Deep Learning with Differential Privacy

Martin Abadi, Andy Chu, Ian Goodfellow et al, Google, 2016

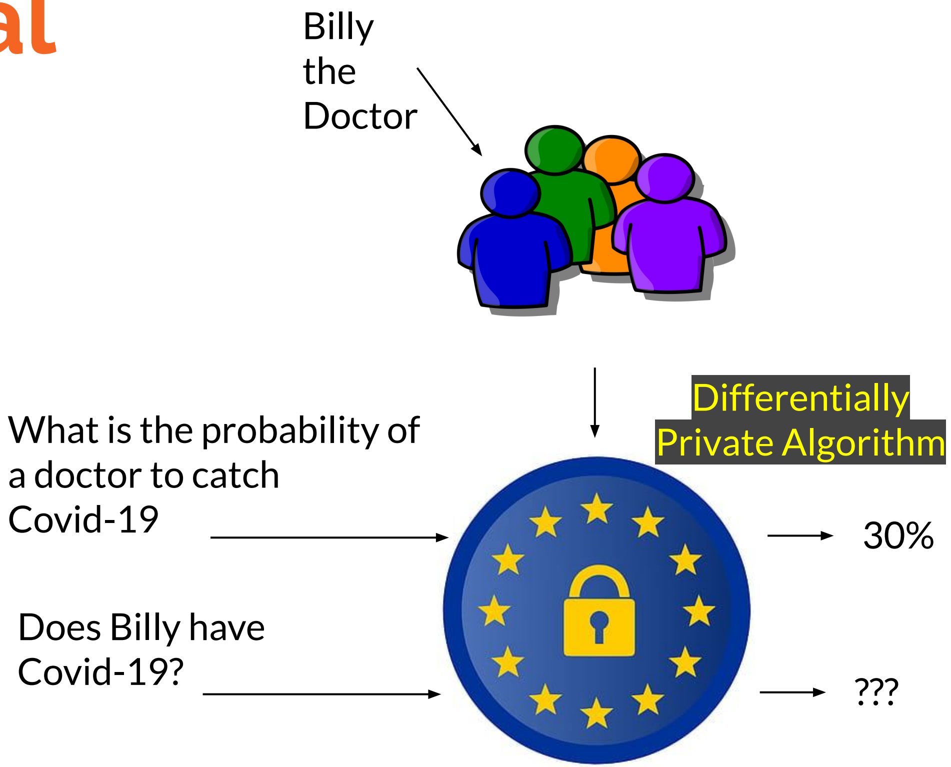
What we will talk about?

- ➔ **What is Differential Privacy**
  Why do i need it? What happens when we don't have it
- ➔ **The Basic Solution**
  Just add noise
- ➔ **How Abadi et al improved?**
  Moments Accountant
- ➔ **Results**

What is Differential Privacy?

- A definition which asserts the paradox of learning something useful about the population, without learning anything about the individual

- Learning statistical data without learning the specific examples



Definition

Given an algorithm that works on database {d}

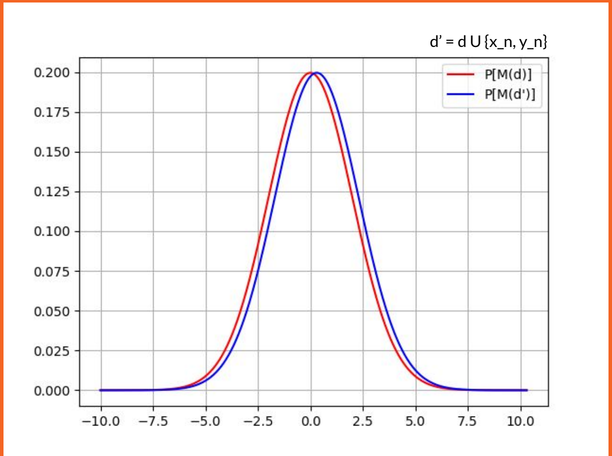
The algorithm is Differentially Private if with high probability the output doesn't change by changing one sample in the database

