Data Poisoning against Differentially-Private Learners: Attacks and Defenses

Yuzhe Ma , **Xiaojin Zhu** and **Justin Hsu**
University of Wisconsin-Madison
{yzm234, jerryzhu, justhsu}@cs.wisc.edu

Abstract

Data poisoning attacks aim to manipulate the model produced by a learning algorithm by adversarially modifying the training set. We consider *differential privacy* as a defensive measure against this type of attack. We show that private learners are resistant to data poisoning attacks when the adversary is only able to poison a small number of items. However, this protection degrades as the adversary is allowed to poison more data. We emprically evaluate this protection by designing attack algorithms targeting *objective* and *output perturbation* learners, two standard approaches to differentially-private machine learning. Experiments show that our methods are effective when the attacker is allowed to poison sufficiently many training items.

1 Introduction

As machine learning is increasingly used for consequential decisions in the real world, their security has received more and more scrutiny. Most machine learning systems were originally designed without much thought to security, but researchers have since identified several kinds of attacks under the umbrella of *adversarial machine learning*. The most well-known example is *adversarial examples* [Szegedy *et al.*, 2013], where inputs are crafted to fool classifiers. More subtle methods aim to recover training examples [Fredrikson *et al.*, 2015], or even extract the model itself [Tramèr *et al.*, 2016].

In *data poisoning* attacks [Biggio *et al.*, 2012; Koh *et al.*, 2018; Mei and Zhu, 2015; Alfeld *et al.*, 2016; Li *et al.*, 2016; Xiao *et al.*, 2015; Muñoz-González *et al.*, 2017; Jun *et al.*, 2018; Zhao *et al.*, 2017; Zhang and Zhu, 2019], the adversary corrupts examples at training time to manipulate the learned model. As is generally the case in adversarial machine learning, the rise of data poisoning attacks has outpaced the development of robust defenses. While attacks are often evaluated against specific target learning procedures, effective defenses should provide protection—ideally guaranteed—against a broad class of attacks. In this paper, we study *differential privacy* [Dwork *et al.*, 2006; Dwork, 2011] as a simple and general defensive technique against data poisoning. While the original motivation of differential privacy was originally capture notions of privacy—the output should not depend too much on any single individual's (private) data—it also provides a defense against data poisoning. Concretely, we establish quantitative bounds on how much an adversary can change the distribution over learned models by manipulating a fixed number of training items. We are not the first to design defenses to data poisoning [Steinhardt *et al.*, 2017; Raghunathan *et al.*, 2018], but our proposal has several notable strengths compared to previous work. First, our defense provides provable protection against a worst-case adversary. Second, our defense is general. Differentially-private learning procedures are known for a wide variety of tasks, and new algorithms are continually being developed. These learners are all resilient against data poisoning.

We complement our theoretical bounds with an empirical evaluation of data poisoning attacks against differentially private learners. Concretely, we design an attack based on stochastic gradient descent to search for effective training examples to corrupt against specific learning algorithms. We evaluate our attack on two private learners: objective perturbation [Kifer *et al.*, 2012] and output perturbation [Chaudhuri *et al.*, 2011]. Our evaluation confirms the theoretical prediction that private learners are vulnerable to data poisoning attacks when the adversary can poison sufficiently many examples. A gap remains between the performance of attack and theoretical limit of data poisoning attacks—it seems like differential privacy provides a substantially stronger defense in practice than in theory. We discuss possible causes and leave further investigation to future work.

2 Preliminaries

Let the data space be $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the label space. We write $\mathcal{D} = \cup_{n=0}^{\infty} \mathcal{Z}^n$ for the space of all training sets, and write $D \in \mathcal{D}$ for a particular training set. A randomized learner $\mathcal{M} : \mathcal{D} \times \mathbb{R}^d \mapsto \Theta$ maps a training set D and noise $b \in \mathbb{R}^d$ drawn from a distribution ν to a model in the model space Θ.

Definition 1. *(Differential Privacy) $\mathcal{M}$ is (ϵ, δ)-*differentially-private *if $\forall D, \tilde{D} \in \mathcal{D}$ that differ by one item, and $\forall \mathcal{S} \subset \Theta$,*

$$\mathbf{P}\left(\mathcal{M}(D, b) \in \mathcal{S}\right) \leq e^{\epsilon} \mathbf{P}\left(\mathcal{M}(\tilde{D}, b) \in \mathcal{S}\right) + \delta, \quad (1)$$

*where the probability is taken over $b \sim \nu$. When $\delta = 0$, we say that $\mathcal{M}$ is ϵ-*differentially private.

Many standard machine learning tasks can be phrased as optimization problems modeling empirical risk minimization (ERM). Broadly speaking, there are two families of techniques in the privacy literature for solving these problems. Objective perturbation injects noise into the objective function of a vanilla machine learner to train a randomized model [Chaudhuri *et al.*, 2011; Kifer *et al.*, 2012; Nozari *et al.*, 2016]. In contrast, output perturbation runs the vanilla learner but after training, it randomly perturbs the output model [Chaudhuri *et al.*, 2011]. We will consider data poisoning attacks against both kinds of learners. We first fix the adversarial attack setting.

Knowledge of the attacker: The attacker has full knowledge of the full training set D and the differentially-private machine learner $\mathcal{M}$, including the noise distribution ν. However, the attacker does *not* know the realized noise b.

Ability of the attacker: The attacker is able to modify the clean training set D by changing the features or labels, adding additional items, or deleting existing items, but the attacker can modify at most k items. Formally, the poisoned training set $\tilde{D}$ must lie in the ball $\mathcal{B}(D,k) \subset \mathcal{D}$, where the center is the clean data D and the radius k is the maximum number of item changes.

