The Limits of Differential Privacy in Online Learning

Bo Li Wei Wang Peng Ye
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Hong Kong SAR, China
bli@ust.hk, weiwa@cse.ust.hk, pyeac@connect.ust.hk

Abstract

Differential privacy (DP) is a formal notion that restricts the privacy leakage of an algorithm when running on sensitive data, in which privacy-utility trade-off is one of the central problems in private data analysis. In this work, we investigate the fundamental limits of differential privacy in online learning algorithms and present evidence that separates three types of constraints: no DP, pure DP, and approximate DP. We first describe a hypothesis class that is online learnable under approximate DP but not online learnable under pure DP under the adaptive adversarial setting. This indicates that approximate DP must be adopted when dealing with adaptive adversaries. We then prove that any private online learner must make an infinite number of mistakes for almost all hypothesis classes. This essentially generalizes previous results and shows a strong separation between private and non-private settings since a finite mistake bound is always attainable (as long as the class is online learnable) when there is no privacy requirement.

1 Introduction

Machine learning has demonstrated extraordinary capabilities in various industries, from healthcare to finance. Yet, it could raise serious privacy concerns as it may require access to a vast amount of personal data. The data used to train machine learning models may also contain sensitive information such as medical records or financial transactions. Therefore, it is crucial to ensure that the private data is well-protected during the training process.

Differential privacy (DP) [24, 23] is a rigorous mathematical definition that quantifies the level of personal data leakage. In a nutshell, an algorithm is said to be *differentially private* if the change of any individual's data won't make the output significantly different. DP has become the standard notion of privacy and has been broadly employed [1, 6, 2].

However, privacy is not a free lunch and usually comes at a cost. Simple tasks may become much harder or even intractable when privacy constraints are imposed. It is crucial to understand the cost to pay for privacy. For probably approximately correct (PAC) learning [44, 11, 43], which is the standard theoretical model of machine learning, there have been many works investigating the cost associated with privacy, which demonstrate a huge discrepancy in terms of the cost under non-private, pure private, and approximate private constraints.

One central requirement in PAC learning is that the data need to be i.i.d. generated and given in advance. Such assumptions fail to capture many scenarios in practice. For example, fraud detection in financial transactions often needs to be handled in real time, which prohibits access to the entire dataset. Moreover, fraudulent patterns can change over time, and new types of fraud can emerge. In such a scenario, the data are clearly not i.i.d. and can even be adaptive to the algorithm's prior predictions, in which the online learning model should be adopted.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-2086

Table 1: Separation between three types of constraints

	Oblivious adversary		Adaptive adversary
	Finite mistakes?	Learnable?	Learnable?
No DP constraints	✓ ([39])	✓ ([39])	✓ ([39])
Approximate DP	✗ (Theorem 4.3)	✓ ([30])	✓ ([30])
Pure DP	✗ (Theorem 4.3)	✓ (Theorem 3.3)	✗ (Theorem 3.5)

Compared with private PAC learning, the limits of private online learning are less understood. For approximate DP, algorithms for Littlestone classes were proposed [30]. But for pure DP, the only known result is the one for point functions given by Dmitriev et al. [22]. They also suggested that one could leverage existing tools of DP continual observation [26, 34] to design algorithms for finite hypothesis classes and asked if generic learners can be constructed for infinite hypothesis classes.

Going beyond qualitative learnability, it is worth quantitatively exploring the number of mistakes made by an online learner. Without privacy, it is possible to achieve a mistake bound of at most the Littlestone dimension of the hypothesis class [39], which is independent of the total rounds T. Therefore, the number of mistakes is always bounded as $T \to \infty$. However, all existing private online learning algorithms suffer from an error count that grows at least logarithmically with T. It was asked by Sanyal and Ramponi [41] whether such a cost is inevitable for DP online learning. In a recent work of Cohen et al. [21], they showed that any private online learning algorithm for the class of point functions over $[T]$ must incur $\Omega(\log T)$ mistakes. However, it remains open whether such cost is unavoidable for generic hypothesis classes, especially for those with a smaller cardinality.

1.1 Main Results

We obtain results that separate three types of constraints: no DP, pure DP, and approximate DP.

Separation between pure and approximate DP. We first perform a systematic study of online learning under pure DP. We prove that every pure privately PAC learnable class is also pure privately online learnable against oblivious adversaries, answering a question raised by Dmitriev et al. [22]. For the stronger adaptive adversaries, we obtain an impossibility result that the class of point functions over $\mathbb{N}$, which can be pure privately learned in the offline model, is not online learnable under pure DP. According to the result of Golowich and Livni [30], it is online learnable under approximate DP. Thus, our conclusion reveals a strong separation between these two privacy definitions.

Separation between private and non-private settings. We next quantitatively investigate the dependence on T in the mistake bound. We show that for any hypothesis class $\mathcal{H}$, any private online learning algorithm must make $\Omega(\log T)$ mistakes unless $\mathcal{H}$ contains only one single hypothesis or exactly two complementary hypotheses (see Section 4 for the definition). This largely generalizes previous results and indicates that such a separation indeed exists universally. We further improve the lower bound to $\Omega(\mathtt{LD}(\mathcal{H}) \log T)$, where $\mathtt{LD}(\mathcal{H})$ represents the Littlestone dimension of $\mathcal{H}$.

To better demonstrate our results, we consider the task of online learning point functions over $\mathbb{N}$ in the oblivious setting and summarize in Table 1 the finiteness of mistakes and learnability under the three types of constraints. Note that for this hypothesis class, the impossibility of making finite mistakes in private online learning can also be derived from the result in [21]. However, our conclusion (Theorem 4.3) is more general – it applies to a much broader family of hypothesis classes. We choose this hypothesis class for illustration because it separates the learnability against adaptive adversaries under pure DP and approximate DP.

1.2 Related Work

The work of [38] initialized the study of PAC learning with differential privacy. A series of subsequent works then showed that privacy constraints have a distinctive impact on the learners. The most remarkable result is the equivalence between approximate private learning and (non-private) online learning [4, 15, 5, 29]. For pure DP, the learnability is characterized by the so-called representation dimension [10]. Both results suggest that private learning is strictly harder than non-private learning. For some specific hypothesis class such as the one-dimensional threshold over finite domain, it was shown that learning with approximate DP enjoys a much lower sample complexity than with pure DP [9, 19, 28, 13, 14, 37, 20], separating these two types of privacy. Another separation result is the huge gap between properly and improperly learning point functions with pure DP [9], which does not exist in non-private and approximate private settings.

The problem of private online learning Littlestone classes was first studied by Golowich and Livni [30]. They proposed private online learning algorithms for hypothesis classes of finite Littlestone dimension in the realizable setting against oblivious and adaptive adversaries, further strengthening the connection between online learning and differential privacy. In contrast with the non-private setting where the mistake bound is always finite, their algorithms exhibit a cost of $\log T$ for data streams of length T. Recently, it was shown by Cohen et al. [21] that this extra cost is unavoidable for point functions over $[T]$. Dmitriev et al. [22] also obtained similar results, but only for algorithms that satisfy certain properties. It was questioned by Sanyal and Ramponi [41] whether an unbounded number of mistakes is necessary for $\varepsilon \approx \sqrt{T}$ (see Section 2.2 for the definition of ε).

There were also a great number of results on private parametric online learning tasks such as online predictions from experts (OPE) and online convex optimization (OCO) [36, 42, 35, 3, 8]. Most of them focus on the agnostic setting. Asi et al. [7] developed algorithms for both problems in the realizable regime with oblivious adversaries, again with a $\log T$ overhead. Asi et al. [8] obtained some hardness results for DP-OPE against adaptive adversaries, but they require the number of experts to be larger than T.

Another related field is differential privacy under continual observation (see, e.g., [26, 18, 34]). While the techniques can be used to design online learning algorithms, it is unclear whether lower bounds for DP continual observation can be transformed to any hardness results for private online learning (see [22] for a detailed discussion).

2 Preliminaries

Notation. Throughout this paper, we use $S = \{z_1, \ldots, z_t\}$ to denote a data stream of length T. We write $S[t]$ to denote the data point comes at time-step t, i.e., z_t. For an algorithm $\mathcal{A}$ that runs on S, we use $\mathcal{A}(S)_t$ to denote the output of $\mathcal{A}$ at time-step t.

2.1 Online Learning

We start by defining online learning as a sequential game played between a learner and an adversary. Let $\mathcal{H} \subseteq \{0,1\}^{\mathcal{X}}$ be a hypothesis class over domain $\mathcal{X}$ and T be a positive integer indicating the total number of rounds. In the t-th round, the learner outputs a hypothesis $h_t \in \{0,1\}^{\mathcal{X}}$ (not required to be in $\mathcal{H}$) while the adversary presents a pair (x_t, y_t). The performance of the learner is measured by the expected *regret*, which is the expected number of additive mistakes made by the learner compared to the best (in hindsight) hypothesis in $\mathcal{H}$:

$$\mathbb{E}\left[\sum_{t=1}^{T} \mathbb{I}[h_t(x_t) \neq y_t] - \min_{h^\star \in \mathcal{H}} \sum_{t=1}^{T} \mathbb{I}[h^\star(x_t) \neq y_t]\right].$$

The above setting is usually referred to as the *agnostic* setting, where we do not make any assumptions on the data. In the *realizable* setting, it is guaranteed that there is some $h^\star \in \mathcal{H}$ so that $y_t = h^\star(x_t)$ for all $t \in [T]$. In this setting, the performance is directly measured by the expected number of mistakes made by the learner, which is called the *mistake bound*, defined as

$$\mathbb{E}\left[\sum_{t=1}^{T} \mathbb{I}[h_t(x_t) \neq y_t]\right].$$

 https://doi.org/10.52202/079017-2086

An online learning algorithm is considered successful if it attains a sublinear regret, i.e., the regret is $o(T)$. In this paper, we mainly focus on the realizable setting. We say a hypothesis class $\mathcal{H}$ is online learnable if there is an online learning algorithm for $\mathcal{H}$ that makes $o(T)$ mistakes in expectation.

We consider two types of adversaries: an *oblivious* adversary chooses the examples in advance (could depend on the learner's strategy, but not on its internal randomness), and (x_t, y_t) is revealed to the learner in the t-th round. An *adaptive* adversary instead, can choose (x_t, y_t) based on past history, i.e., $h_1, \ldots, h_{t-1}$ and $(x_1, y_1), \ldots, (x_{t-1}, y_{t-1})$.

Without privacy, the mistake bound is exactly characterized by the Littlestone dimension [39] even with stronger adversaries that can choose (x_t, y_t) after seeing h_t. To define the Littlestone dimension, we first introduce the notion of a shattered tree.

Definition 2.1 (Shattered Tree)**.** Consider a full binary tree of depth d such that each node is labeled by some $x \in \mathcal{X}$. Every $\{y_1, \ldots, y_d\} \in \{0,1\}^d$ defines a root-to-leaf path $x_1, \ldots, x_d$ obtained by starting at the root, then for each $i = 2, \ldots, d$ choosing x_i to be the left child of x_{i-1} if $y_{i-1} = 0$ and to be the right child otherwise. The tree is said to be shattered by $\mathcal{H}$ if for every root-to-leaf path defined in this way, there exists $h \in \mathcal{H}$ such that $y_i = h(x_i)$ for all $i \in [d]$.

Definition 2.2 (Littlestone Dimension)**.** The Littlestone dimension of $\mathcal{H}$, denoted by $\texttt{LD}(\mathcal{H})$, is the maximal d such that there exists a full binary tree of depth d that is shattered by $\mathcal{H}$.

The problem of online prediction from experts can be viewed as a parametric version of online learning. Let d be the total number of experts. In the t-th round, the algorithm chooses an expert $i_t \in [d]$ while the adversary selects a loss function $\ell_t : [d] \to [0,1]$. Then ℓ_t is revealed and a cost of $\ell_t(i_t)$ is incurred. The goal is to minimize the expected regret

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(i_t) - \min_{i \in [d]} \sum_{t=1}^{T} \ell_t(i)\right].$$

Similar to online learning, an oblivious adversary chooses all ℓ_t in advance, while an adaptive adversary determines ℓ_t based on $i_1, \ldots, i_{t-1}$ and $\ell_1, \ldots, \ell_{t-1}$. In the realizable setting, it is guaranteed that there exists $i^\star \in [d]$ such that $\ell_t(i^\star) = 0$ for all $t \in [T]$.

2.2 Differential Privacy

We first recall the standard definition of differential privacy.

Definition 2.3 (Differential Privacy)**.** An algorithm $\mathcal{A}$ is said to be (ε, δ)-differentially private if for any two sequences S_1 and S_2 that differ in only one entry and any event O, we have

$$\Pr[\mathcal{A}(S_1) \in O] \leq e^{\varepsilon} \Pr[\mathcal{A}(S_2) \in O] + \delta.$$

When $\delta = 0$, we also say $\mathcal{A}$ is ε-differentially private.

Our proofs use the packing argument [32, 9], which heavily relies on the following property of DP.

