

Differentially Private Machine Learning

Gautam Kamath

University of Waterloo

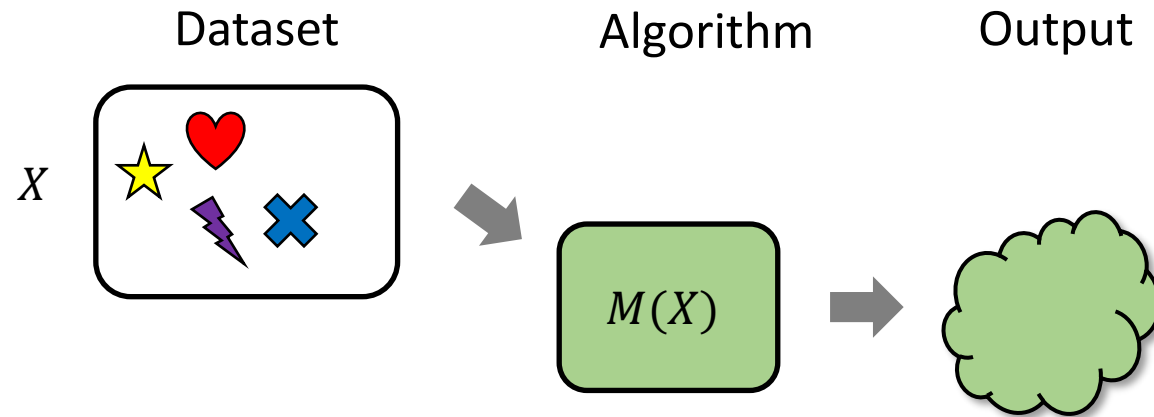


CPSC 471/571: Trustworthy Deep Learning

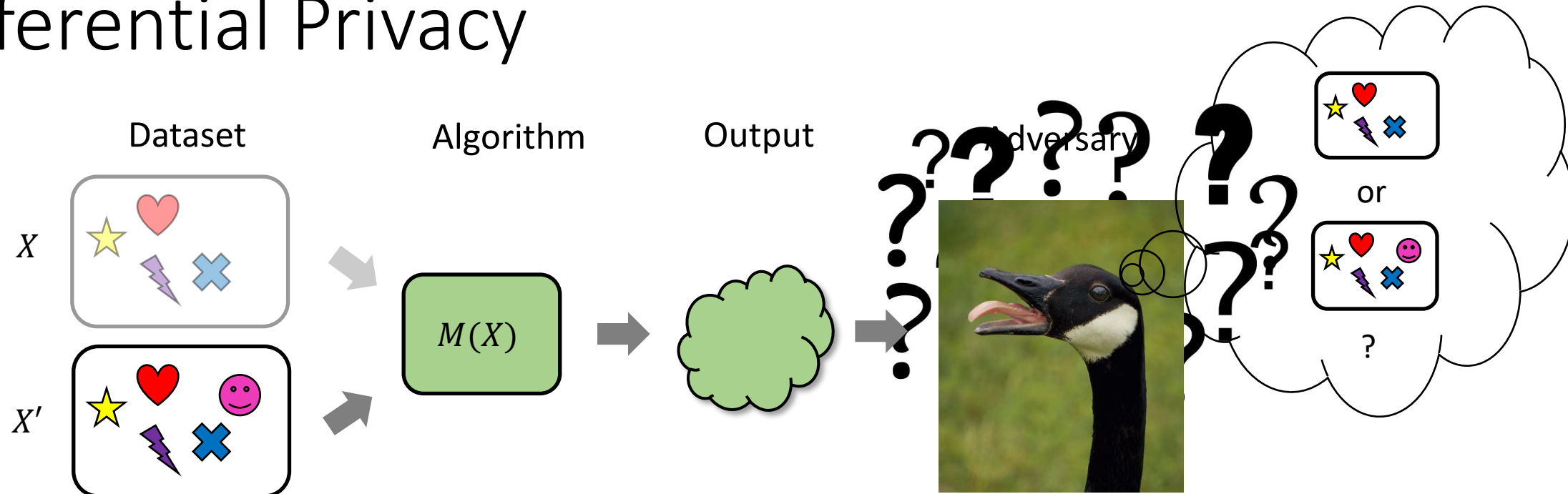
April 1, 2024

Intro, Basic Tools, and Properties

Differential Privacy



Differential Privacy



“An algorithm is differentially private if its distribution over outputs doesn’t change much after adding/removing one point.”

Differential Privacy

- Setup: n datapoints $X = X_1, \dots, X_n$, given to a “trusted curator”
- The curator has an algorithm $M : \mathcal{X}^n \rightarrow \mathcal{Y}$, outputs $M(X)$

Definition: An algorithm M is (ϵ, δ) -differentially private (DP) if for all datasets X and X' which differ in one entry (“neighbouring”), and for all events $S \subseteq \mathcal{Y}$,

$$\Pr[M(X) \in S] \leq e^\epsilon \Pr[M(X') \in S] + \delta.$$

“An algorithm is differentially private if its distribution over outputs doesn’t change much after adding/removing one point.”

What does DP protect against?

- Learning anything about a user that can't be inferred w/o them
 - Database reconstruction
 - Membership inference

What *doesn't* DP do?

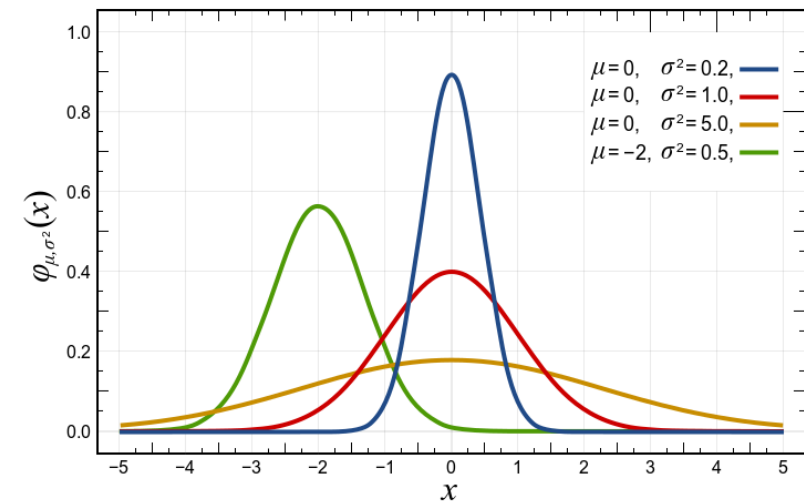
- Important: does **not** prevent inferences (statistics/machine learning)
 - (Public) smoker participates in (differentially private) study investigating whether smoking causes cancer
 - Reveals that smoking causes cancer! Smoker's insurance premiums increase!
 - Was their (differential) privacy violated?
 - No: smoking → cancer could be inferred whether or not they participated
 - Differential privacy: outcome of algorithm is similar, whether or not someone participates
- Not appropriate when individual identities are important
 - “Private” contact tracing