Goal of the attacker: The attacker aims to drive the (stochastic) model learned from the poisoned data $\tilde{\theta} = \mathcal{M}(\tilde{D}, b)$ to achieve a certain goal. Formally, we define a cost function $C : \Theta \mapsto \mathbb{R}$ that measures how much the poisoned model $\tilde{\theta}$ deviates from the attack target. Then the attack problem can be formulated as minimizing an objective function J:

$$\min_{\tilde{D} \in \mathcal{B}(D,k)} \quad J(\tilde{D}) := \mathbf{E}_b \left[C(\mathcal{M}(\tilde{D}, b)) \right]. \tag{2}$$

This formulation differs from previous works (e.g. [Biggio *et al.*, 2012; Mei and Zhu, 2015; Xiao *et al.*, 2015]) because the differentially-private learner produces a stochastic model, so the objective takes the *expected* value of C.

Example 1. *(Parameter-Targeting Attack) If the attacker wants the output model to be close to a target model θ', the attacker can define $C(\tilde{\theta}) = \frac{1}{2}\|\tilde{\theta} - \theta'\|^2$. Minimizing $J(\tilde{D})$ pushes the poisoned model close to θ' in expectation.*

Example 2. *(Label-Targeting Attack) If the attacker wants small prediction error on an evaluation set $\{z_i^*\}_{i\in[m]}$, the attacker can define $C(\tilde{\theta}) = \frac{1}{m}\sum_{i=1}^m \ell(\tilde{\theta}, z_i^*)$ where $\ell(\tilde{\theta}, z_i^*)$ is the loss on item $z_i^* = (x_i^*, y_i^*)$. Minimizing $J(\tilde{D})$ pushes the prediction on x_i^* towards the target label y_i^*.*

Example 3. *(Label-Aversion Attack) If the attacker wants to induce large prediction error on an evaluation set $\{z_i^*\}_{i\in[m]}$, the attacker can define $C(\tilde{\theta}) = -\frac{1}{m}\sum_{i=1}^m \ell(\tilde{\theta}, z_i^*)$, a non-positive cost. Minimizing $J(\tilde{D})$ pushes the prediction on x_i^* away from the true label y_i^*.*

Now, we turn to our two main contributions: i) we show that differentially-private learners have a degree of natural immunity against data poisoning attacks. Specifically, in section 3 we present a lower bound on how much the attacker can reduce $J(\tilde{D})$ by manipulating up to k items; and ii) as k increases, the attacker's power grows and it becomes easier to attack differentially-private learners. In section 4 we propose a stochastic gradient descent algorithm to solve the attack problem (2). To our knowledge, this is the first data poisoning attack targeting specifically differentially-private learners.

3 Differential Privacy Resists Data Poisoning

Fixing the number of poisoned items k and a differentially private learner, can a powerful attacker achieve arbitrarily low attack cost $J(\tilde{D})$? We show that the answer is no: differential privacy implies a lower bound on how far $J(\tilde{D})$ can be reduced, so private learners are inherently resistant to data poisoning attacks. Previous work [Lecuyer *et al.*, 2018] proposed defenses based on differential privacy, but for test-time attacks. While effective, these defenses require the classifier to make randomized predictions. In contrast, our defense against data poisoning only requires the *learning algorithm*, not the classification process, to be randomized.

Our first result is for ϵ-differentially private learners: if an attacker can manipulate at most k items in the clean data, then the attack cost is lower bounded.

Theorem 1. *Let $\mathcal{M}$ be an ϵ-differentially-private learner. Let $J(\tilde{D})$ be the attack cost, where $\tilde{D} \in \mathcal{B}(D,k)$, then*

$$\begin{aligned} J(\tilde{D}) &\geq e^{-k\epsilon} J(D) \quad (C \geq 0), \\ J(\tilde{D}) &\geq e^{k\epsilon} J(D) \quad (C \leq 0). \end{aligned} \tag{3}$$

Note that for $C \geq 0$, Theorem 1 shows that no attacker can achieve the trivial lower bound $J(\tilde{D}) = 0$. For $C \leq 0$, although $J(\tilde{D})$ could be unbounded from below, Theorem 1 shows that no attacker can reduce $J(\tilde{D})$ arbitrarily. These results follow from the differential privacy property of $\mathcal{M}$. Corollary 2 re-states the theorem in terms of the minimum number of items an attacker has to modify in order to sufficiently reduce the attack cost $J(\tilde{D})$.

Corollary 2. *Let $\mathcal{M}$ be an ϵ-differentially-private learner. Assume $J(D) \neq 0$. Let $\tau \geq 1$. Then the attacker has to modify at least $k \geq \lceil \frac{1}{\epsilon} \log \tau \rceil$ items in D in order to achieve*

i). $J(\tilde{D}) \leq \frac{1}{\tau} J(D) \quad (C \geq 0)$.

ii). $J(\tilde{D}) \leq \tau J(D) \quad (C \leq 0)$.

To generalize Theorem 1 to (ϵ, δ)-private learners, we need an additional assumption that C is bounded.