Fact 2.4 (Group Privacy)**.** *Let $\mathcal{A}$ be an (ε, δ)-differentially private algorithm. Then for any two sequences S_1 and S_2 that differ in k entries and any event O, we have*

$$\Pr[\mathcal{A}(S_1) \in O] \leq e^{k\varepsilon} \Pr[\mathcal{A}(S_2) \in O] + \frac{e^{k\varepsilon} - 1}{e^{\varepsilon} - 1} \cdot \delta.$$

Privacy with *adaptive* adversaries. When interacting with adaptive adversaries, the notion of differential privacy becomes a bit trickier.[1] We adopt the definition of adpative differential privacy from [30, 34, 8]. Let $\mathcal{A}$ be an online algorithm, Adv be an adversary[2] who generates two sequences S_1 and S_2 adaptively such that S_1 and S_2 differ in only one entry, and $b \in \{1, 2\}$ be a global parameter that is unknown to $\mathcal{A}$ and Adv. The interactive process $\mathcal{A} \circ \mathsf{Adv}(b)$ works as follows: in each time-step t, Adv generates two data points $S_1[t], S_2[t]$ based on the past history and $\mathcal{A}$ gets $S_b[t]$. The output of $\mathcal{A} \circ \mathsf{Adv}(b)$ is defined to be the entire output of $\mathcal{A}$. We say $\mathcal{A}$ satisfies (ε, δ)-adaptive differential privacy if for any such adversary Adv and any event O, we have

$$\Pr[\mathcal{A} \circ \mathsf{Adv}(1) \in O] \leq e^{\varepsilon} \Pr[\mathcal{A} \circ \mathsf{Adv}(2) \in O] + \delta.$$

[1] It turns out that the standard differential privacy and adaptive differential privacy are equivalent when $\delta = 0$ [26, 42]. But we will use the adaptive version in our proof since it is easier to work with.

[2] Note that the adversary here is different from the one in the definition of online learning.

Choosing the privacy parameters. It is a commonly agreed principle that for the definition of differential privacy to be meaningful, the parameter δ should be much less than the inverse of the dataset size [27]. In this paper, when we say an algorithm $\mathcal{A}$ is private without specifying the privacy parameters, we typically refer to the set-up that ε is a small constant (say 0.01) and $\delta = o(1/T)$.

3 Learning with Pure Differential Privacy

In this section, we study online learning under pure DP constraint. We first propose algorithms for privately offline learnable hypothesis classes against oblivious adversaries via a reduction to OPE using the tool of probabilistic representation. Then we turn to adaptive adversaries and present a hypothesis class that is privately offline learnable but not privately online learnable. Note that according to the results of [30], this class is online learnable under approximate DP with adaptive adversaries. Hence, we manifest a strong separation between pure and approximate DP.

3.1 Learning Against Oblivious Adversaries

In this section, we consider an oblivious adversary. We first recall the notion of representation dimension, which was introduced by Beimel et al. [10] to characterize pure DP offline learnability. Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \{0,1\}$ and $h \in \{0,1\}^{\mathcal{X}}$ be a hypothesis. The error of h with respect to $\mathcal{D}$ is defined as $\mathrm{error}_{\mathcal{D}}(h) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$.

Definition 3.1 (Representation Dimension)**.** A probability distribution $\mathcal{P}$ over hypothesis classes is said to be an (α,β)-probabilistic representation for $\mathcal{H}$ if for any $h^\star \in \mathcal{H}$ and any distribution $\mathcal{D}$ over $\mathcal{X} \times \{0,1\}$ that is labeled by $h^\star$, we have

$$\Pr_{V\sim\mathcal{P}}[\exists v \in V \ s.t. \ \mathrm{error}_{\mathcal{D}}(v) \leq \alpha] \geq 1-\beta.$$

Let $\mathrm{size}(\mathcal{P}) = \max_{V\in\mathrm{supp}(\mathcal{P})} \ln|V|$. The representation dimension of $\mathcal{H}$, denoted by $\texttt{RepDim}(\mathcal{H})$, is defined as

$$\texttt{RepDim}(\mathcal{H}) = \min_{\mathcal{P} \text{ is a } (1/4,1/4)\text{-probabilistic representation for } \mathcal{H}} \mathrm{size}(\mathcal{P}).$$

The following lemma from [10] shows that a constant probabilistic representation can be boosted to an (α,β) one with logarithmic cost in $1/\alpha$ and $1/\beta$.

Lemma 3.2. *There exists an (α,β)-probabilistic representation for $\mathcal{H}$ with*

$$\mathrm{size}(\mathcal{P}) = O(\log(1/\alpha)\cdot(\texttt{RepDim}(\mathcal{H}) + \log\log\log(1/\alpha) + \log\log(1/\beta))).$$

We first consider the realizable setting. Let $S = \{(x_1,y_1),\ldots,(x_T,y_T)\}$ be the sequence chosen by the adversary and $\mathcal{D}_S$ be the empirical distribution of S (i.e., $\Pr_{(x,y)\sim\mathcal{D}_S}[(x,y)=(x_t,y_t)] = 1/T$ for all $t \in [T]$). By sampling a hypothesis class V from an (α,β)-probabilistic representation with $1/\alpha < 1/T$, we know that it holds with probability at least $1-\beta$ that $\mathrm{error}_{\mathcal{D}_S}(v) \leq 1/\alpha < 1/T$ for some $v \in V$. This further implies that v is consistent with all examples in S. By Lemma 3.2, V is finite as long as $\mathcal{H}$ has a finite representation dimension. Thus, it suffices to run the DP-OPE algorithm in [7] with every $v \in V$ as an expert.

Theorem 3.3. *Let $\mathcal{H}$ be a hypothesis class with $\texttt{RepDim}(\mathcal{H}) < \infty$. In the realizable setting, there exists an online learning algorithm that is ε-differentially private and has an expected mistake bound of $O\left(\frac{\log^2 T(\texttt{RepDim}(\mathcal{H})+\log\log T)^2}{\varepsilon}\right)$ with an oblivious adversary.*

The above conclusion directly extends to the agnostic setting by replacing the DP-OPE algorithm with an agnostic one [8].

Theorem 3.4. *Let $\mathcal{H}$ be a hypothesis class with $\texttt{RepDim}(\mathcal{H}) < \infty$. In the agnostic setting, there exists an online learning algorithm that is ε-differentially private and achieves an expected regret of $O\left(\frac{\sqrt{T}\log T(\texttt{RepDim}(\mathcal{H})+\log\log T)}{\varepsilon}\right)$ with an oblivious adversary.*

Note that every online learning algorithm can be transformed to a PAC learner by the online-to-batch conversion [17]. Our result reveals that pure private online learnability against oblivious adversaries is equivalent to pure private PAC learnability in both realizable and agnostic settings.

https://doi.org/10.52202/079017-2086

3.2 Learning Against Adaptive Adversaries

We now turn to adaptive adversaries. For finite hypothesis classes, it is still feasible to employ techniques from DP-OPE [3] or DP continual observation [26, 18, 34] to devise online learning algorithms (in Appendix F, we give an algorithm with a better mistake bound in the realizable setting). One may hope that this can be extended to hypothesis class with finite representation dimension, as we did in the oblivious setting. However, it turns out that our method for oblivious adversaries is not applicable here. Since the examples are not fixed in advance, we cannot guarantee that the sampled hypothesis class V contains a consistent hypothesis. Moreover, the famous oblivious-to-adaptive transformation (see, e.g., [16]), which was used by Golowich and Livni [30] to construct online learners under approximate DP, also fails to give a sublinear mistake bound. This is because pure DP only has the basic composition property, which yields a mistake bound that scales linearly with T (for approximate DP, this can be improved to $\sqrt{T}$ by advanced composition). Therefore, it is not clear if every offline learnable hypothesis class can also be made online learnable against adaptive adversaries under pure DP.

We will show that this is an impossible mission. Let $\texttt{POINT}_d$ be the set of point functions over $[d]$ and $\texttt{POINT}_{\mathbb{N}}$ be the set of point functions over $\mathbb{N}$, where a point function $f_x : \mathcal{X} \to \{0, 1\}$ is a function that maps x to 1 and all other elements to 0. Both $\texttt{POINT}_d$ and $\texttt{POINT}_{\mathbb{N}}$ have a constant representation dimension and thus are offline learnable under pure DP [10]. In the rest of this section, we will prove that for any pure DP online learning algorithm for $\texttt{POINT}_d$, an adaptive adversary can force it to make $\Omega(\min(\log d, T))$ errors. As a direct corollary, $\texttt{POINT}_{\mathbb{N}}$ is not pure privately online learnable against adaptive adversaries.

We now illustrate the idea of our proof. Let us start by considering a simplified version, where the algorithm is constrained to be proper, i.e., $h_t \in \mathcal{H} = \texttt{POINT}_d$ for every $t \in [T]$. Then one can construct a series of data streams $S_i = \{(i, 1), \ldots, (i, 1)\}$ for every $i \in [d]$. An accurate proper learner must output f_i for most of the rounds. This allows us to use the packing argument to derive an $\Omega(\log d)$ lower bound for $T = \Theta(\log d)$.

However, the above argument does not apply to the general case where the learner may be improper since a learner can simply output an all-one function that makes 0 errors on each S_i. Therefore, we have to insert to S_i some examples of the form $(j, 0)$ where $j \neq i$. This prevents h_t from taking 1 on elements other than i. But when should we insert $(j, 0)$? And how do we determine the value of j?

Note that till now, we have not used the adversary's adaptivity. It is necessary to exploit this power since any oblivious construction can be solved by our algorithm in Theorem 3.3. When the adversary acts adaptively, the construction becomes a dual online learning game: in each round, the learner outputs h_t as a "data point" and the adversary chooses $(i, 1)$ or some $(j, 0)$ as the "hypothesis". This inspires us to leverage tools from online learning to construct the adversary.

We now sketch our idea. In each round, we choose $(i, 1)$ as the data point with probability $1/2$, and otherwise sample a $(j, 0)$ from some probability distribution. We maintain the distribution by the multiplicative update rule, which is a widely used method in online decision making. The weight of j is increased by a multiplicative factor whenever $h_t(j) = 1$, and the probability of selecting j is proportional to its weight. We provide a detailed implementation in Algorithm 1.

Using the standard argument of multiplicative update, we can show that an accurate learner must predict $h_t(i) = 1$ for most rounds and $h_t(j) = 1$ for very few rounds. This allows us to apply the packing argument to obtain the following hardness result.

Theorem 3.5. *Let $\varepsilon \leq O(1)$ and $d \geq 2$. Any ε-differentially private online learning algorithm for $\texttt{POINT}_d$ must incur a mistake bound of $\Omega(\min(\log d/\varepsilon, T))$ in the adaptive adversarial setting.*

Since $\texttt{POINT}_d$ is a subset of $\texttt{POINT}_{\mathbb{N}}$ for any d, the above result directly implies that $\texttt{POINT}_{\mathbb{N}}$ is not online learnable with adaptive adversaries under pure DP. This shows a strong separation between pure DP and approximate DP.

Corollary 3.6. *Let $\varepsilon \leq O(1)$. In the adaptive adversarial setting, any ε-differentially private online learning algorithm for $\texttt{POINT}_{\mathbb{N}}$ must make $\Omega(T)$ mistakes.*

https://doi.org/10.52202/079017-2086

Algorithm 1: Adaptive adversary for $\texttt{POINT}_d$

Input: the number of rounds T; online learning algorithm $\mathcal{A}$; input data stream S
Output: hypotheses $h_1, \ldots, h_T$ outputted by $\mathcal{A}$

$w_0(j) \leftarrow 1$ for all $j \in [d]$
for $t = 1, \ldots, T$ **do**
  $(x_t, y_t) \leftarrow S[t]$
  Set $p(j) \leftarrow \frac{w_{t-1}(j)}{\sum_{k \in [d] \setminus \{x_t\}} w_{t-1}(k)}$ for $j \in [d] \setminus \{x_t\}$
  With probability $1/2$, sample $j \sim p$ and set $(x_t, y_t) \leftarrow (j, 0)$
  Present (x_t, y_t) to $\mathcal{A}$ and receive h_t from $\mathcal{A}$
  Update $w_t(j) = w_{t-1}(j) \cdot e^{h_t(j)}$ for all $j \in [d]$
end
return $h_1, \ldots, h_T$

4 A General Lower Bound on the Number of Mistakes

In this section, we prove an $\Omega(\texttt{LD}(\mathcal{H}) \log T)$ lower bound on the number of mistakes made by any private learner for every hypothesis class $\mathcal{H}$ that contains a pair of non-complementary hypotheses.[3] This implies that as $T \to \infty$, any private algorithm will make an infinite number of mistakes. Note that without privacy, the Standard Optimal Algorithm always makes at most $\texttt{LD}(\mathcal{H})$ mistakes [39]. Thus, our lower bound reveals a universal separation between non-private and private models.

Our proof proceeds in two steps. We first show an $\Omega(\log T)$ lower bound in Section 4.1. Then based on this result, we prove the $\Omega(\texttt{LD}(\mathcal{H}) \log T)$ lower bound in Section 4.2.

4.1 A Lower Bound for Non-complementary Hypotheses

We first define the notion of complementary hypotheses.

Definition 4.1. We say two different hypotheses f_1 and f_2 over $\mathcal{X}$ are complementary if $f_1(x) = 1 - f_2(x)$ for all $x \in \mathcal{X}$. Otherwise we say they are non-complementary.

