bound, with probability at least \((1 - \frac{\beta}{5})\) we have that \(\text{error}_S(h^*) \leq \frac{3\alpha}{5}\), provided that \(m \geq O\left( \frac{1}{\alpha} \log(\frac{1}{\beta}) \right)\). Event \(E_1\) occurs, therefore, with probability at least \((1 - \frac{3\beta}{5})\).
All in all, setting $n \geq O\left(\frac{d \cdot VC(C) \log \left(\frac{1}{\alpha} \right)}{\alpha^2 \epsilon^2} + \frac{\log \left(\frac{1}{\alpha} \right)}{\epsilon a}\right)$, and

$$m \geq O\left(\frac{1}{\alpha \epsilon} \cdot \left(VC(C) \log \left(\frac{1}{\alpha} \right) + \log \left(\frac{1}{\beta} \right)\right)\right),$$

ensures that the probability of GenericLearner failing to output an $\alpha$-good hypothesis is at most $\beta$. \hfill $\square$

Note that the labeled sample complexity in Theorem 3.3 is optimal (ignoring the dependency in $\alpha, \beta, \epsilon$), as even without the privacy requirement every PAC learner for a class $C$ must have labeled sample complexity $\Omega(VC(C))$. However, the unlabeled sample complexity is as big as the representation length of domain elements, that is, $O(d \cdot VC(C))$. Such a blowup in the unlabeled sample complexity is unavoidable in any generic construction of pure-private learners.\footnote{Feldman and Xiao \cite{27} showed an example of a concept class $C$ over $X_d$ for which every pure-private learner must have unlabeled sample complexity $\Omega(VC(C) \cdot d)$. Hence, as a function of $d$ and $VC(C)$, the unlabeled sample complexity in Theorem 3.3 is the best possible for a generic construction of pure-private learners.}

To show the usefulness of GenericLearner, we consider the concept class $\text{THRESH}_d$ defined as follows. For $0 \leq j \leq 2^d$ let $c_j : X_d \rightarrow \{0, 1\}$ be defined as $c_j(x) = 1$ if $x < j$ and $c_j(x) = 0$ otherwise. Define the concept class $\text{THRESH}_d = \{c_j : 0 \leq j \leq 2^d\}$. Balcan and Feldman \cite{5} showed an efficient pure-private proper-learner for $\text{THRESH}_d$ with labeled sample complexity $O_{\alpha, \beta, \epsilon}(1)$ and unlabeled sample complexity $O_{a, \beta, \epsilon}(d)$. At the cost of preserving approximate-privacy, we can use GenericLearner to get the following result.

**Corollary 3.4** There exists an efficient approximate-private proper-learner for $\text{THRESH}_d$ with labeled sample complexity $O_{\alpha, \beta, \epsilon}(1)$ and unlabeled sample complexity $\tilde{O}_{a, \beta, \epsilon}(1.5 \cdot (\log^* d)^{1.5})$.

**Proof (Sketch)** Consider the concept class $\text{INTERVAL}_d$ defined as follows. For $0 \leq i < j \leq 2^d$ let $c_{i,j} : X_d \rightarrow \{0, 1\}$ be defined as $c_{i,j}(x) = 1$ if $i \leq x < j$ and $c_{i,j}(x) = 0$ otherwise. Define the concept class $\text{INTERVAL}_d = \{c_{i,j} : 0 \leq i < j \leq 2^d\}$.

Now let $c_i, c_j \in \text{THRESH}_d$ for some $i < j$, and observe that $(c_i \oplus c_j) = 1$ if and only if $i \leq x < j$. We therefore have that $\text{THRESH}_d^{\oplus} = \text{INTERVAL}_d$. So, in order to apply GenericLearner to learn $\text{THRESH}_d$, we need to apply a sanitizer for $\text{INTERVAL}_d$. By the results of \cite{10, 16, 31}, there exists an efficient approximate-private sanitizer for $\text{INTERVAL}_d$ that requires a database of size $\tilde{O}_{a, \beta, \epsilon, \delta}(\log^* d)^{1.5}$.\footnote{These works present sanitizers for $\text{THRESH}_d$, but any sanitizer for $\text{THRESH}_d$ can easily be transformed into a sanitizer for $\text{INTERVAL}_d$.} The corollary follows by using this sanitizer in Step 2 of Algorithm GenericLearner (instead of the sanitizer of \cite{12}). \hfill $\square$
4 Algorithm Relabel

The generic construction from the previous section results in a learner with labeled sample complexity proportional to the VC dimension, and with unlabeled sample complexity proportional to the sanitization complexity of the target class $C$. In the following sections we present a different construction that also results in a learner with labeled sample complexity proportional to the VC dimension, but with unlabeled sample complexity proportional to the private learning complexity of the target class $C$. The private learning complexity is always at most the sanitization complexity [9, 29], and is significantly smaller for many cases of interest [17].

In more detail, in the following sections we present a generic transformation of a private learning algorithm for a class $C$ into a private learner with reduced labeled sample complexity (roughly $\text{VC}(C)$), while maintaining the unlabeled sample complexity of the base learner. This transformation is for a proper learner and could be applied to a learner that preserves pure or approximate differential privacy. Using results of [3] it can also be applied to improper learners. The main ingredient of the transformation is algorithm Relabel (Algorithm 3), which is the focus of this section. We will later show (in Sect. 6) that algorithm Relabel can be used to reduce the labeled sample complexity of private learners. In addition, in Sect. 5 we present an additional application of algorithm Relabel, namely, that it can be used to transform a private learning algorithm that works in the realizable case into a private agnostic-learning algorithm with roughly the same sample complexity.

We now present algorithm Relabel. Given a partially labeled sample $B$ of size $n$, algorithm Relabel chooses a small subset $H$ of $C$ that strongly depends on the points in $B$ so outputting a hypothesis $h \in H$ may breach privacy. Nevertheless, algorithm Relabel does choose a good hypothesis $h \in H$ (using the exponential mechanism) and uses it to relabel the labeled examples of the sample $B$ and label part of the unlabeled examples in $B$. In Lemma 4.1, we analyze the privacy guarantees of algorithm Relabel.

Algorithm 3 Relabel

**Input:** A partially labeled database $B = SoTD \in (X \times \{0, 1, \perp\})^*$.

We assume that the first portion of $B$ (denoted as $S$) contains labeled examples. Our goal is to output a similar database where both $S$ and $T$ are labeled.

1. Initialize $H = \emptyset$.
2. Let $P = \{p_1, \ldots, p_\ell\}$ be the set of all points $p \in X$ appearing at least once in $SoT$.
3. For every $(x_1, \ldots, x_\ell) \in \Pi_C(P) = \{(c(p_1), \ldots, c(p_\ell)) : c \in C\}$, add to $H$ an arbitrary concept $c \in C$ s.t. $c(p_i) = x_i$ for every $1 \leq i \leq \ell$.
4. Choose $h \in H$ using the exponential mechanism with privacy parameter $\epsilon = 1$, solution set $H$, and the database $S$.
5. Relabel $SoT$ using $h$, and denote this relabeled database as $(SoT)^h$ (where $y_i = \perp$ for $x_i \in T$), that is, if $SoT = (x_i, y_i)_{i=1}^\ell$ then $(SoT)^h = (x_i, y_i')_{i=1}^\ell$ where $y_i' = h(x_i)$.
6. Output $(SoT)^h \circ D$.

**Lemma 4.1** Let $A$ be an $(\epsilon, \delta)$-differentially private algorithm operating on partially labeled databases. Construct an algorithm $B$ that on input a
database \(S \circ T \circ D \in (X \times \{0, 1, \bot\})^*\) applies \(A\) on the outcome of algorithm \(\text{Relabel}(S \circ T \circ D)\). Then, \(B\) is \((\epsilon + 3, 4e\delta)\)-differentially private.

**Proof** Consider the executions of \(B\) on two neighboring inputs \(S_1 \circ T_1 \circ D_1\) and \(S_2 \circ T_2 \circ D_2\). If these two neighboring inputs differ (only) on the last portion \(D\) then the executions of \(\text{Relabel}\) on these neighboring inputs are identical, and hence Inequality (1) (approximate differential privacy) follows from the privacy of \(A\). We, therefore, assume that \(D_1 = D_2 = D\) (and that \(S_1 \circ T_1, S_2 \circ T_2\) differ in at most one entry).

Denote by \(H_1, P_1\) and by \(H_2, P_2\) the elements \(H, P\) as they are in the executions of algorithm \(\text{Relabel}\) on \(S_1 \circ T_1 \circ D\) and on \(S_2 \circ T_2 \circ D\). The main difficulty in proving differential privacy is that \(H_1\) and \(H_2\) can significantly differ. We show, however, that the distribution on relabeled databases \((S \circ T)^h\) generated in Step 5 of the two executions are similar in the sense that for each relabeled database in one of the distributions there exist one or two databases in the other s.t. (1) all these databases have, roughly, the same probability, and (2) they differ on at most one entry. Thus, executing the differentially private algorithm \(A\) on \((S \circ T)^h \circ D\) preserves differential privacy. We now make this argument formal.