Theorem 3. *Let $\mathcal{M}$ be (ϵ, δ)-differentially-private and $J(\tilde{D})$ be the attack cost, where $|C| \leq \bar{C}$ and $\tilde{D} \in \mathcal{B}(D,k)$. Then,*

$$\begin{aligned} J(\tilde{D}) &\geq \max\{e^{-k\epsilon}(J(D) + \frac{\bar{C}\delta}{e^\epsilon - 1}) - \frac{\bar{C}\delta}{e^\epsilon - 1}, 0\} (C \geq 0), \\ J(\tilde{D}) &\geq \max\{e^{k\epsilon}(J(D) - \frac{\bar{C}\delta}{e^\epsilon - 1}) + \frac{\bar{C}\delta}{e^\epsilon - 1}, -\bar{C}\} (C \leq 0). \end{aligned} \tag{4}$$

As a corollary, we can lower-bound the minimum number of modifications needed to reduce the attack cost to a certain level.

Corollary 4. *Let $\mathcal{M}$ be an (ϵ, δ)-differentially-private learner. Assume $J(D) \neq 0$. Then*

i). $(0 \le C \le \bar{C})$*: in order to achieve* $J(\tilde{D}) \le \frac{1}{\tau} J(D)$ *for* $\tau \ge 1$*, the attacker has to modify at least the following number of items in* D*.*

$$k \ge \lceil \frac{1}{\epsilon} \log \frac{(e^\epsilon - 1) J(D)\tau + \bar{C}\delta\tau}{(e^\epsilon - 1) J(D) + \bar{C}\delta\tau} \rceil. \quad (5)$$

ii). $(-\bar{C} \le C \le 0)$*: in order to achieve* $J(\tilde{D}) \le \tau J(D)$ *for* $\tau \in [1, -\frac{\bar{C}}{J(D)}]$*, the attacker has to modify at least the following number of items in* D*.*

$$k \ge \lceil \frac{1}{\epsilon} \log \frac{(e^\epsilon - 1) J(D)\tau - \bar{C}\delta}{(e^\epsilon - 1) J(D) - \bar{C}\delta} \rceil. \quad (6)$$

Note that in (5), the lower bound on k is always finite even if $\tau = \infty$, which means an attacker might be able to reduce the attack cost $J(\tilde{D})$ to 0. This is in contrast to ϵ-differentially-private learner, where $J(\tilde{D}) = 0$ can never be achieved. Thus, the weaker (ϵ, δ)-differential privacy guarantee gives weaker protection against attacks.

4 Data Poisoning Attacks on Private Learners

The results in the previous section provide a theoretical bound on how effective a data poisoning attack can be against a private learner. To evaluate how strong the protection is in practice, we propose a range of attacks targeting general differentially-private learners. Our adversary will *modify* (not add or delete) any continuous features (for classification or regression) and the continuous labels (for regression) on at most k items in the training set. Restating the attack problem (2):

$$\min_{\tilde{D} \subset \mathcal{B}(D,k)} \quad J(\tilde{D}) = \mathbf{E}_b \left[C(\mathcal{M}(\tilde{D}, b)) \right] \quad (7)$$

This is a combinatorial problem—the attacker must pick k items to modify. We propose a two-step attack procedure.

step I) use heuristic methods to select k items to poison.

step II) apply stochastic gradient descent to reduce $J(\tilde{D})$.

Step II) could lead to poisoned items taking arbitrary features or labels. However, differentially-private learners typically assume training examples are bounded. To avoid trivially-detectable poisoned examples, we project poisoned features and labels back to a bounded space after each iteration of SGD. We now detail step II), assuming that the attacker has fixed k items to poison. We will return to step I) later.

4.1 SGD on Differentially-Private Victims (DPV)

Our attacker uses stochastic gradient descent to minimize the cost $J(\tilde{D})$. We first show how to compute a stochastic gradient with respect to a training item.

Proposition 5. *Assume* $\mathcal{M}(\tilde{D}, b)$ *is a differentiable function of* $\tilde{D}$*, then under certain conditions (details deferred to the full paper), a stochastic gradient of* $J(\tilde{D})$ *with respect to a particular item* $\tilde{z}_i$ *is* $\frac{\partial C(\mathcal{M}(\tilde{D}, b))}{\partial \tilde{z}_i}$*, where* $b \sim \nu(b)$*.*

The concrete form of the stochastic gradient depends on the target private learner $\mathcal{M}$, as well as the attack cost (which encodes the type of attack). In the following, we derive the stochastic gradient for objective perturbation and output perturbation in the context of parameter-targeting attack with cost function $C(\tilde{\theta}) = \frac{1}{2}\|\tilde{\theta} - \theta'\|^2$, where θ' is the target model.

Instantiating Attack on Objective Perturbation

We first consider objective-perturbed private learners:

$$\mathcal{M}(\tilde{D}, b) = \operatorname*{argmin}_{\theta \in \Theta} \sum_{i=1}^{n} \ell(\theta, \tilde{z}_i) + \lambda\Omega(\theta) + b^\top \theta, \quad (8)$$

where ℓ is some convex learning loss, Ω is a regularization, and Θ is the model space. Note that the noise b enters the objective via linear product with the model θ. By Proposition 5, the stochastic gradient is $\frac{\partial C(\mathcal{M}(\tilde{D}, b))}{\partial \tilde{z}_i} = \frac{\partial C(\tilde{\theta})}{\partial \tilde{z}_i}$, where $\tilde{\theta} = \mathcal{M}(\tilde{D}, b)$ is the poisoned model. By the chain rule, we have

$$\frac{\partial C(\tilde{\theta})}{\partial \tilde{z}_i} = \left(\frac{\partial \tilde{\theta}}{\partial \tilde{z}_i} \right)^\top \frac{dC(\tilde{\theta})}{d\tilde{\theta}}. \quad (9)$$

