

# Auditing Differentially Private Machine Learning: How Private is Private SGD?\*

Matthew Jagielski

Jonathan Ullman

Alina Oprea

*Khoury College of Computer Sciences, Northeastern University*

# Main Idea

1. Though the exact value of  $\epsilon$  can't be determined, it's lower bound can be estimated empirically
2. A novel data poisoning attack can be constructed to estimate a better (i.e. larger) lower bound of  $\epsilon$

# 1. Estimate the lower bound of $\epsilon$ , i.e. $\epsilon_{LB}$

Given clean dataset  $D_0$  and a perturbed dataset  $D_1$ , let  $p_0 = \Pr[A(D_0) \in \mathcal{O}]$ ,  $p_1 = \Pr[A(D_1) \in \mathcal{O}]$ , then  $\epsilon_{LB}$  can be estimated through a Monte Carlo process:

1. Train  $T$  models using algorithm  $A$  on  $D_0$  and  $D_1$ , and count the times when  $A(D_0)$  in  $\mathcal{O}$  and  $A(D_1)$  in  $\mathcal{O}$ .
2. A lower bound for  $p_0$  (i.e.  $\hat{p}_0$ ) and an upper bound for  $p_1$  (i.e.  $\hat{p}_1$ ) can be computed with  $1 - \alpha/2$  certainty, using the so-called Clopper-Pearson Bound.
3. Now we have the lower bound of  $\epsilon$ , i.e.  $\epsilon_{LB} = \ln(\hat{p}_0/\hat{p}_1)/k$  with certainty  $1 - \alpha$

e.g.  $A$  could be DP-SGD

---

**Algorithm 2:** Empirically Measuring  $\epsilon$ 

---

**Data:** Algorithm  $\mathcal{A}$ , datasets  $D_0, D_1$  at distance  $k$ , output set  $\mathcal{O}$ , trial count  $T$ , confidence level  $\alpha$

$ct_0 = 0, ct_1 = 0$

**For**  $i \in [T]$

**If**  $\mathcal{A}(D_0) \in \mathcal{O}$   $ct_0 = ct_0 + 1$   
    **If**  $\mathcal{A}(D_1) \in \mathcal{O}$   $ct_1 = ct_1 + 1$

} Monte Carlo

$\hat{p}_0 =$

CLOPPERPEARSONLOWER( $ct_0, T, \alpha/2$ )

$\hat{p}_1 =$

CLOPPERPEARSONUPPER( $ct_1, T, \alpha/2$ )

**Return**  $\epsilon_{LB} = \ln(\hat{p}_0/\hat{p}_1)/k$

---

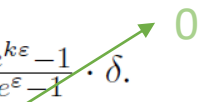
See interpretation of bullet point 3 on next slide.

Note that the perturbed dataset  $D_1$  is generated by a data poisoning attack, which will be introduced later.

# 1. Estimate $\epsilon_{LB}$ (Cont.)

With the following lemma,  $\epsilon_{LB}$  can be computed given  $p_0, p_1$  :

**Lemma 1** (Group Privacy). Let  $D_0, D_1$  be two datasets differing on at most  $k$  rows,  $\mathcal{A}$  is an  $(\epsilon, \delta)$ -differentially private algorithm, and  $\mathcal{O}$  an arbitrary output set. Then

$$\underbrace{\Pr[\mathcal{A}(D_0) \in \mathcal{O}]}_{p_0} \leq e^{k\epsilon} \underbrace{\Pr[\mathcal{A}(D_1) \in \mathcal{O}]}_{p_1} + \frac{e^{k\epsilon}-1}{e^\epsilon-1} \cdot \delta. \quad (2)$$


Group privacy will give guarantees for poisoning attacks that introduce multiple points.

The author mainly considered the case when  $\delta$  is 0. Thus we have:

$$\epsilon \geq \frac{1}{k} \ln \frac{p_0}{p_1} \quad (*)$$

, which implies  $\mathcal{A}$  is not  $\epsilon' - DP$  for any  $\epsilon' < \frac{1}{k} \ln p_0/p_1$ .

According to the Clopper-Pearson bound, with probability at least  $1 - \alpha$ ,  $\widehat{p}_0 \leq p_0$  and  $\widehat{p}_1 \geq p_1$ , which implies

$$\frac{p_0}{p_1} \geq \frac{\widehat{p}_0}{\widehat{p}_1} \quad (**)$$

Combining (\*) and (\*\*),  $\mathcal{A}$  is not  $\epsilon' - DP$  for any  $\epsilon' < \frac{1}{k} \ln \widehat{p}_0/\widehat{p}_1$ .