Note that \(|P_1 \setminus P_2| \in \{0, 1\}\), and let \(p_1\) be the element in \(P_1 \setminus P_2\) if such an element exists. If this is the case, then \(p_1\) appears exactly once in \(S_1 \circ T_1\). Similarly, let \(p_2\) be the element in \(P_2 \setminus P_1\) if such an element exists. Let \(K = P_1 \cap P_2\), hence \(P_i = K \cup \{p_i\}\). Therefore, \(|\Pi_C(K)| \leq |\Pi_C(P_i)| \leq 2|\Pi_C(K)|\). Thus, \(|H_1| \leq 2|H_2|\) and similarly \(|H_2| \leq 2|H_1|\).

More specifically, for every \(z \in \Pi_C(K)\) there are either one or two (but not more) hypotheses in \(H_1\) that agree with \(z\) on \(K\). We denote these one or two hypotheses by \(h_{i,z}\) and \(h'_{i,z}\), which may be identical if only one unique hypothesis exists. Similarly, we denote \(h_{2,z}\) and \(h'_{2,z}\) as the hypotheses corresponding to \(H_2\). Recall that algorithm \(\text{Relabel}\) in Step 4 uses the exponential mechanism with quality function \(q(S, h) = |\{i : h(x_i) = y_i\}|\). For every \(z \in \Pi_C(K)\) we have that \(|q(S_1, h_{i,z}) - q(S_2, h'_{i,z})| \leq 1\) because if \(h_{i,z} = h'_{i,z}\) then the difference is clearly zero and otherwise they differ only on \(p_i\), which appears at most once in \(S_j\). Moreover, for every \(z \in \Pi_C(K)\) we have that \(|q(S_1, h_{1,z}) - q(S_2, h_{2,z})| \leq 1\) because \(h_{1,z}\) and \(h_{2,z}\) disagree on at most two points \(p_1, p_2\) such that at most one of them appears in \(S_1\) and at most one of them appears in \(S_2\). The same is true for every pair in \(\{h_{1,z}, h'_{1,z}\} \times \{h_{2,z}, h'_{2,z}\}\).

Let \(w_{i,z}\) be the probability that the exponential mechanism chooses \(h_{i,z}\) or \(h'_{i,z}\) in Step 4 of the execution on \(S_i \circ T \circ D\). We get that for every \(z \in \Pi_C(K)\),
\[ w_{1,z} \leq \frac{\exp\left(\frac{1}{2} \cdot q(S_1, h_{1,z})\right) + \exp\left(\frac{1}{2} \cdot q(S_1, h'_{1,z})\right)}{\sum_{f \in H_1} \exp\left(\frac{1}{2} \cdot q(S_1, f)\right)} \]
\[ \leq \frac{\exp\left(\frac{1}{2} \cdot q(S_1, h_{1,z})\right) + \exp\left(\frac{1}{2} \cdot q(S_1, h'_{1,z})\right)}{\sum_{r \in H_c(K)} \exp\left(\frac{1}{2} \cdot q(S_1, h_{1,r})\right)} \]
\[ \leq \frac{\exp\left(\frac{1}{2} \cdot [q(S_2, h_{2,x}) + 1]\right) + \exp\left(\frac{1}{2} \cdot [q(S_2, h'_{2,x}) + 1]\right)}{\frac{1}{2} \sum_{r \in H_c(K)} \left(\exp\left(q(S_2, h_{2,x})\right) + \exp\left(\frac{q(S_2, h'_{2,x}) - 1}{2}\right)\right)} \]
\[ \leq 2e \cdot \frac{\exp\left(\frac{1}{2} \cdot [q(S_2, h_{2,x})]\right) + \exp\left(\frac{1}{2} \cdot [q(S_2, h'_{2,x})]\right)}{\sum_{f \in H_2} \exp\left(\frac{1}{2} \cdot q(S_2, f)\right)} \]
\[ \leq 4e \cdot w_{2,x}. \]

We can now conclude the proof by noting that for every \( z \in H_c(K) \) the databases \((S_1 \circ T_1)^{h_{1,z}}\) and \((S_2 \circ T_2)^{h_{2,z}}\) are neighboring, and, therefore, \((S_1 \circ T_1)^{h_{1,z}} \circ D\) and \((S_2 \circ T_2)^{h_{2,z}} \circ D\) are neighboring. For every \( z \in H_c(K) \), let \( h_{i,z} \) denote the event that the exponential mechanism chooses \( h_{i,z} \) or \( h'_{i,z} \) in Step 4 of the execution on \( S_j \circ T_j \circ D \). By the privacy properties of algorithm \( A \) we have that for any set \( F \) of possible outputs of algorithm \( B \)

\[ \Pr[B(S_1 \circ T_1 \circ D) \in F] = \sum_{z \in H_c(K)} w_{1,z} \cdot \Pr[A((S_1 \circ T_1)^h \circ D) \in F | h_{1,z}] \]
\[ \leq \sum_{z \in H_c(K)} 4e w_{2,z} \left( e^e \Pr[A((S_2 \circ T_2)^h \circ D) \in F | h_{2,z}] + \delta \right) \]
\[ \leq e^{e+3} \cdot \Pr[B(S_2 \circ T_2 \circ D) \in F] + 4e \delta. \]

\[ \square \]

We next show that algorithm Relabel can be used to label (and relabel) a big part of the examples in \( B \) without introducing much error. Consider an execution of Relabel on a database \( S \circ T \circ D \), and assume that the examples in \( S \) are labeled by some target concept \( c \in C \). Recall that for every possible labeling \( z \) of the elements in \( S \) and in \( T \), algorithm Relabel adds to \( H \) a hypothesis from \( C \) that agrees with \( z \). In particular, \( H \) contains a hypothesis that agrees with the target concept \( c \) on \( S \) (and on \( T \)). That is, \( \exists f \in H \) s.t. \( \text{error}_S(f) = 0 \). Hence, the exponential mechanism (on Step 4) chooses (w.h.p.) a hypothesis \( h \in H \) s.t. \( \text{error}_S(h) \) is small, provided that \(|S|\) is roughly \( \log |H| \), which is roughly \( \text{VC}(C) \cdot \log(|S| + |T|) \) by Sauer’s lemma. So, algorithm Relabel takes an input database \( B = S \circ T \circ D \in (X \times \{0, 1, \bot\})^n \) where only a small portion of it is labeled (only \( S \)), and returns a similar database \( B' = S' \circ T' \circ D \in (X \times \{0, 1, \bot\})^n \) in which both \( S' \) and \( T' \) are labeled (by a hypothesis with small error w.r.t. the original labels in \( S \)). We will later use this algorithm iteratively in order to “stretch” the labeled portion of our input database without incurring too many errors, where in
every step the labeled portion of the database grows significantly. Before that, we need to understand how big can $T$ be compared to $S$, because this will determinate the “rate” at which we can “stretch” the labels (i.e., determine the number of iterations until the entire database is labeled). We show that $|T|$ can be exponential in $|S|$, provided that $S$ is big enough (roughly bigger than $\text{VC}(C)$).

**Claim 4.2** Fix $\alpha$ and $\beta$, and let $S \circ T \circ D$ be such that $S$ is a labeled database and such that

$$|T| \leq \frac{\beta}{e} \text{VC}(C) \exp\left(\frac{\alpha|S|}{2\text{VC}(C)}\right) - |S|.$$

Consider the execution of algorithm Relabel on $S \circ T \circ D$, and let $h$ denote the hypothesis chosen on Step 4. With probability at least $(1 - \beta)$ we have that $\text{error}_S(h) \leq \alpha + \min_{f \in C} \{\text{error}_S(f)\}$. In particular, if $S$ is consistent with $C$, then with probability at least $(1 - \beta)$ we have that $\text{error}_S(h) \leq \alpha$.

**Proof** Note that by Sauer’s lemma,

$$|H| = |\Pi_C(P)| \leq \left(\frac{e|P|}{\text{VC}(C)}\right)^{\text{VC}(C)}$$

$$\leq \left(\frac{e(|T| + |S|)}{\text{VC}(C)}\right)^{\text{VC}(C)}$$

$$\leq \left(\beta \exp\left(\frac{\alpha|S|}{2\text{VC}(C)}\right)\right)^{\text{VC}(C)}$$

$$\leq \beta \exp\left(\frac{\alpha|S|}{2}\right).$$

For every $(z_1, \ldots, z_\ell) \in \Pi_C(P)$, algorithm Relabel adds to $H$ a hypothesis $f$ s.t. $\forall 1 \leq j \leq \ell$, $f(p_j) = z_j$. In particular, $H$ contains a hypothesis $f^*$ s.t. $\text{error}_S(f^*) = \min_{f \in C} \{\text{error}_S(f)\}$. Hence, Proposition 2.13 (properties of the exponential mechanism) ensures that the probability of the exponential mechanism choosing an $h$ s.t. $\text{error}_S(h) > \alpha + \min_{f \in C} \{\text{error}_S(f)\}$ is at most

$$|H| \cdot \exp\left(-\frac{\alpha|S|}{2}\right) \leq \beta.$$

We also consider the special case of applying algorithm Relabel on a database $S \circ T \circ D$ such that $|T| = 0$. That is, when algorithm Relabel is used only in order to relabel the database $S$, and not in order to “stretch” the labels onto $T$. A similar analysis shows the following.

**Claim 4.3** Fix $\alpha$ and $\beta$, and let $S \circ T \circ D$ be such that $|T| = 0$ and such that
Consider the execution of algorithm Relabel on $S \circ T \circ D$, and let $h$ denote the hypothesis chosen on Step 4. With probability at least $(1 - \frac{1}{u_1D})$ we have that $\text{error}_S(h) \leq \frac{1}{u_1D} + \min_{f \in C} \{\text{error}_S(f)\}$.