Note that $\frac{dC(\tilde{\theta})}{d\tilde{\theta}} = \tilde{\theta} - \theta'$. Next we focus on deriving $\frac{\partial \tilde{\theta}}{\partial \tilde{z}_i}$. Since (8) is convex, the learned model $\tilde{\theta}$ must satisfy the following Karush-Kuhn-Tucker (KKT) condition:

$$f(\tilde{\theta}, \tilde{z}_i) := \sum_{j=1}^{n} \frac{\partial \ell(\tilde{\theta}, \tilde{z}_j)}{\partial \tilde{\theta}} + \lambda \frac{d\Omega(\tilde{\theta})}{d\tilde{\theta}} + b = 0. \quad (10)$$

By using the derivative of implicit function, we have $\frac{\partial \tilde{\theta}}{\partial \tilde{z}_i} = -(\frac{\partial f}{\partial \tilde{\theta}})^{-1} \frac{\partial f}{\partial \tilde{z}_i}$. We now give two examples where the base learner is logistic regression and ridge regression, respectively.

In the case $\ell(\theta, \tilde{z}) = \log(1 + \exp(-\tilde{y}\theta^\top \tilde{x}))$ and $\Omega(\theta) = \frac{1}{2}\|\theta\|^2$, learner (8) is objective-perturbed logistic regression:

$$\mathcal{M}(\tilde{D}, b) = \operatorname*{argmin}_{\theta \in \Theta} \sum_{i=1}^{n} \log(1+\exp(-\tilde{y}_i \theta^\top \tilde{x}_i)) + \frac{\lambda}{2}\|\theta\|^2 + b^\top \theta, \quad (11)$$

where $\Theta = \mathbb{R}^d$. Our attacker will only modify the features, thus $\tilde{y}_i = y_i$. We now derive stochastic gradient for $\tilde{x}_i$. Define $s_j = \exp(y_j \tilde{\theta}^\top \tilde{x}_j)$. By (9), one can verify that $\frac{\partial C(\tilde{\theta})}{\partial \tilde{x}_i} =$

$$\left(\frac{y_i}{1+s_i} I - \frac{s_i \tilde{\theta} \tilde{x}_i^\top}{(1+s_i)^2} \right) \left(\lambda I + \sum_{j=1}^{n} \frac{s_j \tilde{x}_j \tilde{x}_j^\top}{(1+s_j)^2} \right)^{-1} (\tilde{\theta} - \theta').$$

Note that the noise b enters into stochastic gradient through $\tilde{\theta} = \mathcal{M}(\tilde{D}, b)$.

In the case $\ell(\theta, \tilde{z}) = \frac{1}{2}(\tilde{y} - \tilde{x}^\top \theta)^2$ and $\Omega(\theta) = \frac{1}{2}\|\theta\|^2$, learner (8) is the objective-perturbed ridge regression:

$$\mathcal{M}(\tilde{D}, b) = \operatorname*{argmin}_{\theta \in \Theta} \frac{1}{2}\|\tilde{X}\theta - \tilde{y}\|^2 + \frac{\lambda}{2}\|\theta\|^2 + b^\top \theta, \quad (12)$$

where $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2 \le \rho\}$, $\tilde{X} \in \mathbb{R}^{n \times d}$ is the feature matrix, and $\tilde{y} \in \mathbb{R}^n$ is the label vector. Unlike logistic regression, the objective-perturbed ridge regression requires the model space to be bounded (see e.g. [Kifer *et al.*, 2012]). The attacker can modify both the features and the labels. By (9), the stochastic gradient of $\tilde{x}_i$ is $\frac{\partial C(\tilde{\theta})}{\partial \tilde{x}_i} =$

$$\left(\tilde{\theta} \tilde{x}_i^\top + (\tilde{x}_i^\top \tilde{\theta} - \tilde{y}_i) I \right) \left(\tilde{X}^\top \tilde{X} + (\lambda + \mu) I \right)^{-1} (\theta' - \tilde{\theta}),$$

where μ is the dual variable of constraint $\frac{1}{2}\|\theta\|^2 \leq \frac{\rho^2}{2}$ when solving (12). The stochastic gradient of $\tilde{y}_i$ is

$$\frac{\partial C(\tilde{\theta})}{\partial \tilde{y}_i} = \tilde{x}_i^\top \left(\tilde{X}^\top \tilde{X} + (\lambda+\mu)I\right)^{-1} (\tilde{\theta} - \theta').$$

Note that the noise b enters into the stochastic gradient through $\tilde{\theta} = \mathcal{M}(\tilde{D}, b)$.

Instantiating Attack on Output Perturbation

We now consider the output perturbation mechanism:

$$\mathcal{M}(\tilde{D}, b) = b + \operatorname*{argmin}_{\theta\in\Theta} \left\{ \sum_{i=1}^{n} \ell(\theta, \tilde{z}_i) + \lambda\Omega(\theta) \right\}. \quad (13)$$

The stochastic gradients for this target are similar to those for objective perturbation. Again, we instantiate on two examples where the base learner is logistic regression and ridge regression, respectively.

