A private learner is trained on a sample of labeled points and generates a hypothesis that can be used for predicting the labels of newly sampled points while protecting the privacy of the training set [Kasiviswanathan et al., FOCS 2008]. Research uncovered that private learners may need to exhibit significantly higher sample complexity than non-private learners as is the case with, e.g., learning of one-dimensional threshold functions [Bun et al., FOCS 2015, Alon et al., STOC 2019].

We explore prediction as an alternative to learning. Instead of putting forward a hypothesis, a predictor answers a stream of classification queries. Earlier work has considered a private prediction model with just a single classification query [Dwork and Feldman, COLT 2018]. We observe that when answering a stream of queries, a predictor must modify the hypothesis it uses over time, and, furthermore, that it must use the queries for this modification, hence introducing potential privacy risks with respect to the queries themselves.

We introduce private everlasting prediction taking into account the privacy of both the training set and the (adaptively chosen) queries made to the predictor. We then present a generic construction of private everlasting predictors in the PAC model. The sample complexity of the initial training sample in our construction is quadratic (up to polylog factors) in the VC dimension of the concept class. Our construction allows prediction for all concept classes with finite VC dimension, and in particular threshold functions with constant size initial training sample, even when considered over infinite domains, whereas it is known that the sample complexity of privately learning threshold functions must grow as a function of the domain size and hence is impossible for infinite domains.

1 Introduction

A PAC learner is given labeled examples $S = \{(x_i, y_i)\}_{i \in [n]}$ drawn i.i.d. from an unknown underlying probability distribution $D$ over a data domain $X$ and outputs a hypothesis $h$ that can be used for predicting the label of fresh points $x_{n+1}, x_{n+2}, \ldots$ sampled from the same underlying probability distribution $D$ [Valiant, 1984]. It is well known that when points are labeled by a concept selected from a concept class $C = \{c : X \rightarrow \{0,1\}\}$ then learning is possible with sample complexity proportional to the VC dimension of the concept class.

Learning often happens in settings where the underlying training data is related to individuals and privacy-sensitive and where a learner is required, for legal, ethical, or other reasons, to protect personal information from being leaked in the learned hypothesis $h$.

Private learning was introduced by Kasiviswanathan et al. [2011], as a theoretical model for
studying such tasks. A private learner is a PAC learner that preserves differential privacy with respect to its training set \( S \). That is, the learner’s distribution on outcome hypotheses must not depend too strongly on any single example in \( S \). Kasiviswanathan et al. showed via a generic construction that any finite concept class can be learned privately and with sample complexity \( n = O(\log |C|) \). This value \( (O(\log |C|)) \) can be significantly higher than the VC dimension of the concept class \( C \) (see below).

It is now understood that the gap between the sample complexity of private and non-private learners is essential – an important example is private learning of threshold functions (defined over an ordered domain \( X \) as \( C_{\text{thresh}} = \{ c_t \mid t \in \mathbb{R} \} \) where \( c_t(x) = \mathbb{1}_{x \geq t} \)), which requires sample complexity that is asymptotically higher than the (constant) VC dimension of \( C_{\text{thresh}} \).

In more detail, with pure differential privacy, the sample complexity of private learning is characterized by the representation dimension of the concept class [Beimel et al., 2013a].

The representation dimension of \( C_{\text{thresh}} \) (hence, the sample complexity of private learning thresholds) is \( \Theta(\log |X|) \) [Feldman and Xiao, 2015]. With approximate differential privacy, the sample complexity of learning threshold functions is \( \Theta(\log^* |X|) \) [Beimel et al., 2013b, Bun et al., 2015, Alon et al., 2019, Kaplan et al., 2020, Cohen et al., 2022]. Hence, in both the pure and approximate differential privacy cases, the sample complexity grows with the cardinality of the domain \( |X| \) and no private learner exists for threshold functions over infinite domains, such as the integers and the reals, whereas low sample complexity non-private learners exist for these tasks.

**Privacy preserving (black-box) prediction.** Dwork and Feldman [2018] proposed privacy-preserving prediction as an alternative for private learning. Noting that “[i]t is now known that for some basic learning problems [. . .] producing an accurate private model requires much more data than learning without privacy,” they considered a setting where “users may be allowed to query the prediction model on their inputs only through an appropriate interface”. That is, a setting where the learned hypothesis is not made public. Instead, it may be accessed in a “black-box” manner via a privacy-preserving query-answering prediction interface. The prediction interface is required to preserve the privacy of its training set \( S \).

**Definition 1.1** (private prediction interface [Dwork and Feldman, 2018] (rephrased)). A prediction interface \( M \) is \((\epsilon, \delta)\)-differentially private if for every interactive query generating algorithm \( Q \), the output of the interaction between \( Q \) and \( M(S) \) is \((\epsilon, \delta)\)-differentially private with respect to \( S \).

Dwork and Feldman focused on the setting where the entire interaction between \( Q \) and \( M(S) \) consists of issuing a single prediction query and answering it:

**Definition 1.2** (Single query prediction [Dwork and Feldman, 2018]). Let \( M \) be an algorithm that given a set of labeled examples \( S \) and an unlabeled point \( x \) produces a label \( y \). \( M \) is an \((\epsilon, \delta)\)-differentially private prediction algorithm if for every \( x \), the output \( M(S, x) \) is \((\epsilon, \delta)\)-differentially private with respect to \( S \).

W.r.t. answering a single prediction query, Dwork and Feldman showed that the sample complexity of such predictors is proportional to the VC dimension of the concept class.

**1.1 Our contributions**

In this work, we extend private prediction beyond a single query to answering any sequence – unlimited in length – of prediction queries. We refer to this as private everlasting prediction. Our goal is to present a generic private everlasting predictor with low training sample complexity \( |S| \).

**Private prediction interfaces when applied to a large number of queries.** We begin by examining private everlasting prediction under the framework of Definition 1.1. We prove:

**Theorem 1.3** (informal version of Theorem 3.3). Let \( A \) be a private everlasting prediction interface for concept class \( C \) and assume \( A \) bases its predictions solely on the initial training set \( S \), then there exists a private learner for concept class \( C \) with sample complexity \( |S| \).

This means that everlasting predictors that base their prediction solely on the initial training set \( S \) are subject to the same complexity lowerbounds as private learners. Hence, to avoid
private learning lowerbounds, private everlasting predictors need to rely on more than the initial training sample \( S \) as a source of information about the underlying probability distribution and the labeling concept.

In this work, we choose to allow the everlasting predictor to rely on the queries made - which are unlabeled points from the domain \( X \), assuming the queries are drawn from the same distribution the initial training \( S \) is sampled from. This requires changing the privacy definition, as Definition 1.1 does not protect the queries made, yet the classification given to a query can now depend on and hence reveal information provided in queries made earlier.

**A definition of private everlasting predictors.** Our definition of private everlasting predictors is motivated by the observations above. Consider an algorithm \( A \) that is first fed with a training set \( S \) of labeled points and then executes for an unlimited number of rounds, where in round \( i \) algorithm \( A \) receives as input a query point \( x_i \) and produces a label \( \hat{y}_i \). We say that \( A \) is an everlasting predictor if, when the (labeled) training set \( S \) and the (unlabeled) query points are coming from the same underlying distribution, \( A \) answers each query points \( x_i \) with a good hypothesis \( h_i \), and hence the label \( \hat{y}_i \) produced by \( A \) is correct with high probability. We say that \( A \) is a private everlasting predictor if its sequence of predictions \( \hat{y}_1, \hat{y}_2, \hat{y}_3, \ldots \) protects both the privacy of the training set \( S \) and the query points \( x_1, x_2, x_3, \ldots \) in face of any adversary that adaptively chooses the query points.

We emphasize that while private everlasting predictors need to exhibit average-case utility – as good prediction is required only for the case where \( S \) and \( x_1, x_2, x_3, \ldots \) are i.i.d. from the same underlying distribution – our privacy requirement is worst-case, and holds in face of an adaptive adversary that chooses each query point \( x_i \) after receiving the prediction provided for \( (x_1, \ldots, x_{i-1}) \), and not necessarily in accordance with any probability distribution.

**A generic construction of private everlasting predictors.** Our construction, called \texttt{GenericBBL}, executes in rounds. The input to the first round is the initial labeled training set \( S \), where the number of samples in \( S \) is quadratic in the VC dimension of the concept class. Each other round begins with a collection \( S_i \) of labeled examples and ends with newly generated collection of labeled examples \( S_{i+1} \). The set \( S \) is assumed to be consistent with some concept \( c \in C \) and our construction ensures that this is the case also for the sets \( S_i \) for all \( i \). We briefly describe the main computations performed in each round of \texttt{GenericBBL}.

- **Round initialization:** At the outset of a round, the labeled set \( S_i \) is partitioned into sub-sets, each with number of samples which is proportional to the VC dimension of \( C \) (so we have \( \approx \frac{|S|}{\text{VC}(C)} \) sub-sets). Each of the sub-sets is used for training a classifier non-privately, hence creating a collection of classifiers \( F_i = \{ f : X \rightarrow \{0, 1\} \} \) that are used throughout the round.

- **Query answering:** Queries are issued to the predictor in an online manner. Each query is first labeled by each of the classifiers in \( F_i \). Then the predicted label is computed by applying a privacy-preserving majority vote on these intermediate labels. (By standard composition theorems for differential privacy, we could answer roughly \( |F_i|^2 \approx \left( \frac{|S|}{\text{VC}(C)} \right)^2 \) queries without exhausting our privacy budget.) To save on the privacy budget, the majority vote is based on the \texttt{BetweenThresholds} mechanism of Bun et al. [2016] (which in turn is based on the sparse vector technique). The algorithm fails when the privacy budget is exhausted. However, when queries are sampled from the underlying distribution then with a high enough probability the labels produced by the classifiers in \( F_i \) would exhibit a clear majority.