⇔

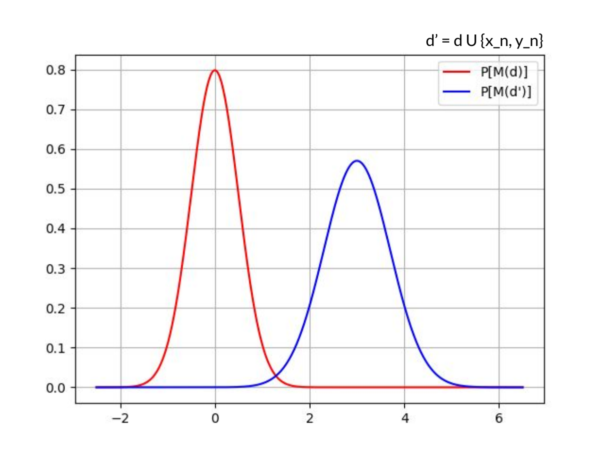
Definition 1. A randomized mechanism $\mathcal{M}\colon \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ε, δ)-differential privacy if for any two adjacent inputs $d, d' \in \mathcal{D}$ and for any subset of outputs $S \subseteq \mathcal{R}$ it holds that

$$\Pr[\mathcal{M}(d) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(d') \in S] + \delta.$$

Differentially Private Algorithm



Non Private Algorithm



Example

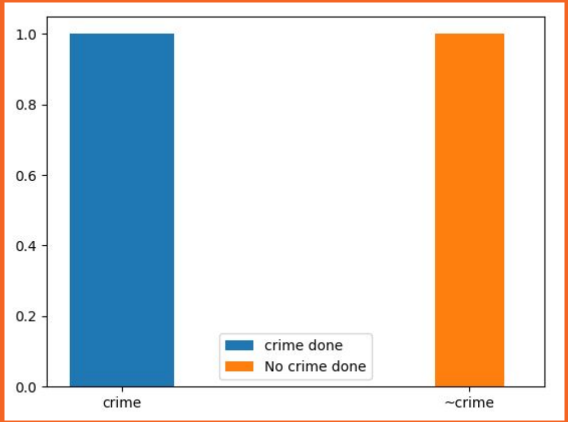
Say that we want people to tell if they did a felony

But we don't want them to incriminate themselves

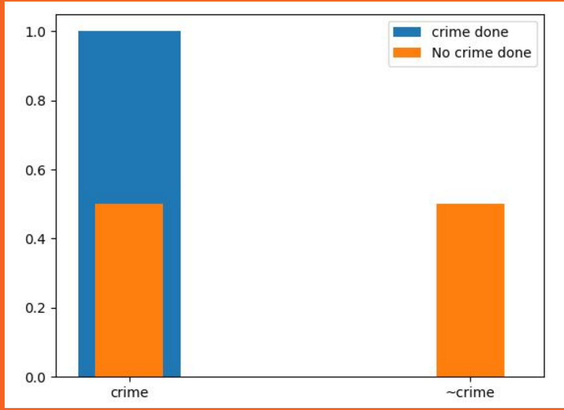
So we will tell them to toss a coin:

- if head -> Tell the truth
- If tail -> say you did the crime

Non Private Algorithm



Differentially Private Algorithm



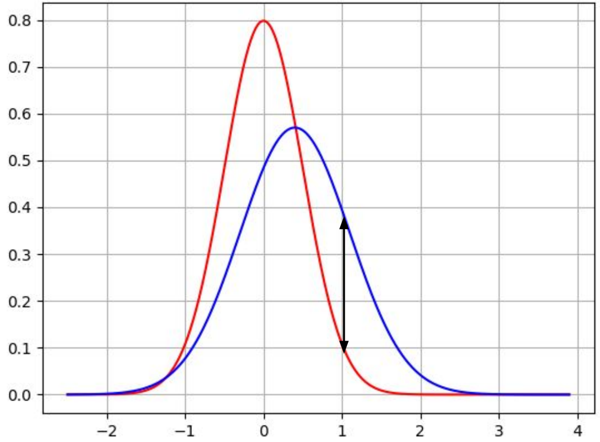
Privacy Loss

Privacy loss is a quantification of how much privacy do we lose when seeing an algorithm output.