The output-perturbed logistic regression takes the following form:

$$\mathcal{M}(\tilde{D}, b) = b + \operatorname*{argmin}_{\theta\in\mathbb{R}^d} \sum_{i=1}^{n} \log(1+\exp(-\tilde{y}_i\theta^\top \tilde{x}_i)) + \frac{\lambda}{2}\|\theta\|^2, \quad (14)$$

Let $s_j = \exp(y_j(\tilde{\theta} - b)^\top \tilde{x}_j)$, then we have $\frac{\partial C(\tilde{\theta})}{\partial \tilde{x}_i} =$

$$\left(\frac{y_i}{1+s_i} I - \frac{s_i(\tilde{\theta}-b)\tilde{x}_i^\top}{(1+s_i)^2}\right) \left(\lambda I + \sum_{j=1}^{n} \frac{s_j \tilde{x}_j \tilde{x}_j^\top}{(1+s_j)^2}\right)^{-1} (\tilde{\theta} - \theta').$$

The output-perturbed ridge regression takes the following form

$$\mathcal{M}(\tilde{D}, b) = b + \operatorname*{argmin}_{\theta\in\Theta} \frac{1}{2}\|\tilde{X}\theta - \tilde{y}\|^2 + \frac{\lambda}{2}\|\theta\|^2, \quad (15)$$

where $\Theta = \{\theta : \|\theta\|_2 \leq \rho\}$. Let μ be the dual variable for the constraint $\frac{1}{2}\|\theta\|^2 \leq \frac{\rho 2}{2}$ when solving the argmin in (15). Then the stochastic gradients of $\tilde{x}_i$ and $\tilde{y}_i$ are

$$\frac{\partial C(\tilde{\theta})}{\partial \tilde{x}_i} = \left((\tilde{\theta} - b)\tilde{x}_i^\top + \tilde{x}_i^\top(\tilde{\theta} - b)I - \tilde{y}_i I\right) \cdot \left(\tilde{X}^\top \tilde{X} + (\lambda+\mu)I)\right)^{-1} (\theta' - \tilde{\theta}).$$

$$\frac{\partial C(\tilde{\theta})}{\partial \tilde{y}_i} = \tilde{x}_i^\top \left(\tilde{X}^\top \tilde{X} + (\lambda+\mu)I\right)^{-1} (\tilde{\theta} - \theta').$$

4.2 SGD on Surrogate Victims (SV)

We also consider an alternative, simpler way to perform step II). When $b = \mathbf{0}$, the differentially-private learning algorithms revert to the base learner which returns a deterministic model $\mathcal{M}(\tilde{D}, \mathbf{0})$. The adversary can simply attack the base learner (without b) as a surrogate for the differentially-private learner $\mathcal{M}$ (with b) by solving the following problem:

$$\min_{\tilde{D}} C(\mathcal{M}(\tilde{D}, \mathbf{0})). \quad (16)$$

Since the objective is deterministic, we can work with its gradient rather than its stochastic gradient, plugging in $b = \mathbf{0}$ into all derivations in section 4.1. Note that the attack found by (16) will still be evaluated with respect to differentially-private learners in our experiments.

4.3 Selecting Items to Poison

Now, we return to step I) of our attack: how can we select the k items for poisoning? We give two heuristic methods: shallow and deep selection.

Shallow selection selects the k items with the largest initial gradient norm $\|\frac{\partial J(\tilde{D})}{\partial \tilde{z}_i} \mid_{\tilde{D}=D} \|$. Modifying these items will reduce the attack cost the most, at least initially. The precise gradients to use during selection depend on the attack in step II). When targeting differentially-private learners directly, computing the gradient of $J(\tilde{D})$ is difficult. We use Monte-Carlo sampling to approximate the gradient g_i of item z_i: $g_i \approx \frac{1}{m}\sum_{s=1}^{m} \frac{\partial}{\partial \tilde{z}_i} C(\mathcal{M}(\tilde{D}, b_s)) \mid_{\tilde{D}=D}$, where b_s are random samples of privacy parameters. Then, the attacker picks the k items with the largest $\|g_i\|$ to poison. When targeting surrogate victim, the objective is $C(\mathcal{M}(\tilde{D}, \mathbf{0}))$, thus the attacker computes the gradient of $C(\mathcal{M}(\tilde{D}, \mathbf{0}))$ of each clean item z_i: $g_i = \frac{d}{d\tilde{z}_i} C(\mathcal{M}(\tilde{D}, \mathbf{0})) \mid_{\tilde{D}=D}$, and then picks the k items with the largest $\|g_i\|$.

Deep selection selects items by estimating the influence of an item on the final attack cost. When targeting a differentially-private victim directly, the attacker first solves the following relaxed attack problem: $\min_{\tilde{D}} J(\tilde{D}) + \alpha R(\tilde{D})$. Importantly, here $\tilde{D}$ allows changes to all training items, not just k of them. $R(\tilde{D})$ is a regularizer penalizing data modification with weight $\alpha > 0$. We take $R(\tilde{D}) = \sum_{i=1}^{n} r(\tilde{z}_i)$, where $r(\tilde{z}_i)$ is the distance between the poisoned item $\tilde{z}_i$ and the clean item z_i. We define $r(\tilde{z}_i) = \frac{1}{2}\|\tilde{x}_i - x_i\|^2$ for logistic regression and $r(\tilde{z}_i) = \frac{1}{2}\|\tilde{x}_i - x_i\|^2 + \frac{1}{2}(\tilde{y}_i - y_i)^2$ for ridge regression. After solving the relaxed attack problem with SGD, the attacker evaluates the amount of change $r(\tilde{z}_i)$ for all training items and pick the top k to poison. When targeting a surrogate victim, the attacker can solve the following relaxed attack first: $\min_{\tilde{D}} C(\mathcal{M}(\tilde{D}, \mathbf{0})) + \alpha R(\tilde{D})$, and then select the k top items.