- **Generating a labeled set for the following round:** The predictions provided in the duration of a round are not guaranteed to be consistent with any concept in \( C \) and hence cannot be used to set the following round. Instead, at the end of the round these points are relabeled consistently with \( C \) using a technique developed by Beimel et al.\(^1\)

\(^1\)Important details, such as privacy amplification via sampling and management of the learning accuracy and error parameters are omitted from the description provided in this section.
GenericBBL is a private everlasting predictor requiring an initial set of labeled examples which is (upto polylogarithmic factors) quadratic in the VC dimension of $C$.

1.2 Related work

Beyond the work of Dwork and Feldman [2018] on private prediction mentioned above, our work is related to private semi-supervised learning and joint differential privacy.

Semi-supervised private learning. As in the model of private semi-supervised learning of Beimel et al. [2021], our predictors depend on both labeled and unlabeled sample. Beyond the obvious difference between the models (outputting a hypothesis vs. providing black-box prediction), a major difference between the settings is that in the work of Beimel et al. [2021] all samples – labeled and unlabeled - are given at once at the outset of the learning process whereas in the setting of everlasting predictors the unlabeled samples are supplied in an online manner. Our construction of private everlasting predictors uses tools developed for the semi-supervised setting, and in particular Algorithm LabelBoost of Beimel et al.

Joint differential privacy. Kearns et al. [2015] introduced joint differential privacy (JDP) as a relaxation of differential privacy applicable for mechanism design and games. For every user $u$, JDP requires that the outputs jointly seen by all other users would preserve differential privacy w.r.t. the input of $u$. Crucially, in JDP users select their inputs ahead of the computation. In our settings, the inputs to a private everlasting predictor are prediction queries which are chosen in an online manner, and hence a query can depend on previous queries and their answers. Yet, similarly to JDP, the outputs provided to queries not performed by a user $u$ should jointly preserve differential privacy w.r.t. the query made by $u$. Our privacy requirement hence extends JDP to an adaptive online setting.

Additional works on private prediction. Bassily et al. [2018] studied a variant of the private prediction problem where the algorithm takes a labeled sample $S$ and is then required to answer $m$ prediction queries (i.e., label a sequence of $m$ unlabeled points sampled from the same underlying distribution). They presented algorithms for this task with sample complexity $|S| \gtrsim \sqrt{m}$. This should be contrasted with our model and results, where the sample complexity is independent of $m$. The bounds presented by Dwork and Feldman [2018] and Bassily et al. [2018] were improved by Dagan and Feldman [2020] and by Nandi and Bassily [2020] who presented algorithms with improved dependency on the accuracy parameter in the agnostic setting.

1.3 Discussion and open problems

We show how to transform any (non-private) learner for the class $C$ (with sample complexity proportional to the VC dimension of $C$) to a private everlasting predictor for $C$. Our construction is not polynomial time due to the use of Algorithm LabelBoost, and requires an initial set $S$ of labeled examples which is quadratic in the VC dimension. We leave open the question whether $|S|$ can be reduced to be linear in the VC dimension and whether the construction can be made polynomial time. A few remarks are in order:

1. Even though our generic construction is not computationally efficient, it does result in efficient learners for several interesting special cases. Specifically, algorithm LabelBoost can be implemented efficiently whenever given an input sample $S$ we could efficiently enumerate all possible dichotomies from the target class $C$ over the points in $S$. In particular, this is the case for the class of 1-dim threshold functions $C_{\text{thresh}}$, as well as additional classes with constant VC dimension. Another notable
example is the class $C_{\text{thresh}}^{\text{enc}}$ which intuitively is an “encrypted” version of $C_{\text{thresh}}$. Bun and Zhandry [2016] showed that (under plausible cryptographic assumptions) the class $C_{\text{thresh}}^{\text{enc}}$ cannot be learned privately and efficiently, while non-private learning is possible efficiently. Our construction can be implemented efficiently for this class. This provides an example where private everlasting prediction can be done efficiently, while (standard) private learning is possible but inefficient.

2. It is now known that some learning tasks require the produced model to memorize parts of the training set in order to achieve good learning rates, which in particular disallows the learning algorithm from satisfying (standard) differential privacy [Brown et al., 2021]. Our notion of private everlasting prediction circumvents this issue, since the model is never publicly released and hence the fact that it must memorize parts of the sample is not of a direct privacy threat. In other words, our work puts forward a private learning model which, in principle, allows memorization. This could have additional applications in broader settings.

3. As we mentioned, in general, private everlasting predictors cannot base their predictions solely on the initial training set, and in this work we choose to rely on the queries presented to the algorithm (in addition to the training set). Our construction can be easily adapted to a setting where the content of the blackbox is updated based on fresh unlabeled samples (whose privacy would be preserved), instead of relying on the query points themselves. This might be beneficial to avoid poisoning attacks via the queries.

2 Preliminaries

2.1 Preliminaries from differential privacy

Definition 2.1 ((\(\epsilon, \delta\))-indistinguishability). Let \(R_0, R_1\) be two random variables over the same support. We say that \(R_0, R_1\) are \((\epsilon, \delta)\)-indistinguishable if for every event \(E\) defined over the support of \(R_0, R_1\),

\[
\Pr[R_0 \in E] \leq e^\epsilon \cdot \Pr[R_1 \in E] + \delta \quad \text{and} \quad \Pr[R_1 \in E] \leq e^\epsilon \cdot \Pr[R_0 \in E] + \delta.
\]

Definition 2.2. Let \(X\) be a data domain. Two datasets \(x, x' \in X^n\) are called neighboring if \(|i : x_i \neq x'_i| = 1\).

Definition 2.3 (differential privacy [Dwork et al., 2006]). A mechanism \(M : X^n \rightarrow Y\) is \((\epsilon, \delta)\)-differentially private if \(M(x)\) and \(M(x')\) are \((\epsilon, \delta)\)-indistinguishable for all neighboring \(x, x' \in X^n\).

In our analysis, we use the post-processing and composition properties of differential privacy, that we cite in their simplest form.

Proposition 2.4 (post-processing). Let \(M_1 : X^n \rightarrow Y\) be an \((\epsilon, \delta)\)-differentially private algorithm and \(M_2 : Y \rightarrow Z\) be any algorithm. Then the algorithm that on input \(x \in X^n\) outputs \(M_2(M_1(x))\) is \((\epsilon, \delta)\)-differentially private.

Proposition 2.5 (composition). Let \(M_1 = (\epsilon_1, \delta_1)\)-differentially private algorithm and let \(M_2 = (\epsilon_2, \delta_2)\)-differentially private algorithm. Then the algorithm that on input \(x \in X^n\) outputs \((M_1(x), M_2(x))\) is \((\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)\)-differentially private.

Definition 2.6 (Exponential mechanism [McSherry and Talwar, 2007]). Let \(q : X^n \times Y \rightarrow \mathbb{R}\) be a score function defined over data domain \(X\) and output domain \(Y\). Define \(\Delta = \max\{|q(x, r) - q(x', r)| : x, x' \in X^n\}\) where the maximum is taken over all \(y \in Y\) and neighboring databases \(x, x' \in X^n\). The exponential mechanism is the \(\epsilon\)-differentially private mechanism which selects an output \(y \in Y\) with probability proportional to \(e^{-\epsilon q(x,y)}/\Delta}\).

Claim 2.7 (Privacy amplification by sub-sampling [Kasiviswanathan et al., 2011]). Let \(A\) be an \((\epsilon', \delta')\)-differentially private algorithm operating on a database of size \(n\). Let \(\epsilon \leq 1\) and let \(t = \frac{\epsilon}{3 + \exp(\epsilon')}\). Construct an algorithm \(B\) operating the database \(D = (z_i)_{i=1}^t\). Algorithm \(B\) randomly selects a subset \(f \subseteq \{1, 2, \ldots, t\}\) of size \(n\), and executes \(A\) on \(D_f = (z_i)_{i\in f}\). Then \(B\) is \((\epsilon, \frac{4\epsilon}{3 + \exp(\epsilon')} \delta')\)-differentially private.
2.2 Preliminaries from PAC learning

A concept class \(C\) over data domain \(X\) is a set of predicates \(c : X \rightarrow \{0, 1\}\) (called concepts) which label points of the domain \(X\) by either 0 or 1. A learner \(A\) for concept class \(C\) is given \(n\) examples sampled i.i.d. from an unknown probability distribution \(D\) over the data domain \(X\) and labeled according to an unknown target concept \(c \in C\). The learner should output a hypothesis \(h : X \rightarrow \{0, 1\}\) that approximates \(c\) for the distribution \(D\). More formally,

**Definition 2.8** (generalization error). The generalization error of a hypothesis \(h : X \rightarrow \{0, 1\}\) with respect to concept \(c\) and distribution \(D\) is defined as error \(\operatorname{Pr}_D[h(x) \neq c(x)]\).

**Definition 2.9** (PAC learning [Valiant, 1984]). Let \(C\) be a concept class over a domain \(X\). Algorithm \(A\) is an \((\alpha, \beta, n)\)-PAC learner for \(C\) if for all \(c \in C\) and all distributions \(D\) on \(X\),

\[
\operatorname{Pr}[\{x_1, \ldots, x_n\} \sim D^n; h \sim A((x_1, c(x_1)), \ldots, (x_n, c(x_n))) ; \operatorname{error}_D(c, h) \leq \alpha] \geq 1 - \beta,
\]

where the probability is over the sampling of \((x_1, \ldots, x_n)\) from \(D\) and the coin tosses of \(A\). The parameter \(n\) is the sample complexity of \(A\).