It is worth noticing the following important fact about non-complementary hypotheses, where the first item directly comes from the above definition and the second is because f_1 and f_2 are different.

Fact 4.2. *Let f_1 and f_2 be two different hypotheses over $\mathcal{X}$ that are non-complementary. Then:*

1. *There exists some $u_0 \in \mathcal{X}$ such that $f_1(u_0) = f_2(u_0)$;*
2. *There exists some $u_1 \in \mathcal{X}$ such that $f_1(u_1) \neq f_2(u_1)$.*

We remark that this fact is also used by Dmitriev et al. [22] to prove a lower bound (in their work, they call it a "distinguishing tuple"). However, they make a strong assumption that when running on a data stream containing $(u_0, f_1(u_0))$ only, with high probability, the algorithm predicts $h_t(u_1) = f_1(u_1)$ simultaneously for all $t \in [T]$. This largely weakens their bound since most DP algorithms clearly do not have such property.

To see how to use Fact 4.2, consider a hypothesis class that contains a pair of non-complementary hypotheses. We will focus on u_0, u_1 and f_1, f_2 only and ignore all other elements and hypotheses. In our proof, we will use $(u_0, f_1(u_0)) = (u_0, f_2(u_0))$ as a dummy input that provides no information about which hypothesis is correct. Let S_0 be a sequence that contains the dummy input only and $\mathcal{A}$ be an online learning algorithm. Without loss of generality, we can assume that $\Pr[\mathcal{A}(S_0)_t(u_1) = f_1(u_1)] \geq 1/2$ for all $t \in [T]$ (we can make this hold for half of the rounds by swapping f_1 and f_2, and ignore the rounds that it does not hold). We will insert $(u_1, f_2(u_1))$'s to make algorithm error.

Our proof relies on the classical packing argument. For ease of presentation, we only consider pure DP here, but the proof strategy easily extends to approximate DP via group privacy under approximate DP. In the framework of packing argument, we will construct a series of input sequences $S_1, \ldots, S_m$

[3] Such type of classes is equivalent to the notion of non-trivial classes in learning with data poisoning. See, e.g., [12] and [31].

https://doi.org/10.52202/079017-2086

from S_0 and disjoint subsets of output $O_1, \ldots, O_m$ such that S_0 and S_i differ by at most k elements for every $i \in [m]$, and any algorithm will make $\Omega(k)$ mistakes on S_i. Then by group privacy, for any ε-differentially private algorithm Alg we have

$$1 \geq \sum_{i=1}^{m} \Pr[\mathsf{Alg}(S_0) \in O_i] \geq e^{-k\varepsilon} \sum_{i=1}^{m} \Pr[\mathsf{Alg}(S_i) \in O_i].$$

Thus, a lower bound on $\Pr[\mathsf{Alg}(S_i) \in O_i]$ implies a lower bound on k by the above inequality.

The first challenge here is the construction of S_i. By our assumption, we can insert a $(u_1, f_2(u_1))$ at any position of S_0 to cause a loss of $1/2$. However, when inserting the second one, the loss may decrease by a multiplicative factor of e^{ε}. Following this argument, no matter how many $(u_1, f_2(u_1))$'s are inserted, we can only bound the expected number of mistakes by

$$\frac{1}{2}\left(1 + e^{-\varepsilon} + e^{-2\varepsilon} + \cdots\right) = constant,$$

failing to give an $\Omega(k)$ bound for $k = \log T$.

We overcome this by constructing them according to the given algorithm $\mathcal{A}$ instead of arbitrary algorithms. We will assume $\mathcal{A}$ has a mistake bound of $O(\log T)$ and seek to derive a contradiction. We now depict our construction. For S_1, we let $S = S_0$ be the initial data stream. We then go through every $t \in [T]$ in an increasing order, insert a $(u_1, f_2(u_1))$ at time-step t whenever $\Pr[\mathcal{A}(S)_t(u_1) = f_1(u_1)] \geq 1/3$, and let $S_1 = S$ at the end. By our assumption, the number of $(u_1, f_2(u_1))$'s should not exceed $k = 3 \cdot O(\log T)$. Hence, S_1 and S_0 differ by at most $k = O(\log T)$ points. Moreover, by our construction, for each $t \in [T]$ such that $S_1[t] = (u_0, f_1(u_0))$, we must have $\Pr[\mathcal{A}(S_1)_t(u_1) = f_1(u_1)] < 1/3$.

Now let us construct S_2. We find the earliest round t_1 such that $\Pr[\mathcal{A}(S_1)_{t_1}(u_1) = f_1(u_1)] < 1/3$. The property we mentioned above ensures the existence of such t_1 as long as $k < T$. We then perform a similar procedure as in the construction of S_1, but instead of starting from $t = 1$ and going over the entire time span $[T]$, we start from $t = t_1$. The online nature of $\mathcal{A}$ allows us to use t_1 to distinguish S_1 and S_2 (as well as $S_3, \ldots, S_m$, which we will construct later) since

$$\Pr[\mathcal{A}(S_1)_{t_1}(u_1) = f_1(u_1)] < 1/3 < 1/2 \leq \Pr[\mathcal{A}(S_2)_{t_1}(u_1) = f_1(u_1)].$$

In other words, $\mathcal{A}$ is more likely to predict $h_{t_1}(u_1) = f_1(u_1)$ on S_2 but is less likely to do so on S_1.

We repeat the construction for $i = 3, \ldots, m$. For each i, we first identify the minimal t_{i-1} such that $\Pr[\mathcal{A}(S_j)_{t_{i-1}}] < 1/3$ for every $j < i$. Then we insert $(u_1, f_2(u_1))$'s starting from $t = t_{i-1}$. By the same argument, t_{i-1} can be used to distinguish $S_1, \ldots, S_{i-1}$ and $S_i, \ldots, S_m$. We formally describe the construction procedure in Algorithm 2.

At the end, we will have m sequences $S_1, \ldots, S_m$ and $m - 1$ time-steps $t_1, \ldots, t_{m-1}$ such that $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] < 1/3$ for any $j \geq i$ and $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] \geq 1/2$ for any $j < i$. It can be proved that $m = \Omega(T/k)$, which is sufficiently large for $k = O(\log T)$. Now we run $\mathcal{A}$ on some $S = S_i$. Suppose we can figure out the index i, we can apply the packing argument to derive an $\Omega(\log m) = \Omega(\log T)$ lower bound.

Here comes the second challenge. Though we can use the output of $\mathcal{A}$ to estimate $\Pr[\mathcal{A}(S)_{t_j}(u_1) = f_1(u_1)]$ for a given j, we only have a constant success probability. To make the estimate accurate for every $j \in [m-1]$, one has to achieve a success probability of $1 - 1/m$ for each j. This requires running $\mathcal{A}$ for $O(\log m) = O(\log T)$ times and taking the average, which is prohibited since the resulting algorithm would be $O(\varepsilon \log T)$-DP, yielding a meaningless $\Omega(1)$ lower bound.

We address this issue by using binary search. We start with $\{t_1, \ldots, t_{m-1}\}$ and select the middle point t_{mid} in each iteration. By averaging over multiple copies of $\mathcal{A}(S)$, we can figure out whether we should go left or go right. This can be done in $O(\log m) = O(\log T)$ iterations, and we only require the decision made on each middle point to be correct. Thus, the number of independent copies can be reduced to $O(\log \log T)$, which leads to a lower bound of $\Omega(\log T / \log \log T)$.

The above approach is already sufficient to show an unbounded number of mistakes, but we can further refine our method to achieve an $\Omega(\log T)$ bound. The key observation here is that we do not need the probability of outputting i on S_i to be a constant. In fact, a success probability of $1/m^{1-\Omega(1)}$ is enough to obtain $k \geq \Omega(\log(m/m^{1-\Omega(1)})) = \Omega(\log T)$.

Algorithm 2: Constructing $S_0, S_1, \ldots, S_m$ and $t_1, \ldots, t_{m-1}$

Input: the number of rounds T; online learning algorithm $\mathcal{A}$; threshold k; f_1, f_2 and u_0, u_1
Output: a single data stream S_i, or a collection of m data streams $S_1, \ldots, S_m$ along with $m-1$ time-steps $t_1, \ldots, t_{m-1}$

$S_0 \leftarrow \{(u_0, f_1(u_0)), \ldots, (u_0, f_1(u_0))\}$
$m \leftarrow \lceil T/k \rceil$
for $i = 1, \ldots, m$ **do**
  $S_i \leftarrow S_0$
  Find the smallest t_{i-1} such that $\forall j \in [i-1], \Pr[\mathcal{A}(S_j)_{t_{i-1}}(u_1) = f_1(u_1)] < 1/3$
  for $t = t_{i-1}, \ldots, T$ **do**
    if $\Pr[\mathcal{A}(S_i)_t(u_1) = f_1(u_1)] \geq 1/3$ **then**
      $S_i[t] \leftarrow (u_1, f_2(u_1))$
    end
  end
  if $|\{t \in [T] : S_i[t] = (u_1, f_2(u_1))\}| > k$ **then**
    return S_i
  end
end
return $S_1, \ldots, S_m$ *and* $t_1, \ldots, t_{m-1}$

We thus "smooth" our binary search. In each iteration, instead of going left or right deterministically, we go to the side that is more likely to be correct with some probability $p > 1/2$. We show that, by choosing p appropriately, this approach will output i on S_i with probability $1/m^{1-\Omega(1)}$. Moreover, it only requires running the online learning algorithm $O(1)$ times, avoiding the $\log \log T$ blow-up of privacy parameters. The $\Omega(\log T)$ lower bound then follows by applying the packing argument. We illustrate this approach in Algorithm 3.

Theorem 4.3. *Let $c \in (0,1)$ be some constant. Suppose $\varepsilon \geq \ln T / T^{1-c}$ and $\delta \leq \varepsilon/T$. If $\mathcal{H}$ is a hypothesis class that contains two non-complementary hypotheses, then any (ε, δ)-differentially private online learning algorithm for $\mathcal{H}$ must incur a mistake bound of $\Omega(\log T/\varepsilon)$ even in the oblivious adversarial setting.*

One may ask whether the existence of a non-complementary pair is a necessary condition for the number of mistakes to be unbounded. Note that there are only two cases that $\mathcal{H}$ contains no non-complementary pairs: either $|\mathcal{H}| = 1$ or $\mathcal{H} = \{f_1, f_2\}$ such that $f_1 = 1 - f_2$. The former is definitely online learnable with zero mistakes. For the latter one, we give an algorithm with a finite expected mistake bound in Appendix F, showing that the condition is indeed necessary and sufficient.

Algorithm 3: Distinguishing $S_1, \ldots, S_m$

Input: the number of rounds T; online learning algorithm $\mathcal{A}$; time-steps $t_1, \ldots, t_{m-1}$; input data stream $S \in \{S_1, \ldots, S_m\}$; f_1, f_2 and u_0, u_1 used in Algorithm 2
Output: an index $i \in [m]$

Run $\mathcal{A}$ on S for 360 times, obtain 360 copies of output $\{h_1^{(w)}, \ldots, h_T^{(w)}\}$ for $w \in [360]$
$l \leftarrow 1, r \leftarrow m$
while $l < r$ **do**
  $mid \leftarrow \lfloor \frac{l+r}{2} \rfloor$
  if $|\{h_{t_{mid}}^{(w)}(u_1) = f_1(u_1) : w \in [360]\}| < 150$ **then**
    Let $r \leftarrow mid$ with probability $3/4$, and $l \leftarrow mid + 1$ otherwise
  else
    Let $l \leftarrow mid + 1$ with probability $3/4$, and $r \leftarrow mid$ otherwise
  end
end
return l

https://doi.org/10.52202/079017-2086

4.2 Incorporating the Littlestone Dimension

Building upon the $\Omega(\log T)$ lower bound, we are now ready to show an $\Omega(\texttt{LD}(\mathcal{H}) \log T)$ lower bound for general hypothesis classes. Let $\mathcal{A}$ be a private online learning algorithm for $\mathcal{H}$. Consider a shattered tree of depth $\texttt{LD}(\mathcal{H}) \geq 2$. Let u_0 denote its root and u_1 be its left child. By the definition of shattered tree, there exists $f_1, f_2 \in \mathcal{H}$ such that $f_1(u_0) = f_2(u_0) = 0$ and $0 = f_1(u_1) \neq f_2(u_1) = 1$. Note that f_1, f_2 and u_0, u_1 satisfy the property mentioned in Fact 4.2. We can thus apply Theorem 4.3 to find a sequence S_1 of length T' on which $\mathcal{A}$ makes $\Omega(\log T')$ mistakes.

Till now, only the true labels of u_0 and u_1 are revealed to the learner. Therefore, we can go into the corresponding subtree of u_1 and reiterate the above operation. After repeating it $\texttt{LD}(\mathcal{H})/2$ times, we obtain a series of completely non-overlapping sequences $S_1, \ldots, S_{\texttt{LD}(\mathcal{H})/2}$ and on any one of them $\mathcal{A}$ makes $\Omega(\log T')$ mistakes. By concatenating them together and setting $T' = T/\texttt{LD}(\mathcal{H})$, we arrive at the $\Omega(\texttt{LD}(\mathcal{H}) \log T)$ lower bound assuming $T > \texttt{LD}(\mathcal{H})^{1+c}$.