In summary, we have four attack algorithms based on combinations of methods used in step I) and step II): shallow-SV, shallow-DPV, deep-SV and deep-DPV. In the following, we evaluate the performance of these four algorithms.

5 Experiments

We now evaluate our attack with experiments, taking objective-perturbed learners [Kifer *et al.*, 2012] and output-perturbed learners [Chaudhuri *et al.*, 2011] as our victims. Throughout, we use $\alpha = 10^{-4}$ for deep selection and fix a constant step size $\eta = 1$ for (stochastic) gradient descent. After each iteration of SGD, we project poisoned items to ensure feature norms are at most 1 and labels are in $[-1, 1]$.

5.1 Attacks Intelligently Modify Data

As a first example, we use label-aversion attack to illustrate that our attack modifies data intelligently. The victim is an objective-perturbed learner for ϵ-differentially private logistic regression, with $\epsilon = 0.1$ and regularizer $\lambda = 10$. The training set contains $n = 21$ items uniformly sampled in the interval $[-1, 1]$ with labels $y_i = \mathbb{1}[x_i \geq 0]$, see Figure 1(a). To generate the evaluation set, we construct $m = 21$ evenly-spaced items in the interval $[-1, 1]$, i.e., $x_i^* = 0.1i, -10 \leq i \leq 10$

and labeled as $y_i^* = \mathbb{1}[x_i^* \geq 0]$. The cost function is defined as $C(\tilde{\theta}) = -\frac{1}{m}\sum_{i=1}^{m} \log(1 + \exp(-y_i^* x_i^{*\top}\tilde{\theta}))$. To achieve small attack cost J, a negative model $\tilde{\theta} < 0$ is desired.[1] We run deep-DPV with $k = n$ (attacker can modify all items) for $T = 300$ iterations. In Figure 1(a), we show the position of the poisoned items on the vertical axis as the SGD iteration, t, grows. As expected, the attacker flips the positions of positive (blue) items and negative (red) items. As a result, our attack causes the learner to find a model with $\tilde{\theta} < 0$.

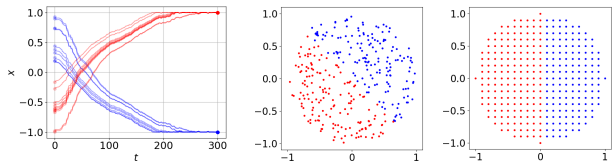
5.2 Attacks Can Achieve Different Goals

We now study a 2D example to show that our attack is effective for different attack goals. The victim is the same as in section 5.1. The clean training set in Figure 1(b) contains $n = 317$ items uniformly sampled in a unit sphere with labels $y_i = \mathbb{1}\left[x_i^\top \theta^* \geq 0\right]$ where $\theta^* = (1, 1)$. We now describe the attack settings for different attack goals.

For label-aversion attack, we generate an evaluation set containing $m = 317$ grid points (x_i^*, y_i^*) lying in a unit sphere and labeled by a vertical decision boundary, see Figure 1(c). The cost function is $C(\tilde{\theta}) = -\frac{1}{m}\sum_{i=1}^{m} \log(1 + \exp(-y_i^* x_i^{*\top}\tilde{\theta}))$.

For label-targeting attack, the evaluation set remains the same as in the label-aversion attack. The cost function is $C(\tilde{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + \exp(-y_i^* x_i^{*\top}\tilde{\theta}))$.

For parameter-targeting attack, we first train vanilla logistic regression on the evaluation set to obtain the target model $\theta' = (2.6, 0)$, then define cost function as $C(\tilde{\theta}) = \frac{1}{2}\|\tilde{\theta} - \theta'\|^2$.



(a) poisoning trajectory (b) training set (c) evaluation set

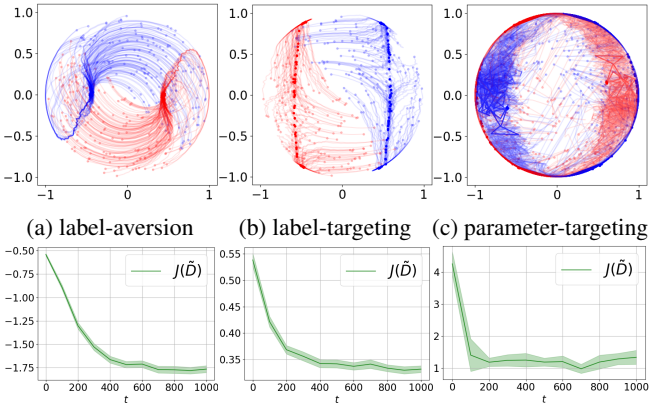
Figure 1: (a) 1D example (b, c) 2D example

We run deep-DPV with $k = n$ for $T = 5 \times 10^3$ iterations. In Figure 2(a)-(c), we show the poisoning trajectories for the three attack goals respectively. Each translucent point is an original training point, and the connected solid point is its final position after attack. The curve connecting them shows the trajectory as the attack algorithm gradually poisons the data.

In label-aversion attack the attacker aims to maximize the logistic loss on the evaluation set. It ends up moving positive (negative) items to the left (right), so that the poisoned data deviates from the evaluation set.

In contrast, the label-targeting attack tries to minimize the logistic loss, thus items are moved to produce a poisoned data aligned with the evaluation set. However, the attack does not produce exactly the evaluation set. To understand that, we compute the model learnt by vanilla logistic regression on the evaluation set and the poisoned data, which are (2.60,0) and (2.94,0.01). Note that both models are aligned with the evaluation set, but the latter has larger norm, which leads to smaller attack cost, thus our result is a better attack.