Privacy Loss at o: $c(o; \mathcal{M}, \mathsf{aux}, d, d') \triangleq \log \frac{\Pr[\mathcal{M}(\mathsf{aux}, d) = o]}{\Pr[\mathcal{M}(\mathsf{aux}, d') = o]}.$

For ε,δ DP algorithm, privacy loss is bounded by ε

$$\Pr[\mathcal{M}(d) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(d') \in S] + \delta.$$



Why not just anonymize the data?

Top Secret

Name: [redacted]

Medical Condition: Very rare disease

Social Security Number: [redacted]

Date of Hospitalization: 04.02.20

“Only You, Your Doctor, and Many Others May Know” - L. Sweeny, 2015

De-identified public health records were linked together with newspaper reports to get the Massachusetts Governor health records

“Robust De-anonymization of Large Datasets” - A.Narayanan, V.Shmatikov, 2008

In 2006 Netflix released 100M records anonymized dataset created by 500K users. Writers used IMDB as side data and managed to associate records to specific users, thus gaining personal data

What it promises

- The utility of a participant will most likely won’t change by providing it’s personal data.

$$\begin{aligned}\mathbb{E}_{a\sim f(\mathcal{M}(x))}[u_i(a)] &= \sum_{a\in\mathcal{A}} u_i(a)\cdot \Pr_{f(\mathcal{M}(x))}[a] \\ &\le \sum_{a\in\mathcal{A}} u_i(a)\cdot \exp(\varepsilon)\Pr_{f(\mathcal{M}(y))}[a] \\ &= \exp(\varepsilon)\mathbb{E}_{a\sim f(\mathcal{M}(y))}[u_i(a)]\end{aligned}$$

- Immunity to Post-Processing

What it doesn’t promise

- Preventing the algorithm result from harming an individual. If the public results have high correlation with a personal data then a person can be harmed

  e.g. public results: smoking cause cancer with high probability, personal data: Timmy smokes. In this case an insurance company that knows Timmy smokes will also know it probably have cancer

How to (simply) make a mechanism differentially private?

Simple Solution Part 1: The Gaussian Mechanism

For function $f : \mathbb{N}^{|\mathcal{X}|} \rightarrow R$, the l_2 sensitivity is $\Delta_2 f = \max_{adjacent\ x,y} ||f(x) - f(y)||_2$

A Gaussian Mechanism output: $f(x) + \xi$ where $\xi \sim N(0, \sigma^2)$

if $\sigma \geq \frac{\sqrt{2\ln(\frac{1.25}{\delta})}\Delta_2 f}{\epsilon}$ then GM is (ϵ, δ) private

Simple Solution Part 2: Composability

Basically say that you can concatenate differentially private functions

if $f : \mathbb{N}^{|x|} \rightarrow R^m$ is (ϵ, δ) private, and $g : R^m \rightarrow R^d$ is (ϵ, δ) private,

then $g \circ f$ is $(2\epsilon, 2\delta)$ private.

Differential Privacy in Neural Networks

What is the mechanism we will make private?