See Appendix A for additional preliminaries on PAC learning.

2.3 Preliminaries from private learning

**Definition 2.10** (private PAC learning [Kasiviswanathan et al., 2011]). Algorithm \(A\) is a \((\alpha, \beta, \epsilon, \delta, n)\)-private PAC learner if (i) \(A\) is an \((\alpha, \beta, n)\)-PAC learner and (ii) \(A\) is \((\epsilon, \delta)\)-differentially private.

Kasiviswanathan et al. [2011] provided a generic private learner with \(O(V(C) \log(|X|))\) labeled samples. Beimel et al. [2013a] introduced the representation dimension and showed that any concept class \(C\) can be privately learned with \(\Theta(\text{RepDim}(C))\) samples.\(^2\) For the sample complexity of \((\epsilon, \delta)\)-differentially private learning of threshold functions over domain \(X\), Bun et al. [2015] give a lower bound of \(\Omega(|X|)\). Recently, Cohen et al. [2022] give a (nearly) matching upper bound of \(\tilde{O}(\log^2 |X|)\).

3 Towards private everlasting prediction

In this work, we extend private prediction beyond a single query to answering any sequence – unlimited in length – of prediction queries. Our goal is to present a generic private everlasting predictor with low training sample complexity \(|S|\).

**Definition 3.1** (everlasting prediction). Let \(A\) be an algorithm with the following properties:

1. Algorithm \(A\) receives as input \(n\) labeled examples \(S = \{(x_i, y_i)\}_{i=1}^n \in (X \times \{0, 1\})^n\) and selects a hypothesis \(h_0 : X \rightarrow \{0, 1\}\).

2. For round \(r \in \mathbb{N}\), algorithm \(A\) gets a query, which is an unlabeled element \(x_{n+r} \in X\), outputs \(h_{r-1}(x_{n+r})\) and selects a hypothesis \(h_r : X \rightarrow \{0, 1\}\).

We say that \(A\) is an \((\alpha, \beta, n)\)-everlasting predictor for a concept class \(C\) over a domain \(X\) if the following holds for every concept \(c \in C\) and for every distribution \(D\) over \(X\). If \(x_1, x_2, \ldots\) are sampled i.i.d. from \(D\), and the labels of the \(n\) initial samples \(S\) are correct, i.e., \(y_i = c(x_i)\) for \(i \in [n]\), then \(\operatorname{Pr}[\exists r \geq 0 \text{ s.t. } \operatorname{error}_D(c, h_r) > \alpha] \leq \beta\), where the probability is over the sampling of \(x_1, x_2, \ldots\) from \(D\) and the randomness of \(A\).

**Definition 3.2.** An algorithm \(A\) is an \((\alpha, \beta, \epsilon, \delta, n)\)-everlasting differentially private prediction interface if (i) \(A\) is an \((\epsilon, \delta)\)-differentially private prediction interface \(M\) (as in Definition 1.1), and (ii) \(A\) is an \((\alpha, \beta, n)\)-everlasting predictor.

As a warmup, consider an \((\alpha, \beta, \epsilon, \delta, n)\)-everlasting differentially private prediction interface \(A\) for concept class \(C\) over (finite) domain \(X\) (as in Definition 3.2 above). Assume that \(A\) does not vary its hypotheses, i.e. (in the language of Definition 3.1) \(h_r = h_0\) for all \(r > 0\).\(^3\) Note

\(^2\)We omit the dependency on \(\epsilon, \delta, \alpha, \beta\) in this brief review.

\(^3\)Formally, \(A\) can be thought of as two mechanisms \((M_0, M_1)\) where \(M_0\) is \((\epsilon, \delta)\)-differentially private. (i) On input a labeled training sample \(S\) mechanism \(M_0\) computes a hypothesis \(h_0\). (ii) On a query \(x \in X\) mechanism \(M_1\) replies \(h_0(x)\).
that a computationally unlimited adversarial querying algorithm can recover the hypothesis
$h_0$ by issuing all queries $x \in X$. Hence, in using $A$ indefinitely we lose any potential benefits
to sample complexity of restricting access to $h_0$ to being black-box and getting to the point
where the lower-bounds on $n$ from private learning apply. A consequence of this simple
observation is that a private everlasting predictor cannot answer all prediction queries with
a single hypothesis – it must modify its hypothesis over time as it processes new queries.

We now take this observation a step further, showing that a private everlasting predictor
that answers prediction queries solely based on its training sample $S$ is subject to the same
sample complexity lowerbounds as private learners.

Consider an $(\alpha, \beta < 1/8, \epsilon, \delta, n)$-everlasting differentially private prediction interface $A$ for
concept class $C$ over (finite) domain $X$ that upon receiving the training set $S \in (X \times \{0,1\})^n$
selects an infinite sequence of hypotheses $\{h_r\}_{r \geq 0}$ where $h_r : X \to \{0,1\}$. Formally, we
can think of $A$ as composed of three mechanisms $A = (M_0, M_1, M_2)$ where $M_0$ is $(\epsilon, \delta)$-
differentially private:

- On input a labeled training sample $S \in (X \times \{0,1\})^n$ mechanism $M_0$ computes an
  initial state and an initial hypothesis $(\sigma_0, h_0) = M_0(S)$.
- On a query $x_{n+r}$ mechanism $M_1$ produces an answer $M_1(x_{n+r}) = h_r(x_{n+r})$ and mech-
  anism $M_2$ updates the hypothesis-state pair $(h_{r+1}, \sigma_{r+1}) = M_2(\sigma_r)$.

Note that as $M_0$ and $M_2$ do not receive the sequence $\{x_{n+r}\}_{r \geq 0}$ as input, the sequence $\{h_r\}_{r \geq 0}$
depends solely on $S$. Furthermore as $M_1$ and $M_2$ post-process the outcome of $M_0$, i.e., the
sequence of queries and predictions $\{(x_r, h_r(x_r))\}_{r \geq 0}$ preserves $(\epsilon, \delta)$-differential privacy with
respect to the training set $S$. In Appendix B we prove:

**Theorem 3.3.** $A$ can be transformed into a $(O(\alpha), O(\beta), \epsilon, \delta, O(n \log(1/\beta)))$-private PAC learner
for $C$.  

### 3.1 A definition of private everlasting prediction

Theorem 3.3 requires us to seek private predictors whose prediction relies on more inform-
ation than what is provided by the initial labeled sample. Possibilities include requiring
the input of additional labeled or unlabeled examples during the lifetime of the predictor,
while protecting the privacy of these examples. In this work we choose to rely on the queries
for updating the predictor’s internal state. This introduces a potential privacy risk for these
queries as sensitive information about a query may be leaked in the predictions following it.
Furthermore, we need take into account that a privacy attacker may choose their queries
adversarily and adaptively.

**Definition 3.4** (private everlasting black-box prediction). An algorithm $A$ is an $(\alpha, \beta, \epsilon, \delta, n)$-
private everlasting black-box predictor for a concept class $C$ if

1. **Prediction:** $A$ is an $(\alpha, \beta, n)$-everlasting predictor for $C$ (as in Definition 3.1).
2. **Privacy:** For every adversary $B$ and every $t \geq 1$, the random variables $\text{View}^0_{B,t}$ and
   $\text{View}^1_{B,t}$ (defined in Figure 1) are $(\epsilon, \delta)$-indistinguishable.

### 4 Tools from prior works

We briefly describe tools from prior works that we use in our construction. See Appendix C
for a more detailed account.

**Algorithm LabelBoost** [Beimel et al., 2021]: Algorithm LabelBoost takes as input a
partially labeled database $S \cup T \in (X \times \{0,1\})^n$ (where the first portion of the database,
$S$, contains labeled examples) and outputs a similar database where both $S$ and $T$ are
(re)labeled. We use the following lemmata from Beimel et al. [2021]:

**Lemma 4.1** (privacy of Algorithm LabelBoost). Let $A$ be an $(\epsilon, \delta)$-differentially private al-
gorithm operating on labeled databases. Construct an algorithm $B$ that on input a partially
Parameters: \( b \in \{0,1\}, \ t \in \mathbb{N} \).

Training Phase:

1. The adversary \( B \) chooses two sets of \( n \) labeled elements \((x_0^0, y_0^0), \ldots, (x_n^0, y_n^0)\) and \((x_1^1, y_1^1), \ldots, (x_n^1, y_n^1)\), subject to the restriction \( \left| \{ i \in [n] : (x_i^0, y_i^0) \neq (x_i^1, y_i^1) \} \right| \leq 0.1 \).
2. If \( \exists i \) s.t. \((x_i^0, y_i^0) \neq (x_i^1, y_i^1)\) then set Flag = 1. Otherwise set Flag = 0.
3. Algorithm \( A \) gets \((x_1^b, y_1^b), \ldots, (x_n^b, y_n^b)\) and selects a hypothesis \( h_0 : X \to \{0,1\} \).
\( * \) the adversary \( B \) does not get to see the hypothesis \( h_0 \).

Prediction phase:

4. For round \( r = 1, 2, \ldots, t \):
   (a) If Flag = 1 then the adversary \( B \) chooses two elements \( x_{n+r}^0 = x_{n+r}^1 \in X \). Otherwise, the adversary \( B \) chooses two elements \( x_{n+r}^0, x_{n+r}^1 \in X \).
   (b) If \( x_{n+r}^0 \neq x_{n+r}^1 \) then Flag is set to 1.
   (c) If \( x_{n+r}^0 = x_{n+r}^1 \) then the adversary \( B \) gets \( h_{r-1}(x_{n+r}) \).
\( * \) the adversary \( B \) does not get to see the label if \( x_{n+r}^0 \neq x_{n+r}^1 \).
   (d) Algorithm \( A \) gets \( x_{n+r}^b \) and selects a hypothesis \( h_r : X \to \{0,1\} \).
\( * \) the adversary \( B \) does not get to see the hypothesis \( h_r \).