[1]Differentially-private machine learning typically considers homogeneous learners, thus the 1D model is just a scalar.



(a) label-aversion (b) label-targeting (c) parameter-targeting

(d) label-aversion (e) label-targeting (f) parameter-targeting

Figure 2: Poisoning trajectories in 2D.

In parameter-targeting attack, we compute the model learnt by vanilla logistic regression on the clean data and poisoned data, which are (1.86,1.85) and $(2.21, 0.04)$. Note that the attack pushes the model closer to the target model (2.6,0).

To quantify the attack performance, we evaluate the attack cost $J(\tilde{D})$ via Monte-Carlo sampling. We run private learners $T_e = 10^3$ times, each time with a random b_s to produce a cost $C_s = C(\mathcal{M}(\tilde{D}, b_s))$. Then we average to obtain an estimate of the cost $J(\tilde{D}) \approx \frac{1}{T_e}\sum_{s=1}^{T_e} C_s$. In Figure 2(d)-(f), we show the attack cost $J(\tilde{D})$ and its 95% confidence interval (green, $\pm 2 *$stderr) up to $t = 10^3$ iterations. $J(\tilde{D})$ decreases in all plots, showing that our attack is effective for all attack goals.

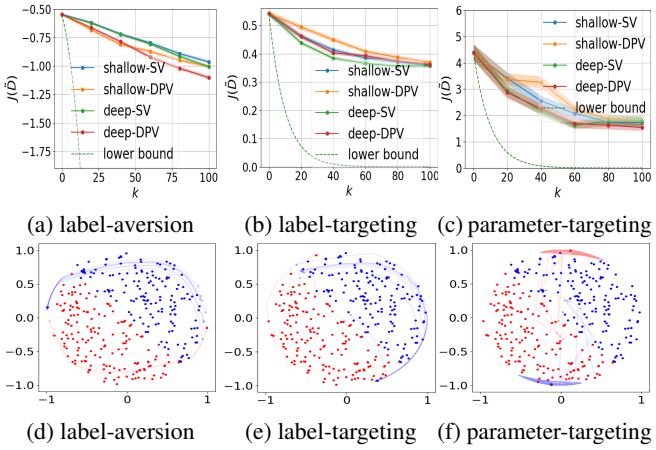
5.3 Attacks Become More Effective as k Increases

The theoretical protection provided by differential privacy weakens as the adversary is allowed to poison more training items. We can show empirically that our attacks indeed become more effective as the number of poisoned items k increases. The data set is the same as in section 5.2. We consider k from 20 to 100 in steps of 20. For each k, we run our four attack algorithms for $T = 5 \times 10^3$ iterations and estimate the attack cost $J(\tilde{D})$ on $T_e = 2 \times 10^3$ samples. Figure 3(a)-(c) show that the attack cost $J(\tilde{D})$ decreases as k grows, indicating better attack performance. We also show the poisoning trajectory produced by deep-DPV with $k = 10$ in Figure 3(d)-(f). The corresponding attack costs $J(\tilde{D})$ are $-0.60, 0.49$ and 3.43 respectively. Compared to that of $k = n$, -1.79, 0.32 and 1.20, we see that poisoning only 10 items is not enough to reduce the attack cost $J(\tilde{D})$ significantly, although the attacker is having some effect.

5.4 Attacks Effective on Both Privacy Mechanisms

We now show experiments on real data to illustrate that our attacks are effective on both objective and output perturbation mechanisms. We focus on label-targeting attack.

The first real data set is vertebral column from the UCI Machine Learning Repository [Dua and Karra Taniskidou, 2017].

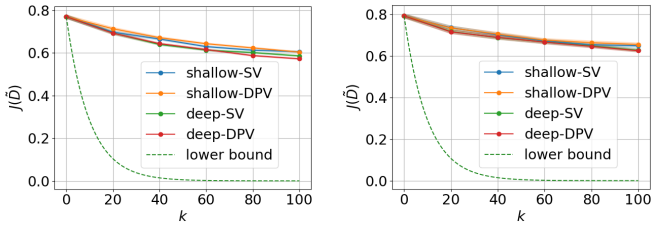


(a) label-aversion (b) label-targeting (c) parameter-targeting

(d) label-aversion (e) label-targeting (f) parameter-targeting

Figure 3: (a-c) the attack cost $J(\tilde{D})$ decreases as k grows. (d-f) the sparse attack trajectories for $k = 10$.

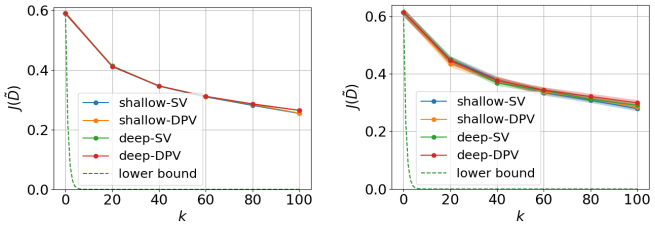
The data set contains 6 features of 310 orthopaedic patients, and the task is to predict if the vertebra of any patient is abnormal (positive class) or not (negative class). We normalize the features so that the norm of any item is at most 1. Since this is a classification task, we use private logistic regression with $\epsilon = 0.1$ and $\lambda = 10$ as the victim. To generate the evaluation set $\{(x_i^*, y_i^*)\}$, we randomly pick one positive item and find its 10 nearest neighbors within the positive class, and set $y_i^* = -1$ for all 10. Intuitively, this attack targets a small cluster of abnormal patients and the goal is to mislead the learner to classify them as normal. The other experimental parameters remain the same as in section 5.2. Note that in order to push the prediction on x_i^* toward y_i^*, the attacker requires $y_i^* x_i^{*\top} \tilde{\theta} > 0$, which indicates $C(\tilde{\theta}) = \log(1 + \exp(-y_i^* x_i^{*\top} \tilde{\theta})) < \log 2 \approx 0.69$, so should $J(\tilde{D})$. As shown in Figure 4, the attacker indeed reduces the attack cost $J(\tilde{D})$ below 0.69 for both privacy mechanisms, thus our attack is successful.