Therefore we have  $\epsilon_{LB} = \ln(\widehat{p}_0/\widehat{p}_1)/k$

Quote from the original paper:

*Proof of Theorem 2.* First, the guarantee of the Clopper-Pearson confidence intervals is that, with probability at least  $1 - \alpha$ ,  $\widehat{p}_0 \leq p_0$  and  $\widehat{p}_1 \geq p_1$ , which implies  $p_0/p_1 \geq \widehat{p}_0/\widehat{p}_1$ . Second, if  $\mathcal{A}$  is  $\epsilon$ -DP, then by group privacy we would have  $p_0/p_1 \leq \exp(k\epsilon)$ , meaning  $\mathcal{A}$  is *not*  $\epsilon'$ -DP for any  $\epsilon' < \frac{1}{k} \ln(p_0/p_1)$ . Combining the two statements,  $\mathcal{A}$  is *not*  $\epsilon'$  for any  $\epsilon' < \frac{1}{k} \ln(\widehat{p}_0/\widehat{p}_1) = \epsilon_{LB}$ .  $\square$

## 2. A novel poisoning attack

The back door attack is used to construct the dataset  $D_0$ ,  $D_1$  and the output set  $O$  in Algorithm 2.

### Problem with the existing data poisoning attacks:

they are vulnerable against not only the Gaussian perturbation in DP-SGD, but also against gradient clipping.

Quote from the paper:

*“We find that existing data poisoning attacks, as well as membership inference attacks proposed by prior work, **have poor or trivial performance not only against DP-SGD, but even against SGD with gradient clipping** (i.e. rescaling gradients to have norm no larger than some  $C$ ). Gradient clipping is an important part of DP-SGD, but does not provide any formal privacy guarantees on its own. Thus, we develop a novel data poisoning attack that is more robust to gradient clipping, and also performs much better against DP-SGD.”*

### Analysis of the problem:

existing attacks modify the model's gradient in a random direction.

Quote from the paper:

*“Intuitively, data poisoning attacks introduce new points whose gradients will change the model in a certain direction, and the attack impact increases when adding poisoning points of larger gradients. **Existing attacks modify the model in a random direction**, and have to push far enough that the original distribution on model parameters and the new distribution become distinguishable. To be effective, these attacks use points which induce large gradients, making the attack sensitive to gradient clipping.”*

## 2. A novel poisoning attack

### Proposed solution to the problem:

Design a perturbation which pushes the gradient towards the direction of lowest variance.

Quote from the paper:

*“our attack improves by finding the direction where the model parameters have the lowest variance, and select poisoning points that modify the model in that direction. Therefore, we achieve the same effect of model poisoning with poisoning points of smaller gradients, thereby making the attack more robust to clipping.”*

*“To be more effective in the presence of clipping, the attack must produce not only large gradients, but distinguishable gradients. That is, the distribution of gradients arising from poisoned and cleaned data must be significantly different. To analyze distinguishability, we consider the variance of gradients, illustrated in Figure 1, and seek a poisoning point  $(x_p; y_p)$  minimizing  $\text{Var}_{(x,y) \in D} [\nabla_w \ell(w, b; x_p, y_p) \cdot \nabla_w \ell(w, b; x, y)]$ ”*

The above minimization problem can be solved with approximation using SVD, for more details, see Section 3.2 in the paper. The resulting poisoning attack is shown on next slide.

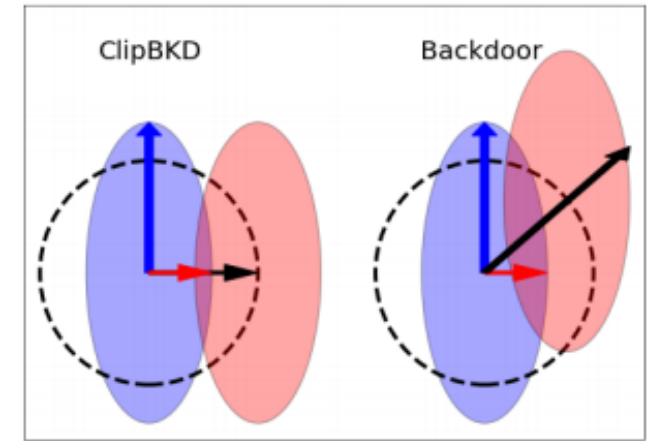


Figure 1: The distribution of gradients from an iteration of DP-SGD under a clean dataset (blue ellipse) and a poisoned dataset (red ellipse). The right pair depicts traditional backdoors while the left pair depicts our backdoors. Our attack pushes in the direction of least variance, so is impacted less by gradient clipping, which is indicated by the two distributions overlapping less.

