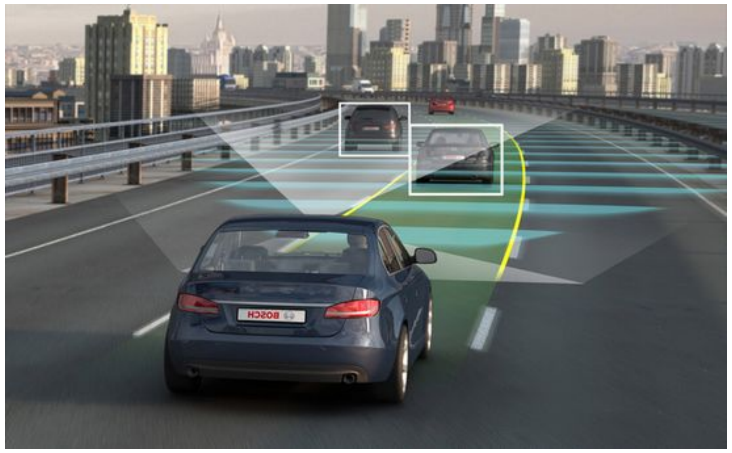
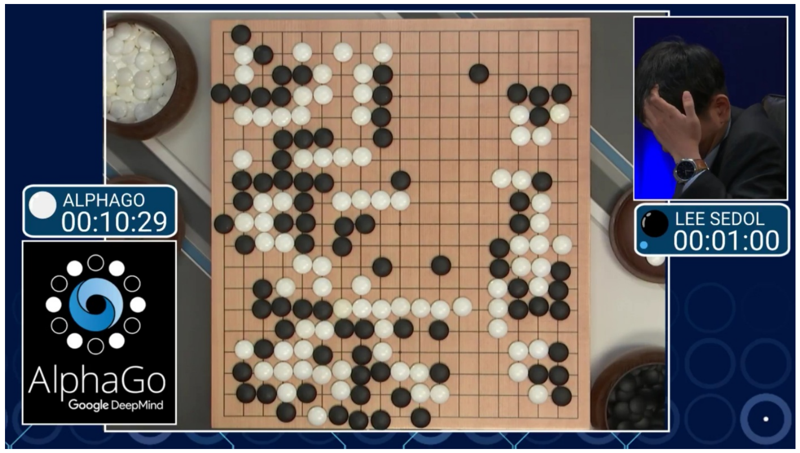
Deep Learning With Differential Privacy