(a) objective perturbation (b) output perturbation

Figure 4: Attack on objective and output-perturbed logistic regression.

The second data set is red wine quality from UCI. The data set contains 11 features of 1598 wine samples, and the task is predict the wine quality, a number between 0 and 10. We normalize the features so that the norm of any item is at most 1. Labels are also normalized to ensure the value is in $[-1, 1]$. The victim is private ridge regression with $\epsilon = 1$, $\lambda = 10$, and bound on model space $\rho = 2$. To generate the evaluation set, we pick an item x_i^* with the smallest quality value, and set the target label to $y_i^* = 1$. The cost function is defined as $C(\theta) = \frac{1}{2}(x_i^{*\top}\theta - y_i^*)^2$. The other experimental parameters are the same as in section 5.2. This attack aims to force the learner to predict a low-quality wine as having a high quality. Figure 5 shows the result. Note that the attack cost can be significantly reduced for both privacy mechanisms even if the attacker poisons only $100/1598 \approx 6.3\%$ items of the data set.



(a) objective perturbation (b) output perturbation

Figure 5: Attack on objective and output-perturbed ridge regression.

We make two overall observations. First, deep-DPV is in general the most effective method among our four attacks. Second, there remains a gap between the performance of deep-DPV and the theoretical bound, leaving space to design more efficient attacks or to tighten the theoretical bound.

5.5 Attack Is Easier with Weaker Privacy

Finally, we show that the attack is more effective as the privacy guarantee becomes weaker, i.e., as we increase ϵ. To illustrate, we run experiments on the data set in section 5.2, where we fix $k = 100$ and vary ϵ from 0.1 to 0.5. In Figure 6, the attack cost approaches the lower bound as ϵ grows. This means weaker privacy guarantees (smaller ϵ) lead to easier attacks.

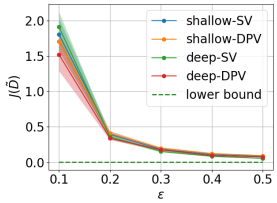


Figure 6: Attacker can reduce cost $J(\tilde{D})$ more as ϵ grows.

6 Conclusion and Future Work

We showed that differentially private learners are resistant to data poisoning attacks, with the protection degrading as the attacker poisons more items. We proposed attack algorithms and demonstrated that our attacks are effective on different privacy mechanisms and machine learners. There remains a gap between the theoretical limit and the empirical performance of our attacks; closing the gap remains future work.

Acknowledgments

We thank Steve Wright for helpful discussions. This work is supported in part by NSF 1545481, 1561512, 1623605, 1704117, 1836978, the MADLab AF Center of Excellence FA9550-18-1-0166, a Facebook TAV grant, and the University of Wisconsin.

References

[Alfeld *et al.*, 2016] Scott Alfeld, Xiaojin Zhu, and Paul Barford. Data poisoning attacks against autoregressive models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[Biggio *et al.*, 2012] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[Chaudhuri *et al.*, 2011] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

[Dua and Karra Taniskidou, 2017] Dheeru Dua and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[Dwork *et al.*, 2006] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[Dwork, 2011] Cynthia Dwork. Differential privacy. *Encyclopedia of Cryptography and Security*, pages 338–340, 2011.

[Fredrikson *et al.*, 2015] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015.

[Jun *et al.*, 2018] Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Jerry Zhu. Adversarial attacks on stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 3640–3649, 2018.

[Kifer *et al.*, 2012] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pages 25–1, 2012.

[Koh *et al.*, 2018] Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*, 2018.

[Lecuyer *et al.*, 2018] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *Certified Robustness to Adversarial Examples with Differential Privacy*. IEEE, 2018.

[Li *et al.*, 2016] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1885–1893, 2016.

[Mei and Zhu, 2015] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *The 29th AAAI Conference on Artificial Intelligence*, 2015.

[Muñoz-González *et al.*, 2017] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 27–38. ACM, 2017.

[Nozari *et al.*, 2016] Erfan Nozari, Pavankumar Tallapragada, and Jorge Cortés. Differentially private distributed convex optimization via objective perturbation. In *2016 American Control Conference (ACC)*, pages 2061–2066. IEEE, 2016.

[Raghunathan *et al.*, 2018] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[Steinhardt *et al.*, 2017] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3517–3529, 2017.

[Szegedy *et al.*, 2013] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[Tramèr *et al.*, 2016] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *25th USENIX Security Symposium*, pages 601–618, 2016.

[Xiao *et al.*, 2015] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *International Conference on Machine Learning*, pages 1689–1698, 2015.

[Zhang and Zhu, 2019] Xuezhou Zhang and Xiaojin Zhu. Online data poisoning attack. *arXiv preprint arXiv:1903.01666*, 2019.

[Zhao *et al.*, 2017] Mengchen Zhao, Bo An, Wei Gao, and Teng Zhang. Efficient label contamination attacks against black-box learning models. In *IJCAI*, pages 3945–3951, 2017.