## 2. A novel poisoning attack

### The original poisoning attack:

No specific way to generate the perturbed data  $X'_p$ , and thus the model parameters trained on the perturbed dataset  $D_1$  do not differ significantly with the model parameters trained on the clean dataset  $D_0$ .

The perturbation function  $Pert()$  adds a pattern in the corner of an image. The poisoning attack takes clean data  $(x; y)$ , perturbs the image to  $Pert(x)$ , and changes the class to some  $y_p$ .

### The proposed poisoning attack:

The perturbation is designed to maximize the change on the model parameters in terms of maximize the perturbed change on the updating gradient: select  $x_p$  as the singular vector corresponding to the smallest singular value and scale  $x_p$  to a similar norm to the rest of the dataset; select  $y_p$  to be the smallest probability class on  $x_p$ ; then insert  $k$  copies of the poisoning point  $(x_p; y_p)$

---

**Algorithm 3:** Baseline Backdoor Poisoning Attack and Test Statistic (Section 3.1)

---

**Data:** Dataset  $X, Y$ , poison size  $k$ , perturbation function  $Pert$ , target class  $y_p$

**Function** BACKDOOR( $X, Y, k, Pert, y_p$ ):

$X_p = \text{GETRANDOMROWS}(X, k)$

$X'_p = Pert(X_p)$

$X_{tr}^p = \text{REPLACERANDOMROWS}(X, X'_p)$

$Y_{tr}^p = \text{REPLACERANDOMROWS}(Y, y_p)$

**return**  $D_0 = (X, Y), D_1 = (X_{tr}^p, Y_{tr}^p)$

**Data:** Model  $f$ , dataset  $(X, Y)$ , pert. function  $Pert$ , target class  $y_p$ , loss function  $\ell$ , threshold  $Z$

**Function** BACKDOORTEST( $f, X, Y, Pert, y_p, \ell, Z$ ):

$X_p = Pert(X)$

**If**  $\sum_{x_p \in X_p} \ell(f; (x_p, y_p)) > Z$  **Return** Backdoored

**Return** Not Backdoored

---

Backdoor: The original backdoor poisoning attack.

---

**Algorithm 4:** Clipping-Aware Backdoor Poisoning Attack and Test Statistic (Section 3.2)

---

**Data:** Dataset  $X, Y$ , pretrained model  $f$ , poison size  $k$ , dataset dimension  $d$ , norm  $m$

**Function** CLIPBKD( $X, Y, k, f, m$ ):

$U, D, V = \text{SVD}(X)$

▷ Singular value decomposition

$x_p = mV_d$

▷  $V_d$  is the singular vector for smallest singular value

$y_p = \arg \min_i f(x_p)$

▷ Pick class maximizing gradient norm

$X_{tr}^p = \text{REPLACERANDOMROWS}(X, [x_p] * k)$

▷ Add poisoning point  $k$  times

$Y_{tr}^p = \text{REPLACERANDOMROWS}(Y, [y_p] * k)$

▷ Add targeted class  $k$  times

**return**  $D_0 = (X, Y), D_1 = (X_{tr}^p, Y_{tr}^p)$

**Data:** Model  $f$ , Poison Data  $x_p, y_p$ , Threshold  $Z$

**Function** CLIPBKDTEST( $f, x_p, y_p, Z$ ):

**If**  $(f(x_p) - f(0^d)) \cdot y_p > Z$  **Return** Backdoored

**Return** Not Backdoored

---

ClipBKD: The proposed backdoor poisoning attack.