Theorem 4.4. *Let $c_1 \in (0, 1)$ and $c_2 > 0$ be two constants. Suppose $\varepsilon \geq \ln T / T^{(1-c_1)c_2/(1+c_2)}$ and $\delta \leq \varepsilon/T$. If $\mathcal{H}$ is a hypothesis class that contains two non-complementary hypotheses, then any (ε, δ)-differentially private online learning algorithm for $\mathcal{H}$ must incur a mistake bound of $\Omega(\texttt{LD}(\mathcal{H}) \log T/\varepsilon)$ even in the oblivious adversarial setting given that $T > \texttt{LD}(\mathcal{H})^{1+c_2}$.*

Note that the class of all hypotheses over $[d]$ has a Littlestone dimension of $\lfloor \log_2 d \rfloor$. The above theorem directly implies the following lower bound for the OPE problem. This improves the lower bound in [7] by a $\log T$ factor.

Corollary 4.5. *Let $c_1 \in (0, 1)$ and $c_2 > 0$ be two constants. Suppose $\varepsilon \geq \ln T / T^{(1-c_1)c_2/(1+c_2)}$ and $\delta \leq \varepsilon/T$. In the realizable setting, any (ε, δ)-differentially private algorithm for OPE has a regret of $\Omega(\log d \log T/\varepsilon)$ even against oblivious adversaries given that $T > \lfloor \log_2 d \rfloor^{1+c_2}$.*

4.3 Comparing to the Upper Bounds

We have shown an $\Omega_{\mathcal{H}}(\log T)$ lower bound on the number of mistakes made by any private online learning algorithm. We now compare it to existing upper bounds.

For pure DP, we provide an upper bound of $O_{\mathcal{H}}(\log^2 T \cdot (\log \log T)^2)$. This is larger than our lower bound by a factor of $\log T \cdot (\log \log T)^2$. In Appendix F, we show that $O_{\mathcal{H}}(\log T)$ is achievable for some specific hypothesis classes. Whether $O_{\mathcal{H}}(\log T)$ is attainable for generic hypothesis classes remains open.

For approximate DP, Golowich and Livni [30] proposed an algorithm with $O_{\mathcal{H}}(\log T)$ mistakes against oblivious adversaries. Thus, our lower bound is tight assuming a constant Littlestone dimension. However, their algorithm exhibits an $O_{\mathcal{H}}(\sqrt{T})$ upper bound against upper bound against adaptive adversaries. Whether this can also be improved to $O_{\mathcal{H}}(\log T)$ is an interesting open question.

Acknowledgments and Disclosure of Funding

The research was supported in part by an RGC RIF grant under the contract R6021-20, an RGC TRS grant under the contract T43-513/23N-2, RGC CRF grants under the contracts C7513, C7004-22G, C1029-22G and C6015-23G, and RGC GRF grants under the contracts 16200221, 16207922 and 16207423. The authors would like to thank the anonymous reviewers for their feedback and suggestions.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 23rd ACM Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] John M Abowd. The U.S. Census Bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 2867, 2018.

[3] Naman Agarwal and Karan Singh. The price of differential privacy for online learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 32–40, 2017.

[4] Noga Alon, Roi Livni, Maryanthe Malliaris, and Shay Moran. Private PAC learning implies finite littlestone dimension. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, pages 852–860, 2019.

[5] Noga Alon, Mark Bun, Roi Livni, Maryanthe Malliaris, and Shay Moran. Private and online learnability are equivalent. *Journal of the ACM*, 69(4):1–34, 2022.

[6] Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 2017.

[7] Hilal Asi, Vitaly Feldman, Tomer Koren, and Kunal Talwar. Near-optimal algorithms for private online optimization in the realizable regime. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 1107–1120, 2023.

[8] Hilal Asi, Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private online prediction from experts: Separations and faster rates. In *Proceedings of the 36th Annual Conference on Learning Theory*, volume 195, pages 674–699, 2023.

[9] Amos Beimel, Hai Brenner, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. Bounds on the sample complexity for private learning and private data release. *Machine learning*, 94(3): 401–437, 2014.

[10] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of pure private learners. *Journal of Machine Learning Research*, 20:1–33, 2019.

[11] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[12] Nader H Bshouty, Nadav Eiron, and Eyal Kushilevitz. Pac learning with nasty noise. *Theoretical Computer Science*, 288(2):255–275, 2002.

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

[14] Mark Bun, Cynthia Dwork, Guy N Rothblum, and Thomas Steinke. Composable and versatile privacy via truncated cdp. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing*, pages 74–86, 2018.

[15] Mark Bun, Roi Livni, and Shay Moran. An equivalence between private classification and online prediction. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science*, pages 389–402, 2020.

[16] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

[17] Nicolo Cesa-Bianchi, Alex Conconi, and Claudio Gentile. On the generalization ability of online learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

[18] T-H Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security*, 14(3):1–24, 2011.

[19] Kamalika Chaudhuri and Daniel Hsu. Sample complexity bounds for differentially private learning. In *Proceedings of the 24th Annual Conference on Learning Theory*, volume 19, pages 155–186, 2011.

[20] Edith Cohen, Xin Lyu, Jelani Nelson, Tamás Sarlós, and Uri Stemmer. Õptimal differentially private learning of thresholds and quasi-concave optimization. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 472–482, 2023.

 https://doi.org/10.52202/079017-2086

[21] Edith Cohen, Xin Lyu, Jelani Nelson, Tamás Sarlós, and Uri Stemmer. Lower bounds for differential privacy under continual observation and online threshold queries. In *Proceedings of the 37th Annual Conference on Learning Theory*, volume 247, pages 1200–1222, 2024.

[22] Daniil Dmitriev, Kristóf Szabó, and Amartya Sanyal. On the growth of mistakes in differentially private online learning: A lower bound perspective. In *Proceedings of the 37th Annual Conference on Learning Theory*, volume 247, pages 1379–1398, 2024.

[23] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Proceedings of the 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503, 2006.

[24] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography*, pages 265–284, 2006.

[25] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 381–390, 2009.

[26] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 715–724, 2010.

[27] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[28] Vitaly Feldman and David Xiao. Sample complexity bounds on differentially private learning via communication complexity. In *Proceedings of the 27th Annual Conference on Learning Theory*, volume 35, pages 1000–1019, 2014.

[29] Badih Ghazi, Noah Golowich, Ravi Kumar, and Pasin Manurangsi. Sample-efficient proper PAC learning with approximate differential privacy. In *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing*, pages 183–196, 2021.

[30] Noah Golowich and Roi Livni. Littlestone classes are privately online learnable. In *Advances in Neural Information Processing Systems*, volume 34, pages 11462–11473, 2021.

[31] Steve Hanneke, Amin Karbasi, Mohammad Mahmoody, Idan Mehalel, and Shay Moran. On optimal learning under targeted data poisoning. In *Advances in Neural Information Processing Systems*, volume 35, pages 30770–30782, 2022.

[32] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 705–714, 2010.

[33] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[34] Palak Jain, Sofya Raskhodnikova, Satchit Sivakumar, and Adam Smith. The price of differential privacy under continual observation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 14654–14678, 2023.

[35] Prateek Jain and Abhradeep Guha Thakurta. (Near) dimension independent risk bounds for differentially private learning. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 476–484, 2014.

[36] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23, pages 24.1–24.34, 2012.

[37] Haim Kaplan, Katrina Ligett, Yishay Mansour, Moni Naor, and Uri Stemmer. Privately learning thresholds: Closing the exponential gap. In *Proceedings of the 33rd Annual Conference on Learning Theory*, volume 125, pages 2263–2285, 2020.

[38] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[39] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2:285–318, 1988.

[40] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 94–103. IEEE Computer Society, 2007.

[41] Amartya Sanyal and Giorgia Ramponi. Open problem: Do you pay for privacy in online learning? In *Proceedings of the 35th Annual Conference on Learning Theory*, volume 178, pages 5633–5637, 2022.

[42] Adam Smith and Abhradeep Thakurta. (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In *Advances in Neural Information Processing Systems*, volume 26, pages 2733–2741, 2013.

[43] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[44] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, 16(2):264–280, 1971.

 https://doi.org/10.52202/079017-2086

A Discussion

In this work, we investigate online learning with differential privacy and provide separation results that distinguish non-private, pure private, and approximate private constraints. Below, we discuss some limitations and future work.

Tighter dependence on T under pure DP. Our algorithm for pure private online learning with oblivious adversaries exhibits a $\log^2 T \cdot (\log\log T)^2$ dependence on T (Theorem D.1). We also provide algorithms for $\texttt{POINT}_N$ (Theorem F.1) and $\texttt{Threshold}_d$ (Theorem F.3) that have a $O_{\mathcal{H}}(\log T)$ mistake bound. It is interesting to find out if a $\log T$ dependence is achievable for generic hypothesis classes.

Broader range of privacy parameters. Our $\Omega(\log T)$ lower bound requires $\delta < 1/T$ (Theorem 4.3) for constant ε, while the result in [21] only needs $\delta < 1/\log T$. Although it is a commonly accepted criterion to select $\delta = o(1/T)$, we still wonder whether our bound also holds for $\delta < 1/\log T$. Moreover, our results do not cover the cases where ε or T are extremely small. Is it possible to cover the entire range?

Mistake bound against stochastic adversaries. One benefit of online learning is that it does not require the data to be i.i.d. generated. But in some scenarios, we may still have i.i.d. data but have to make online predictions. Clearly, such *stochastic adversaries* are weaker than oblivious ones. Our construction in Algorithm 2 does not apply to stochastic adversaries. Can we overcome the $\Omega(\log T)$ barrier assuming stochastic adversaries?

Lower bound on learning with constant success probability. In Section 4.1, we show that the *expected* number of mistakes incurred by any algorithm is $\Omega(\log T)$. It is unclear whether the $\Omega(\log T)$ cost remains inevitable or a mistake bound of $o(\log T)$ can be achieved if we only require the learner to succeed with a *constant probability* (e.g., 0.99).

B Broader Impacts

Privacy has become a fundamental concern in today's machine learning community. In this work, we show several lower bounds and upper bounds on private online learning tasks. While our contributions are theoretical in nature, and we do not see any direct societal implications, we hope that our results will reveal the intrinsic structures of the problems and provide insights for the development of practical online learning algorithms with better privacy-utility trade-offs.

C Additional Preliminaries

Theorem C.1 (Hoeffding's Inequality [33])**.** *Let $Z_1, \ldots, Z_n$ be independent bounded random variables with $Z_i \in [a, b]$. Then*

$$\Pr\left[\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right] \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

for all $t \geq 0$.

The Laplace mechanism ensures privacy by adding Laplace noise.

Definition C.2 (Sensitivity)**.** Let $f : \mathcal{Z}^n \to \mathbb{R}$ be a function. The sensitivity of f is defined by

$$\Delta_f = \max_{S_1 \text{ and } S_2 \text{ differ in one entry}} |f(S_1) - f(S_2)|.$$

Definition C.3 (Laplace Distribution)**.** The Laplace distribution with parameter b and mean 0, denoted by $\mathrm{Lap}(b)$, is defined by the following probability density function:

$$f(x) = \frac{1}{2b}\exp(-|x|/b).$$

https://doi.org/10.52202/079017-2086

Lemma C.4 (The Laplace Mechanism [24]). *Let $r \sim \mathrm{Lap}(\Delta_f/\varepsilon)$ be a Laplace random variable. The algorithm that outputs $f(S) + r$ satisfies ε-differential privacy. Moreover, with probability $1-\beta$ it holds that $|r| \le \ln(1/\beta)\Delta_f/\varepsilon$.*

We also need the following AboveThreshold algorithm (aka the sparse vector technique) [25].

Algorithm 4: AboveThreshold

Input: database S; privacy parameter ε; threshold L; a series of online and adaptively chosen sensitivity-1 queries $q_1, \ldots$

Output: a stream of response $a_1, \ldots$

$\hat{L} \leftarrow L + \mathrm{Lap}(2/\varepsilon)$
for $i = 1, \ldots,$ **do**
  $\hat{q}_i \leftarrow q_i(S) + \mathrm{Lap}(4/\varepsilon)$
  if $\hat{q}_i \ge \hat{L}$ **then**
    $a_i \leftarrow \top$
    halt
  else
    $a_i \leftarrow \bot$
  end
end

Lemma C.5. *Algorithm 4 is ε-differently private.*

If we only want to identify a query with a large value instead of the values of all queries, the report-noisy-max mechanism gives a much better utility guarantee. It can be implemented by adding Laplace noise or directly applying the exponential mechanism [40].

Theorem C.6 (Report-Noisy-Max). *Let S be a database and $q_1, \ldots, q_d$ be d sensitivity-1 queries. There exists an ε-differentially private algorithm that outputs an index i such that*

$$q_i(S) \ge \max_{j \in [d]} q_j(S) - \frac{2(\ln(d) + \ln(1/\beta))}{\varepsilon}$$

with probability at least $1-\beta$.

The composition property allows us to combine multiple differentially private algorithms into one, even if they are executed adaptively.