**Proof (Sketch)** The only modification from the analysis of Claim 4.2 is in the bound on $|H|$. Specifically, now we have that $|H| \leq \left(\frac{e|S|}{\text{VC}(C)}\right)^{\text{VC}(C)} \cdot \exp\left(-\frac{a|S|}{2}\right)$, which is at most $\beta$ for

$$|S| \geq O\left(\frac{1}{\alpha} \left(\text{VC}(C) \cdot \log\left(\frac{1}{\alpha}\right) + \log\left(\frac{1}{\beta}\right)\right)\right).$$



5 A Generic Transformation from Private Learning in the Realizable Case to Private Agnostic Learning

In this section we show that algorithm Relabel can be useful also in the standard setting of private learning (i.e., not semi-supervised), where the learning algorithm gets one labeled sample (and there is no unlabeled sample). Specifically, we show that algorithm Relabel can be used to transform a private learning algorithm $\mathcal{A}$ that works in the realizable case into a private agnostic-learner $\mathcal{A}'$ with roughly the same sample complexity.

Our transformation works only in the case that the base learner $\mathcal{A}$ is a proper learner. Following our work, Alon et al. [3] showed that, by introducing small modifications to algorithm Relabel, the same transformation could be applied also when the base learner $\mathcal{A}$ is improper.

**Theorem 5.1** For every $\alpha, \beta, \delta, n$, if there exists a $(1, \delta)$-differentially private $(\alpha, \beta)$-PAC proper-learner $\mathcal{A}$ for a concept class $C$ with sample complexity $n$, then there exists an $O(1), O(\delta))$-differentially private $(O(\alpha), O(\beta))$-agnostic proper-learner $\mathcal{B}$ using $C$ with sample complexity

$$m = \min \left\{ n, \ O\left(\frac{1}{\alpha^2} \left(\text{VC}(C) + \ln\left(\frac{1}{\beta}\right)\right)\right) \right\}.$$
In particular, if $A$ is 1-differentially private, then $B$ is $O(1)$-differentially private (i.e., if $A$ guarantees pure differential privacy, then $B$ guarantees pure differential privacy).

**Algorithm 4 PrivateAgnostic**

**Setting:** Algorithm $A$ with (labeled and unlabeled) sample complexity $n$.

**Input:** A labeled database $S \in (X \times \{0, 1\})^m$.

1. Let $S' \leftarrow \text{Relabel}(S)$.
   
   % Formally, denote $T = D = \emptyset$ and let $S' \leftarrow \text{Relabel}(S)$. That is, we use algorithm Relabel to order to relabel $S$ using a hypothesis from $C$.

2. Let $\hat{S}$ denote the outcome of $n$ i.i.d. samples from $S'$.

3. Execute $A$ on $\hat{S}$.

**Proof** The proof is via the construction of algorithm PrivateAgnostic. The privacy properties of PrivateAgnostic follow from the privacy properties of algorithm Relabel (Lemma 4.1) and from Claim 2.18 (i.i.d. sampling). We now proceed with the utility analysis.

Fix a distribution $\mu$ on $X \times \{0, 1\}$, and let $S$ be a database containing $m$ i.i.d. samples from $\mu$. Consider the execution of PrivateAgnostic on $S$. Let $h \in C$ denote the hypothesis used by algorithm Relabel in order to relabel $S$ in Step 1 of algorithm PrivateAgnostic (the relabeled database is called $S'$). By Claim 4.3, with probability at least $(1 - \beta)$ it holds that $\text{error}_S(h) \leq \alpha + \min_{f \in C} \{\text{error}_S(f)\}$. Let $h_{\text{fin}}$ denote the hypothesis returned by $A$. Since $A$ is executed on a random sample (containing i.i.d. uniformly distributed examples from $S$) in which every element is labeled by $h$, by the utility properties of $A$, with probability at least $(1 - \beta)$ we have that $\text{error}_S(h_{\text{fin}}, h) \leq \alpha$. By the triangle inequality we therefore get that with probability at least $(1 - 2\beta)$

$$\text{error}_S(h_{\text{fin}}) \leq 2\alpha + \min_{f \in C} \{\text{error}_S(f)\}.$$ 

Now, by Theorem 2.12 (VC generalization bound), with probability at least $(1 - \beta)$, for every $f \in C$ we have that $|\text{error}_S(f) - \text{error}_\mu(f)| \leq \alpha$, provided that $m \geq O\left(\frac{1}{\alpha^2} \left(\text{VC}(C) + \ln\left(\frac{1}{\beta}\right)\right)\right)$. Hence, with probability at least $(1 - 3\beta)$

$$\text{error}_S(h_{\text{fin}}) \leq 2\alpha + \min_{f \in C} \{\text{error}_S(f)\} \leq 3\alpha + \min_{f \in C} \{\text{error}_\mu(f)\}.$$ 

Therefore (using Theorem 2.12 again) we have that with probability at least $(1 - 4\beta)$

$$\text{error}_\mu(h_{\text{fin}}) \leq \alpha + \text{error}_S(h_{\text{fin}}) \leq 4\alpha + \min_{f \in C} \{\text{error}_\mu(f)\}. \quad (4)$$

The point at which we used the assumption that $A$ is a proper learner is in Inequality (4). Specifically, if $A$ is an improper learner, then $h_{\text{fin}}$ is not necessarily in
C, and we cannot use Theorem 2.12 (the VC generalization bound) to argue that error\(_n(h_{\text{fin}})\) is small, because \(h_{\text{fin}}\) might be coming from a hypothesis class with a much larger VC dimension.

Following our work, Alon et al. [3] extended our transformation from realizable to agnostic learning also to the case where the base learner is improper. Instead of using VC arguments, Alon et al. [3] used the fact that differential privacy guarantees generalization [7, 21, 30] in order to argue about the error of the returned hypothesis. We refer the reader to [3] for more details.

6 Algorithm LabelBoost

In this section we show that algorithm Relabel can be used to reduce the labeled sample complexity of private learners. In more detail, we describe algorithm LabelBoost that iteratively applies algorithm Relabel in order to enlarge the labeled portion of the database. Every such application deteriorates the privacy parameters, and hence, every iteration includes a sub-sampling step, which compensates for those privacy losses. In a nutshell, the learner LabelBoost could be described as follows. It starts by training on the given labeled data. In each step, a part of the unlabeled points is labeled using the current hypothesis (previously labeled points are also relabeled); then the learner retrains using its own predictions as a (larger) labeled sample. Variants of this idea (known as self-training) have appeared in the literature for non-private learners (e.g., [2, 28, 39]). As we will see, in the context of private learners, this technique provably reduces the labeled sample complexity (while maintaining utility).

Consider algorithm LabelBoost (described in Algorithm 5). We begin with the privacy analysis.

**Lemma 6.1** If \( A \) is \((1, \delta)\)-differentially private, then algorithm LabelBoost is \((1, 41\delta)\)-differentially private.

**Proof** We think of the input of LabelBoost as one database

\[
B \in (X \times \{0, 1, \perp\})^{90000n + m}.
\]

Note that the number of iterations performed on neighboring databases is identical (determined by the parameters \(\alpha, \beta, n, m\)), and denote this number as \(N\). Throughout the execution, random elements from the input database are deleted (on Step 2c). Note however, that the size of the database at any moment throughout the execution does not depend on the database content (determined by the parameters \(\alpha, \beta, n, m\)). We denote the size of the database at the beginning of the \(i\)th iteration as \(n(i)\), e.g., \(n(1) = 90000n + m\).
Let $L_t$ denote an algorithm similar to algorithm LabelBoost, except that only the last $t$ iterations are performed. The input of $L_t$ consists of a database $D \in X \times \{0, 1\}$ and a labeled database $S \in (X \times \{0, 1\})^m$.

1. Set $i = 1$.
2. While $|S| < 300n$:
   a. Denote $\alpha_i = \frac{\alpha}{10^{2t-1}}$, and $\beta_i = \frac{\beta}{2^{10^2}}$.
   b. Set $\nu = \min\left\{30000n, \beta_i \text{VC}(C) e^{-\frac{\alpha_i |S|}{20 \text{VC}(C)}} - |S|\right\}$. Let $T$ be the first $\nu$ elements of $D$, and remove $T$ from $D$. Fail if there are not enough elements in $D$.
   c. Delete (permanently) $\frac{99}{100} |T|$ random entries from $T$, and $\frac{99}{100} |S|$ random entries from $S$.
   d. $S' \leftarrow \text{Relabel}(S \cup T)$.
   e. Add every element of $T$ to $S$.
   f. Set $i = i + 1$.
3. Delete $\frac{99}{300} |S|$ random entries from $S$.
4. Let $S'$ denote the outcome of $|S|$ i.i.d. samples from $S$.
5. Execute $A$ on $S'$.

Let $L_t$ denote an algorithm similar to algorithm LabelBoost, except that only the last $t$ iterations are performed. The input of $L_t$ consists of a database in $(X \times \{0, 1, \bot\})^n$ and a labeled database $S \in (X \times \{0, 1\})^m$.