Let \( \text{View}^b_{B,t} \) be \( B \)'s entire view of the execution, i.e., the adversary's randomness and the sequence of predictions in Step 4c.

Figure 1: Definition of \( \text{View}^0_{B,t} \) and \( \text{View}^1_{B,t} \).

\( \text{Lemma 4.2 (Utility of Algorithm LabelBoost).} \) Fix \( \alpha \) and \( k \), and let \( S; T \) be s.t. \( S \) is labeled by some target concept \( c \in C \), and s.t. \( |T| \leq \frac{k}{2} \text{VC}(C) \text{exp}\left(\frac{\alpha|S|}{2\text{VC}(C)}\right) - |S| \). Consider the execution of LabelBoost on \( S; T \), and let \( h \) denote the hypothesis chosen by LabelBoost to relabel \( S; T \).
With probability at least \((1 - \beta) \) we have that \( \text{error}_T(h) \leq \alpha \).

\( \text{Algorithm BetweenThresholds [Bun et al., 2016]:} \) Algorithm BetweenThresholds takes as input a database \( S \in X^n \) and thresholds \( t_{\ell}, t_u \). It applies the sparse vector technique to answer noisy threshold queries with \( L \) (below threshold) \( R \) (above threshold) and \( \top \) (halt).
We use the following lemmata by Bun et al. [2016] and observe that, using standard privacy amplification theorems, Algorithm BetweenThresholds can be modified to allow for \( c \) times of outputting \( \top \) before halting, with a (roughly) \( \sqrt{c} \) growth in its privacy parameter.

\( \text{Lemma 4.3 (Privacy for BetweenThresholds).} \) Let \( \epsilon, \delta \in (0, 1) \) and \( n \in \mathbb{N} \). Then algorithm BetweenThresholds is \((\epsilon, \delta)\)-differentially private for any adaptively-chosen sequence of queries as long as the gap between the thresholds \( t_{\ell}, t_u \) satisfies \( t_u - t_{\ell} \geq \frac{12}{\epsilon n} (\log(10/\epsilon) + \log(1/\delta) + 1) \).

\( \text{Lemma 4.4 (Accuracy of BetweenThresholds).} \) Let \( \alpha, \beta, \epsilon, t_{\ell}, t_u \in (0, 1) \) and \( n, k \in \mathbb{N} \) satisfy \( n \geq \frac{\beta}{\delta^2} (\log(k + 1) + \log(1/\beta)) \). Then, for any input \( x \in X^n \) and any adaptively-chosen sequence of queries \( q_1, q_2, \ldots, q_k \), the answers \( a_1, a_2, \ldots, a_k \) produced by BetweenThresholds on input \( x \) satisfy the following with probability at least \( 1 - \beta \). For any \( j \in [k] \) such that \( a_j \) is returned before BetweenThresholds halts, (i) \( a_j = L \) \( \implies \) \( q_j(x) \leq t_{\ell} + \alpha \), (ii) \( a_j = R \) \( \implies \) \( q_j(x) \geq t_u - \alpha \), and (iii) \( a_j = \top \) \( \implies \) \( t_u - \alpha \leq q_j(x) \leq t_n + \alpha \).

\( \text{Observation 1.} \) Using standard composition theorems for differential privacy (see, e.g., Dwork et al. [2010]), we can assume that algorithm BetweenThresholds takes another parameter \( c \), and halts after \( c \) times of outputting \( \top \). In this case, the algorithm satisfies \((\epsilon', 2c\delta)\)-differential privacy, for \( \epsilon' = \sqrt{2c \ln(\frac{1}{c\delta})} \epsilon + c\epsilon(\epsilon^c - 1) \).
5 A Generic Construction

Our generic construction Algorithm GenericBBL transforms a (non-private) learner for a concept class C into a private everlasting predictor for C. The proof of the following theorem follows from Theorem 5.2 and Claim 5.3 which are proved in Appendix E.

**Theorem 5.1.** Given $a, \beta, \delta < 1/16, \epsilon < 1$, Algorithm GenericBBL is a $(6a, 4\beta, \epsilon, \delta, n)$-private everlasting predictor, where $n$ is set as in Algorithm GenericBBL.

Algorithm GenericBBL

**Initial input:** A labeled database $S \in (X \times \{0, 1\})^n$ where $n = \frac{8r}{\alpha^2 \epsilon^2} \cdot (8VC(C) \log(\frac{26}{\alpha}) + 4 \log(\frac{1}{\beta}) \cdot \log^2(\frac{64VC(C) \log(\frac{26}{\alpha}) + 32 \log(\frac{4}{\beta})}{\epsilon a^2 \beta \delta}) \cdot (3 + \exp(\epsilon + 4))).$

1. Let $\tau > 1.1 \times 10^{10}$. Set $\alpha_1 = \alpha/2$, $\beta_1 = \beta/2$. Define $\lambda_i = \frac{8VC(C) \log(\frac{26}{\alpha}) + 32 \log(\frac{4}{\beta})}{\alpha i}$.

2. By Theorem A.2 $\lambda_i$ samples suffice for PAC learning C with parameters $\alpha_i, \beta_i$.

3. Let $S_1 \subseteq S$ be a random subset of size $n \cdot \frac{\epsilon}{3 + \exp(\epsilon + 4)} = \frac{\tau \cdot \lambda_i^2 \cdot \log^2(\frac{1}{\alpha_i \beta_i})}{\alpha_i \epsilon}$.

4. Repeat for $i = 1, 2, 3, \ldots$

   a. Divide $S_i$ into $T_i = \frac{\tau \cdot \lambda_i^2 \cdot \log^2(\frac{1}{\alpha_i \beta_i})}{\alpha_i \epsilon}$ disjoint databases $S_{i,1}, \ldots, S_{i,T_i}$ of size $\lambda_i$.

   b. For $t \in [T_i]$ let $f_t \in C$ be a hypothesis minimizing error $\epsilon_{S_{i,t}}$. Define $F_i = \{f_1, \ldots, f_{T_i}\}$.

   c. Set $R_i = \frac{25600\|S_i\|}{16}$. Set $t_u = 1/2 + \alpha_i, t_r = 1/2 - \alpha_i$. Set the privacy parameters $\epsilon_i = \frac{\epsilon}{3 \sqrt{c_i \ln(\frac{1}{\gamma_i})}}$ and $\delta_i = \frac{\delta}{\epsilon_i}$, where $c_i = 64 \alpha_i R_i$. Instantiate algorithm BetweenThresholds on the database of hypotheses $F_i$ allowing for $c_i = 64 \alpha_i R_i$ rounds of $\tau$ while satisfying $(1, \delta)$-differential privacy (as in Observation 2).

   d. For $\ell = 1$ to $R_i$:

      i. Receive as input a prediction query $x_{i,\ell} \in X$.

      ii. Give BetweenThresholds the query $q_{x_{i,\ell}}$ where $q_{x_{i,\ell}}(F_i) = \sum_{t \in [T_i]} f_t(x_{i,\ell})$, and obtain an outcome $y_{i,\ell} \in \{L, T, R\}$.

      iii. Respond with the label 0 if $y_{i,\ell} = L$ and 1 if $y_{i,\ell} \in \{T, R\}$.

      iv. If BetweenThresholds halts, then halt and fail (recall that BetweenThresholds only halts if $c_i$ copies of $\tau$ were encountered during the current iteration).

   e. Denote $D_i = (x_{i,1}, \ldots, x_{i,R_i})$.

   f. Let $\tilde{S}_i \subseteq S_i$ and $\tilde{D}_i \subseteq D_i$ be random subsets of size $\frac{\epsilon |S_i|}{3 + \exp(\epsilon + 4)}$ and $\frac{\epsilon |D_i|}{3 + \exp(\epsilon + 4)}$ respectively, and let $\tilde{S}'_i \circ \tilde{D}'_i \leftarrow \text{LabelBoost}(\tilde{S}_i \circ \tilde{D}_i)$. Let $S_{i+1} \subseteq D'_i$ be a random subset of size $\lambda_{i+1} T_{i+1}$.

   g. Set $\alpha_{i+1} \leftarrow \alpha_i / 2$ and $\beta_{i+1} \leftarrow \beta_i / 2.$

**Theorem 5.2 (accuracy of algorithm GenericBBL).** Given $a, \beta, \delta < 1/16$, $\epsilon < 1$, for any concept $c$ and any round $r$, algorithm GenericBBL can predict the label of $x_r$ as $h_r(x_r)$, such that $Pr[\text{error}_D(c(x_r) \neq h_r(x_r))] \leq 6a \cdot [1 - 4\beta].$

**Claim 5.3.** GenericBBL is $(\epsilon, \delta)$-differentially private.

**Remark 5.4.** For simplicity, we analyzed GenericBBL in the realizable setting, i.e., under the assumption that the training set $S$ is consistent with the target class C. Our construction carries over to the agnostic setting via standard arguments (ignoring computational efficiency). We refer the reader to [Beimel et al., 2021] and [Alon et al., 2020] for generic agnostic-to-realizable reductions in the context of private learning.
References


URL https://doi.org/10.1145/3313276.3316312.