- The network
- The learning algorithm

Model inversion attack by M.Fredrikson et al, 2015

<u>We want the weights to be private</u>

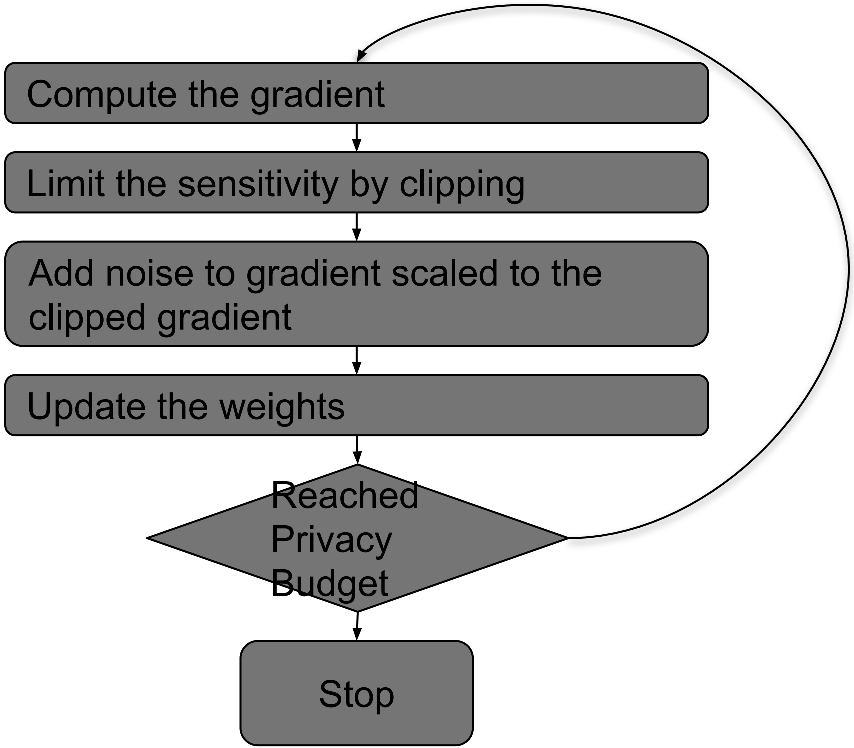
Simple Solution

Make θ’ = θ + ∇ℒ(θ, x) Differentially Private

Use Gaussian Mechanism + Composability

Sensitivity = |Gradient|, can be infinitely high, so clip it.

Compute gradient
For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
Clip gradient
$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
Add noise
$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$
Descent
$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$



Abadi et al

Algorithm 1 Differentially private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/\max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

The magic is in bounding the noise needed

Abadi et al

Algorithm 1 Differentially private SGD (Outline)

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

Using Simple Solution

$$\sigma \geq qT\sqrt{2\ln(\frac{1.25qT}{\delta})}/\epsilon$$

Using Strong Composition Theorem

$$\sigma = \Omega(q\sqrt{T\log(1/\delta)\log(T/\delta)}/\varepsilon)$$

Paper Result - Using Moments Accountant

$$\sigma \geq c_2\frac{q\sqrt{T\log(1/\delta)}}{\varepsilon}.$$

Log Moment of Privacy Loss

Privacy Loss at o

$$c(o; \mathcal{M}, \mathsf{aux}, d, d') \triangleq \log \frac{\Pr[\mathcal{M}(\mathsf{aux}, d) = o]}{\Pr[\mathcal{M}(\mathsf{aux}, d') = o]}.$$

Log Moment of Privacy loss

$$\alpha_{\mathcal{M}}(\lambda; \mathsf{aux}, d, d') \triangleq \log \mathbb{E}_{o \sim \mathcal{M}(\mathsf{aux}, d)}[\exp(\lambda c(o; \mathcal{M}, \mathsf{aux}, d, d'))].$$

Maximum over all adjacent data bases

$$\alpha_{\mathcal{M}}(\lambda) \triangleq \max_{\mathsf{aux}, d, d'} \alpha_{\mathcal{M}}(\lambda; \mathsf{aux}, d, d'),$$

Proof Sketch

➔ Privacy Loss' Log Moment is Composable

➔ We can bound the Log Moment for each step

➔ Find Condition on Total Log Moment such that algorithm is private

➔ Get a bound on the noise needed

$$\alpha_{\mathcal{M}}(\lambda) \leq \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda)$$

$$\alpha_{\mathcal{M}}(\lambda) \leq \frac{q^2\lambda(\lambda+1)}{(1-q)\sigma^2} + O(q^3\lambda^3/\sigma^3)$$