We next show (by induction on $t$) that $L_t$ is $(1, 41\delta)$-differentially private. To this end, note that an execution of $L_0$ consists of sub-sampling (as in Claim 2.16, i.i.d. sampling (as in Claim 2.18), and applying the $(1, \delta)$-private algorithm $A$. By Claim 2.18, Steps 4–5 preserve $(\ln(244), 2476)$-differential privacy, and, hence, by Claim 2.16, we have that $L_0$ is $(1, 41\delta)$-differentially private.

Assume that $L_{t-1}$ is $(1, 41\delta)$-differentially private, and observe that $L_t$ could be restated as an algorithm that first performs one iteration of algorithm LabelBoost and then applies $L_{t-1}$ on the databases $D, S$ as they are at the end of that iteration. Now fix two neighboring databases $B_1, B_2$ and consider the execution of $L_t$ on $B_1$ and on $B_2$.

Let $S_1^b, T_1^b, D_1^b$ and $S_2^b, T_2^b, D_2^b$ be the databases $S, T, D$ after Step 2b of the first iteration of $L_t$ on $B_1$ and on $B_2$ (note that $B_1 = S_1^b \circ T_1^b \circ D_1^b$ and $B_2 = S_2^b \circ T_2^b \circ D_2^b$). If $B_1$ and $B_2$ differ (only) on their last portion, denoted as $D_1^b, D_2^b$, then the execution of $L_t$ on these neighboring inputs differs only in the execution of $L_{t-1}$, and hence Inequality (1) (approximate differential privacy) follows from the privacy of $L_{t-1}$. We, therefore, assume that $D_1^b = D_2^b$ (and that $S_1^b \circ T_1^b$ and $S_2^b \circ T_2^b$ differ in at most one entry).
Now, note that an execution of $\mathcal{L}_i$ consists of sub-sampling (as in Claim 2.16), applying algorithm Relabel on the inputs, and executing the $(1, 41\delta)$-private algorithm $\mathcal{L}_{i-1}$. By Lemma 4.1 (privacy properties of Relabel), the application of $\mathcal{L}_{i-1}$ on top of Relabel preserves $(4, 446\delta)$-differential privacy, and, hence, by Claim 2.16 (sub-sampling), we have that $\mathcal{L}_i$ is $(1, 41\delta)$-differentially private.

Before proceeding with the utility analysis, we introduce to following notations.

Notation Consider the $i$th iteration of LabelBoost. We let $S^b_i, T^b_i$ and $S^c_i, T^c_i$ denote the elements $S, T$ as they are after Steps 2b and 2c, and let $h_i$ denote the hypothesis $h$ chosen in the execution of Relabel in the $i$th iteration.

We first make the following observation, which follows from Claim 4.2.

Observation 6.2 In every iteration $i$, with probability at least $(1 - \beta_i)$ we have that $\text{error}_{S^b_i}(h_i) \leq \alpha_i$.

Recall that algorithm LabelBoost might fail on Step 2b. We now show that if $|D|$ is large enough, then this does not happen.

Claim 6.3 Assume that algorithm LabelBoost is executed with a base learner with sample complexity $n$, and on databases $D, S$. If $|D| \geq 90000n$, then algorithm LabelBoost never fails on Step 2b.

Proof Denote the number of iterations throughout the execution as $N$. We need to show that $\sum_{i=1}^{N} T^b_i \leq 90000n$. Clearly, $|T^b_N|, |T^b_{N-1}| \leq 30000n$. Moreover, for every $1 < i < N$ we have that $|T^b_i| \geq 2|T^b_{i-1}|$. Hence,

$$\sum_{i=1}^{N} T^b_i \leq 30000n + 30000n \sum_{i=0}^{\infty} \frac{1}{2^i} = 90000n.$$  

Recall that the number of labeled examples $|S_i|$ grows significantly with each iteration (roughly $|S_i| \geq 2^{|S_{i-1}|}$), and that the learning parameters $\alpha_i, \beta_i$ are decreased with each iteration ($\alpha_i = \frac{a}{10^2i}$, and $\beta_i = \frac{6}{42^i}$). The following claim shows that $|S_i|$ is always big enough w.r.t. $\frac{1}{\alpha_i}$ and $\frac{1}{\beta_i}$ (this follows since $|S_i|$ grows much faster than $\frac{1}{\alpha_i}, \frac{1}{\beta_i}$).

Claim 6.4 Fix $\alpha, \beta$. Assume that algorithm LabelBoost is executed on a base learner with sample complexity $n$, and on databases $D, S$, where $|D| \geq 90000n$ and $|S| \geq \frac{960000}{\alpha} \text{VC}(C) \log(\frac{2440}{\alpha\beta})$. In every iteration $i$

$$|S^b_i| \geq \frac{4800}{\alpha_i} \text{VC}(C) \log(\frac{14}{\alpha_i \beta_i}).$$  

Proof The proof is by induction on $i$. Note that the base case (for $i = 1$) trivially holds, and assume that the claim holds for $i - 1$. We have that
\[ |S_i^b| = |S_{i-1}^c| + |T_{i-1}^c| = \frac{1}{100} (|S_{i-1}^b| + |T_{i-1}^b|) \]
\[ = \frac{1}{100} \beta_{i-1} \text{VC}(C) \exp \left( \frac{\alpha_{i-1} |S_{i-1}^b|}{200 \text{VC}(C)} \right) \]
\[ \geq \frac{1}{100} \beta_{i-1} \text{VC}(C) \exp \left( 24 \log \left( \frac{14}{\alpha_{i-1} \beta_{i-1}} \right) \right) \]
\[ \geq \frac{1}{100} \beta_{i-1} \text{VC}(C) \cdot \left( \frac{14}{\alpha_{i-1} \beta_{i-1}} \right)^{24} \]
\[ \geq 4800 \alpha_i \text{VC}(C) \log \left( \frac{14}{\alpha_i \beta_i} \right). \]

\[ \square \]

**Remark 6.5** The above analysis could easily be strengthened to show that \(|S_i^b|\) grows as an exponentiation tower in \(i\). This implies that there are at most \(O(\log^* n)\) iterations throughout the execution of LabelBoost on a base learner \(A\) with sample complexity \(n\).

Recall that in each iteration we use algorithm Relabel to identify a hypothesis \(h_i\) and use it to relabel part of the data. In the next claim we show that (w.h.p.) all of these hypotheses have small generalization error.

**Claim 6.6** Assume that algorithm LabelBoost is executed on databases \(D, S\) containing i.i.d. samples from a fixed distribution \(\mu\), where the examples in \(S\) are labeled by some fixed target concept \(c \in C\), and \(|S_i| \geq 36000 \text{VC}(C) \log \left( \frac{2540}{\alpha_i \beta_i} \right)\). For every \(i\), the probability that \(\text{error}_\mu(c, h_i) > 10 \sum_{j=1}^i \alpha_j\) is at most \(2 \sum_{j=1}^i \beta_j\).

**Proof** The proof is by induction on \(i\). Note that for \(i = 1\) we have that \(S_i^c\) contains \(\frac{48}{\alpha_i} \text{VC}(C) \log \left( \frac{14}{\alpha_i \beta_i} \right)\) i.i.d. samples from \(\mu\) that are labeled by the target concept \(c\). By Observation 6.2, with probability at least \((1 - \beta_1)\), we have that \(\text{error}_{S_i^c}(h_1) \leq \alpha_1\). In that case, Theorem 2.11 (the VC dimension bound) states that with probability at least \((1 - \beta_1)\) it holds that \(\text{error}_\mu(c, h_1) \leq 10 \alpha_1\).

Now assume that the claim holds for \((i - 1)\), and consider the \(i\)th iteration. Note that \(S_i^c\) contains i.i.d. samples from \(\mu\) that are labeled by \(h_{i-1}\). Moreover, by Claim 6.4, we have that \(|S_i^c| = \frac{1}{100} |S_{i-1}^b| \geq \frac{48}{\alpha_i} \text{VC}(C) \log \left( \frac{14}{\alpha_i \beta_i} \right)\). By Observation 6.2, with probability at least \((1 - \beta_i)\), we have that \(\text{error}_{S_i^c}(h_i) \leq \alpha_i\). If that is the case, Theorem 2.11 states that with probability at least \((1 - \beta_i)\) it holds that \(\text{error}_\mu(h_{i-1}, h_i) \leq 10 \alpha_i\). So, with probability at least \((1 - 2 \beta_i)\) we have that \(\text{error}_\mu(h_{i-1}, h_i) \leq 10 \alpha_i\). Using the inductive assumption, the probability that \(\text{error}_\mu(c, h_i) \leq \text{error}_\mu(c, h_{i-1}) + \text{error}_\mu(h_{i-1}, h_i) \leq 10 \sum_{j=1}^i \alpha_j\) is at least \((1 - 2 \sum_{j=1}^i \beta_j)\). \(\square\)
We are now ready to complete the utility analysis of algorithm LabelBoost. In the next lemma we analyze the utility of LabelBoost under the assumption that the base learner $A$ is a proper learner. The extension to the case where $A$ is an improper learner is given at the end of this section.