A Additional Preliminaries from PAC Learning

It is well known that the sample size $\Theta(VC(C))$ is necessary and sufficient for the PAC learning of a concept class $C$, where the Vapnik-Chervonenkis (VC) dimension of a class $C$ is defined as follows:

**Definition A.1** (VC-Dimension [Vapnik and Chervonenkis, 1971]). Let $C$ be a concept class over a domain $X$. For a set $B = \{b_1, \ldots, b_\ell\} \subseteq X$ of $\ell$ points, let $\Pi_C(B) = \{(c(b_1), \ldots, c(b_\ell)) : c \in C\}$ be the set of all dichotomies that are realized by $C$ on $B$. We say that the set $B \subseteq X$ is shattered by $C$ if $C$ realizes all possible dichotomies over $B$, in which case we have $|\Pi_C(B)| = 2^{|B|}$.

The VC dimension of the class $C$, denoted $VC(C)$, is the cardinality of the largest set $B \subseteq X$ shattered by $C$.

**Theorem A.2** (VC bound). Let $C$ be a concept class over a domain $X$. For $\alpha, \beta < 1/2$, there exists an $(\alpha, \beta, n)$-PAC learner for $C$, where $n = \frac{\thetaVC(C)\log(\frac{12}{\alpha})+4\log(\frac{1}{\beta})}{\alpha}$.

B Proof of Theorem 3.3

The proof of Theorem 3.3 follows from algorithms HypothesisLearner, AccuracyBoost and claims B.1, B.2, all described below.

In Algorithm HypothesisLearner we assume that the everlasting differentially private prediction interface $A$ was fed with $n$ i.i.d. samples taken from some (unknown) distribution $D$ and labeled by an unknown concept $c \in C$. Assuming the sequence of hypotheses $\{h_r\}_{r \geq 0}$ produced by $A$ satisfies

$$\forall r \text{ error}_D(c, h_r) \leq \alpha \quad (1)$$

we use it to construct – with constant probability – a hypothesis $h$ with error bounded by $O(\alpha)$.

**Algorithm HypothesisLearner**

**Parameters:** $0 < \beta \leq 1/8$, $R = |X|\log(|X|)\log(1/\beta)$

**Input:** hypothesis sequence $\{h_r\}_{r \geq 0}$

1. for all $x \in X$ let $L_x = \emptyset$
2. for $r = 0, 1, 2, \ldots, R$
   (a) select $x$ uniformly at random from $X$ and let $L_x = L_x \cup \{h_r(x)\}$
3. if $L_x = \emptyset$ for some $x \in X$ then fail, output an arbitrary hypothesis, and halt
   /* Pr[\exists x \text{ such that } L_x = \emptyset] \leq |X|(1 - \frac{1}{|X|})^R = |X|e^{-R/|X|} = \beta */
4. for all $x \in X$ let $r_x$ be sampled uniformly at random from $L_x$
5. construct the hypothesis $h$, where $h(x) = r_x$

**Claim B.1.** If executed on a hypothesis sequence satisfying Equation 1 then with probability at least $3/4$ Algorithm HypothesisLearner outputs a hypothesis $h$ satisfying $\text{error}_D(c, h) \leq 8\alpha$.

**Proof.** Having $D, c \in C$ fixed, and given a hypothesis $h$, we define $e_h(x)$ to be 1 if $h(x) \neq c(x)$ and 0 otherwise. Thus, we can write $\text{error}_D(c, h) = \mathbb{E}_{x \sim D}[e_h(x)]$.

Observe that when Algorithm HypothesisLearner does not fail, $r_x$ (and hence $h(x)$) is chosen with equal probability among $(h_1(x), h_2(x), \ldots, h_R(x))$ and hence $\mathbb{E}_D[e_h(x)] = \mathbb{E}_{i \sim R}[e_h(x)]$ where $\theta$ denotes the randomness of HypothesisLearner. We get:

$$\mathbb{E}_{\theta}[\text{error}_D(c, h)] = \mathbb{E}_{\theta}\mathbb{E}_{x \sim D}[e_h(x)] = \mathbb{E}_{x \sim D}\mathbb{E}_{\theta}[e_h(x)]$$
$$= \mathbb{E}_{x \sim D}\mathbb{E}_{i \sim R}[e_h(x)] = \mathbb{E}_{i \sim R}\mathbb{E}_{x \sim D}[e_h(x)]$$
$$\leq \mathbb{E}_{i \sim R}[\alpha] = \alpha.$$

By Markov inequality, we have $\Pr_{\theta}[\text{error}_D(c, h) \geq 8\alpha] \leq 1/8$. The claim follows noting that Algorithm HypothesisLearner fails with probability at most $\beta \leq 1/8$. \qed
The second part of the transformation is Algorithm AccuracyBoost that applies Algorithm HypothesisLearner \(O(\log(1/\beta))\) times to obtain with high probability a hypothesis with \(O(\alpha)\) error.

Algorithm AccuracyBoost

Parameters: \(\beta, R = 104\ln \frac{1}{\beta}\)

Input: \(R\) labeled samples with \(n\) examples each \((S_1, \ldots, S_R)\) where \(S_i \in (X \times \{0, 1\})^n\)

1. for \(i = 1, 2 \ldots R\)
   (a) execute \(A(S_i)\) to obtain a hypothesis sequence \(\{h_i^t\}_{t \geq 0}\)
   (b) execute Algorithm WeakHypothesisLearner on \(\{h_i^t\}_{t \geq 0}\) to obtain hypothesis \(h^t\)

2. construct the hypothesis \(\hat{h}\), where \(\hat{h}(x) = \text{maj}(h^1(x), \ldots, h^R(x))\).

Claim B.2. With probability \(1 - \beta\), Algorithm AccuracyBoost output a \(24\alpha\)-good hypothesis over distribution \(D\).

Proof. Define \(B_i\) to be the event where the sequence of hypotheses \(\{h_i^t\}_{t \geq 0}\) produced in Step 1a of AccuracyBoost does not satisfy Equation 1. We have,

\[
\Pr[\text{error}_D(c, h_i) > 8\alpha] \leq \Pr[B] + (1 - \Pr[B]) \cdot \Pr[\text{error}_D(c, h) > 8\alpha] \leq \beta + 1/4 < 3/8.
\]

Hence, by the Chernoff bound, when \(R \geq 104\ln \frac{1}{\beta}\), we have at least \(7R/8\) hypotheses are \(8\alpha\)-good over distribution \(D\). Consider the worst case, in which \(R/8\) hypotheses always output wrong labels. To output a wrong label of \(x\), we require at least \(3R/8\) hypotheses to output wrong labels. Thus \(h\) is \(24\alpha\)-good over distribution \(D\).

C Tools from Prior Works

C.1 Algorithm LabelBoost [Beimel et al., 2021]

Algorithm LabelBoost [Beimel et al., 2021]

Parameters: A concept class \(C\).

Input: A partially labeled database \(S \circ T \in (X \times \{0, 1, \perp\})^*\).

We assume that the first portion of the database (denoted \(S\)) contains labeled examples.

The algorithm outputs a similar database where both \(S\) and \(T\) are (re)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 \(S \circ T\).
   Let \(\Pi_C(P) = \{(c(p_1), \ldots, c(p_\ell)) : c \in C\}\) be the set of all dichotomies generated by \(C\) on \(P\).
3. For every \((z_1, \ldots, z_\ell) \in \Pi_C(P)\), add to \(H\) an arbitrary concept \(c \in C\) s.t. \(c(p_i) = z_i\) 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. (Re)label \(S \circ T\) using \(h\), and denote the resulting database \((S \circ T)^h\), that is, if \(S \circ T = (x_i, y_i)_{i=1}^n\) then \((S \circ T)^h = (x_i, y_i')_{i=1}^n\) where \(y'_i = h(x_i)\).
6. Output \((S \circ T)^h\).

Lemma C.1 (privacy of Algorithm LabelBoost [Beimel et al., 2021]). Let \(A\) be an \((\epsilon, \delta)\)-differentially private algorithm operating on partially labeled databases. Construct an algorithm \(B\) that on input a partially labeled database \(S \circ T \in (X \times \{0, 1, \perp\})^*\) applies \(A\) on the outcome of \(LabelBoost(S \circ T)\). Then, \(B\) is \((\epsilon + 3, 4\epsilon\delta)\)-differentially private.
Consider an execution of LabelBoost on a database SoT, 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 LabelBoost adds to $H$ a hypothesis from $C$ that agrees with $Z$. In particular, $H$ has 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 LabelBoost takes an input database where only a small portion of it is labeled, and returns a similar database in which the labeled portion grows exponentially.

**Lemma C.2** (utility of Algorithm LabelBoost [Beimel et al., 2021]). Fix $\alpha$ and $\beta$, and let SoT be s.t. $S$ is labeled by some target concept $c \in C$, and s.t.

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

Consider the execution of LabelBoost on SoT, and let $h$ denote the hypothesis chosen on Step 4. With probability at least $(1 - \beta)$ we have that $\text{error}_S(h) \leq \alpha$. 

### C.2 Algorithm BetweenThresholds [Bun et al., 2016]

**Algorithm BetweenThresholds** [Bun et al., 2016]

**Input:** Database $S \in X^n$.

**Parameters:** $\varepsilon, t, \mu, \nu \in (0, 1)$ and $n, k \in \mathbb{N}$. 

1. Sample $\mu \sim \text{Lap}(2/\varepsilon n)$ and initialize noisy thresholds $\hat{t}_\ell = t_\ell + \mu$ and $\hat{t}_u = t_u - \mu$.
2. For $j = 1, 2, \ldots, k$:
   a. Receive query $q_j : X^n \rightarrow [0, 1]$.
   b. Set $c_j = q_j(S) + v_j$ where $v_j \sim \text{Lap}(6/\varepsilon n)$.
   c. If $c_j < \hat{t}_\ell$, output $L$ and continue.
   d. If $c_j > \hat{t}_u$, output $R$ and continue.
   e. If $c_j \in [\hat{t}_\ell, \hat{t}_u]$, output $T$ and halt.