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\varepsilon)$$

$$\sigma \geq c_2 \frac{q\sqrt{T\log(1/\delta)}}{\varepsilon}$$

Results On MNIST

60 - dimensional PCA input layer

1000-unit ReLU Hidden Layer

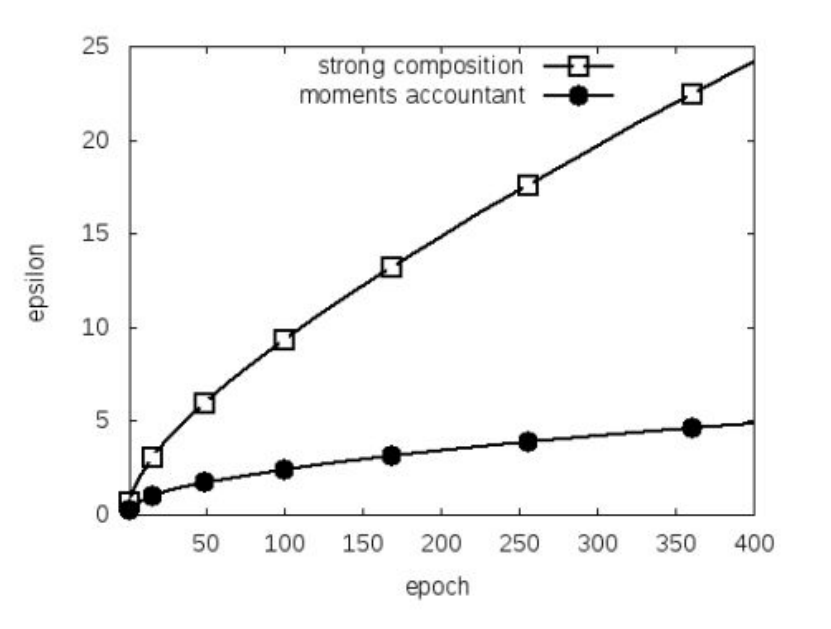
10 Outputs

MNIST Results

Comparing Moments Accountant and Strong Composition

Privacy Loss (ε) as function of Epochs.

Moments Accountant allows more Epochs for the same budget!

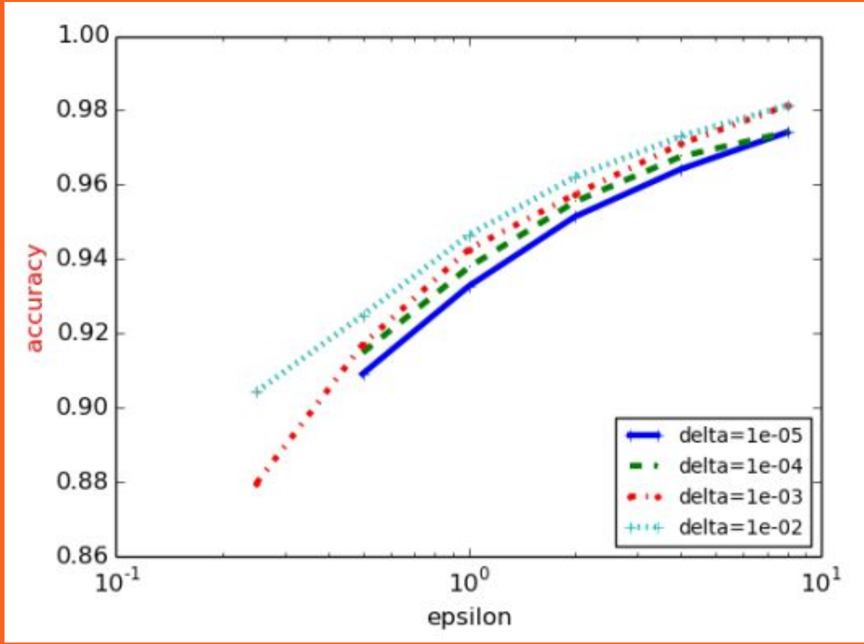


MNIST Results

Best Accuracy, as function of ε, δ

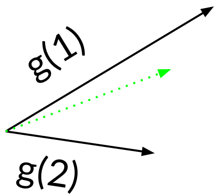
Non private version
Reaches **98.3%**

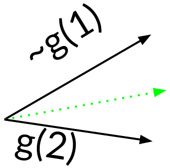
Private Version:
δ = 0.01, ε = 0.25 reaches **90%**