**Lemma 6.7** Fix $\alpha, \beta$. Applying algorithm LabelBoost with a proper $(\alpha, \beta)$-PAC learner for a class $C$ with sample complexity $n$ as the base learner $A$ results in a proper $(11\alpha, 2\beta, O(n + m), m)$-SSL for $C$, where

$$m = O\left(\frac{1}{\alpha} \text{VC}(C) \log\left(\frac{1}{\alpha \beta}\right)\right).$$

**Proof** Assume that algorithm LabelBoost is executed on databases $D, S$ containing i.i.d. samples from a fixed distribution $\mu$, where $|D| \geq 90000n$ and $|S| \geq \frac{96000}{\alpha} \text{VC}(C) \log\left(\frac{2240}{\alpha \beta}\right)$. Moreover, assume that the examples in $S$ are labeled by some fixed target concept $c \in C$.

Consider the last iteration of Algorithm LabelBoost (say $i = N$) on these inputs. The intuition is that after the last iteration, when reaching Step 4, the database $S$ is big enough s.t. $A$ returns (w.h.p.) a hypothesis with small error on $S$. This hypothesis also has small generalization error as $S$ is labeled by $h_N$ which is close to the target concept (by Claim 6.6).

Formally, let $S^3$ denote the database $S$ as it is after Step 3 of the execution, and let $h_{\text{fin}}$ denote the hypothesis returned by the base learner $A$ on Step 5. By the while condition on Step 2, we have that $|S^3| \geq n$. Hence, by the utility guarantees of the base learner $A$, with probability at least $(1 - \beta)$ we have that $\text{error}_S(h_{\text{fin}}) \leq \alpha$. As $|S^3| \geq \frac{1}{300} |S| \geq \frac{640}{\alpha} \text{VC}(C) \log\left(\frac{480}{\alpha \beta}\right)$, and as $S^3$ contains i.i.d. samples from $\mu$ labeled by $h_N$, Theorem 2.11 states that with probability at least $(1 - \frac{\beta}{2})$ it holds that

$$\text{error}_\mu(h_{\text{fin}}, h_N) \leq 10\alpha. \tag{5}$$

By Claim 6.6, with probability at least $(1 - 2 \sum_{i=1}^{N} \beta_i) \geq (1 - \frac{\beta}{2})$ it holds that $\text{error}_\mu(c, h_{\text{fin}}) \leq 10 \sum_{i=1}^{N} a_i \leq \alpha$. All in all (using the triangle inequality), with probability at least $(1 - 2\beta)$ we get that $\text{error}_\mu(c, h_{\text{fin}}) \leq 11\alpha$. \hfill $\square$

**Remark 6.8** The point at which we used the assumption that $A$ is a proper learner is in Inequality (5). Specifically, if $A$ is an improper learner, then $h_{\text{fin}}$ is not necessarily in $C$, and we cannot use Theorem 2.11 (the VC generalization bound) to argue that $\text{error}_\mu(h_{\text{fin}}, h_N)$ is small.

Combining Lemmas 6.1 and 6.7 we get the following theorem.

**Theorem 6.9** Fix $\alpha, \beta, \delta$. Applying LabelBoost with a $(1, \delta)$-differentially private proper $(\alpha, \beta)$-PAC learner for a class $C$ with sample complexity $n$ as the base learner $A$ results in a proper $(11\alpha, 2\beta, \epsilon = 1, 41\delta, O(n + m), m)$-PSSL for $C$, where

$$m = O\left(\frac{1}{\alpha} \text{VC}(C) \log\left(\frac{1}{\alpha \beta}\right)\right).$$

Springer
Using Claim 2.16 to boost the privacy guarantees of the learner resulting from Theorem 6.9, proves Theorem 6.10:

**Theorem 6.10** For every $\alpha, \beta, \epsilon, \delta, n$, if there exists a $(1, \delta)$-differentially private proper $(\alpha, \beta)$-PAC learner for a concept class $C$ with sample complexity $n$, then there exists a proper $(O(\alpha), O(\beta), \epsilon, \delta, O(\frac{2}{\epsilon} + m), m)$-PSSL for $C$, where

$$
m = O\left(\frac{1}{\alpha \epsilon} \text{VC}(C) \log\left(\frac{1}{\alpha \beta}\right)\right).
$$

**Remark 6.11** Notice that the labeled sample complexity in Theorem 6.10 has no dependency in $\delta$. (The unlabeled sample complexity depends on $\delta$ as $n$ depends on $\delta$.) It would be helpful if we could also reduce the dependency on $\epsilon$. As we will later see, this can be achieved in the active learning model.

To show the usefulness of Theorem 6.10, we consider (a discrete version of) the class of all axis-aligned rectangles (or hyperrectangles) in $\ell$ dimensions. Formally, let $X_d^\ell = \{\{0, 1\}^d\}^\ell$ denote a discrete $\ell$-dimensional domain, in which every axis consists of $2^d$ points. For every $a = (a_1, \ldots, a_\ell), b = (b_1, \ldots, b_\ell) \in X_d^\ell$ define the concept $c_{[a,b]} : X_d^\ell \to \{0, 1\}$ where $c_{[a,b]}(x) = 1$ if and only if for every $1 \leq i \leq \ell$ it holds that $a_i \leq x_i \leq b_i$. Define the concept class of all axis-aligned rectangles over $X_d^\ell$ as $\text{RECTANGLE}_d^{\ell} = \{c_{[a,b]}\}_{a,b \in X_d^\ell}$. The VC dimension of this class is $2\ell$, and, thus, it can be learned non-privately with (labeled and unlabeled) sample complexity $O_{a,\beta}(\ell)$. Using LabelBoost with the construction of [10], which has (labeled and unlabeled) sample complexity $O_{a,\beta,\epsilon}(\ell^3 \cdot 8^{\log^* d})$, reduces the labeled sample complexity while maintaining the unlabeled sample complexity.

**Corollary 6.12** There exists a private semi-supervised learner for $\text{RECTANGLE}_d^{\ell}$ with unlabeled sample complexity $\widetilde{O}_{a,\beta,\epsilon,\delta}(\ell^3 \cdot 8^{\log^* d})$ and labeled sample complexity $O_{a,\beta,\epsilon}(\ell')$. The learner is efficient (runs in polynomial time) whenever the dimension $\ell$ is small enough (roughly, $\ell' \leq \log^5 d$).

**Extension to the improper case** Following our work, Alon et al. [3] introduced a small modification to algorithm Relabel that allowed them to extended our transformation from realizable to agnostic learning (stated in Theorem 5.1) also to the case where the base learner is improper. Recall that the difficulty in this case is that we cannot use standard VC arguments to argue that the hypothesis returned by the base learner has small generalization error. The modification that Alon et al. [3] introduced into algorithm Relabel allowed them the use the fact that differential privacy guarantees generalization [7, 21, 30] in order to bound the generalization error of the returned hypothesis. (In a nutshell, differential privacy can be viewed as a form of robustness that ensures generalization.) We refer the reader to [3] for more details.

In our context of semi-supervised learning, we can use the modified Relabel procedure of Alon et al. [3] in algorithm LabelBoost to handle the case that the base learner $A$ is improper. Specifically, we only need to replace the last application
of algorithm \textit{Relabel} with the modified procedure of Alon et al. [3]. To see this, recall that by Claim 6.6 it holds that (w.h.p.) all of the hypotheses $h_i$ used by \textit{Relabel} throughout the execution have small generalization error, and the difficulty was only in bounding $\epsilon^{-1} \left( \frac{a}{\alpha \beta} \right) \log \left( \frac{1}{\alpha \beta} \right)$. In order to use the techniques of Alon et al. [3] to bound $\epsilon^{-1} \left( \frac{a}{\alpha \beta} \right) \log \left( \frac{1}{\alpha \beta} \right)$ using the generalization properties of differential privacy [7, 21, 30], it suffices to replace the last application of \textit{Relabel} in algorithm LabelBoost with the modified relabeling procedure of [3]. This results in the following theorem (extending Theorem 6.10 to the improper case).

\textbf{Theorem 6.13} For every $\alpha, \beta, \delta, n$, if there exists a $(1, \delta)$-differentially private (proper or improper) $(\alpha, \beta)$-PAC learner for a concept class $C$ with sample complexity $n$, then there exists an $(O(\alpha), O(\beta + \delta(m + n)), O(1), O(\delta), O(n + m), m)$-PSSL for $C$, where

$$m = O \left( \frac{1}{\alpha} \text{VC}(C) \log \left( \frac{1}{\alpha \beta} \right) \right).$$

Extension to the agnostic case As in Sect. 5, algorithm LabelBoost can also be used as an agnostic learner, even if the base learner only works in the realizable case. A similar analysis to that of Sect. 5 shows the following result.

\textbf{Theorem 6.14} For every $\alpha, \beta, \delta, n$, if there exists a $(1, \delta)$-differentially private (proper or improper) $(\alpha, \beta)$-PAC learner for a concept class $C$ with sample complexity $n$, then there exists an $(O(\alpha), O(\beta + \delta(m + n)), O(1), O(\delta), O(n + m), m)$-agnostic-PSSL using $C$, where

$$m = O \left( \frac{1}{\alpha^2} \text{VC}(C) \log \left( \frac{1}{\alpha \beta} \right) \right).$$

\section{Private Active Learners}

Semi-supervised learners are a subset of the larger family, called \textit{active learners}, that can adaptively request to reveal the labels of specific examples. An active learner is given access to a pool of $n$ unlabeled examples, and adaptively chooses to label $m$ examples.