The convenience of differential privacy

- A rigorous and quantitative definition
- Toolbox of algorithmic building blocks
- Nice definitional properties that make it easy to put them together

The Gaussian Mechanism

- Let $f(X) : \mathcal{X}^n \rightarrow \mathbf{R}^d$ be a function of interest
- Let $\Delta_2^f = \max_{X, X' \text{ differ in one entry}} \|f(X) - f(X')\|_2$ be ℓ_2 -sensitivity of f
 - “How much can the function change by modifying one datapoint?”
- Theorem (roughly):



The **Gaussian** Mechanism $f(X) + N\left(0, \left(\frac{\Delta_2^f \log(1/\delta)}{\epsilon}\right)^2\right)^{\otimes d}$ is (ϵ, δ) -DP

- “Add **Gaussian** noise to each coordinate, proportional to the ℓ_2 -sensitivity”

Post-Processing

- You can't “undo” privatization of something which is released differentially privately
- Post-processing theorem: Let $M : \mathcal{X}^n \rightarrow \mathcal{Y}$ be (ϵ, δ) -differentially private, and $F : \mathcal{Y} \rightarrow \mathcal{Z}$ be an arbitrary randomized mapping. Then $F \circ M$ is (ϵ, δ) -differentially private

Composition

- Answering multiple questions about the same sensitive dataset will “intuitively” leak more privacy
 - We know “more things” about the same dataset
- Differential privacy allows us to quantify!
- “Basic” Composition Theorem: Let $M = (M_1, \dots, M_k)$ be a sequence of (ϵ, δ) -differentially private algorithms. Then M is $(k\epsilon, k\delta)$ -differentially private.
- If you release multiple differentially private statistics of a dataset, the privacy parameters “add up”
- Composition allows us to build complex procedures out of basic tools

Advanced Composition

- Basic Composition Theorem: Let $M = (M_1, \dots, M_k)$ be a sequence of (ϵ, δ) -differentially private algorithms. Then M is $(k\epsilon, k\delta)$ -differentially private.
 - If you do k private analyses, you pay k times the privacy cost of one analysis
- Advanced Composition Theorem (informal): Let $M = (M_1, \dots, M_k)$ be a sequence of (ϵ, δ) -differentially private algorithms. Then M is $(O(\sqrt{k}\epsilon), k\delta)$ -differentially private.
 - If you do k private analyses, you pay \sqrt{k} times the privacy cost of one analysis
- 10,000 queries (advanced comp) vs 100 queries (basic comp)

Basic tools

- Gaussian Mechanism
 - Add Gaussian noise to every coordinate of a function, proportional to sensitivity
- Post-Processing
 - You can't “undo” privatization
- Composition
 - It's possible to account for the privacy cost when doing multiple private queries on the same dataset

Differentially Private SGD

Gradient Descent

- The standard method* to train ML models (non-privately)
- Given: Dataset (x_i, y_i) for $i \in [n]$
 1. Compute the average gradient $\frac{1}{n} \sum_{i=1}^n \nabla \ell(\theta_t, x_i, y_i)$
 2. Take a step in the negative direction of the gradient
 3. Repeat k times

Differentially Private Gradient Descent (DPGD)

1. For each point (x_i, y_i) in the dataset, compute the gradient $\nabla \ell(\theta_t, x_i, y_i)$ and “clip” it to have ℓ_2 norm at most C

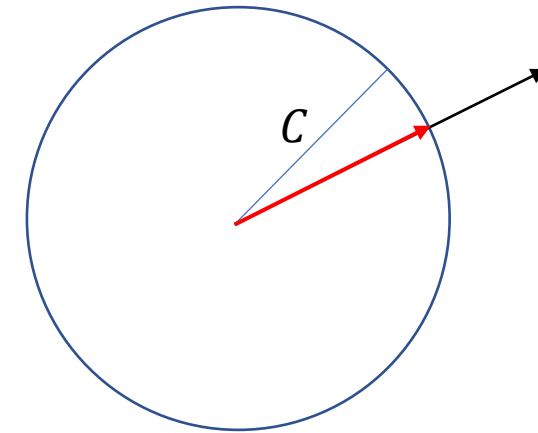
- Limits sensitivity

Gaussian Mechanism

2. Average the clipped gradients and add Gaussian noise

3. Take a step **(Advanced) Composition** altering vector

4. Repeat k times



- Very general! Works for privately training any ML model.
- In particular, works for non-convex models, e.g., neural networks

Analysis of DPGD

1. For each point (x_i, y_i) in the dataset, compute the gradient $\nabla \ell(\theta_t, x_i, y_i)$ and “clip” it to have ℓ_2 norm at most C
 - Limits sensitivity of the average gradient
2. Average the clipped gradients and add Gaussian noise

- Consider $f(X) = \frac{1}{n} \sum \text{clip}_C(\nabla \ell(\theta_t, x_i, y_i))$
- ℓ_2 -sensitivity: C/n
- Adding Gaussian noise $\approx N\left(0, \left(\frac{C\sqrt{\log(1/\delta)}}{n\varepsilon}\right)^2\right)$ guarantees (ε, δ) -DP (Gaussian Mechanism)
 - Adding Gaussian noise $\approx N\left(0, \left(\frac{C\sqrt{k \log(k/\delta)}}{n\varepsilon}\right)^2\right)$ guarantees $(\varepsilon/\sqrt{k}, \delta/k)$ -DP

Analysis of DPGD

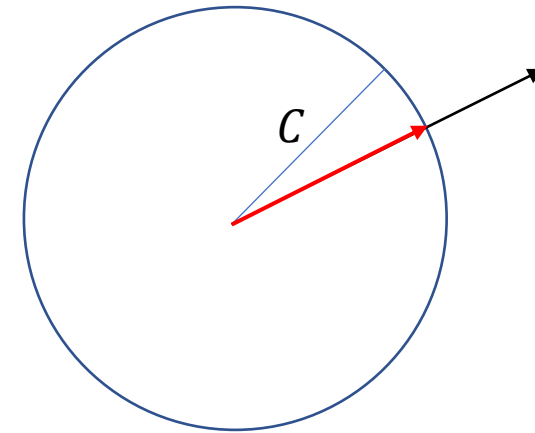
- Suppose one step of DPGD is $(\varepsilon/\sqrt{k}, \delta/k)$ -DP
 - Gaussian Mechanism
- k steps of DPGD will be $\approx (\varepsilon, \delta)$ -DP
 - Advanced composition
- Releasing final model is private
 - Post-processing
- Surprisingly simple!
 - Relies upon simple building blocks in DP
- Problem: full-batch gradient descent is slow...