# 3. Experiment results

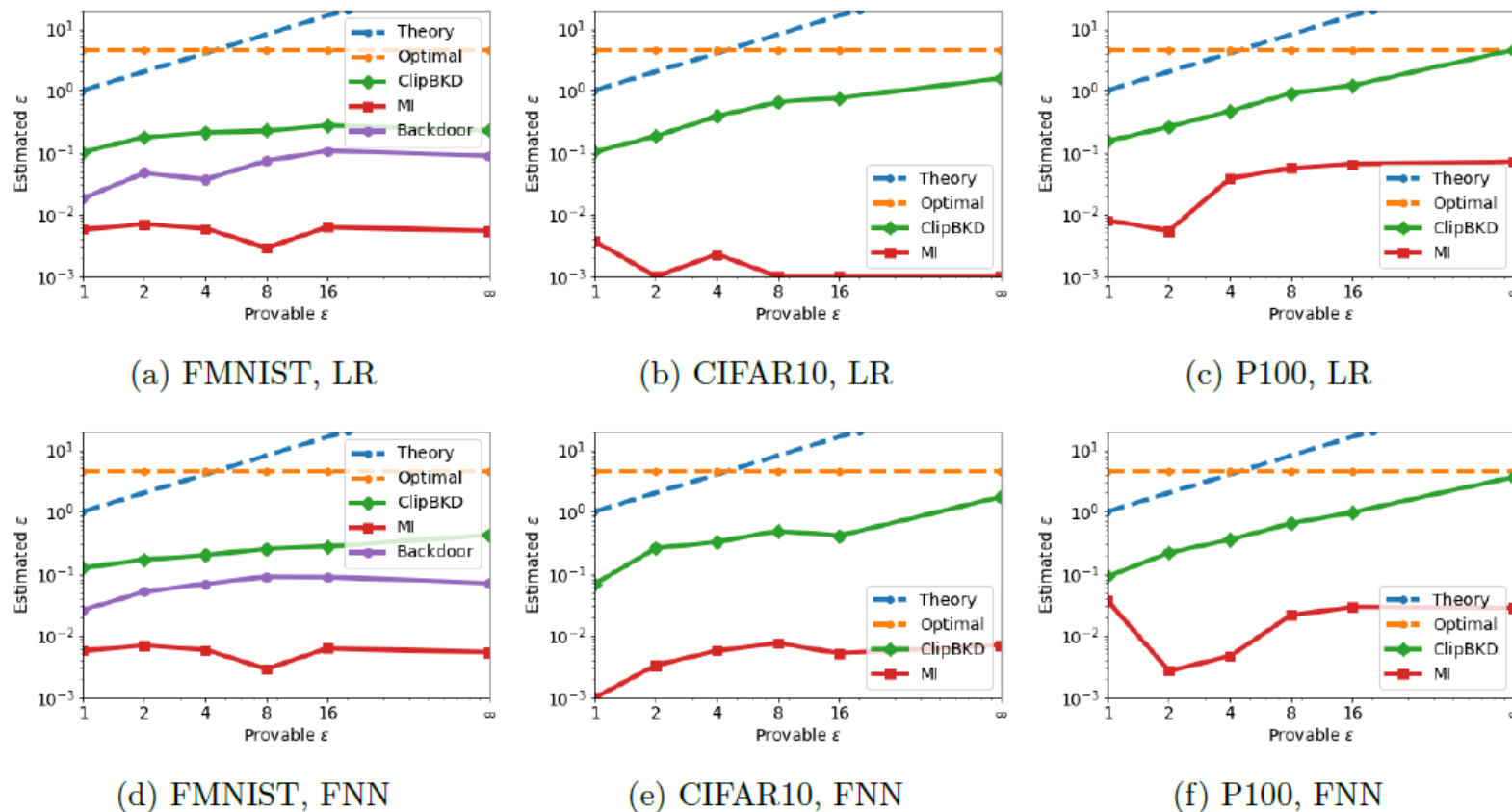


Figure 2: Performance of privacy attacks MI, Backdoor, and ClipBKD on our datasets. LR = logistic regression, FNN = two-layer neural network. Backdoor attacks have not been developed for Purchase-100, so only MI and Clip-BKD were run. Backdoors do not provide positive  $\epsilon_{LB}$  on CIFAR10 due to difficulty with the pretrained model.

The results show that, ClipBKD gives a larger  $\epsilon_{LB}$ , leading to tighter bound of  $\epsilon$

x axis: provable  $\epsilon$ , the theoretical upper bound of  $\epsilon$   
y axis: the estimated  $\epsilon_{LB}$  given by algorithm 2 (see slide 3)

The blue “Theory” curve is the  $y=x$  line.

The yellow “optimal” line is the maximal  $\epsilon_{LB}$  which algorithm 2 can output, i.e. when we can perfectly distinguish clean data and attacked data, i.e  $ct_0 = T$  and  $ct_1 = 0$

The MI, Backdoor and ClipBKD curve are obtained by train DP-SGD with linear regression or a two-layer neural network, under different DP-SGD parameters, compute the corresponding provable  $\epsilon$  and estimated  $\epsilon_{LB}$



# Conclusion

- The novel ClipBKD can give better lower bound of  $\epsilon$

The rationale:

ClipBKD produces more powerful perturbations on the dataset, i.e D0 and D1 are very different

=> the trained models on clean and perturbed dataset differ a lot, i.e.  $A(D0)$  and  $A(D1)$  are very different, where  $A$  is the training mechanism, e.g. DP-SGD

=> a larger  $\epsilon$  is needed to guarantee  $\epsilon - DP$ , i.e better  $\epsilon_{LB}$  is obtained