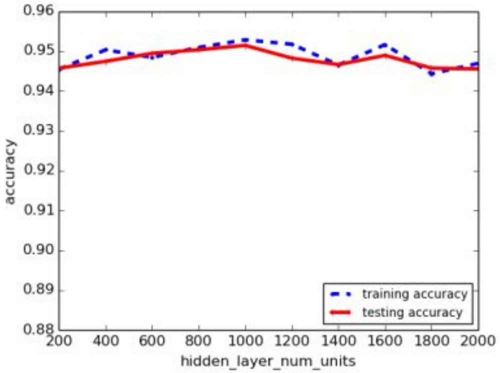


More MNIST Results

Stops training at (2, 10^-5) Differentially Privacy



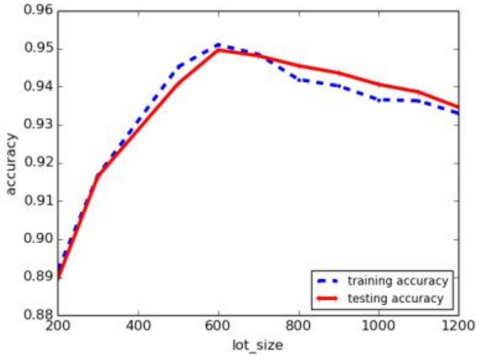




Hidden Layer

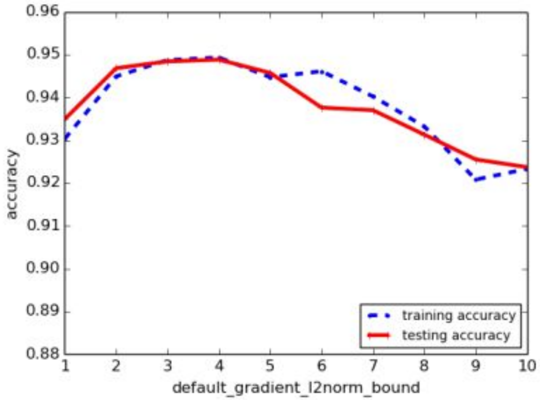
Hidden layer size doesn't affect accuracy by much

Might be because more units increase the sensitivity of the gradient



Lot Size

Best lot size is N^0.5
Balance between # of epochs and effect of noise



Clipping Threshold

$$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i)/\max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$$

$$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$$

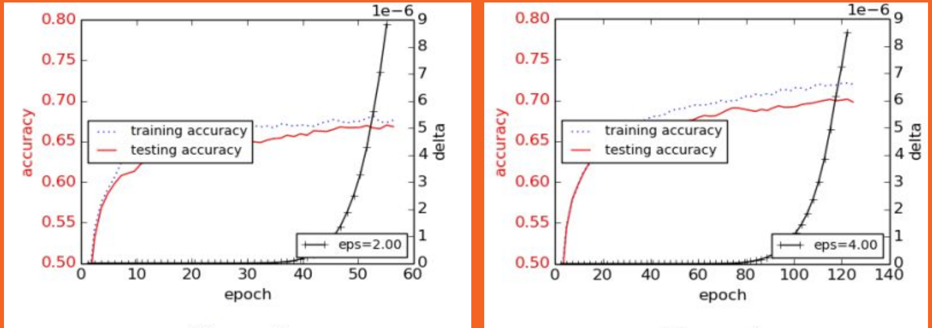
Destroy Direction/Add more noise

Results On CIFAR-10

- 2 Convolutional Layers
  - 5x5, stride 1, 64 Channels
  - ReLU
  - MaxPool
- 2 Fully Connected Layers
  - 384 Units

Interesting Parts

- Use CIFAR-100 as a “Public Dataset” to learn more easily
- Small ε brings test and training accuracy to be close ⇔ DP generalize well



(1) $\varepsilon = 2$ (2) $\varepsilon = 4$

Questions?