Stochastic Gradient Descent (SGD)

1. Choose a random minibatch B of points from the dataset
2. Compute the average gradient $\frac{1}{|B|} \sum_{(x,y) \in B} \nabla \ell(\theta_t, x, y)$
3. Take a step in the negative direction of the gradient
4. Repeat k times

Differentially Private **Stochastic** Gradient Descent (DP**S**GD)

1. **Sample a batch of points of size B**
2. For each point in the lot, compute the gradient $\nabla \ell(\theta_t, x, y)$ and “clip” it to have ℓ_2 norm at most C
3. Average the clipped gradients and add Gaussian noise
 - Apply the Gaussian Mechanism
4. Take a step in the negative direction of resulting vector
5. Repeat k times

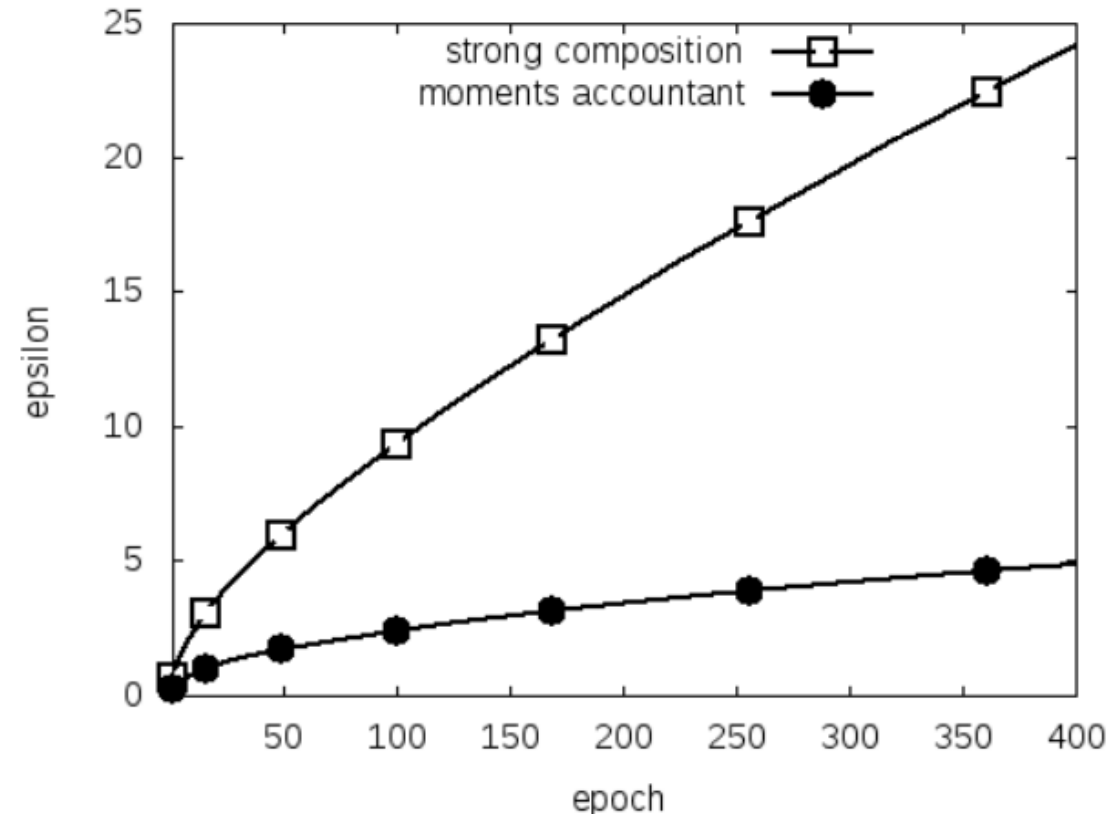


Analysis of DP^SGD

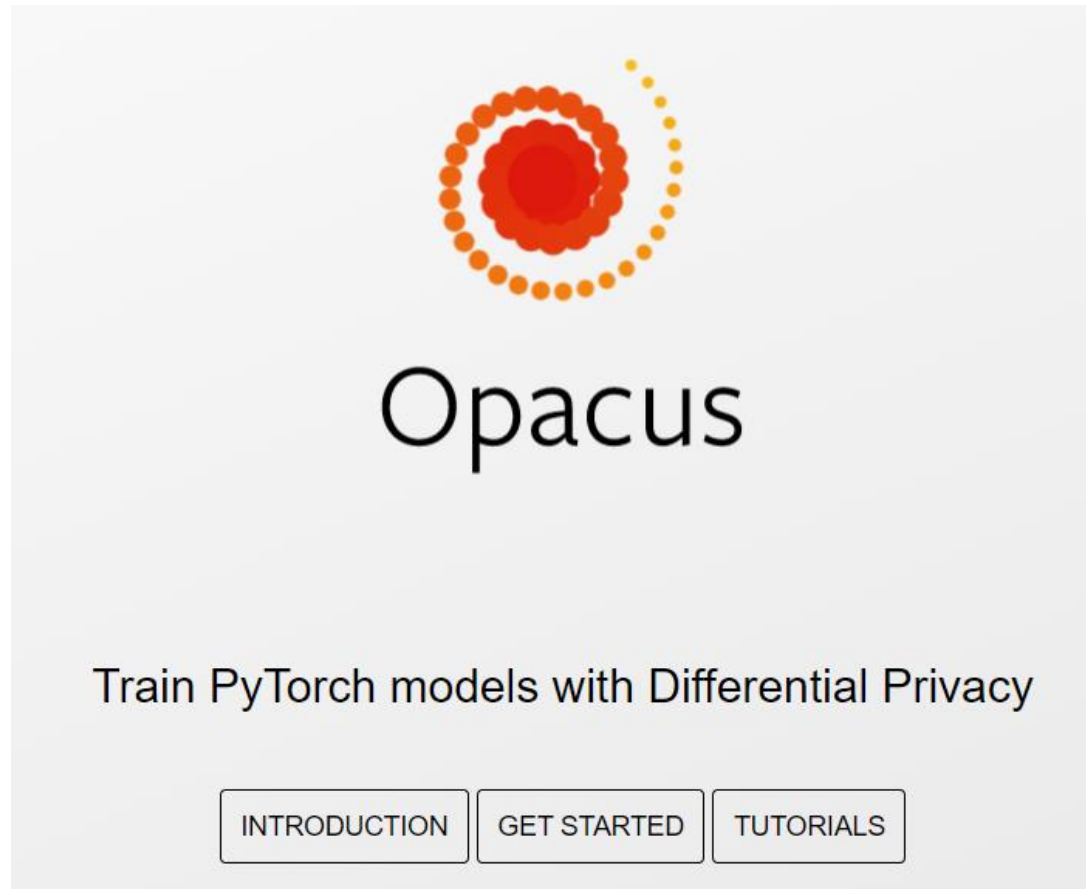
- Suppose one step of DPGD is (ϵ, δ) -DP
 - Gaussian Mechanism
- Choosing a random batch of size L makes it $\approx \left(\frac{\epsilon B}{n}, \frac{\delta B}{n}\right)$ -DP
 - Privacy amplification by subsampling
- k steps of DPGD will be $\approx \left(\frac{\epsilon \sqrt{k} L}{n}, \frac{\delta' L}{n}\right)$ -DP
 - Advanced composition