Lemma C.7 (Basic Composition [23, 27]). *Let $\mathcal{A}_1 : \mathcal{Z}^n \to \mathcal{R}_1$ be an algorithm that satisfies $(\varepsilon_1, \delta_1)$-DP, and for $2 \le i \le k$ let $\mathcal{A}_i : \mathcal{R}_1 \times \cdots \times \mathcal{R}_{i-1} \times \mathcal{Z}^n \to \mathcal{R}_i$ be an algorithm that satisfies $(\varepsilon_i, \delta_i)$-DP for any given $(r_1, \ldots, r_{i-1}) \in \mathcal{R}_1 \times \cdots \mathcal{R}_{i-1}$. Let $\mathcal{A}$ be an algorithm that*

1. *Computes $r_1 \leftarrow \mathcal{A}_1(S)$;*
2. *For each $i = 2, \ldots, k$, computes $r_i \leftarrow \mathcal{A}_i(r_1, \ldots, r_{i-1}, S)$;*
3. *Outputs $r_1, \ldots, r_k$.*

Then $\mathcal{A}$ is $(\sum_{i=1}^k \varepsilon_i, \sum_{i=1}^k \delta_i)$-DP.

D Proofs from Section 3

D.1 Proof of Theorem 3.3

We use the following DP-OPE algorithm from [7].

Theorem D.1. *For any $0 < \beta < 1/2$, there exists an ε-differentially private algorithm such that with probability $1-\beta$ it has regret*

$$O\left(\frac{\log^2 d + \log(T/\beta)\log(d/\beta)}{\varepsilon}\right)$$

against oblivious adversaries in the realizable setting.

https://doi.org/10.52202/079017-2086

Proof of Theorem 3.3. Let $\alpha = \beta = 1/2T$. By Lemma 3.2, there exists an (α, β)-probabilistic representation of $\mathcal{H}$ with

$$\text{size}(\mathcal{P}) = O(\log(T) \cdot (\texttt{RepDim}(\mathcal{H}) + \log\log T)).$$

Let $S = \{(x_1, y_1), \ldots, (x_T, y_T)\}$ be the sequence chosen by the adversary and $\mathcal{D}_S$ be the empirical distribution of S, namely, $\Pr_{(x,y)\sim\mathcal{D}_S}[(x, y) = (x_t, y_t)] = 1/T$ for all $t \in [T]$. Then we have

$$\begin{aligned}\Pr_{V\sim\mathcal{P}}[\exists v \in V \ s.t. \ y_t = v(x_t) \ \forall t \in [T]] &= \Pr_{V\sim\mathcal{P}}[\exists v \in V \ s.t. \ \text{error}_{\mathcal{D}_S} \leq 1/\alpha = 1/2T] \\ &\geq 1 - \beta \\ &= 1 - 1/2T.\end{aligned}$$

Conditioning on this event, we then run the algorithm in Theorem D.1 with V being the set of experts and $\ell_t(v) = 1[v(x_t) \neq y_t]$. With probability $1 - 1/2T$, the number of mistakes is at most

$$O\left(\frac{\log^2|V| + \log T \log|V| + \log^2 T}{\varepsilon}\right) = O\left(\frac{\log^2 T(\texttt{RepDim}(\mathcal{H}) + \log\log T)^2}{\varepsilon}\right).$$

By the union bound, the expected number of mistakes is bounded by

$$1/T \cdot T + (1 - 1/T) \cdot O\left(\frac{\log^2 T(\texttt{RepDim}(\mathcal{H}) + \log\log T)^2}{\varepsilon}\right).$$

□

D.2 Proof of Theorem 3.4

The proof follows the same steps as the proof of Theorem 3.3. The only difference is that we use the following DP-OPE algorithm from [8] in the agnostic setting.

Theorem D.2. *There exists an ε-differentially private algorithm that has an expected regret of*

$$\mathbb{E}\left[\sum_{t=1}^{T} \ell_t(i_t) - \min_{i\in[d]} \sum_{t=1}^{T} \ell_t(i)\right] = O\left(\frac{\sqrt{T}\log d}{\varepsilon}\right)$$

against oblivious adversaries in the agnostic setting.

Proof of Theorem 3.4. We can use the same argument as in the proof of Theorem 3.3 to sample a hypothesis class V from $\mathcal{P}$ such that $\ln|V| = O(\log(T) \cdot (\texttt{RepDim}(\mathcal{H}) + \log\log T))$ and

$$\Pr_{V\sim\mathcal{P}}[\exists v \in V \ s.t. \ v(x_t) = h^\star(x_t)] \geq 1 - 1/2T,$$

where $h^\star = \text{argmin}_{h\in\mathcal{H}}\text{error}_{\mathcal{D}_S}(h)$ is a minimizer of the error on S. Running the algorithm in Theorem D.2 gives an expected regret of at most

$$(1 - 1/2T) \cdot O\left(\frac{\sqrt{T}\log|V|}{\varepsilon}\right) + T \cdot 1/2T = O\left(\frac{\sqrt{T}\log T(\texttt{RepDim}(\mathcal{H}) + \log\log T)}{\varepsilon}\right).$$

□

D.3 Proof of Theorem 3.5

We start with the following claim, which states that Algorithm 1 is an adversary that preserves privacy.

Claim D.3. *Suppose $\mathcal{A}$ satisfies ε-adaptive differential privacy. Let $\mathcal{B}$ be the algorithm that runs Algorithm 1 with $\mathcal{A}$. Then $\mathcal{B}$ is ε-differentially private.*

Proof. Let S_1 and S_2 be two input sequences that differ in only one entry. Consider an adversary Adv that runs Algorithm 1 on S_1 and S_2 simultanously using the same randomness, and interacts with $\mathcal{A}$ using S_b for some $b \in \{1, 2\}$. Let S_1' and S_2' be the resulting sequences. Note that when $S_1[t] = S_2[t]$, with the same randomness we have $S_1'[t] = S_2'[t]$ since the weights depend on $h_1, \ldots, h_{t-1}$ only. Thus, S_1' and S_2' also differ in only one entry. By the definition of adaptive differential privacy, it holds that for any event O,

$$\Pr[\mathcal{A} \circ \mathsf{Adv}(1) \in O] \leq e^\varepsilon \Pr[\mathcal{A} \circ \mathsf{Adv}(2) \in O].$$

The conclusion follows by observing that the output distributions of $\mathcal{A} \circ \mathsf{Adv}(b)$ and $\mathcal{B}(S_b)$ are identical. □

Proof of Theorem 3.5. Let $\varepsilon \leq 0.01$. We first consider $T \in [0.1 \ln d/\varepsilon, 0.2 \ln d/\varepsilon]$ and prove a lower bound of $\Omega(T) = \Omega(\log d/\varepsilon)$. To this end, we will assume that there exists an online learning algorithm $\mathcal{A}$ with an expected mistake bound of $0.01T$ that is ε-adaptive DP and derive a contradiction.

Let $\mathcal{B}$ be the algorithm that runs Algorithm 1 with $\mathcal{A}$. By Claim D.3, we know that $\mathcal{B}$ is ε-differentially private. Now suppose we are running $\mathcal{B}$ on $S_i = \{(i,1), \ldots, (i,1)\}$ for some $i \in [d]$. Let $c_t(j) = \sum_{r=1}^{t} h_r(j)$, $w_t(j) = e^{c_t(j)}$, $\Phi_t = \sum_{j \in [d] \setminus \{i\}} w_t(j)$, and $p_t(j) = w_t(j)/\Phi_t$. The expected number of mistakes made by $\mathcal{A}$ can be expressed as

$$\sum_{t=1}^{T} \mathbb{E}\left[\frac{1}{2} \cdot \mathbb{I}[h_t(i) = 0] + \frac{1}{2} \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) \cdot h_t(j)\right] \leq 0.01T. \tag{1}$$

Now consider the potential Φ_t. At the beginning, we have $\Phi_0 = d - 1$. We can upper bound Φ_t by

$$\begin{aligned}
\Phi_t &= \sum_{j \in [d] \setminus \{i\}} w_t(j) \\
&= \sum_{j \in [d] \setminus \{i\}} w_{t-1}(j) e^{h_t(j)} \\
&= \Phi_{t-1} \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) e^{h_t(j)} \\
&\leq \Phi_{t-1} \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j)(1 + 2h_t(j)) \\
&= \Phi_{t-1} \left(1 + 2 \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) h_t(j)\right) \\
&\leq \Phi_{t-1} \exp\left(2 \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) h_t(j)\right),
\end{aligned}$$

where we use $e^x \leq 1 + 2x$ for $0 \leq x \leq 1$ in the forth line and $1 + x \leq e^x$ for $x \in \mathbb{R}$ in the last line. Then it follows by induction that

$$\Phi_T \leq \Phi_0 \exp\left(2 \sum_{t=1}^{T} \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) h_t(j)\right).$$

Taking the logarithm on both sides, the linearity of expectation gives

$$\begin{aligned}
\mathbb{E}[\ln \Phi_T] &\leq \mathbb{E}\left[\ln \Phi_0 + 2 \sum_{t=1}^{T} \sum_{j \in [d] \setminus \{i\}} p_{t-1}(j) h_t(j)\right] \\
&= \ln(d-1) + 2 \sum_{t=1}^{T} \sum_{j \in [d] \setminus \{i\}} \mathbb{E}[p_{t-1}(j) h_t(j)] \\
&\leq \ln(d-1) + 0.04T \\
&\leq 0.14T,
\end{aligned}$$

where the third line is due to (1) and the last inequality uses the facts that $0.1 \ln d/\varepsilon \leq T$ and $\varepsilon \leq 0.01$.

By Markov's inequality, with probability at least $5/6$ we have $\ln \Phi_T \leq 0.84T$. This implies that, for every $j \neq i$, we have

$$c_T(j) = \ln w_T(j) \leq \ln \Phi_T \leq 0.84T.$$

We then bound $c_T(i)$. Note that by (1) we have

$$\mathbb{E}[T - c_T(i)] = \mathbb{E}\left[\sum_{t=1}^{T} \mathbb{I}[h_t(i) = 0]\right] \leq 0.02T.$$

 https://doi.org/10.52202/079017-2086

Applying Markov's inequality again, with probability at least $5/6$ we have $T - c_T(i) \le 0.12T$, or equivalently, $c_T(i) \ge 0.88T$. By the union bound, it holds with probability $2/3$ that $c_T(i) \ge 0.88T > 0.84T \ge c_T(j)$ for every $j \ne i$.

Let O_i be the event that $c_T(i) > c_T(j)$ for every $j \ne i$, then $O_1, \dots, O_d$ are disjoint. Then by group privacy,

$$1 \ge \sum_{i=1}^{d} \Pr[\mathcal{B}(S_1) \in O_i] \ge e^{-T\varepsilon} \sum_{i=1}^{d} \Pr[\mathcal{B}(S_i) \in O_i] \ge 2/3 \cdot de^{-T\varepsilon}.$$

Rearranging the inequality yields $T \ge (\ln d - \ln 1.5)/\varepsilon > 0.2 \ln d/\varepsilon$ when $d \ge 2$, a contradiction.

Now, let us deal with the remaining range. When $T > 0.2 \ln d/\varepsilon$, the algorithm must make $\Omega(\log d/\varepsilon)$ in the first $\lfloor 0.2 \ln d/\varepsilon \rfloor$ rounds. For $T < 0.1 \ln d/\varepsilon$, suppose $\mathcal{A}$ makes no more than $0.005T$ mistakes in expectation. Then we can repeatedly initiate $\mathcal{A}$ after every T round to obtain an algorithm that makes at most $0.005kT$ mistakes in expectation for a total of kT rounds, where

$$k = \left\lfloor \frac{0.2 \ln d/\varepsilon}{T} \right\rfloor \ge 0.5 \cdot \frac{0.2 \ln d/\varepsilon}{T} = \frac{0.1 \ln d/\varepsilon}{T}.$$

This contradicts our previous conclusion since $kT \in [0.1 \ln d/\varepsilon, 0.2 \ln d/\varepsilon]$. We thus obtain the $\Omega(T)$ lower bound as desired. □

E Proofs from Section 4

E.1 Proof of Theorem 4.3

We start by analyzing Algorithm 2. As we discussed in Section 4.1, it constructs a series of data sequences and a list of time-steps that can be used to distinguish the sequences. We formalize this in the following lemma.

Lemma E.1. *For any threshold k and online learning algorithm $\mathcal{A}$ such that $\Pr[\mathcal{A}(S_0)_t(u_1) = f_1(u_1)] \ge 1/2$ for all $t \in [T]$, where $S_0 = \{(u_0, f_1(u_0)), \dots, (u_0, f_1(u_0))\}$ and f_1, f_2, u_0, u_1 satisfy the property listed in Fact 4.2, Algorithm 2 either outputs an S on which $\mathcal{A}$ makes more than $k/3$ mistakes in expectation, or $S_1, \dots, S_m$ and $t_1, \dots, t_{m-1}$ such that*

1. *$m = \lceil T/k \rceil$.*
2. *For each $i \in [m]$, we have $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] < 1/3$ for all $j \ge i$ and $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] \ge 1/2$ for all $j \le i - 1$.*

Proof. If Algorithm 2 outputs a single $S = S_i$, it inserts at least $k + 1$ $(u_1, f_2(u_1))$'s to S_i. By our construction, each of them incurs a mistake with a probability of at least $1/3$. Therefore, the expected number of mistakes made by $\mathcal{A}$ on S_i is at least $(k + 1)/3 > k/3$. Now suppose Algorithm 2 does not output a single data stream, this means for every i, Algorithm 2 replaces at most k elements when constructing S_i from S_0.