\textbf{Definition 7.1} [Active Learning [34]] Let $C$ be a concept class over a domain $X$. Let $\mathcal{A}$ be an interactive (stateful) algorithm that holds an initial input database $D = (x_i)_{i=1}^n \in (X)^n$. For at most $m$ rounds, algorithm $\mathcal{A}$ outputs an index $i \in \{1, 2, \ldots, n\}$ and receives an answer $y_i \in \{0, 1\}$. Afterwards, algorithm $\mathcal{A}$ outputs a hypothesis $h$, and terminates.

Algorithm $\mathcal{A}$ is an $(\alpha, \beta, n, m)$-AL (Active learner) for $C$ if for all concepts $c \in C$ and all distributions $\mu$ on $X$: If $\mathcal{A}$ is initiated on an input $D = (x_i)_{i=1}^n$, where each $x_i$ is drawn i.i.d. from $\mu$, and if every index $i$ queried by $\mathcal{A}$ is answered by $y_i = c(x_i)$,
then algorithm $A$ outputs a hypothesis $h$ satisfying $\Pr[\text{error}_\mu(c, h) \leq \alpha] \geq 1 - \beta$. The probability is taken over the random choice of the samples from $\mu$ and the coin tosses of the learner $A$.

**Remark 7.2** In the standard definition of active learners, the learners specify examples by their value (whereas in Definition 7.1 the learner queries the labels of examples by their index). E.g., if $x_5 = x_9 = p$ then instead of asking for the label of $p$, algorithm $A$ asks for the label example 5 (or 9). This deviation from the standard definition is because when privacy is introduced, every entry in $D$ corresponds to a single individual, and can be changed arbitrarily (and regardless of the other entries).

**Definition 7.3** (Private active learner [5]) An algorithm $A$ is an $(\alpha, \beta, \epsilon, \delta, n, m)$-PAL (Private Active Learner) for a concept class $C$ if $A$ is an $(\alpha, \beta, n, m)$-active learner for $C$ and $A$ is $(\epsilon, \delta)$-differentially private, where in the definition of privacy we consider the input of $A$ to be a fully labeled sample $S = (x_i, y_i)_{i=1}^n \in (X \times \{0, 1\})^n$ (and limit the number of labels $y_i$ it can access to $m$).

Note that the queries that an active learner makes depend on individuals’ data. Hence, if the indices that are queried are exposed, they may breach privacy. An example of how such an exposure may occur is a medical research of a new disease—a hospital may possess background information about individuals and hence can access a large pool of unlabeled examples, but to label an example an actual medical test is needed. Partial information about the labeling queries would hence be leaked to the tested individuals. More information about the queries may be leaked to an observer of the testing site. The following definition remedies this potential breach of privacy.

**Definition 7.4** We define the transcript in an execution of an active learner $A$ as the ordered sequence $L = (i_j)_{j=1}^m \in \{1, 2, \ldots, n\}^m$ of indices that $A$ outputs throughout the execution. We say that a learner $A$ is $(\epsilon, \delta)$-transcript-differentially private if the algorithm whose input is the labeled sample and whose output is the output of $A$ together with the transcript of the execution is $(\epsilon, \delta)$-differentially private. An algorithm $A$ is an $(\alpha, \beta, \epsilon, \delta, n, m)$-TPAL (transcript-private active-learner) for a concept class $C$ if Algorithm $A$ is an $(\alpha, \beta, n, m)$-Active learner for $C$ and $A$ is $(\epsilon, \delta)$-transcript-differentially private.

Recall that a semi-supervised learner has no control over which of its examples are labeled, and the indices of the labeled examples are publicly known. Hence, a private semi-supervised learner is, in particular, a transcript-private active learner.

**Theorem 7.5** If $A$ is an $(\alpha, \beta, \epsilon, \delta, n, m)$-PSSL, then $A$ is an $(\alpha, \beta, \epsilon, \delta, n, m)$-TPAL.

In particular, our algorithms from Sects. 3 and 6 satisfy Definition 7.4, suggesting that the strong privacy guarantees of Definition 7.4 are achievable. However, as we will now see, this comes with a price. The work on (non-private) active
learning has mainly focused on reducing the dependency of the labeled sample complexity in $\alpha$ (the approximation parameter). The classic result in this regime states that the labeled sample complexity of learning $\text{THRESH}_d$ without privacy is $O(\log(1/\alpha))$, exhibiting an exponential improvement over the $\Omega(1/\alpha)$ labeled sample complexity in the non-active model. As the next theorem states, the labeled sample complexity of every transcript-private active-learner for $\text{THRESH}_d$ is lower bounded by $\Omega(1/\alpha)$.

**Theorem 7.6** Let $\alpha \leq 1/9$ and $\beta \leq 1/4$. In every $(\alpha, \beta, \varepsilon, \delta, n, m)$-TPAL for $\text{THRESH}_d$ the labeled sample complexity satisfies $m = \Omega\left(\frac{1}{\alpha}\right)$.

**Proof** Let $A$ be an $(\alpha, \beta, \varepsilon, \delta, n, m)$-TPAL for $\text{THRESH}_d$ with $\alpha \leq 1/9$ and $\beta \leq 1/4$. Without loss of generality, we can assume that $n \geq 1001/\alpha^2 \ln(1/\alpha\beta)$ (since $A$ can ignore part of the sample). Denote $B = \{1, 2, \ldots, 8\alpha 2^d\} \subseteq X_d$, and consider the following thought experiment for randomly generating a labeled sample of size $n$.

1. Let $D = (x_1, x_2, \ldots, x_n)$ denote the outcome of $n$ uniform i.i.d. draws from $X_d$.
2. Uniformly at random choose $t \in B$, and let $c_t \in \text{THRESH}_d$ be s.t. $c_t(x) = 1$ iff $x < t$.
3. Return $S = (x_i, c_t(x_i))_{i=1}^n$.

The above process induces a distribution on labeled samples of size $n$, denoted as $P$. Let $S \sim P$, and consider the execution of $A$ on $S$. Recall that $A$ operates on the unlabeled portion of $S$ and actively queries for labels. Let $b$ denote the number of elements from $B$ in the database $S$. Standard arguments in learning theory (see Theorem 2.12) state that with all but probability $\beta \leq 1/4$ it holds that $7\alpha n \leq b \leq 9\alpha n$. We continue with the proof assuming that this is the case. We first show that $A$ must (w.h.p.) ask for the label of at least one example in $B$. To this end, note that even given the labels of all $x \in B$, the target concept is distributed uniformly on $B$, and the probability that $A$ fails to output an $\alpha$-good hypothesis is at least $\frac{3}{4}$. Hence,
\[
\beta \geq \Pr_{S, A}[A \text{ fails}]
\]
\[
\geq \Pr_{S, A}[A \text{ does not ask for the label of any point in } B \text{ and fails}]
\]
\[
= \Pr_{S, A}[A \text{ does not ask for the label of any point in } B] \cdot \Pr_{S, A}[A \text{ fails} | A \text{ does not ask for the label of any point in } B]
\]
\[
\geq \Pr_{S, A}[A \text{ does not ask for the label of any point in } B] \cdot \frac{3}{4}
\]
\[
\geq \Pr[b \leq 9an] \cdot \Pr_{S, A}[A \text{ does not ask for the label of any point in } B | b \leq 9an] \cdot \frac{3}{4}
\]
\[
\geq \frac{9}{16} \cdot \Pr_{S, A}[A \text{ does not ask for the label of any point in } B | b \leq 9an].
\]

Thus, assuming that \( b \leq 9an \), the probability that \( A \) asks for the label of a point in \( B \) is at least \( (1 - \frac{16}{9} \beta) \). Now choose a random \( x^* \) from \( S \) s.t. \( x^* \in B \). Note that

\[
\Pr_{S, x^*, \hat{x}, A}[A(S) \text{ asks for the label of } x^*]
\]
\[
\geq \Pr[b \leq 9an] \cdot \Pr_{S, x^*, \hat{x}, A}[A(S) \text{ asks for the label of } x^* | b \leq 9an]
\]
\[
\geq (1 - \beta) \cdot \frac{(1 - \frac{16}{9} \beta)}{9an}
\]
\[
\geq \frac{1 - \frac{25}{9} \beta}{9an}.
\]

Choose a random \( \hat{x} \) from \( S \) (uniformly), and construct a labeled sample \( S' \) by swapping the entries \((x^*, c(x^*))\) and \((\hat{x}, c(\hat{x}))\) in \( S \). Note that \( S' \) is also distributed according to \( \mathcal{P} \), and that \( \hat{x} \) is a uniformly random element of \( S' \). Therefore,

\[
\Pr_{S, x^*, \hat{x}, A}[A(S') \text{ asks for the label of } \hat{x}] \leq \frac{m}{n}.
\]

As \( S \) and \( S' \) differ in at most 2 entries, differential privacy states that

\[
\frac{m}{n} \geq \Pr_{S, x^*, \hat{x}, A}[A(S') \text{ asks for the label of } \hat{x}]
\]
\[
= \sum_{S, x^*, \hat{x}} \Pr[S, x^*, \hat{x}] \cdot \Pr[A(S') \text{ asks for the label of } \hat{x}]
\]
\[
\geq \sum_{S, x^*, \hat{x}} \Pr[S, x^*, \hat{x}] \cdot e^{-2e} \cdot \Pr[A(S) \text{ asks for the label of } x^*] - \delta(1 + e^{-\epsilon})
\]
\[
= e^{-2e} \cdot \Pr_{S, x^*, A}[A(S) \text{ asks for the label of } x^*] - \delta(1 + e^{-\epsilon})
\]
\[
\geq e^{-2e} \cdot \frac{1 - \frac{25}{9} \beta}{9an} - \delta(1 + e^{-\epsilon}).
\]
Solving for $m$, and assuming that $\beta, \epsilon, \delta$ are smaller than some absolute constant $0 < c < 1$, this yields $m = \Omega(\frac{1}{a})$. \hfill \Box

The private active learners presented in 5] as well as the algorithm described in the next section only satisfy the weaker Definition 7.3.