Better Analysis

- “Moments accountant”
 - Track moments of privacy loss RV, choose best one
 - “Rényi DP”
- Even better analysis using FFT
 - Computes exact privacy guarantees



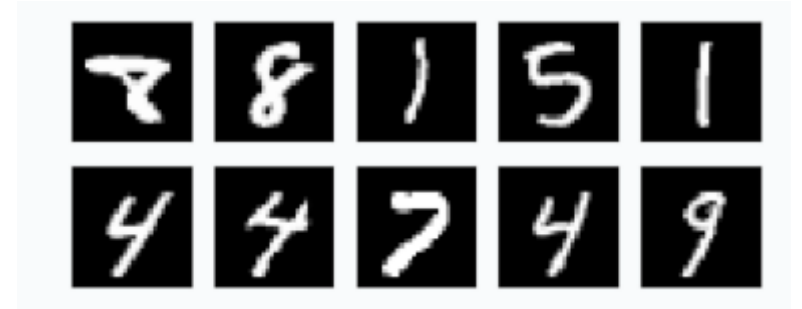
Mature tools for private ML



Introducing TensorFlow Privacy: Learning with Differential Privacy for Training Data

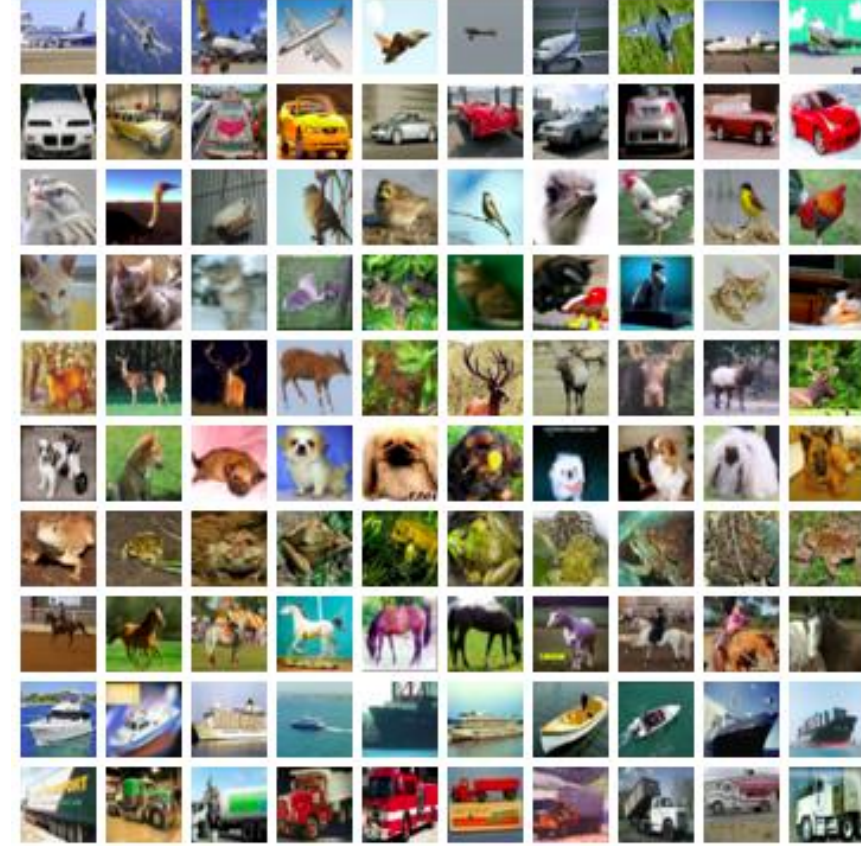
Does it work?

- MNIST: black and white image classification
 - Canonical “easy” ML task
- Non-private test accuracy: $\approx 100\%$
- Private (ϵ from 1 to 3): **98% - 99%**
 - [Tramer-Boneh, '21]
- Works pretty well for “easy” datasets!



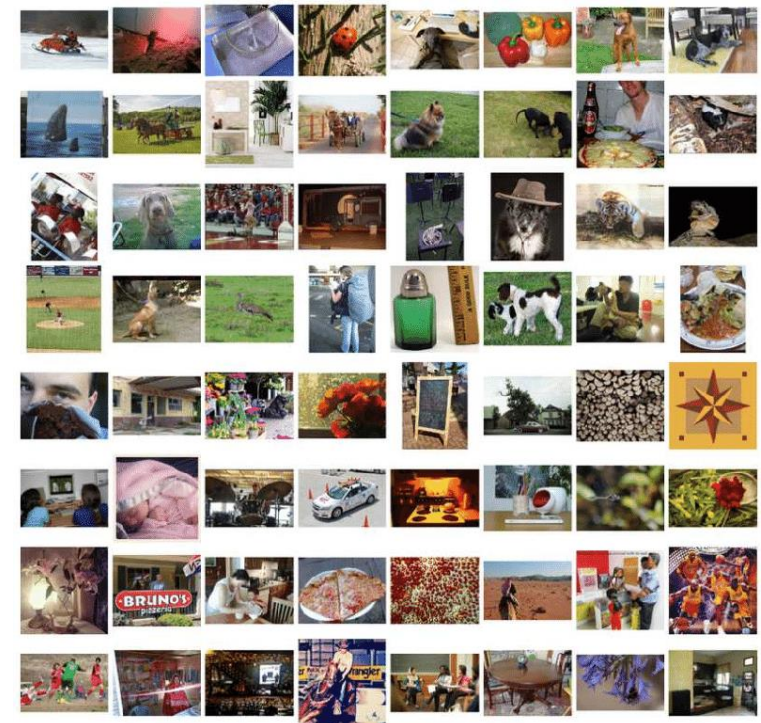
Does it work?

- CIFAR-10: Low resolution images
 - Same size as MNIST, but harder
- Non-privately: 98%+
- Privately ($\epsilon = 3$): 69%
 - [Tramer-Boneh, '21]
 - Much worse!
- Recent results: 73.5% for $\epsilon = 4$ and 82.5% for $\epsilon = 8$
 - [De-Berrada-Hayes-Smith-Balle, '22], [Klause-Ziller-Rueckert-Hammernik-Kaissis, '22]



Does it work?