For each $i \in [m]$, the algorithm will first find the minimal t_{i-1} such that $\Pr[\mathcal{A}(S_j)_{t_{i-1}}(u_1) = f_1(u_1)] < 1/3$ for all $1 \le j \le i - 1$, which is exactly the first half of item 2. Moreover, this also suggests that $t_1 \le \cdots \le t_m$. Due to the construction process, the first t_{i-1} entries of S_i will be the same as S_0. Therefore, we have $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] \ge 1/2$ for any $1 \le j \le i - 1$ since $\mathcal{A}$ is an online algorithm. This proves the second half of item 2.

Now it remains to show the construction actually runs for $\lceil T/k \rceil$ rounds, i.e., we can find t_{i-1} for any $i \le \lceil T/k \rceil$. Since S_0 and any one of $S_1, \dots, S_{i-1}$ differ at most k entries, the pigeonhole principle suggests that there exists some $t \in [(i - 1)k + 1]$ such that $S_1[t] = \cdots = S_{i-1}[t] = (u_0, f_1(u_0))$. We next prove that $t > t_j$ for every $1 \le j < i - 1$. For the sake of contradiction, assume that j is the smallest index such that $t_j > t$ and $j < i - 1$ (it is impossible for t to be equal to any t_j since we always have $S_{j+1}[t_j] = (u_1, f_2(u_1))$ by our construction). Then we have $t_1 \le \cdots \le t_{j-1} < t$. This implies $\Pr[\mathcal{A}(S_l)_t(u_1) = f_1(u_1)] < 1/3$ for every $l \le j$, otherwise Algorithm 2 will put a $(u_1, f_2(u_1))$ at time-step t when constructing S_l. By the minimality of t_j, we should choose t_j to be t when constructing S_{j+1}, contradicting our assumption that $t_j > t$.

Therefore, we have $t > t_j$ for every $1 \le j < i-1$. As we just argued, it follows that $\Pr[\mathcal{A}(S_l)_t(u_1) = f_1(u_1)] < 1/3$ for all $l \in [i-1]$ otherwise Algorithm 2 will put a $(u_1, f_2(u_1))$ at this location. Thus, we have $t_{i-1} \le t \le (i-1)k+1 \le (\lceil T/k \rceil - 1)k+1 \le T$ by the minimality of t_{i-1}. This completes the proof. □

The following lemma suggests that the condition in the Lemma E.1 can be assumed without loss of generality.

Lemma E.2. *Given f_1, f_2 and u_0, u_1 as in Fact 4.2. Let $\mathcal{A}$ be a randomized online learning algorithm and $S_0 = \{(u_0, f_1(u_0)), \ldots, (u_0, f_1(u_0))\}$. There exists $b \in \{1, 2\}$ such that*

$$|\{t : \Pr[\mathcal{A}(S_0)_t(u_1) = f_b(u_1)] \ge 1/2\}| \ge T/2.$$

Proof. Let $n_b = |\{t : \Pr[\mathcal{A}(S_0)_t(u_1) = f_b(u_1)] \ge 1/2\}|$ for $b \in \{1, 2\}$. We have $n_1 + n_2 \ge T$. Therefore, either $n_1 \ge T/2$ or $n_2 \ge T/2$. □

Proof of Theorem 4.3. Let $\mathcal{A}$ be an (ε, δ)-differentially private online learning algorithm and $S_0 = \{(u_0, f_1(u_0)), \ldots, (u_0, f_1(u_0))\}$. By Lemma E.2, we can without loss of generality assume that $\Pr[\mathcal{A}(S_0)_t(u_1) = f_1(u_1)] \ge 1/2$ for all $t \in [T]$ by ignoring all time-steps that do not have such property.

Let $k = \lfloor c \ln T/(3600\varepsilon) \rfloor$. We will assume that $\mathcal{A}$ makes no more than $k/3$ mistakes in expectation and derive a contradiction. By Lemma E.1, Algorithm 2 will output $S_1, \ldots, S_m$ and $t_1, \ldots, t_{m-1}$ with $m = \lceil T/k \rceil$. Moreover, $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] < 1/3$ for every $j \ge i$ and $\Pr[\mathcal{A}(S_i)_{t_j}(u_1) = f_1(u_1)] \ge 1/2$ for every $j \le i-1$.

Let $\mathcal{B}$ be the algorithm that runs Algorithm 3 with $\mathcal{A}$. The basic composition property immediately guarantees that $\mathcal{B}$ is $(360\varepsilon, 360\delta)$-differentially private.

We now examine the probability that $\mathcal{B}$ outputs i on S_i. If so, $\mathcal{B}$ must go through a series of time-step $t_{i_1}, \ldots, t_{i_n}$ in the binary search, where $n \le \lceil \log_2 m \rceil \le \log_2 m + 1 = \log_2(2m)$. For each t_{i_j}, let X_j be a random variable that takes 1 if

$$\mathbb{I}\left[\left|\left\{h^{(w)}_{t_{i_j}}(u_1) = f_1(u_1) : w \in [360]\right\}\right| < 150\right] = \mathbb{I}\left[i_j \ge i\right].$$

and takes 0 otherwise. Conditioning on X_j's, the probability that $\mathcal{B}$ outputs i can be expressed as

$$\Pr[\mathcal{B}(S_i) = i \mid X_1, \ldots, X_n] = \Pi_{j=1}^{n} \left(0.75X_j + 0.25(1 - X_j)\right) = 0.75^{\sum_{j=1}^{n} X_j} 0.25^{n - \sum_{j=1}^{n} X_j}.$$

By Hoeffding's inequality, for each X_j we have

$$\mathbb{E}[X_j] = \Pr[X_j = 1] \ge 1 - \exp\left(-2 \cdot 360 \cdot (30/360)^2\right) = 1 - e^{-5} > 0.99.$$

It then follows by the linearity of expectation and Markov's inequality that

$$\Pr[X_1 + \cdots + X_n \ge 0.97n] \ge 2/3.$$

Putting it all together gives

$$\begin{aligned}
\Pr[\mathcal{B}(S_i) = i] &\ge \Pr[\mathcal{B}(S_i) = i \mid X_1 + \cdots + X_n \ge 0.97n] \cdot \Pr[X_1 + \cdots + X_n \ge 0.97n] \\
&\ge 2/3 \cdot 0.75^{0.97n} \cdot 0.25^{0.03n} \\
&\ge 2/3 \cdot \left(\frac{1}{2} \cdot 2^{1/2}\right)^{0.97n} \cdot \left(\frac{1}{2} \cdot 2^{-1}\right)^{0.03n} \\
&= 2/3 \cdot 2^{-n} \cdot 2^{(0.97/2 - 0.03)n} \\
&= 2/3 \cdot 2^{-0.545n} \\
&\ge 2/3 \cdot (2m)^{-0.545}.
\end{aligned}$$

 https://doi.org/10.52202/079017-2086

We now apply the packing argument. Since $\mathcal{B}$ is $(360\varepsilon, 360\delta)$-differentially private, it follows by group privacy that

$$\begin{aligned}
1 &= \sum_{i=1}^{m} \Pr[\mathcal{B}(S_0) = i] \\
&\geq \sum_{i=1}^{m} \left(e^{-360k\varepsilon} \Pr[\mathcal{B}(S_i) = i] - \frac{1 - e^{-360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta \right) \\
&\geq 2/3 \cdot 2^{-0.545} \cdot m^{0.455} \cdot e^{-360k\varepsilon} - m \cdot \frac{1 - e^{-360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta \\
&\geq 0.45 m^{0.455} \cdot e^{-360k\varepsilon} - m \cdot \frac{1 - e^{-360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta.
\end{aligned}$$

If $0.45 m^{0.455} \cdot e^{-360k\varepsilon} \geq 2m \cdot \frac{1 - e^{-360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta$, then we have

$$0.225 m^{0.455} \cdot e^{-360k\varepsilon} \leq 0.45 m^{0.455} \cdot e^{-360k\varepsilon} - m \cdot \frac{1 - e^{360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta \leq 1.$$

Rearranging the above gives

$$\begin{aligned}
k &\geq \frac{\ln 0.225 + 0.455 \ln m}{360\varepsilon} \\
&\geq \frac{\ln 0.225 + 0.455 \ln(T/k)}{360\varepsilon} \\
&\geq \frac{\ln 0.225 + 0.455 \ln\left(\frac{3600T\varepsilon}{c \ln T}\right)}{360\varepsilon} \\
&\geq \frac{\ln 0.225 + 0.455c \ln T + 0.455 \ln(3600/c)}{360\varepsilon} \\
&\geq \frac{0.455c \ln T}{360\varepsilon},
\end{aligned}$$

where in the forth inequality we use the condition $\varepsilon \geq \ln T / T^{1-c}$. This contradicts our assumption that $k \leq c \ln T / 3600\varepsilon$.

If otherwise $0.45 m^{0.455} \cdot e^{-360k\varepsilon} < 2m \cdot \frac{1 - e^{-360k\varepsilon}}{e^{360\varepsilon} - 1} \cdot 360\delta$, we have

$$\begin{aligned}
360k\varepsilon &> \ln\left(\frac{e^{360\varepsilon} - 1}{360\delta} \cdot 0.225 m^{-0.545} + 1 \right) \\
&> \ln\left(\frac{e^{360\varepsilon} - 1}{360\delta} \right) + \ln 0.225 - 0.545 \ln m \\
&\geq \ln(\varepsilon/\delta) - 0.545 \ln T + \ln 0.225 \\
&\geq 0.455 \ln T + \ln 0.225 \\
&\geq 0.1 \ln T
\end{aligned}$$

when $T \geq 100$. Then it follows that $k > 0.1 \ln T / (360\varepsilon) > c \ln T / 3600\varepsilon$, again a contradiction.

In conclusion, any (ε, δ)-differentially private algorithm makes $(k+1)/3 \geq c \ln T / (10800\varepsilon)$ mistakes in expectation when $T \geq 100$. This gives the $\Omega(\log T / \varepsilon)$ lower bound. □

E.2 Proof of Theorem 4.4

Proof. When $\texttt{LD}(\mathcal{H}) = 1$, the result is directly implied by Theorem 4.3. From now on, we will assume $\texttt{LD}(\mathcal{H}) \geq 2$.

Let $m = \lfloor \texttt{LD}(\mathcal{H})/2 \rfloor$ and $T' = \lfloor T/m \rfloor \geq \lfloor 2T/\texttt{LD}(\mathcal{H}) \rfloor \geq T/\texttt{LD}(\mathcal{H}) > T^{1-1/(1+c_2)} = T^{c_2/(1+c_2)}$. Pick a shattered tree for $\mathcal{H}$ of depth $\texttt{LD}(\mathcal{H})$. Let u_0 be its root and u_1 be the left child of u_0. The definition of shattered tree indicates that there exists $f_1, f_2 \in \mathcal{H}$ such that $f_1(u_0) = f_2(u_0) = 0$ and $0 = f_1(u_1) \neq f_2(u_1) = 1$. By Theorem 4.3, for any (ε, δ)-differentially private online learner with

$\varepsilon \geq \ln T/T^{c_2(1-c_1)/(1+c_2)} > \ln T'/T'^{1-c_1}$ and $\delta \leq \varepsilon/T < \varepsilon/T'$, we can construct a sequence S_1 such that it makes in expectation $\Omega(\log T'/\varepsilon) = \Omega(\log T/\varepsilon)$ mistakes on S_1. Moreover, S_1 only contains $(u_0, 0)$ and (u_1, b) for some $b \in \{0, 1\}$.

Let $\mathcal{H}' = \{h \in \mathcal{H} : h(u_0) = 0 \text{ and } h(u_1) = b\}$. If $\texttt{LD}(\mathcal{H}) \geq 4$, we then have $\texttt{LD}(\mathcal{H}') \geq \texttt{LD}(\mathcal{H}) - 2 \geq 2$ since the subtree rooted at the child (left if $b = 0$, right if $b = 1$) of u_1 is shattered by $\mathcal{H}'$ and has a depth of $\texttt{LD}(\mathcal{H}) - 2$. Thus, we can similarly construct a sequence S_2 such that the algorithm makes $\Omega(\log T/\varepsilon)$ errors. Importantly, the entries in S_2 are completely different from those in S_1.

We repeat the above process until we reach a hypothesis class with Littlestone dimension ≤ 1 and construct a series of sequences $S_1, \ldots, S_m$ that have non-overlapping entries. On each of them, the learner makes $\Omega(\log T/\varepsilon)$ mistakes in expectation. Let S be the stream formed by concatenating $S_1, \ldots, S_m$ together. The length of S is at most $mT' \leq T$ while the expected number of mistakes on S is $\Omega(\texttt{LD}(\mathcal{H}) \log T)$. □

F Additional Algorithms

In this section, we provide several algorithms for various tasks. Note that some of the upper bounds hold even against strong adaptive adversaries, i.e., adversaries that can see the prediction made by the learner in the current round.