**Lemma C.3** (Privacy for BetweenThresholds [Bun et al., 2016]). Let $\varepsilon, \delta \in (0, 1)$ and $n \in \mathbb{N}$. Then algorithm BetweenThresholds is $(\varepsilon, \delta)$-differentially private for any adaptively-chosen sequence of queries as long as the gap between the thresholds $t_\ell, t_u$ satisfies

$$t_u - t_\ell \geq \frac{12 \varepsilon n}{\varepsilon} (\log(10/\varepsilon) + \log(1/\delta) + 1).$$

**Lemma C.4** (Accuracy for BetweenThresholds [Bun et al., 2016]). Let $\alpha, \beta, \varepsilon, t_\ell, t_u \in (0, 1)$ and $n, k \in \mathbb{N}$ satisfy

$$n \geq \frac{8}{\alpha \varepsilon} (\log(k + 1) + \log(1/\beta)).$$

Then, for any input $x \in X^n$ and any adaptively-chosen sequence of queries $q_1, q_2, \ldots, q_k$, the answers $a_1, a_2, \ldots, a_k$ produced by BetweenThresholds on input $x$ satisfy the following with probability at least $1 - \beta$. For any $j \in [k]$ such that $a_j$ is returned before BetweenThresholds halts,

- $a_j = L \Rightarrow q_j(x) \leq t_\ell + \alpha$,
- $a_j = R \Rightarrow q_j(x) \geq t_u - \alpha$, and
- $a_j = T \Rightarrow t_\ell - \alpha \leq q_j(x) \leq t_u + \alpha$.

**Observation 2.** Using standard composition theorems for differential privacy (see, e.g., Dwork et al. [2010]), we can assume that algorithm BetweenThresholds takes another parameter $\delta$, and halts after $c$ times of outputting $T$. In this case, the algorithm satisfies $(\varepsilon', 2c\delta)$-differential privacy, for $\varepsilon' = \sqrt{2c \ln(1/\delta) \varepsilon + c\varepsilon (\varepsilon^c - 1)}$. 

14
D Some Technical Facts

We refer to the execution of steps 3a-3g of algorithm GenericBBL as a phase of the algorithm, indexed by \( i = 1, 2, 3, \ldots \).

The original BetweenThresholds needs to halt when it outputs \( T \). In GenericBBL, we tolerate it to halt at most \( c_i \) times in the phase \( i \). We prove BetweenThresholds in GenericBBL is \((1, \delta)\)-differentially private.

**Claim D.1.** For \( \delta < 1 \), Mechanism BetweenThresholds used in step 3c in the \( i \)-th iteration, is \((1, \delta)\)-differentially private.

**Proof.** Let \( \epsilon'_i, \delta'_i \) be as in Step 3c. Since \( \epsilon'_i - 1 < 2 \epsilon'_i \) for \( 0 < \epsilon'_i < 1 \), we have

\[
\sqrt{2c_i \ln\left(\frac{1}{\epsilon'_i \delta'_i}\right)} \cdot \epsilon'_i + c_i \epsilon'_i (\epsilon'_i - 1) \leq \sqrt{2c_i \ln\left(\frac{2}{\delta'}\right)} \cdot \epsilon'_i + 2c_i \epsilon'_i^2 = \frac{\sqrt{2}}{3} + \frac{2}{9\ln(\frac{2}{\delta})} \leq 1.
\]

The proof is concluded by using observation 2.

In Claim D.2- D.5, we prove that with high probability, BetweenThresholds in step 3d halts within \( 64a_i \) times. We prove it by 4 steps:

1.prove that with high probability, most hypothesis in step 3b have high accuracy (Claim D.2).
2.prove that if most hypothesis in step 3b have high accuracy, then with high probability, the queries in BetweenThresholds are closed to 0 or 1 (Claim D.3).
3.prove that if the queries in BetweenThresholds are closed to 0 or 1, then the Output of the algorithm, is \((1, \delta)\)-differentially private (Claim D.5).
4.prove that if BetweenThresholds outputs L or R, then every single phase fails with low probability(Claim D.4).

**Claim D.2.** If \( \beta_i \leq 1/32 \) and \( T_i \geq 96 \ln \frac{1}{\alpha_i} \), then with probability \( 1 - \alpha_i, \frac{157}{16} \) hypotheses in step 3b are \( \alpha_i \)-good with respect to \( g_i \), where \( g_i \) is the concept of \( S_i \).

**Proof.** By the VC bound (Theorem A.2), for each \( t \in [T_i] \), we have

\[
\Pr[\text{error}_D(f_t, g_i) \leq \alpha_i] \geq 1 - \beta_i.
\]

By Chernoff bound, if \( T_i \geq \frac{16+256\beta_i}{(1-16\beta_i)^2} \ln \frac{1}{\alpha_i} \), then with probability \( 1 - \alpha_i \), we have \( \frac{157}{16} \) hypotheses have error \( D \) of \( f_t, g_i \) \( \leq \alpha_i \). When \( \beta_i \leq 1/32 \), it is sufficient to set \( T_i \geq 96 \ln \frac{1}{\alpha_i} \).

**Claim D.3.** If \( \alpha_i \leq 1/16 \) and \( \frac{157}{16} \) hypotheses in step 3b are \( \alpha_i \)-good with respect to \( g_i \), where \( g_i \) is the concept of \( S_i \), then \( \Pr_{x \sim D}[q(x) - \frac{1}{2} I] \leq \frac{3}{8} \leq 15\alpha_i \).

**Proof.** W.l.o.g. assume \( g_i(x) = 1 \), where \( g_i \) is the concept of \( S_i \), so it is sufficient to prove \( \Pr_{x \sim D}[q(x) \leq \frac{3}{8}] \leq 8\alpha_i \). Consider the worst case that \( \frac{T_i}{16} \) “bad” hypotheses output 0. In that case, \( q(x) \leq \frac{3}{8} \) when \( \frac{T_i}{16} \) of \( \alpha_i \)-good hypotheses output 0. So that with probability \( 15\alpha_i \), we have \( q(x) \leq \frac{3}{8} \) (see Figure 2).

**Claim D.4.** Let \( t_q < 1/2 + 1/8 \) and \( t_q > 1/2 - 1/8 \). For a query \( q \) such that \( q(S) > 7/8 \) (similarly, for \( q(S) < 1/8 \), Algorithm BetweenThresholds outputs R (similarly, L) with probability at least \( 1 - \exp\left(\frac{-T_i}{144\sqrt{c_i \ln(\frac{2}{\delta})}}\right) \).

**Proof.** Wlog assume \( q(S) > 7/8 \), it is sufficient to show

15
Figure 2: The horizontal represents the input point. The vertical represents the hypothesis. The red parts represent the incorrect prediction. We let $\frac{T_i}{16}$ hypothesis predict all labels as 0. To let $q(x) \leq \frac{7}{8}$, there must exist $\frac{T_i}{16}$ hypothesis output 0. In the worst case, at most $15\alpha_i$ of points are labeled as 0.

\[
\Pr[\text{BetweenThresholds outputs } R] = \Pr[q(S) + \text{Lap}(6/\varepsilon'T_i) > t_u + \text{Lap}(2/\varepsilon'T_i)] \\
> \Pr[\text{Lap}(6/\varepsilon'T_i) > -1/8] \cdot \Pr[\text{Lap}(2/\varepsilon'T_i) < 1/8] \\
= \left(1 - \frac{1}{2} \exp\left(-\frac{T_i}{144 \sqrt{c_i \ln\left(\frac{2}{\beta_i}\right)}}\right)\right) \cdot \left(1 - \frac{1}{2} \exp\left(-\frac{T_i}{48 \sqrt{c_i \ln\left(\frac{2}{\beta_i}\right)}}\right)\right) \\
> 1 - \exp\left(-\frac{T_i}{144 \sqrt{c_i \ln\left(\frac{2}{\beta_i}\right)}}\right).
\]

\textbf{Claim D.5.} For any phase $i$, BetweenThresholds outputs $\top$ at most $64\alpha_i R_i$ times with probability at most $\beta_i$.

\textbf{Proof.} For a single query, if $t_u < 1/2 + 1/8$ and $q(S) > 7/8$ (similarly, $t_\ell > 1/2 - 1/8$ and $q(S) < 1/8$), by Claim D.4, BetweenThresholds outputs $\top$ with probability at most $\exp\left(-\frac{T_i}{144 \sqrt{c_i \ln\left(\frac{2}{\beta_i}\right)}}\right) < \alpha_i$. Combine Claim D.2 and D.3, BetweenThresholds outputs $\top$ with probability at most $32\alpha_i$. By the Chernoff bound and $R_i \geq \frac{3\ln(\frac{1}{\alpha_i})}{\alpha_i}$, BetweenThresholds outputs $\top$ more than $64\alpha_i R_i$ times with probability at most $\beta_i$. □

In step 3f, GenericBBL takes a random subset of size $\lambda_i T_{i+1}$ from $\hat{D}'_i$. We show that the size of $\hat{D}'_i$ is at least $\lambda_{i+1} T_{i+1}$.

\textbf{Claim D.6.} When $\varepsilon \leq 1$, for any $i \geq 1$, we always have $|\hat{D}'_i| \geq \lambda_{i+1} T_{i+1}$.