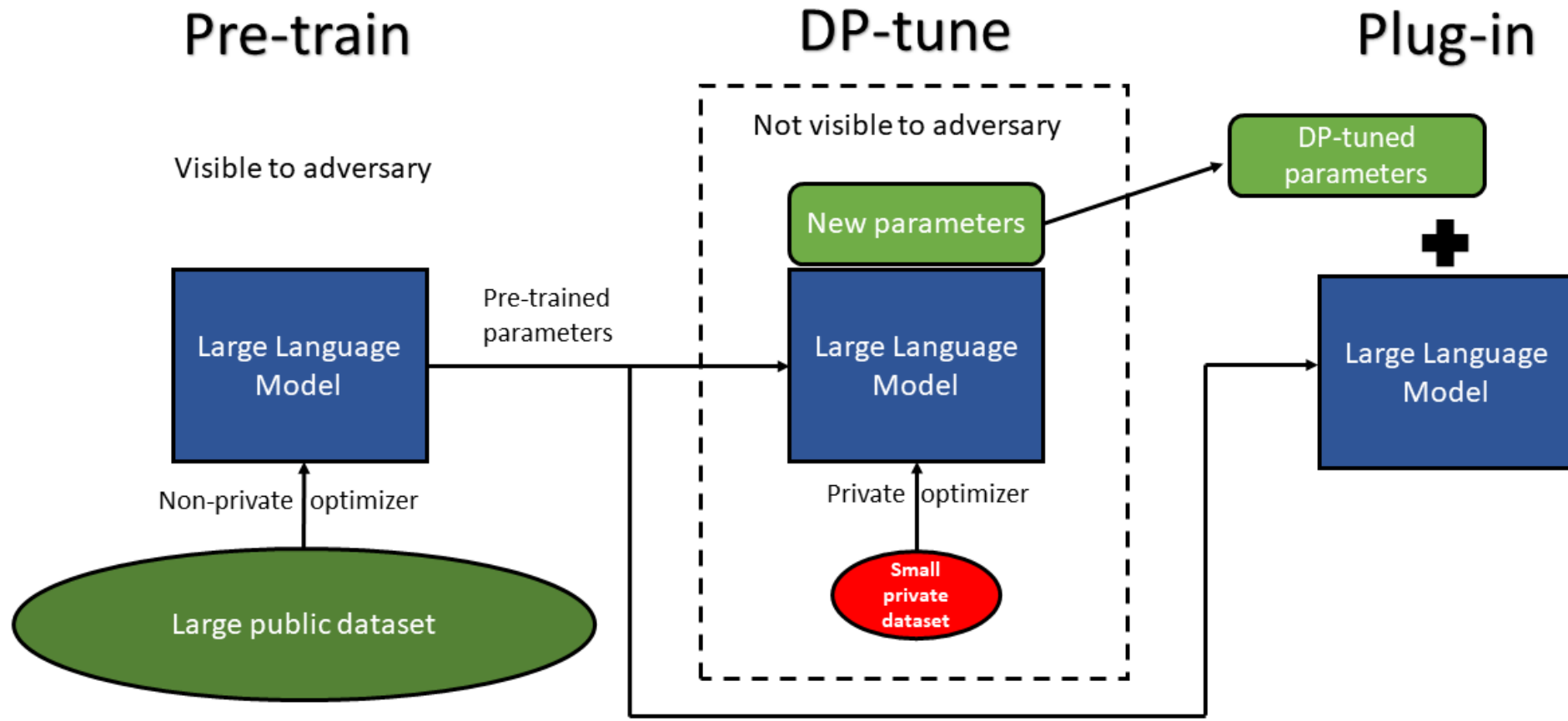
- ImageNet: a hard dataset
 - Millions of higher resolution images, 1000 classes
- Non-privately: $\sim 87\%$
- Privately ($\epsilon = 8$): **32.4%**
 - [De-Berrada-Hayes-Smith-Balle, '22]
 - Very tough to achieve (compute, expertise, etc.)
- Not quite there yet...



Public Data for Private ML

- Pretrain the model with public data
 - Very large amounts of (public) data
 - Often scraped from the Internet
 - E.g., Google Image search results
- Fine-tune model (privately) with sensitive data
 - Smaller and task specific dataset
 - E.g., chest x-ray images

Using public data




Using public data (NLU on RoBERTa-Large)

- Can train privately while approaching the non-private accuracy
 - Language model setting
 - [Yu, Naik, Backurs, Gopi, Inan, K., Kulkarni, Lee, Manoel, Wutschitz, Yekhanin, Zhang, '22], [Li, Tramer, Liang, Hashimoto, '22]

Method		MNLI	SST-2	QQP	QNLI	Avg.	Trained params
Full	w/o DP	90.2	96.4	92.2	94.7	93.4	100%
RGP	DP	86.1	93.0	86.7	90.0	88.9	100%
Adapter	DP	87.7	93.9	86.3	90.7	89.7	1.4% ($r = 48$)
Compacter	DP	87.5	94.2	86.2	90.2	89.5	0.053% ($r = 96, n = 8$)
LoRA	DP	87.8	95.3	87.4	90.8	90.3	0.94% ($r = 16$)

Using public data (CIFAR-10)

Dataset	Weaker privacy 			
	Top-1 Accuracy (%)			
	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$
CIFAR-10	56.8	65.9	73.5	81.4

Dataset	Pre-Training Data	Top-1 Accuracy (%)				δ	Section
		$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$		
CIFAR-10	ImageNet	94.7	95.4	96.1	96.7	10^{-5}	4.1
CIFAR-100	ImageNet	70.3	74.7	79.2	81.8	10^{-5}	4.1
ImageNet	JFT-4B	84.4	85.6	86.0	86.7	$8 \cdot 10^{-7}$	4.2
Places-365	JFT-300M	–	–	–	55.1	$5 \cdot 10^{-7}$	4.3

Using public data (ImageNet)