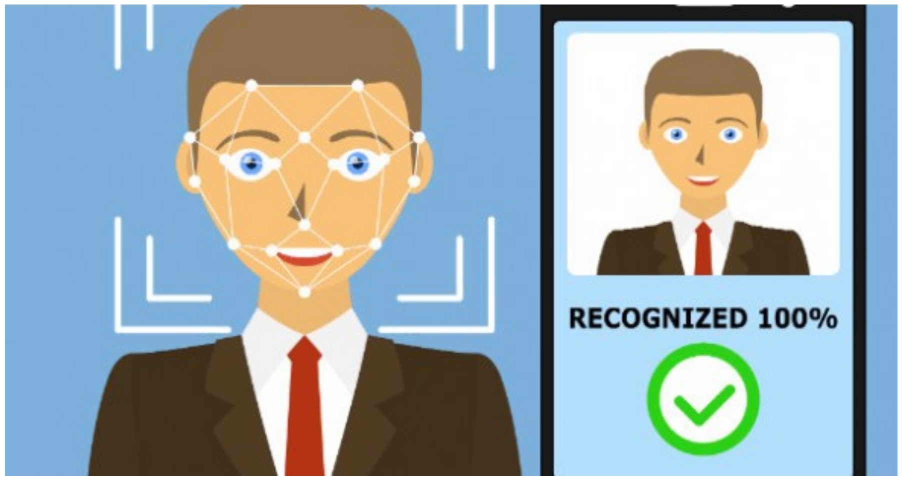
Presenter: Xiaojun Xu

Deep Learning Framework


Autonomous Driving