\textbf{Proof.} Let $m = 3 + \exp(\varepsilon + 4) < 200$. By the step 3c, step 3e and step 3f, $|\hat{D}_i| = \frac{\epsilon|D_i|}{m} = \frac{25600|S_i|}{m} \geq 128|S_i| = 128\lambda_j T_j$. Then it is sufficient to verify $128\lambda_j T_j \geq \lambda_{j+1} T_{j+1}$.
We can verify that
\[ 4\lambda_j = 4 \cdot \frac{8VC(C)\log(\frac{13}{\alpha_i}) + 4\log(\frac{2}{\beta_i})}{\alpha_i} = 4 \cdot \frac{8VC(C)(\log(\frac{13}{\alpha_{j+1}}) - 1) + 4(\log(\frac{2}{\beta_{j+1}}) - 1)}{2\alpha_{j+1}} \geq \lambda_{j+1} \]
and
\[ 32T_j = \frac{32\tau \cdot \lambda_i \cdot \log(\frac{1}{\alpha_i \epsilon}) \cdot \log^2\left(\frac{\lambda_i}{\epsilon \alpha_i \beta_i \delta}\right)}{\alpha_i \epsilon} \geq \frac{32\tau \cdot \lambda_i \cdot \log(\frac{1}{\alpha_i \epsilon}) \cdot \log^2\left(\frac{\lambda_{j+1}}{\epsilon \alpha_i \beta_i \delta}\right)}{8\alpha_{j+1} \epsilon} \geq \lambda_{j+1} T_{j+1}. \]
The last inequality holds because \( \lambda_j \geq 4 \) and \( \alpha_j, \beta_j \leq 1/2. \)

To apply the privacy and accuracy of LabelBoost and BetweenThresholds, the sizes of the databases need to satisfy the inequalities in lemma C.2, C.3 and C.4. We verify that in each phase, the sizes of the databases always satisfy the requirement.

**Claim D.7.** Let \( \alpha, \beta, \delta < 1/16, \epsilon \leq 1, \) and \( VC(C) \geq 1. \) Then for any \( i \geq 1, \) we have
\[ T_i \geq \frac{8}{\alpha_i \epsilon} (\log(|D_i|) + 1) + \log(1/\beta_i)). \]

**Proof.** By claim D.6 and step 3c, \(|D_i| = \frac{25600|S_i|}{\epsilon} = \frac{25600\lambda_i T_i}{\epsilon} \). Since
\[ \frac{8}{\alpha_i \epsilon} (\log(|D_i|) + 1) + \log(1/\beta_i)) = \frac{24 \sqrt{64\alpha_i |D_i| \ln(\frac{2}{\delta})}}{\sqrt{2\alpha_i}} \cdot (\log(|D_i|) + 1) + \log(1/\beta_i)) = O \left( \frac{\lambda_i \cdot T_i \cdot \log(\frac{1}{\alpha_i \epsilon})}{\alpha_i \epsilon} \right) \cdot \log\left( \frac{\lambda_i \cdot T_i \cdot \log(\frac{1}{\alpha_i \epsilon})}{\alpha_i \epsilon} \right) \]
and \( T_i = \frac{\tau \cdot \lambda_i \cdot \log(\frac{1}{\alpha_i \epsilon}) \cdot \log^2\left(\frac{\lambda_i}{\epsilon \alpha_i \beta_i \delta}\right)}{\alpha_i \epsilon} \), where \( \tau \geq 1.1 \cdot 10^{10} \), the inequality always holds.

**Claim D.8.** When \( \epsilon \leq 1, \) for any \( i \geq 1, \) we have \(|\hat{D}_i| \leq \frac{\epsilon}{\epsilon} VC(C) \exp\left( \frac{\alpha_i |S_i|}{2VC(C)} \right) - |S_i| \).

**Proof.** By claim D.6, step 3c and step 3f,
\[ |\hat{D}_i| = \frac{\epsilon |D_i|}{m} = O(\lambda_i T_i) = O\left( VC(C) \log^2(VC(C)) \cdot \text{poly}\left( \frac{1}{\alpha_i}, \log(\frac{1}{\beta_i}), \frac{1}{\epsilon}, \log(\frac{1}{\delta}) \right) \right) \]
and
\[ |\hat{S}_i| = \frac{\epsilon |S_i|}{m} \]
\[ = O(\epsilon \lambda_i T_i) = O(\lambda_i T_i) \]
\[ = O\left( VC(C) \log^2(VC(C)) \cdot \text{poly}\left( \frac{1}{\alpha_i}, \log(\frac{1}{\beta_i}), \frac{1}{\epsilon}, \log(\frac{1}{\delta}) \right) \right). \]

Note that
\[ \frac{\beta_i}{\epsilon} VC(C) \exp\left( \frac{\alpha_i |S_i|}{2VC(C)} \right) = \Omega\left( VC^2(C) \cdot \exp\left( \text{poly}\left( \frac{1}{\alpha_i}, \log(\frac{1}{\beta_i}), \frac{1}{\epsilon}, \log(\frac{1}{\delta}) \right) \right) \right), \]
for \( T_i = \frac{\tau \cdot \lambda_i \cdot \log(\frac{1}{\alpha_i \epsilon}) \cdot \log^2\left(\frac{\lambda_i}{\epsilon \alpha_i \beta_i \delta}\right)}{\alpha_i \epsilon} \), the inequality holds when \( \tau \geq 1. \)
We give some technical facts in Appendix D. In Claim E.1, we show that in each phase, the execution of steps 3a-3g of algorithm GenericBBL fails with low probability. In Claim E.2, we prove that algorithm GenericBBL predicts the labels with high accuracy. In Claim E.4, we prove that algorithm GenericBBL does not fail on phases 1 to i, then for phase $i + 1$ we have

$$\Pr \left[ \exists g_{i+1} \in C \text{ s.t. error}_{S_{i+1}}(g_{i+1}) = 0 \text{ and error}_D(g_{i+1}, c) \leq \sum_{j=1}^{i+1} \alpha_j \right] \geq 1 - 2 \sum_{j=0}^{i+1} \beta_j.$$ 

**Proof.** The proof is by induction on $i$. The base case for $i = 1$ is trivial, with $g_1 = c$. Assume the claim holds for all $j \leq i$. By the properties of LabelBoost (Lemma C.2) and Claim D.8, with probability at least $1 - \beta_{i+1}$ we have that $S_{i+1}$ is labeled by a hypothesis $g_{i+1} \in C$ s.t. error$_S(g_{i+1}, g_{i+1}) \leq \alpha_{i+1}$. Observe that the points in $S_i$ (without their labels) are chosen i.i.d. from $D$, and hence, By Theorem A.2 (VC bounds) and $|S_i| \geq 128 \lambda_i \geq \lambda_{i+1}$, with probability at least $1 - \beta_{i+1}$ we have that error$_D(g_{i+1}, g_{i+1}) \leq \alpha_{i+1}$. Hence, with probability $1 - 2\beta_{i+1}$, we have error$_D(g_{i+1}, g_{i+1}) \leq \alpha_{i+1}$. Finally, by the triangle inequality, error$_D(g_{i+1}, c) \leq \sum_{j=0}^{i+1} \alpha_j$, except with probability $2\sum_{j=1}^{i+1} \beta_j$.

Define the following good event.

**Event $E_1$:** Algorithm GenericBBL never fails on the execution of BetweenThresholds in step 3(d)iv.

**Claim E.2.** Event $E_1$ occurs with probability at least $1 - \beta$.

**Proof.** Using to union bound and Claim D.5,

$$\Pr[\text{Event $E_1$ occurs}] \geq 1 - \beta.$$ 

Combining claims E.1 and E.2, we get:

**Claim E.3.** Let $D$ be an underlying distribution and let $c \in C$ be a target concept. Then

$$\Pr[\forall i \exists g_i \in C \text{ s.t. error}_{S_i}(g_i) = 0 \text{ and error}_D(g_i, c) \leq \alpha] \geq 1 - 3\beta.$$
Notations. Consider the $i$th phase of Algorithm GenericBBL, and focus on the $j$-th iteration of Step 3. Fix all of the randomness in BetweenThresholds. Now observe that the output on step 3(d)iii is a deterministic function of the input $x_{i,j}$. This defines a hypothesis which we denote as $h_{i,j}$.

Claim E.4. For $\beta < 1/16$, with probability at least $1 - 4\beta$, all of the hypotheses defined above are $6\alpha$-good w.r.t. $D$ and $c$.

Proof. In the phase $i$, by Claim E.3, with probability at least $1 - 3\beta$ we have that $S_i$ is labeled by a hypothesis $g_i \in C$ satisfying $\text{error}_D(g_i, c) \leq \alpha$. We continue with the analysis assuming that this is the case.

On step 3a of the $i$th phase we divide $S_i$ into $T_i$ subsamples of size $\lambda_i$ each, identify a consistent hypothesis $f_t \in C$ for every subsample $S_{i,t}$, and denote $F_i = (f_1, \ldots, f_{T_i})$. By Theorem A.2 (VC bounds), every hypothesis in $F_i$ satisfies $\text{error}_D(f_t, g_i) \leq \alpha$ with probability $3/4$, in which case, by the triangle inequality we have that $\text{error}_D(f_t, c) \leq 2\alpha$.

Set $T_i \geq \frac{512(1-4\beta)\ln(\frac{1}{\beta_i})}{(1-64\beta_i)^2}$, using Chernoff bound, it holds that for at least $15T_i/16$ of the hypotheses in $F_i$, have error $\text{error}_D(f_t, g_i) \leq 2\alpha$ with probability at least $1 - \beta_i$. These hypotheses have error $\text{error}_D(f_t, c) \leq 3\alpha$.

Let $m : X \rightarrow \{0, 1\}$ defined as $m(x) = \text{maj}_{f_t \in F_i}(f_t(x))$. For $m$ to err on a point $x$ (w.r.t. the target concept $c$), it must be that at least $7/16$-fraction of the $3\alpha$-good hypotheses in $F_i$ err on $x$. Consider the worst case in Figure 4, we have $\text{error}_D(m,c) \leq 6\alpha$.