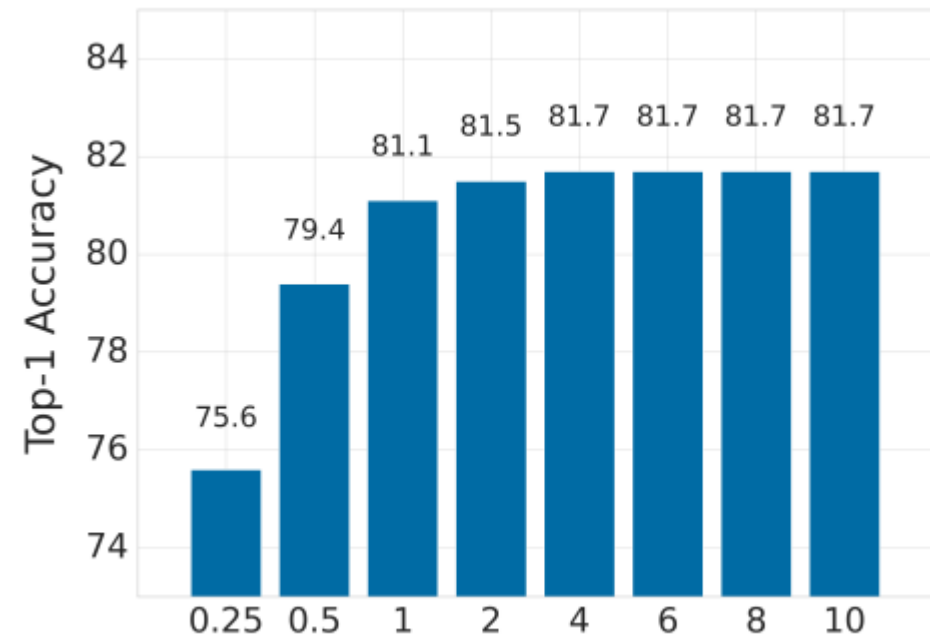
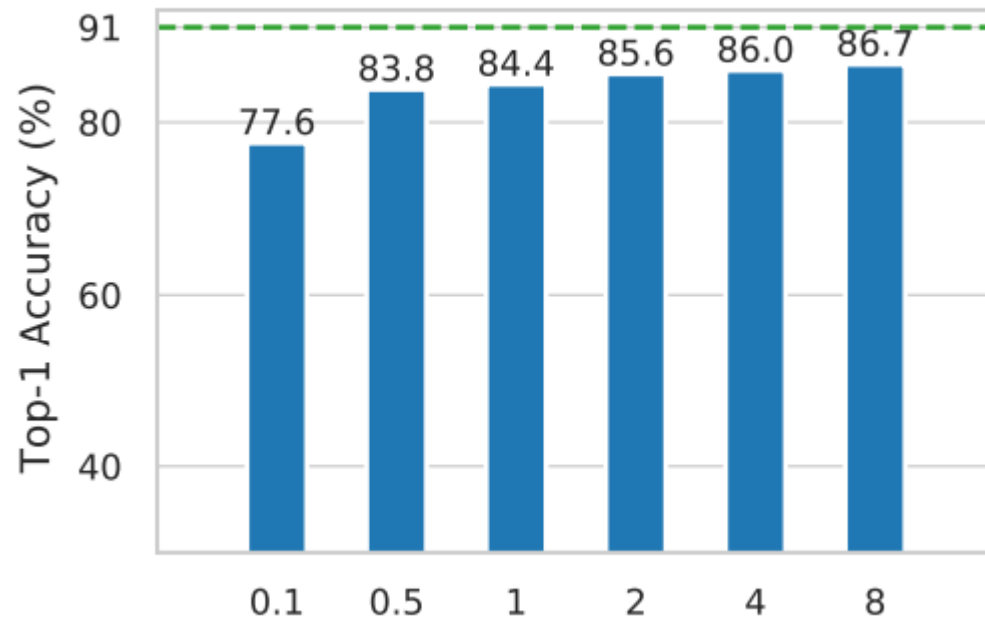
- [De, Berrada, Hayes, Smith, Balle, '22]

- Pretrain with JFT-4B

- [Mehta, Thakurta, Kurakin, Cutkosky, '22]

- Pretrain with JFT-4B

“public” dataset = proprietary Google dataset!



Public pre-training and private fine-tuning has some important caveats [Tramer, K., Carlini], 2022

Private Aggregation of Teacher Ensembles (PATE)

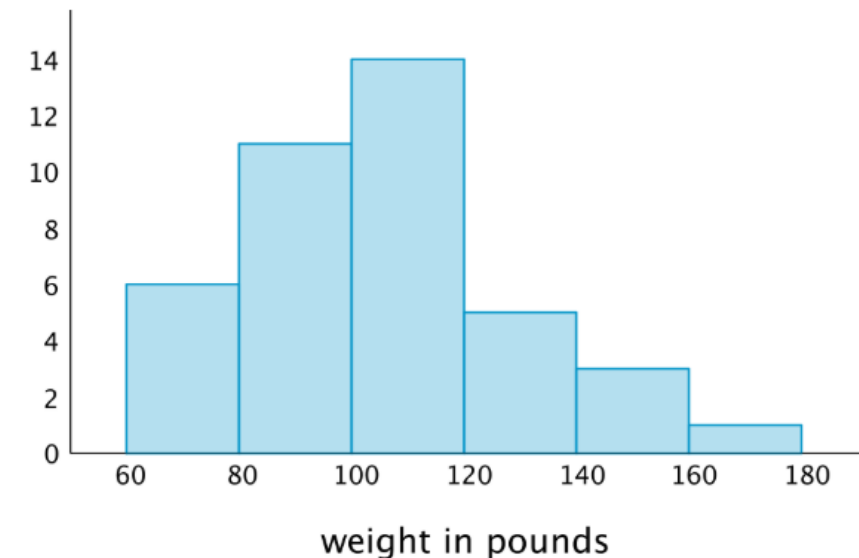
How to make a single prediction privately?

“Sample and aggregate” paradigm

1. Split data into k parts. Train a model (non-privately) on each.
 - Note: each datapoint affects only a single model. Limits sensitivity.
2. Privately aggregate results of the k models to generate a prediction.