### 7.1 Removing the Dependency on the Privacy Parameters

We next show how to transform a semi-supervised private learner $\mathcal{A}$ into an active learner $\mathcal{B}$ with better privacy guarantees without increasing the labeled sample complexity. Algorithm $\mathcal{B}$, takes an unlabeled database $D$ as input, randomly chooses a subset of the inputs $D' \subseteq D$ and asks for the labels of the examples in $D'$ (denote the resulting labeled database as $S$). Algorithm $\mathcal{B}$ then applies $\mathcal{A}$ on $D, S$. As the next claim states, this eliminates the $\frac{1}{\epsilon}$ factor from the labeled sample complexity as the (perhaps adversarial) choice for the input database is independent of the queries chosen.

**Claim 7.7** If there exists an $(\alpha, \beta, \epsilon^*, \delta, n, m)$-PSSL for a concept class $C$, then for every $\epsilon$ there exists an $\left(\alpha, \beta, \epsilon, \frac{7+e^{c}}{3+e^{c}}, \epsilon\delta, t, m\right)$-PAL (private active learner) for $C$, where $t = \frac{n}{\epsilon}(3 + \exp(2e^c))$.

To interpret this claim, consider an $(\alpha, \beta, \epsilon^*=1, \delta, n, m)$-PSSL $\mathcal{A}$ for a concept class $C$. Then, Claim 7.7 states that for every $0 < \epsilon < 1$ there is an $\left(\alpha, \beta, \epsilon, \epsilon\delta, O(\frac{n}{\epsilon}), m\right)$-PAL $\mathcal{B}$ for $C$. That is, only the unlabeled sample complexity of $\mathcal{B}$ grows with $\frac{1}{\epsilon}$, and its labeled sample complexity is independent of the privacy parameter $\epsilon$.

**Algorithm 6 SubSampling**

**Inputs:** Base learner $\mathcal{A}$, privacy parameters $\epsilon^*, \epsilon$, and a database $D = (x_i)_{i=1}^{t}$ of $t$ unlabeled examples.

1. Uniformly at random select a subset $J \subseteq \{1, 2, \ldots, t\}$ of size $n$, and let $K \subseteq J$ denote the smallest $m$ indices in $J$.
2. Request the label of every index $i \in K$, and let $\{y_i : i \in K\}$ denote the received answers.
3. Run $\mathcal{A}$ on the multiset $D_J = \{(x_i, \bot) : i \in J \setminus K\} \cup \{(x_i, y_i) : i \in K\}$.

**Proof** The proof is via the construction of Algorithm SubSampling (Algorithm 6). The utility analysis is straight forward. Fix a target concept $c$ and a distribution $\mu$. Assume that $D$ contains $t$ i.i.d. samples from $\mu$ and that every query on an index $i$ is answered by $c(x_i)$. Therefore, algorithm $\mathcal{A}$ is executed on a multiset $D_J$ containing $n$ i.i.d. samples from $\mu$ where $m$ of those samples are labeled by $c$. By the utility properties of $\mathcal{A}$, an $\alpha$-good hypothesis is returned with probability at least $(1 - \beta)$. 

\copyright Springer
For the privacy analysis, fix two neighboring databases \( S, S' \in (X \times \{0, 1\})^t \) differing on their \( i \)-th entry, and let \( D, D' \in X^t \) denote the restriction of these two databases to \( X \) (that is, \( D \) contains an entry \( x \) for every entry \( (x, y) \) in \( S \)). Consider an execution of SubSampling on \( D \) (and on \( D' \)), and let \( J \subseteq \{1, \ldots, t\} \) denote the random subset of size \( n \) chosen on Step 1. Moreover, and let \( D_J \) denote the multiset on which \( A \) in executed.

Since \( S \) and \( S' \) differ in just the \( i \)-th entry, for any set of outcomes \( F \) it holds that \( \Pr[A(D_J) \in F | i \notin J] = \Pr[A(D'_J) \in F | i \notin J] \). When \( i \in J \) we have that

\[
\Pr[\text{SubSampling}(D) \in F \land i \in J] = \sum_{R \subseteq [t] \setminus \{i\}} \frac{1}{t-n} \sum_{|R| = n-1} \Pr[J = R \cup \{i\}] \cdot \Pr[A(D_J) \in F | J = R \cup \{i\}].
\]

Note that for every choice of \( R \subseteq [t] \setminus \{i\} \) s.t. \( |R| = (n-1) \), there are exactly \((t-n)\) choices for \( Q \subseteq [t] \setminus \{i\} \) s.t. \( |Q| = n \) and \( R \subseteq Q \). Hence,

\[
\Pr[\text{SubSampling}(D) \in F \land i \in J] \leq \sum_{R \subseteq [t] \setminus \{i\}} \frac{1}{t-n} \sum_{|R| = n-1} \Pr[J = Q] \left( e^{2e^r} \Pr[A(D_J) \in F | J = Q] + \delta + \delta e^{e^r} \right).
\]

For the last inequality, note that \( D_Q \) and \( D_{R \cup \{i\}} \) differ in at most two entries, as they differ in one unlabeled example, and possibly one other example that is labeled in one multiset and unlabeled on the other. Now note that every choice of \( Q \) will appear in the above sum exactly \( n \) times (as the number of choices for appropriate \( R \)'s s.t. \( R \subseteq Q \)). Hence,

\[
\Pr \left[ \left( \text{SubSampling}(D) \in F \right) \land \{i \in J\} \right] \leq \frac{n}{t-n} \sum_{Q \subseteq [t] \setminus \{i\}} \Pr[J = Q] \left( e^{2e^r} \Pr[A(D_J) \in F | J = Q] + \delta + \delta e^{e^r} \right) = \frac{n}{t-n} \Pr[i \notin J] \left( e^{2e^r} \Pr[A(D_J) \in F | i \notin J] + \delta + \delta e^{e^r} \right) = \frac{n}{t} e^{2e^r} \cdot \Pr[A(D_J) \in F | i \notin J] + \frac{n}{t} (1 + e^{e^r}) \delta = \frac{n}{t} e^{2e^r} \cdot \Pr[A(D'_J) \in F | i \notin J] + \frac{n}{t} (1 + e^{e^r}) \delta.
\]
Therefore,
\[
\Pr[\text{SubSampling}(D) \in F] = \Pr\left[\{\text{SubSampling} \in F\} \land \{i \in J\}\right] + \Pr[i \notin J] \cdot \Pr[A(D') \in F | i \notin J]
\leq \left(\frac{n}{t} e^{2\varepsilon} + \frac{I-n}{t}\right) \cdot \Pr[A(D') \in F | i \notin J] + \frac{n}{t} (1 + e^{\varepsilon}) \delta.
\]

Similar arguments show that
\[
\Pr[\text{SubSampling}(D') \in F] \geq \left(\frac{n}{t} e^{-2\varepsilon} + \frac{I-n}{t}\right) \cdot \Pr[A(D') \in F | i \notin J] - \frac{n}{t} 2\delta.
\]

For \( t \geq \frac{n}{\varepsilon}(3 + \exp(2\varepsilon)) \), this yields
\[
\Pr[\text{SubSampling}(D) \in F] \leq e^\varepsilon \cdot \Pr[\text{SubSampling}(D') \in F] + \frac{7 + e^{e^\varepsilon} \varepsilon\delta}{3 + e^{2e^\varepsilon}}.
\]

The transformation of Claim 7.7 preserves the efficiency of the base (non-active) learner. Hence, a given (efficient) non-active private learner could always be transformed into an (efficient) active private learner whose labeled sample complexity does not depend on \( \varepsilon \). Applying Claim 7.7 to our semi-supervised learner from Sect. 6 results in the following theorem, showing that the labeled sample complexity of private active learners has no dependency in the privacy parameters \( \varepsilon \) and \( \delta \).

**Theorem 7.8** For every \( \alpha, \beta, \varepsilon, \delta, n \), if there exists a \((1, \delta)\)-differentialy private \((\alpha, \beta)\) -PAC learner for a concept class \( C \) with sample complexity \( n \), then there exists an \((O(\alpha), O(\beta), \varepsilon, \delta, O(\frac{n}{\varepsilon}), m)\)-PAL for \( C \), where
\[
m = O\left(\frac{1}{\alpha \text{VC}(C) \log(\frac{1}{\alpha\beta})}\right).
\]

**8 Conclusions**

Our main result in this paper is that the labeled sample complexity of private learners is characterized by the VC dimension. We obtain this result by presenting two generic constructions of private semi-supervised learners—algorithm GenericLearner and algorithm LabelBoost. Our constructions are (generally) not computationally efficient. Constructing an efficiency-preserving variant for algorithm LabelBoostis left open.