By Lemma C.4 and Claim D.7, with probability at least $1 - \beta_i$, all of the hypotheses defined during the $i$th iteration satisfy this condition, and are hence $6\alpha$-good w.r.t. $c$ and $D$. By the union bound, with probability $1 - 4\beta$, all the hypotheses are $6\alpha$-good.

E.1 Privacy analysis – proof of Claim 5.3

Fix $t \in \mathbb{N}$ and the adversary $B$. We need to show that $\text{View}^0_{B,t}$ and $\text{View}^1_{B,t}$ (defined in Figure 1) are $(\epsilon, \delta)$-indistinguishable. We will consider separately the case where the executions differ in the training phase (Claim E.5) and the case where the difference occurs during the prediction phase (Claim E.6).
Figure 4: The horizontal represents the input point. The vertical represents the hypothesis. The red parts represent the incorrect prediction. We let $T_{16}$ hypothesis predict all labels incorrectly. To output an incorrect label, there must exist $T_{16}$ hypothesis output the incorrect label. In the worst case, at most $6\alpha$ of points are incorrectly classified.

Privacy of the initial training set $S$. Let $S^0, S^1 \in (X \times \{0, 1\})^n$ be neighboring datasets of labeled examples and let $\text{View}_{B,t}^0$ and $\text{View}_{B,t}^1$ be as in Figure 1 where $(x^0_1, y^0_1), \ldots, (x^0_n, y^0_n)) = S^0$ and $(x^1_1, y^1_1), \ldots, (x^1_n, y^1_n)) = S^1$.

Claim E.5. For all adversaries $B$, for all $t > 0$, and for any two neighbouring database $S^0$ and $S^1$ selected by $B$, $\text{View}_{B,t}^0$ and $\text{View}_{B,t}^1$ are $(\epsilon, \delta)$-indistinguishable.

Proof. Let $R'_1 = \min(t, R_1)$. Note that $\text{View}_{B,R'_1}^b$ is a prefix of $\text{View}_{B,t}^b$ which includes the labels Algorithm $\text{GenericBBL}$ produces in Step 3(d)iii for the $R'_1$ first unlabeled points selected by $B$. Let $S_{2}^b$ be the result of the first application of algorithm $\text{LabelBoost}$ in Step 3f of $\text{GenericBBL}$ (if $t < R_1$ we set $S_{2}^b$ as $\bot$). The creation of these random variables is depicted in Figure 5, where $D_1^L$ denotes the labels Algorithm $\text{GenericBBL}$ produces for the unlabeled points $D_1$.

Figure 5: Privacy of the labeled sample $S$
Observe that View\(^b\)\(_{B,t}\) results from a post-processing (jointly by the adversary \(B\) and Algorithm \texttt{GenericBBL}) of the random variable \((\text{View}\(^b\)\(_{B,R_1'}, S\text{\(_2\)'})\), and hence it suffices to show that 
\[
\left(\text{View}\(^0\)\(_{B,R_1'}, S\text{\(_2\)'}\right) \text{ and } \left(\text{View}\(^1\)\(_{B,R_1'}, S\text{\(_2\)'}\right)
\]
are (\(\epsilon, \delta\))-indistinguishable.

We follow the processes creating View\(^b\)\(_{B,t}\) and \(S\text{\(_2\)'}\) in Figure 5: (i) The mechanism \(M_1\) corresponds to the loop in Step 3d of \texttt{GenericBBL} where labels are produced for the adversarially chosen points \(D_i\). By application of Lemma \texttt{C.3}, \(M_1\) is \((1, \delta)\)-differentially private. 
(ii) The mechanism \(M_2\), corresponds to the subsampling of \(\hat{S}_i\) from \(S_i\) and the application of procedure \texttt{LabelBoost} on the subsample in Step 3f of \texttt{GenericBBL} resulting in \(S\text{\(_2\)'}\). By application of Claim \texttt{2.7} and Lemma \texttt{C.1}, \(M_2\) is \((\epsilon, 0)\)-differentially private. Thus \((M_1, M_2)\) is \((\epsilon + 1, \delta)\)-differentially private. (iii) The mechanism \(M_3\) with input of \(S\) and output \((D_i^{b,l}, S\text{\(_2\)'})\) applies \((M_1, M_2)\) on the sub-sample \(S_i\) obtained from \(S\) in Step 2 of \texttt{GenericBBL}. By application of Claim \texttt{2.7} \(M_3\) is \((\epsilon, \frac{4\epsilon\delta}{\exp(\epsilon+1)})\)-differentially private. Since \(\frac{4\epsilon\delta}{\exp(\epsilon+1)} \leq \delta\) for any \(\epsilon\), hence \((\text{View}\(^0\)\(_{B,R_1'}, S\text{\(_2\)'}\)) and \((\text{View}\(^1\)\(_{B,R_1'}, S\text{\(_2\)'}\)) are (\(\epsilon, \delta\))-indistinguishable \(\Box\)

**Privacy of the unlabeled points \(D\).** Let \(D^0, D^1 \in X^t\) be neighboring datasets of unlabeled examples and let View\(^0\)\(_{B,t}\) and View\(^1\)\(_{B,t}\) be as in Figure 1 where \((x^0_i, \ldots, x^0_i) = D^0\) and \((x^1_i, \ldots, x^1_i) = D^1\).

**Claim E.6.** For all adversaries \(B\), for all \(t > 0\), and for any two neighbouring databases \(D^0\) and \(D^1\) selected by \(B\), View\(^0\)\(_{B,t}\) and View\(^1\)\(_{B,t}\) are (\(\epsilon, \delta\))-indistinguishable.

**Proof.** Let \(D_i^0, D_i^0, \ldots, D_i^0\) and \(D_i^1, D_i^1, \ldots, D_i^1\) be the set of unlabeled databases in step 3e of \texttt{GenericBBL}. Without loss of generality, we assume \(D_i^0\) and \(D_i^1\) differ on one entry. When \(i = k\), View\(^0\)\(_{B,t}\) = View\(^1\)\(_{B,t}\) because all selected hypothesis are the same. When \(i < k\), let 
\(R' = \min \left( \sum_{j=1}^{i+1} R_j, t \right)\).

Similar to the analysis if Claim \texttt{E.5}, View\(^b\)\(_{B,t}\) results from a post-processing of the random variable \((\text{View}\(^b\)\(_{B,R'}, S\text{\(_2\)'}\text{\(_{t+2}\)})\) (if \(t < \sum_{j=1}^{i+1} R_j\) we set \(S\text{\(_2\)'}_{t+2}\) as \(\perp\)). Note that View\(^b\)\(_{B,R'}\) = \((D_i^{b,l}, \ldots, D_i^{b,L}, D_i^{b,L}_{i+1})\) and \((D_i^{1,l}, \ldots, D_i^{1,L}, D_i^{1,L}_{i+1})\) follow the same distribution for \(b \in \{0, 1\}\), where \(D_i^{b,l}\) is the labels of points in \(D_i\) expect the different point. So that it suffices to show that \((D_i^{b,l}_{i+1}, S\text{\(_2\)'}_{t+2})\) and \((D_i^{1,l}_{i+1}, S\text{\(_2\)'}_{t+2})\) are \((\epsilon, \delta)\)-indistinguishable.
We follow the processes creating \( D_{i+1}^{b,L} \) and \( S_{i+2}^b \) in Figure 6: (i) The mechanism \( M_1 \) corresponds to the loop in Step 3d of GenericBBL where labels are produced for the adversarially chosen points \( D_{i+1}^b \). By application of Lemma C.3, \( M_1 \) is \((1,\delta)\)-differentially private. (ii) The mechanism \( M_2 \), corresponds to the subsampling of \( \hat{S}^b_i \) from \( S^b_i \) and the application of procedure LabelBoost on the subsample in Step 3f of GenericBBL resulting in \( S_{i+2}^b \). By application of Claim 2.7 and Lemma C.1, \( M_2 \) is \((\varepsilon,0)\)-differentially private. Thus \((M_1,M_2)\) is \((\varepsilon + 1,\delta)\)-differentially private. (iii) The mechanism \( M_3 \) with input of \( \hat{D}^b_i \) and output \((D_{i+1}^{b,L},S_{i+2}^b)\) applies \((M_2,M_3)\) on \( S_{i+1,i+2} \), which is generated from \( D_{i+1}^b \) and in Step 3f of GenericBBL. By application of Claim C.1, \( M_3 \) is \((\varepsilon + 4,4\varepsilon\delta)\)-differentially private. (iv) The mechanism \( M_4 \), corresponds to the subsampling \( \hat{D}^b_i \) from \( D_i^b \) and the application of \( M_4 \) on \( \hat{D}^b_i \). By application of Claim 2.7, \( M_4 \) is \((\varepsilon,\frac{16\varepsilon\delta}{3+\exp(\varepsilon+4)})\)-differentially private. Since \( \frac{16\varepsilon\delta}{3+\exp(\varepsilon+4)} \leq 1 \) for any \( \varepsilon \), \((D_{i+1}^{b,L},S_{i+2}^0)\) and \((D_{i+1}^{1,L},S_{i+2}^1)\) are \((\varepsilon,\delta)\)-indistinguishable.

Remark E.7. The above proofs work on the adversarially selected \( D \) because: (i) Lemma C.3 works on the adaptively selected queries. (We treat the hypothesis class \( F_i \) as the database, the unlabelled points \( x_{i,\ell} \) as the query parameters.) (ii) LabelBoost generates labels by applying one private hypothesis on points. The labels are differentially private by post-processing.