$$f(x) = \arg \max_j \{n_j(x) + \text{Laplace}(1/\gamma)\}$$

Return majority vote of the k trained models



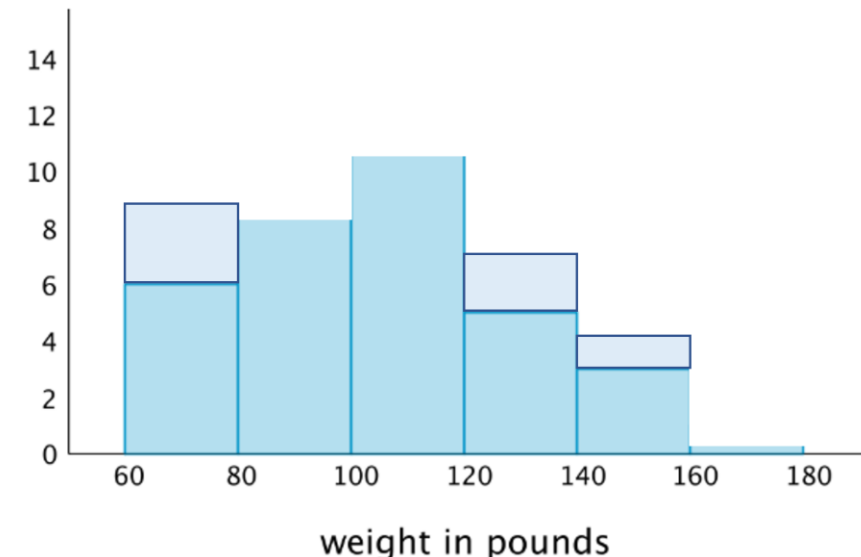
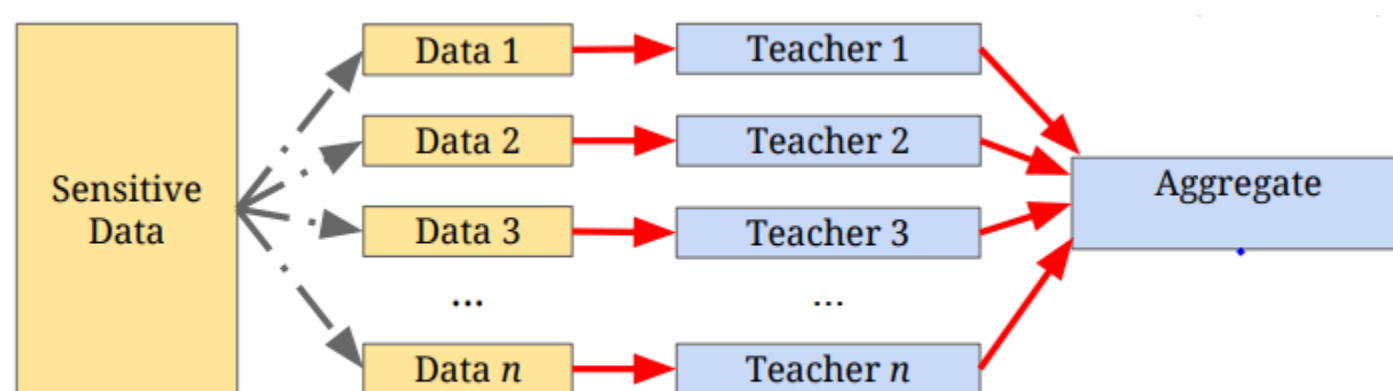
How to make a single prediction privately?

“Sample and aggregate” paradigm

1. Split data into k parts. Train a model (non-privately) on each.
 - Note: each datapoint affects only a single model. Limits sensitivity.
2. Privately aggregate results of the k models to generate a prediction.

$$f(x) = \arg \max_j \{n_j(x) + \text{Laplace}(1/\gamma)\}$$

Return **noisy** majority vote of the k trained models

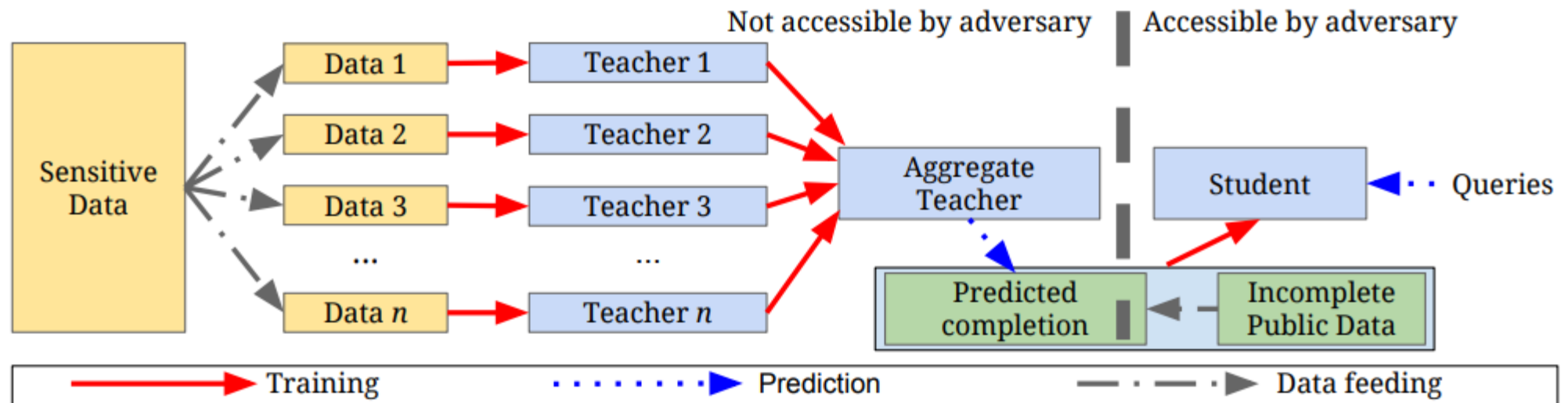


Caveats

- This allows us to do *one* query. How do we account for many queries?
 - Composition!
- But composition costs us: the more queries, the more noise for each
 - Limits how many classifications you can perform
 - Compare with DPSGD: can perform infinitely many classifications after train

PATE

- Given: sensitive, labeled data AND public, unlabeled data
1. Get private predictions for all public, unlabeled data
 2. Use resulting labeled data to train a student model



PATE Results

Dataset	ϵ	δ	Queries	Non-Private Baseline	Student Accuracy
MNIST	2.04	10^{-5}	100	99.18%	98.00%
MNIST	8.03	10^{-5}	1000	99.18%	98.10%
SVHN	5.04	10^{-6}	500	92.80%	82.72%
SVHN	8.19	10^{-6}	1000	92.80%	90.66%

Best paper award, ICLR 2017

Auditing

Privacy Auditing

- Differential privacy guarantees
 - Of the form: “For all possible input datasets, we have $\varepsilon \leq 4.52$ ”
 - Mathematically proven guarantees
 - Worst case over all possible datasets and events
 - May be overly pessimistic
- Privacy auditing
 - Of the form “Looking at this pair of datasets X and X' and event E , we have $\varepsilon \geq 1.58$ ”
 - Empirically demonstrated
 - Construct a pair of datasets and some particular event
- If privacy proof (UB on ε) matches auditing (LB on ε), analysis is tight

Intuition

Definition: An algorithm M is ε -differentially private (DP) if for all datasets X and X' which differ in one entry (“neighbouring”), and for all events $S \subseteq \mathcal{Y}$,

$$\Pr[M(X) \in S] \leq e^\varepsilon \Pr[M(X') \in S].$$

- Fix some value of ε . How can we show that this guarantee is false?
- Specify some neighbouring datasets X and X' , some event S . Show that the inequality is not true in this case.
- Compute $\Pr[M(X) \in S]$, $e^\varepsilon \Pr[M(X') \in S]$, compare
- Can't compute probabilities explicitly: estimate by running repeatedly

Auditing Algorithm

Algorithm 2: Empirically Measuring ε

Data: Algorithm \mathcal{A} , datasets D_0, D_1 at distance 1, output set \mathcal{O} , trial count T , confidence level α

Neighbouring datasets

Event of interest

$ct_0 = 0, ct_1 = 0$

For $i \in [T]$

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

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

Run the algorithm on both datasets many times, count how many times the event happens under each

$\hat{p}_0 = \text{CLOPPERPEARSONLOWER}(ct_0, T, \alpha/2)$

$\hat{p}_1 = \text{CLOPPERPEARSONUPPER}(ct_1, T, \alpha/2)$

Draw some confidence intervals around these estimates (account for error due to randomness)

Return $\varepsilon_{LB} = \ln(\hat{p}_0/\hat{p}_1)$

Rearrange to find out what value of ε is realized in this instance

How to choose datasets X, X' , event S ?