F.1 Learning Point Functions Against Oblivious Adversaries

We show how to improve the $\log^2 T$ dependence in Theorem 3.3 to $\log T$ for $\texttt{POINT}_{\mathbb{N}}$. In the beginning, we keep outputting an all-zero function and use the sparse vector technique to monitor the mistakes. Then we sample hypotheses from the probabilistic representation constructed in [10] and again apply the sparse vector technique to find one with low error on the past data. Following their argument, we can show that the construction guarantees that the hypothesis we found won't make too many mistakes in the future. The result is stated as follows.

Theorem F.1. *In the realizable setting, there is an ε-differentially private online learning algorithm for $\texttt{POINT}_{\mathbb{N}}$ that makes in expectation $O(\log T/\varepsilon)$ mistakes against oblivious adversaries.*

Proof. We run Algorithm 5. Note that it is in fact composed of two instances of AboveThreshold with privacy parameter $\varepsilon/2$, the algorithm is ε-DP.

Consider the first AboveThreshold, there are $T+1 \leq 2T$ Laplace random variables. With probability $1 - 1/3T$, all of them have absolute values no larger than $4\ln(6T^2)/\varepsilon'$. Thus, when it halts, there are at least $\hat{L} - 4\ln(6T^2)/\varepsilon' \geq 20\ln(12T^3)/\varepsilon'$ data points of the form $(x^\star, 1)$, where $f_{x^\star}$ is the target hypothesis. Moreover, it makes at most $20\ln(12T^3)/\varepsilon' + 8\ln(6T^2)/\varepsilon' + 8\ln(6T^2)/\varepsilon' + 1$ mistakes.

Now we show that the sampling won't last for too long. We consider the first $3T^2$ iterations. Then with probability $1 - 1/3T$, every random noise has an amplitude of at most $4\ln(12T^3)/\varepsilon'$ in the second AboveThreshold. Note that $\sum_{r=1}^{t} \mathbb{I}[h(x_r) \neq y_r \text{ and } y_r = 1] \geq 20\ln(12T^3)/\varepsilon'$ if $h(x^\star) = 0$ and is 0 if $h(x^\star) = 1$. Thus, the algorithm will exit the loop if and only if $h(x^\star) = 1$. The probability that this happens in the first $3T^2$ iterations is at least $1 - (1 - 1/T)^{3T^2} \geq 1 - e^{-3T} \geq 1 - 1/3T$. Moreover, this h makes in expectation less than $(T-1)/T < 1$ mistakes on data points other than $(x^\star, 1)$ in the sequence.

Putting it all together, the expected number of mistakes is less than

$$1/T \cdot T + (1 - 1/T) \cdot (20\ln(12T^3)/\varepsilon' + 8\ln(6T^2)/\varepsilon' + 8\ln(6T^2)/\varepsilon' + 1 + 1) = O(\log T/\varepsilon).$$

□

F.2 Learning Point Functions Against Adaptive Adversaries

Theorem F.2. *In the realizable setting, there is an ε-differentially private online learning algorithm for $\texttt{POINT}_d$ that makes in expectation $O((\log d + \log T)/\varepsilon)$ mistakes against strong adaptive adversaries.*

https://doi.org/10.52202/079017-2086

Algorithm 5: Learning point functions over $\mathbb{N}$

Input: the number of rounds T; privacy parameter ε; sequence S
Output: hypotheses $h_1, \ldots, h_T$

$\varepsilon' \leftarrow \varepsilon/2$
$\hat{L} \leftarrow 20\ln(12T^3)/\varepsilon' + 8\ln(6T^2)/\varepsilon' + \mathrm{Lap}(2/\varepsilon')$
for $t = 1, \ldots, T$ **do**
  Let h_t be an all-zero function such that $h_t(x) = 0$ for all $x \in \mathbb{N}$
  $(x_t, y_t) \leftarrow S[t]$
  if $\sum_{r=1}^{t} y_r + \mathrm{Lap}(4/\varepsilon') \geq \hat{L}$ **then**
    $\hat{R} \leftarrow 12\ln(12T^3)/\varepsilon' + \mathrm{Lap}(2/\varepsilon')$
    repeat
      Sample an h such that $\Pr[h(x) = 1] = 1/T$ for every $x \in \mathbb{N}$ independently
    until $\sum_{r=1}^{t} \mathbb{I}[h(x_r) \neq y_r \textit{ and } y_r = 1] + \mathrm{Lap}(4/\varepsilon') \leq \hat{R}$
    Output $h_{t+1} = \cdots = h_T = h$ for the remaining rounds and halt
  end
end

Proof. Let $\varepsilon' = \varepsilon/2$. The algorithm works as follows: it keeps outputting an all-zero function and runs an ε'-differentially private sparse vector technique (Algorithm 4) with $L = 3(\ln d + \ln(2T))/\varepsilon' + 8\ln(4T^2)/\varepsilon'$ to monitor the number of mistakes. Once the (noisy) number of mistakes exceeds $\hat{L}$ at round k, it computes $c_i = |\{t \in [k] : (x_t, y_t) = (i, 1)\}|$ and apply the report-noisy-max mechanism with privacy parameter ε' to find an index i with a large c_i. After that, it persistently outputs f_i till the end. The privacy directly follows from the basic composition.

With probability $1 - 1/2T$, the amplitude of every noise added in the sparse vector technique is no larger than $4\ln(4T^2)/\varepsilon'$. Thus, at round k, the number of mistakes must be in the range $[\hat{L} - 4\ln(4T^2)/\varepsilon', \hat{L} + 4\ln(4T^2)/\varepsilon' + 1] \subseteq [L - 8\ln(4T^2)/\varepsilon', L + 8\ln(4T^2)/\varepsilon' + 1]$. Hence, we have $c_i \geq L - 8\ln(4T^2)/\varepsilon' = 3(\ln d + \ln(2T))/\varepsilon'$ for some i and $c_j = 0$ for all $j \neq i$. With probability $1 - 1/2T$, the report-noisy-max algorithm will identify i correctly.

Thus, the expected number of mistakes is bounded by

$$1/T \cdot T + (1 - 1/T) \cdot (L + 8\ln(4T^2)/\varepsilon' + 1) = O((\log d + \log T)/\varepsilon).$$

□

F.3 Learning Threshold Functions

A threshold function over $[d]$ is a function f_i such that $f_i(x) = 0$ for all $x \leq i$ and $f_i(x) = 1$ for all $x > i$, where $i \in [d] \cup \{0\}$. The class of threshold functions over $[d]$, denote by $\texttt{Threshold}_d$, is the set $\{f_i : i \in [d] \cup \{0\}\}$.

Theorem F.3. *In the realizable setting, there is an ε-differentially private online learning algorithm for $\texttt{Threshold}_d$ that makes in expectation $O(\log d \log T/\varepsilon)$ mistakes against strong adaptive adversaries.*

Proof. We run Algorithm 6. Since the counters are refreshed once we switch the current hypothesis, we are actually running AboveThreshold on disjoint datasets. Moreover, the comparisons of c_0 and c_1 are also performed on disjoint datasets. Therefore, the overall algorithm is ε-DP.

Since the binary search runs for at most $\lceil \log_2(d+1) \rceil$ iterations, it follows by the privacy of AboveThreshold and the basic composition that the overall algorithm is ε-DP since the counters are refreshed once we switch the current hypothesis.

By the property of Laplace distribution and the union bound, with probability $1 - 1/T$, every random noise that appears in the algorithm has an amplitude no larger than $4\ln(4T^2)/\varepsilon'$. Conditioning on this event, once we change the current hypothesis, we must have $c_0 + c_1 \geq \hat{L} - 4\ln(4T^2)/\varepsilon' \geq 8\ln(4T^2)/\varepsilon'$ and $c_0 + c_1 \leq \hat{L} + 4\ln(4T^2)/\varepsilon' + 1 \leq 24\ln(4T^2)/\varepsilon' + 1$. Therefore, we can identify

which one is zero. Since the binary search runs at most $O(\log d)$ iterations, we will make at most

$$O(\log d) \cdot (24 \ln(4T^2)/\varepsilon' + 1) = O(\log d \log T/\varepsilon)$$

mistakes with probability $1 - 1/T$. This implies an $1/T \cdot T + O(\log d \log T/\varepsilon) = O(\log d \log T/\varepsilon)$ bound in expectation. □

Algorithm 6: Learning threshold functions over $[d]$

Input: the number of rounds T; privacy parameter ε; sequence S
Output: hypotheses $h_1, \ldots, h_T$

$\varepsilon' = \varepsilon/2$
$\hat{L} \leftarrow 16 \ln(4T^2)/\varepsilon' + \mathrm{Lap}(2/\varepsilon')$
$l \leftarrow 0, r \leftarrow d, mid \leftarrow \lfloor (l+r)/2 \rfloor$
$c_0 \leftarrow 0, c_1 \leftarrow 0$
for $t = 1, \ldots, T$ **do**
  $h_t \leftarrow f_{mid}$
  $(x_t, y_t) \leftarrow S[t]$
  $c_{y_t} \leftarrow c_{y_t} + \mathbb{I}[h(x_t) \neq y_t]$
  if $l < r$ *and* $c_0 + c_1 + \mathrm{Lap}(4/\varepsilon') \geq \hat{L}$ **then**
    if $c_0 + \mathrm{Lap}(1/\varepsilon') > c_1$ **then**
      $l \leftarrow mid + 1$
    else
      $r \leftarrow mid - 1$
    end
    $mid \leftarrow \lfloor (l+r)/2 \rfloor$
    $c_1 \leftarrow 0, c_2 \leftarrow 0$
    $\hat{L} \leftarrow 16 \ln(4T^2)/\varepsilon' + \mathrm{Lap}(2/\varepsilon')$
  end
end

F.4 Online Prediction from Experts Against Adaptive Adversaries

It is easy to come up with an algorithm with a regret of $O(d \log T/\varepsilon)$ even against strong adaptive adversaries. The idea is similar to Algorithm 6. The only difference is that we try the experts one by one instead of running a binary search.

Theorem F.4. *There is an ε-differentially private algorithm that solves the OPE problem with an expected regret of $O\left(\frac{d \log T}{\varepsilon}\right)$ even against strong adaptive adversaries in the realizable setting.*

Proof. We iterate over the set of experts. For each expert, we keep choosing it and run an ε-differentially private sparse vector technique to monitor the loss incurred by the current expert. Once the sparse vector technique halts, we switch to the next expert and restart the sparse vector technique. Since all instances of the sparse vector technique are run on disjoint data, the entire algorithm is ε-DP.

With probability $1 - 1/T$, all the Laplace noises added are bounded by $4 \ln(2T^2)/\varepsilon$. By choosing $L = 9 \ln(2T^2)/\varepsilon$, we will make at most $O(\log T/\varepsilon)$ mistakes on each expert. Moreover, for an expert that makes no errors, we will not switch to the next one. Therefore, the overall expected regret is

$$1/T \cdot T + (1 - 1/T) \cdot d \cdot O(\log T/\varepsilon) = O(d \log T/\varepsilon).$$

□

Note that there is a multiplicative factor of d on the $\log T$ term. We will then show how to improve this dependence to $\log d$ for (weak) adaptive adversaries. Due to Corollary 4.5, such a dependence is tight even for oblivious adversaries. It is interesting to find out if this can also be achieved for strong adaptive adversaries, and we leave this as future work.

https://doi.org/10.52202/079017-2086

Moreover, our algorithm has a regret of $O\left(d\log^2 d + \log d \log T\right)$. Since it was shown by Asi et al. [8] that an $\Omega(d)$ cost is inevitable, our algorithm is near optimal.

The idea is to select a uniformly random expert rather than keep choosing a fixed one. Suppose we are at the beginning. Let $c_t(i) = \sum_{r=1}^{t} \ell_r(i)$ for expert i. The benefit of such random selection is that it only incurs a loss of $\sum_{i\in[d]} c_t(i)/d$, which is much less than the $\sum_{i\in[d]} c_t(i)$ loss if we always choose the same expert since the adversary can let this expert make an error all the time.

We use the sparse vector technique to track the maximum of c_t. Once it exceeds $O(\log T + d\log d)$, we then apply the report-noisy-max algorithm to remove every expert j with $a_t(j) = \sum_{r=1}^{t} \ell_r(j) \geq O(d\log d)$ (we do not write c_t here since we will reset c_t to be 0 later). After that, we reiterate the sampling and monitoring process using the remaining experts. Observe that if expert j is the i-th one being removed, the loss incurred by j is at most $O(\log T + d\log d)/i$. This gives an upper bound of $O(\log T + d\log d)\cdot(1 + 1/2 + \cdots + 1/d) = O(d\log^2 d + \log d\log T)$. The details are depicted in Algorithm 7.