**Acknowledgements** We thank Aryeh Kontorovich, Adam Smith, and Salil Vadhan for helpful discussions of ideas in this work. We thank the anonymous reviewers for their helpful comments and suggestions. Work of A. B. was supported in part by the Israeli Ministry of Science and Technology, by the Israel Science Foundation (Grants 544/13 and 152/17), by the Frankel Center for Computer Science, by ERC Grant 742754 (project NTSC), by the Cyber Security Research Center at Ben-Gurion University of
the Negev, and by NSF Grant No. 1565387, TWC: Large: Collaborative: Computing Over Distributed Sensitive Data. Work of K. N. was done in part while the author was visiting in the Center for Research on Computation and Society, Harvard University, and was initially supported by the Israel Science Foundation (Grant 276/12) and by NSF Grant CNS-1237235 and later by NSF Grant No. 1565387 TWC: Large: Collaborative: Computing Over Distributed Sensitive Data. Work of U. S. was supported in part by the Israeli Ministry of Science and Technology, by the Check Point Institute for Information Security, by the IBM PhD Fellowship Awards Program, by the Frankel Center for Computer Science, by the Israel Science Foundation (Grant 1871/19), and by the Cyber Security Research Center at Ben-Gurion University of the Negev.

References

1. Abadi, M., Chu, A., Goodfellow, I.J., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24–28, 2016, pp. 308–318. ACM (2016). https://doi.org/10.1145/2976749.2978318
2. Agrawala, A.: Learning with a probabilistic teacher. IEEE Trans. Inf. Theory 16(4), 373–379 (1970). https://doi.org/10.1016/S0018-9448(70)90013-7
3. Alon, N., Beimel, A., Moran, S., Stemmer, U.: Closure properties for private classification and online prediction. In: COLT (2020)
4. Alon, N., Livni, R., Malliaris, M., Moran, S.: Private PAC learning implies finite littlestone dimension. In: Charikar, M., Cohen, E. (eds.) Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23–26, 2019, pp. 852–860. ACM (2019). https://doi.org/10.1145/3313276.3316312
5. Balcan, M.F., Feldman, V.: Statistical active learning algorithms. In: Advances in Neural Information Processing Systems, vol. 26, pp. 1295–1303 (2013)
6. Bassily, R., Feldman, V., Talwar, K., Thakurta, A.G.: Private stochastic convex optimization with optimal rates. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8–14 December 2019, Vancouver, BC, Canada, pp. 11279–11288 (2019). http://papers.nips.cc/paper/9306-private-stochastic-convex-optimization-with-optimal-rates
7. Bassily, R., Nissim, K., Smith, A.D., Steinke, T., Stemmer, U., Ullman, J.: Algorithmic stability for adaptive data analysis. In: Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18–21, 2016, pp. 1046–1059 (2016). https://doi.org/10.1145/2897518.2897566
8. Beimel, A., Brenner, H., Kasiviswanathan, S.P., Nissim, K.: Bounds on the sample complexity for private learning and private data release. Mach. Learn. 94(3), 401–437 (2014)
9. Beimel, A., Nissim, K., Stemmer, U.: Characterizing the sample complexity of private learners. In: ITCS, pp. 97–110. ACM (2013)
10. Beimel, A., Nissim, K., Stemmer, U.: Private learning and sanitization: pure vs. approximate differential privacy. Theory Comput. 12(1), 1–61 (2016). https://doi.org/10.4086/toc.2016.v012a001
11. Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: the SuLQ framework. In: Li, C. (ed.) PODS, pp. 128–138. ACM (2005)
12. Blum, A., Ligett, K., Roth, A.: A learning theory approach to noninteractive database privacy. J. ACM 60(2), 12 (2013)
13. Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M.K.: Learnability and the Vapnik-Chervonenkis dimension. J. ACM 36(4), 929–965 (1989)
14. Bun, M., Livni, R., Moran, S.: An equivalence between private classification and online prediction. CoRR arxiv:abs/2003.00563 (2020)
15. Bun, M., Nissim, K., Stemmer, U.: Simultaneous private learning of multiple concepts. In: ITCS, pp. 369–380. ACM (2016)
16. Bun, M., Nissim, K., Stemmer, U., Vadhan, S.P.: Differentially private release and learning of threshold functions. In: FOCS, pp. 634–649 (2015).
17. Bun, M., Ullman, J., Vadhan, S.P.: Fingerprinting codes and the price of approximate differential privacy. In: Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31–June 03, 2014, pp. 1–10 (2014). https://doi.org/10.1145/2591796.2591877
18. Chaudhuri, K., Hsu, D.: Sample complexity bounds for differentially private learning. In: Kakade, S.M., von Luxburg, U. (eds.) COLT, JMLR Proceedings, vol. 19, pp. 155–186. JMLR.org (2011)
19. Chaudhuri, K., Monteleoni, C.: Privacy-preserving logistic regression. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) NIPS. MIT Press (2008)
20. Chaudhuri, K., Monteleoni, C., Sarwate, A.D.: Differentially private empirical risk minimization. J. Mach. Learn. Res. 12, 1069–1109 (2011). http://dl.acm.org/citation.cfm?id=1953048.2021036
21. Dwork, C., Feldman, V., Hardt, M., Pitassi, T., Reingold, O., Roth, A.L.: Preserving statistical validity in adaptive data analysis. In: STOC, pp. 117–126. ACM (2015)
22. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: privacy via distributed noise generation. In: Vaudenay, S. (ed.) EUROCRYPT, Lecture Notes in Computer Science, vol. 4004, pp. 486–503. Springer (2006)
23. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: TCC, Lecture Notes in Computer Science, vol. 3876, pp. 265–284. Springer (2006)
24. Dwork, C., Rothblum, G.N., Vadhan, S.P.: Boosting and differential privacy. In: FOCS, pp. 51–60. IEEE Computer Society (2010)
25. Ehrenfeucht, A., Haussler, D., Kearns, M.J., Valiant, L.G.: A general lower bound on the number of examples needed for learning. Inf. Comput. 82(3), 247–261 (1989)
26. Feldman, V., Koren, T., Talwar, K.: Private stochastic convex optimization: optimal rates in linear time. In: Makarychev, K., Makarychev, Y., Tulsiani, M., Kamath, G., Chuzhoy, J. (eds.) Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22–26, 2020, pp. 439–449. ACM (2020). https://doi.org/10.1145/3357713.3384335
27. Feldman, V., Xiao, D.: Sample complexity bounds on differentially private learning via communication complexity. SIAM J. Comput. 44(6), 1740–1764 (2015). https://doi.org/10.1137/140991844
28. Fralick, S.: Learning to recognize patterns without a teacher. IEEE Trans. Inf. Theor. 13(1), 57–64 (2006). https://doi.org/10.1109/TIT.1967.1053952
29. Gupta, A., Hardt, M., Roth, A., Ullman, J.: Privately releasing conjunctions and the statistical query barrier. In: Fortnow, L., Vadhan, S.P. (eds.) STOC, pp. 803–812. ACM (2011)
30. Hardt, M., Ullman, J.: Preventing false discovery in interactive data analysis is hard. In: FOCS. IEEE (2014)
31. Kaplan, H., Ligett, K., Mansour, Y., Naor, M., Stemmer, U.: Privately learning thresholds: closing the exponential gap. In: COLT (2020)
32. Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? SIAM J. Comput. 40(3), 793–826 (2011)
33. Kearns, M.J.: Efficient noise-tolerant learning from statistical queries. J. ACM 45(6), 983–1006 (1998)
34. McCallum, A., Nigam, K.: Employing em and pool-based active learning for text classification. In: Proceedings of the Fifteenth International Conference on Machine Learning, ICML’98, pp. 350–358. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998). http://dl.acm.org/citation.cfm?id=645527.757765
35. McSherry, F., Talwar, K.: Mechanism design via differential privacy. In: FOCS, pp. 94–103. IEEE Computer Society (2007)
36. Papernot, N., Abadi, M., Erlingsson, Ú., Goodfellow, I.J., Talwar, K.: Semi-supervised knowledge transfer for deep learning from private training data. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings. OpenReview.net (2017), https://openreview.net/forum?id=HkwoSDPgg
37. Rubinstein, B.I.P., Bartlett, P.L., Huang, L., Taft, N.: Learning in a large function space: privacy-preserving mechanisms for svm learning. CoRR arXiv:abs/0911.5708 (2009)
38. Sauer, N.: On the density of families of sets. J. Comb. Theory Ser. A 13(1), 145–147 (1972). https://doi.org/10.1016/0097-3165(72)90019-2
39. Scudder, H.: Probability of error of some adaptive pattern-recognition machines. IEEE Trans. Inf. Theory 11(3), 363–371 (1965). https://doi.org/10.1109/TIT.1965.1053799
40. Valiant, L.G.: A theory of the learnable. Commun. ACM 27(11), 1134–1142 (1984). https://doi.org/10.1145/1968.1972
41. Vapnik, V., Chervonenkis, A.: Theory of Pattern Recognition [in Russian]. Nauka, Moscow (1974)
42. Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. Theory Probab. Appl. 16(2), 264–280 (1971)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Amos Beimel¹ · Kobbi Nissim² · Uri Stemmer¹

Amos Beimel
amos.beimel@gmail.com

Kobbi Nissim
kobbi.nissim@georgetown.edu

¹ Ben-Gurion University, Beer-Sheva, Israel
² Georgetown University, Washington, D.C., USA