- We can choose any neighbouring datasets X and X' , and any event S
 - Differential privacy is worst-case over *all* possibilities
- How do we choose?
 - Try to make probability of event S as discrepant as possible
- Use ideas from backdoor data poisoning attacks
- Dataset X is natural (e.g., MNIST, CIFAR-10)
- Dataset X' has a single training image x backdoored
 - E.g., add a pattern to the corner of x
- Event S : Compare loss of model $M(X)$ on x versus loss of model $M(X')$ on x . Intuition: latter should have much lower loss.

A case study

We present backpropagation clipping, a novel variant of differentially private stochastic gradient descent (DP-SGD) for privacy-preserving deep learning. Our approach clips each trainable layer's inputs (during the forward pass) and its upstream gradients (during the backward pass) to ensure bounded global sensitivity for the layer's gradient; this combination replaces the gradient clipping step in existing DP-SGD variants. Our approach is simple to implement in existing deep learning frameworks. The results of our empirical evaluation demonstrate that backpropagation clipping provides higher accuracy at lower values for the privacy parameter ϵ compared to previous work. We achieve 98.7% accuracy for MNIST with $\epsilon = 0.07$ and 74% accuracy for CIFAR-10 with $\epsilon = 3.64$.



A case study

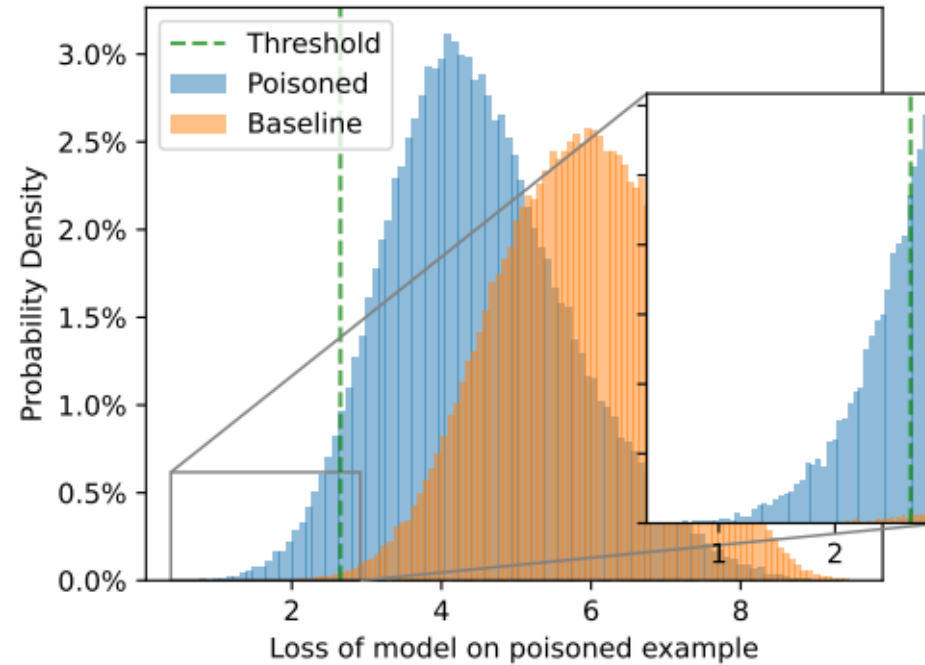


Figure 1: The distribution of loss values using a claimed $(0.21, 10^{-5})$ -DP algorithm. We train 100,000 models on one dataset D (in orange), and another 100,000 models on another $D' = D \cup \{x_p\}$ (in blue). With a threshold $\tau = 2.64$, our attack has a true positive rate of 4.922% and false positive rate of 0.174%. The Clopper-Pearson bounds allow us to show that $\epsilon > 2.79$ with probability at least $1 - 10^{-10}$. This refutes the claim the algorithm is in fact $(0.21, 10^{-5})$ -DP

Privacy Auditing in One Run

- Training 100,000 models is prohibitively expensive
- Ideal: can we just train the model *once*?
- Yes! **NeurIPS 2023 Outstanding Paper: Privacy auditing in just one run**

NeurIPS 2023 just wrapped up, and one of the two [outstanding paper awards](#) went to [Privacy Auditing with One \(1\) Training Run](#), by [Thomas Steinke](#), [Milad Nasr](#), and [Matthew Jagielski](#). The main result of this paper is a method for auditing the (differential) privacy guarantees of an algorithm, but much faster and more practically than previous methods. In this post, we'll dive into what this all means.

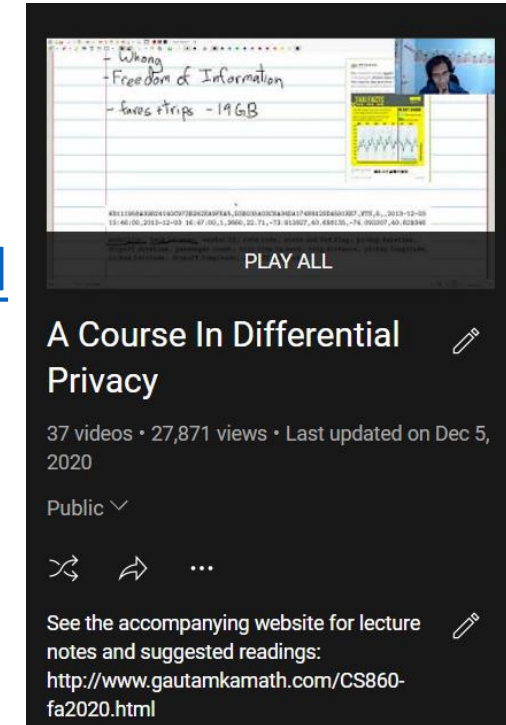
<https://differentialprivacy.org/neurips23-op/>

Overview

- Differentially Private SGD
- Public data for private ML
- PATE
- Auditing

For more, see

- My course
 - <http://www.gautamkamath.com/CS860-fa2020.html>
 - Full set of lecture notes and videos
- DifferentialPrivacy.org
 - Blog posts, as well as links to resources



DifferentialPrivacy.org