Algorithm 7: DP-OPE against adaptive adversaries

Input: the number of rounds T; privacy parameter ε; sequence S
Output: indices of experts $i_1, \ldots, i_T$

Set $a_0(i) \leftarrow 0$ and $c_0(i) \leftarrow 0$ for all $i \in [d]$
$E \leftarrow [d]$
$\varepsilon_1 = \varepsilon/2$, $\varepsilon_2 = \varepsilon/(6d)$
$\hat{L} \leftarrow 8\ln(2T^2)/\varepsilon_1 + 3(\ln d + \ln(3d^2))/\varepsilon_2 + \mathrm{Lap}(2/\varepsilon_1)$
for $t = 1, \ldots, T$ **do**
    Sample i_t uniformly from E
    $\ell_t \leftarrow S[t]$
    Update $a_t(i) \leftarrow a_{t-1}(i) + \ell_t(i)$ and $c_t(i) \leftarrow c_{t-1}(i) + \ell_t(i)$ for all $i \in [d]$
    if $|E| > 1$ *and* $\max_{i\in E} c_t(i) + \mathrm{Lap}(4/\varepsilon_1) \geq \hat{L}$ **then**
        Run report-noisy-max with privacy parameter ε_2 on $\{a_t(j)\}_{j\in E}$ and obtain some index i
        Update $E \leftarrow E \setminus \{i\}$
        while $i \notin E$ *and* $|E| > 1$ **do**
            Run report-noisy-max with privacy parameter ε_2 on $\{a_t(j)\}_{j\in E}$ and obtain some index i
            if $a_t(i) + \mathrm{Lap}(1/\varepsilon_2) > \ln(3d^2)/\varepsilon_2$ **then**
                Update $E \leftarrow E \setminus \{i\}$
            end
        end
        $\hat{L} \leftarrow 8\ln(2T^2)/\varepsilon_1 + 3(\ln d + \ln(3d^2))/\varepsilon_2 + \mathrm{Lap}(2/\varepsilon_1)$
        $c_t(i) \leftarrow 0$ for all $i \in [d]$
    end
end

Here is a subtle issue: the report-noisy-max mechanism only succeeds with probability $1 - 1/d$. This does not imply an upper bound in expectation. We cannot simply raise the success probability to $1 - 1/T$. Otherwise, the $d\log^2 d$ term will become $d\log d\log T$, which is even higher than the brute force. To fix this, we run an extra sparse vector technique on top of the algorithm to inspect the loss. Once the loss is greater than the upper bound, we then run the algorithm in Theorem F.4 for the rest of the rounds. This reduces the expected cost from T to $O(d\log T)$ when it fails, hence successfully bound the expected loss.

Theorem F.5. *There is an ε-differentially private algorithm that solves the OPE problem with an expected regret of $O\left(\frac{d\log^2 d + \log d\log T}{\varepsilon}\right)$ against adaptive adversaries in the realizable setting.*

Proof. We first show the privacy and utility guarantee for Algorithm 7. For privacy, the sparse vector technique is $\varepsilon/2$-DP. The report-noisy-max outside the while loop will be executed at most d times, and the same for the one inside the while loop. Also, the if clause inside the while loop will also

be executed at most d times. All of these cost a privacy budget of $3d\varepsilon_2 = \varepsilon/2$. Thus, the overall algorithm is ε-DP.

We next analyze the regret. With probability $1 - 1/T$, all the noises added by AboveThreshold are bounded by $4\ln(2T^2)/\varepsilon_1$. Therefore, once $c_t(i) \geq \frac{16\ln(2T^2)}{\varepsilon_1} + \frac{3(\ln d + \ln(3d^2))}{\varepsilon_2}$ for some $i \in E$, we must invoke the report-noisy-max mechanism. On the other hand, once we invoke the report-noisy-max mechanism outside the while loop, we must have $c_t(i) \geq \frac{3(\ln d + \ln(3d^2))}{\varepsilon_2}$ for some i. With probability $1 - 1/3d$, it always identifies an i with $a_t(i) \geq \frac{3(\ln d + \ln(3d^2))}{\varepsilon_2} - \frac{2(\ln d + \ln(3d^2))}{\varepsilon_2} > 0$ and delete it.

Now let us move into the while loop. With probability $1 - 1/3d$ the report-noisy-max always returns an i with $a_t(i) > \frac{2\ln(3d^2)}{\varepsilon_2} + \frac{2(\ln d + \ln(3d^2))}{\varepsilon_2} - \frac{2(\ln d + \ln(3d^2))}{\varepsilon_2} = \frac{2\ln(3d^2)}{\varepsilon_2}$ whenever $\max_{j\in[E]} a_t(j) > \frac{2(\ln d + \ln(3d^2))}{\varepsilon_2} + \frac{2\ln(3d^2)}{\varepsilon_2}$. Moreover, the noise added in the if clause is less than $\frac{\ln(3d^2)}{\varepsilon_2}$ for the entire time span with probability $1 - 1/3d$. Thus, expert i will be removed. Conversely, if it identifies an i such that $a_t(i) = 0$, we will not remove i and exit the loop.

Let E_t be the set E at the beginning of round t and E_{T+1} be the set E after the algorithm terminates. Suppose i is removed from E at time-step T_i (if it still remains in E at the end, we define $T_i = T+1$). From the above analysis, we know that with probability $1 - 1/d - 1/T$, for every expert i we have $a_{T_i}(i) \leq \frac{3(\ln d + \ln(3d^2))}{\varepsilon_2} + \frac{2\ln(3d^2)}{\varepsilon_2} + \frac{16\ln(2T^2)}{\varepsilon_1} + 1$. Conditioning on the event that $a_{T_i}(i) \leq M$ for some M, we can bound the expected loss by

$$\begin{aligned}
\sum_{t=1}^{T} \mathbb{E}\left[\sum_{i\in E_t} \ell_t(i)/|E_t|\right] &= \sum_{t=1}^{T} \mathbb{E}\left[\sum_{i\in[d]} \mathbb{I}[i \in E_t]\ell_t(i)/|E_t|\right] \\
&= \mathbb{E}\left[\sum_{i\in[d]} \sum_{t=1}^{T} \mathbb{I}[i \in E_t]\ell_t(i)/|E_t|\right] \\
&= \mathbb{E}\left[\sum_{i\in[d]} \sum_{t=1}^{T_i} \ell_t(i)/|E_t|\right] \\
&\leq \mathbb{E}\left[\sum_{i\in[d]} \sum_{t=1}^{T_i} \ell_t(i)/|E_{T_i}|\right] \\
&= \mathbb{E}\left[\sum_{i\in[d]} a_{T_i}(i)/|E_{T_i}|\right] \\
&\leq \mathbb{E}\left[\sum_{i\in[d]} \frac{1}{|E_{T_i}|}\right] \cdot M \\
&\leq \left(\sum_{j=1}^{d} \frac{1}{j}\right) \cdot M \\
&\leq 2\ln d \cdot M
\end{aligned}$$

for $d \geq 2$ (the case that $d = 1$ is trivial, so we just ignore it).

We already have $M = O\left(\frac{d\log d + \log T}{\varepsilon}\right)$ with probability $1 - 1/d - 1/T$. However, this is not enough to show an expected bound. To get an expected bound, we run Algorithm 7 with privacy parameter $\varepsilon' = \varepsilon/2$ and an ε'-differentially private AboveThreshold to monitor $\max_{i\in E_t} a_t(i)$ (note that it won't affect the privacy and utility even if we release E_t publicly) with threshold

$$L = \left(\frac{2\ln(3d^2)}{\varepsilon'} + \frac{3(\ln d + \ln(3d^2))}{\varepsilon'} + \frac{16\ln(2T^2)}{\varepsilon'} + 1\right) + \frac{4\ln(T^2)}{\varepsilon'}.$$

https://doi.org/10.52202/079017-2086

Once AboveThreshold halts, we then stop Algorithm 7 and switch to the algorithm in Theorem F.4 with privacy parameter ε.

It is not hard to see that the overall algorithm is ε-DP. For utility, we have that with probability $1 - 1/T$ AboveThreshold halts once $\max_{i \in E_t} a_t(i)$ is greater than $\overline{L} = L + \frac{4\ln(T^2)}{\varepsilon'}$ and never halts if the value is always no more than $\underline{L} = L - \frac{4\ln(T^2)}{\varepsilon'}$. Conditioning on this, with probability $1 - 1/d - 1/T$ we have $a_{T_i} \le \underline{L}$ for every i and thus AboveThreshold never halts. In this case, the expected loss is at most $\underline{L} \cdot 2\ln d = O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right)$. If AboveThreshold halts, the expected loss incurred before that moment should be at most $(\overline{L}+1) \cdot 2\ln d = O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right)$. And after that, the regret should be $O\left(\frac{d\log T}{\varepsilon}\right)$ by Theorem F.4. Putting these two cases together gives an expected regret of

$$\begin{aligned}
&\left(1 - \frac{1}{d} - \frac{1}{T}\right) \cdot \underline{L} \cdot 2\ln d + \left(\frac{1}{d} + \frac{1}{T}\right)\left((\overline{L}+1) \cdot 2\ln d + O\left(\frac{d\log T}{\varepsilon}\right)\right) \\
&= O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right) + \left(\frac{1}{d} + \frac{1}{T}\right) O\left(\frac{d\log T}{\varepsilon}\right) \\
&= O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right).
\end{aligned}$$

Thus, the expected regret of the entire algorithm is

$$\left(1 - \frac{1}{T}\right) \cdot O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right) + \frac{1}{T} \cdot T = O\left(\frac{d\log^2 d + \log d \log T}{\varepsilon}\right).$$

□

F.5 Learning Two Complementary Hypotheses

Let $\mathcal{H} = \{f_1, f_2\}$ such that $f_1 = 1 - f_2$. We now give an algorithm that achieves a mistake bound of $O(1/\varepsilon)$.

It is not hard to come up with an algorithm that makes $O(1/\varepsilon)$ mistakes with constant probability. Since the incorrect hypothesis makes an error on every input sample, we can output arbitrarily in the first $O(1/\varepsilon)$ rounds. Then we use the $O(1/\varepsilon)$ data points to figure out which hypothesis is the correct one. This can be done using the Laplace mechanism.

However, a constant probability bound does not imply an expected bound. Note that once the Laplace mechanism fails, we may make $\Omega(T)$ errors on the entire sequence. Thus, if we follow the above framework, we have to achieve a success probability of $1 - 1/T$. This requires $O(\log T/\varepsilon)$ samples, which exceed our target.

To reduce the expected number of errors, we split the entire sequence into buckets of length $O(1/\varepsilon)$. We perform the Laplace mechanism on every bucket. Then for the i-th bucket, instead of just using the result on the last bucket to predict, we take the majority over all previous buckets. This makes the fail probability decrease exponentially. The mistake bound then converges to $O(1/\varepsilon)$ as desired.

We describe the procedure in Algorithm 8 and provide a formal statement of our result in the following theorem.

Theorem F.6. *Let $\mathcal{H} = \{f_1, f_2\}$ be a hypothesis such that f_1 and f_2 are complementary. In the realizable setting, there exists an ε-differentially private algorithm that online learns $\mathcal{H}$ with a mistake bound of $O(1/\varepsilon)$ even against strong adaptive adversaries.*

Proof. It is easy to see that Algorithm 8 is ε-differentially private since we add Laplace noise on disjoint buckets. For each bucket, with probability $2/3$, the noise added to the counter is less than $\ln(3)/\varepsilon < s/2$. This means we can identify the correct hypothesis with probability $2/3$.

For the n-th bucket, we will make s errors only if we wrongly identify the target hypothesis on at least half of the previous buckets. By Hoeffding's inequality, this happens with probability at most

$$\exp\left(-2(n-1)(1/6)^2\right) = \exp(-(n-1)/18).$$

Thus, the expected number of mistakes is bounded by

$$s \sum_{n=1}^{\infty} \exp(-(n-1)/18) = O(1/\varepsilon).$$

□

Algorithm 8: Learning complementary hypotheses

Input: the number of rounds T; hypothesis class $\mathcal{H} = \{f_1, f_2\}$ such that f_1 and f_2 are complementary; privacy parameter ε; sequence S

Output: hypotheses $h_1, \ldots, h_T$

$c_1 \leftarrow 0, c_2 \leftarrow 0$
$s \leftarrow \lceil 2\ln(3)/\varepsilon \rceil$
for $t = 1, \ldots, T$ **do**
  $(x_t, y_t) \leftarrow S[t]$
  if $c_1 > c_2$ **then**
    $h_t \leftarrow f_1$
  else
    $h_t \leftarrow f_2$
  end
  if $t \bmod s = 0$ **then**
    if $\sum_{r=t-s+1}^{t} \mathbb{I}[f_1(x_r) \neq y_r] + \mathrm{Lap}(1/\varepsilon) < s/2$ **then**
      $c_1 \leftarrow c_1 + 1$
    else
      $c_2 \leftarrow c_2 + 1$
    end
  end
end

https://doi.org/10.52202/079017-2086

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract and introduction, we state our two main contributions: a separation result between pure and approximate DP, and the other one between private and non-private settings.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations and future work in Appendix A.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The proofs of the results in Section 3 and 4 can be found in Appendix D and E respectively. We also provide some additional results together with their formal proofs in Appendix F. For each theorem in the paper, we clearly state the assumptions that it relies on.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [NA]

   Justification: The paper does not include experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

https://doi.org/10.52202/079017-2086

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: The paper does not include experiments requiring code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [NA]

   Justification: The paper does not include experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: The paper does not include experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: The paper does not include experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and make sure that the research in this paper conforms with it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the potential impacts of this paper in Appendix B.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

https://doi.org/10.52202/079017-2086

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not have such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: The paper does not use existing assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-2086