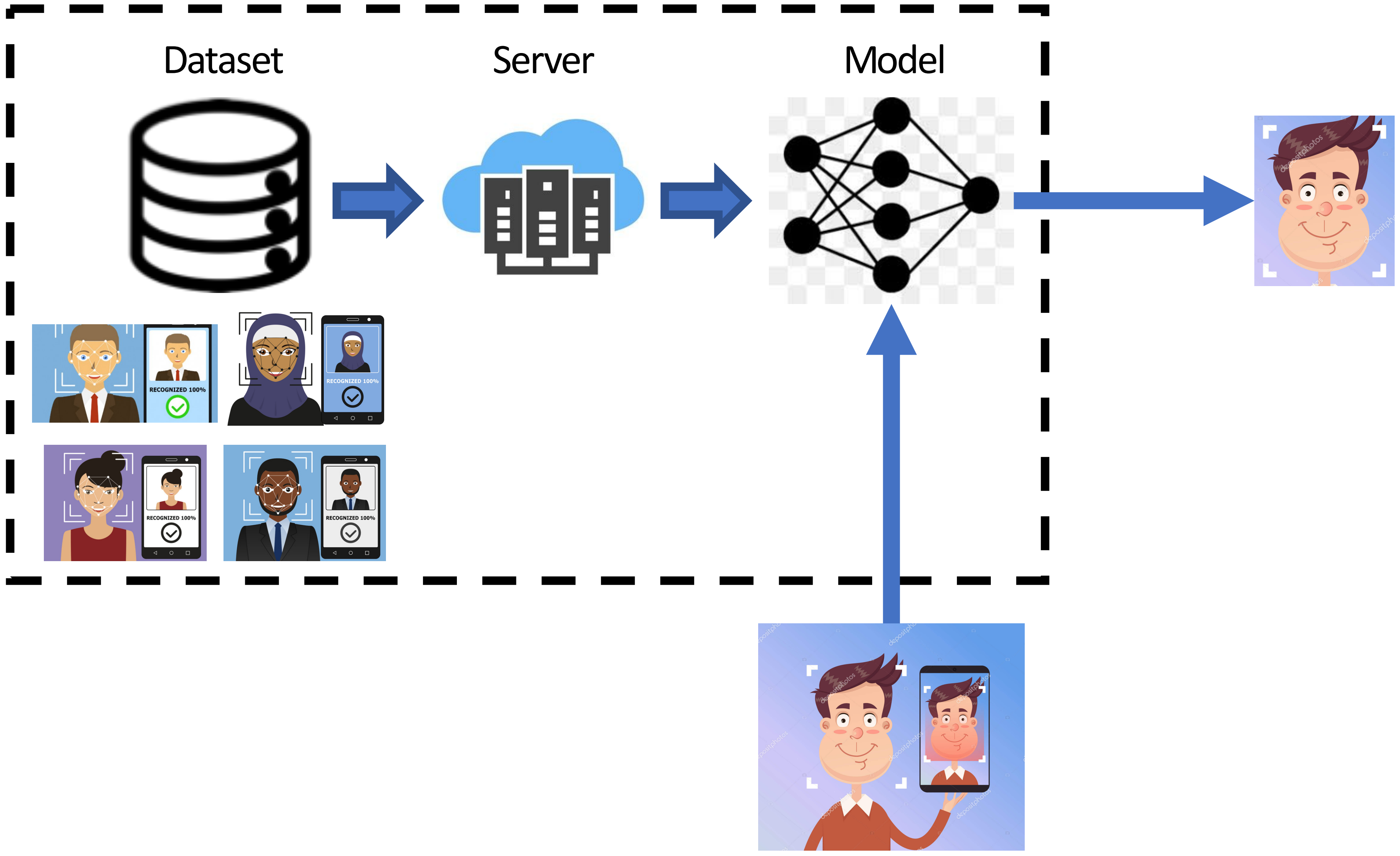
Gaming



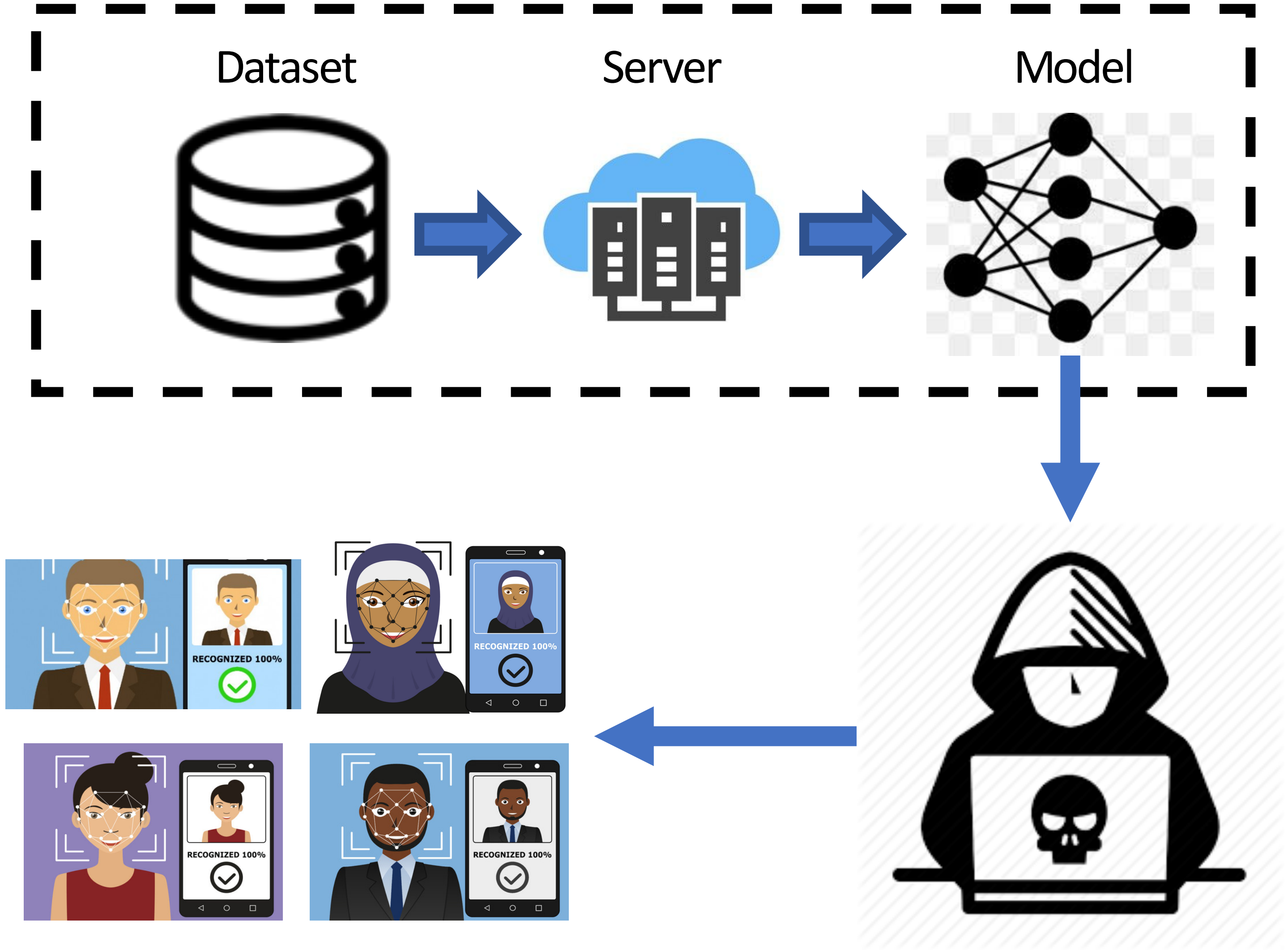
Face Recognition

Healthcare

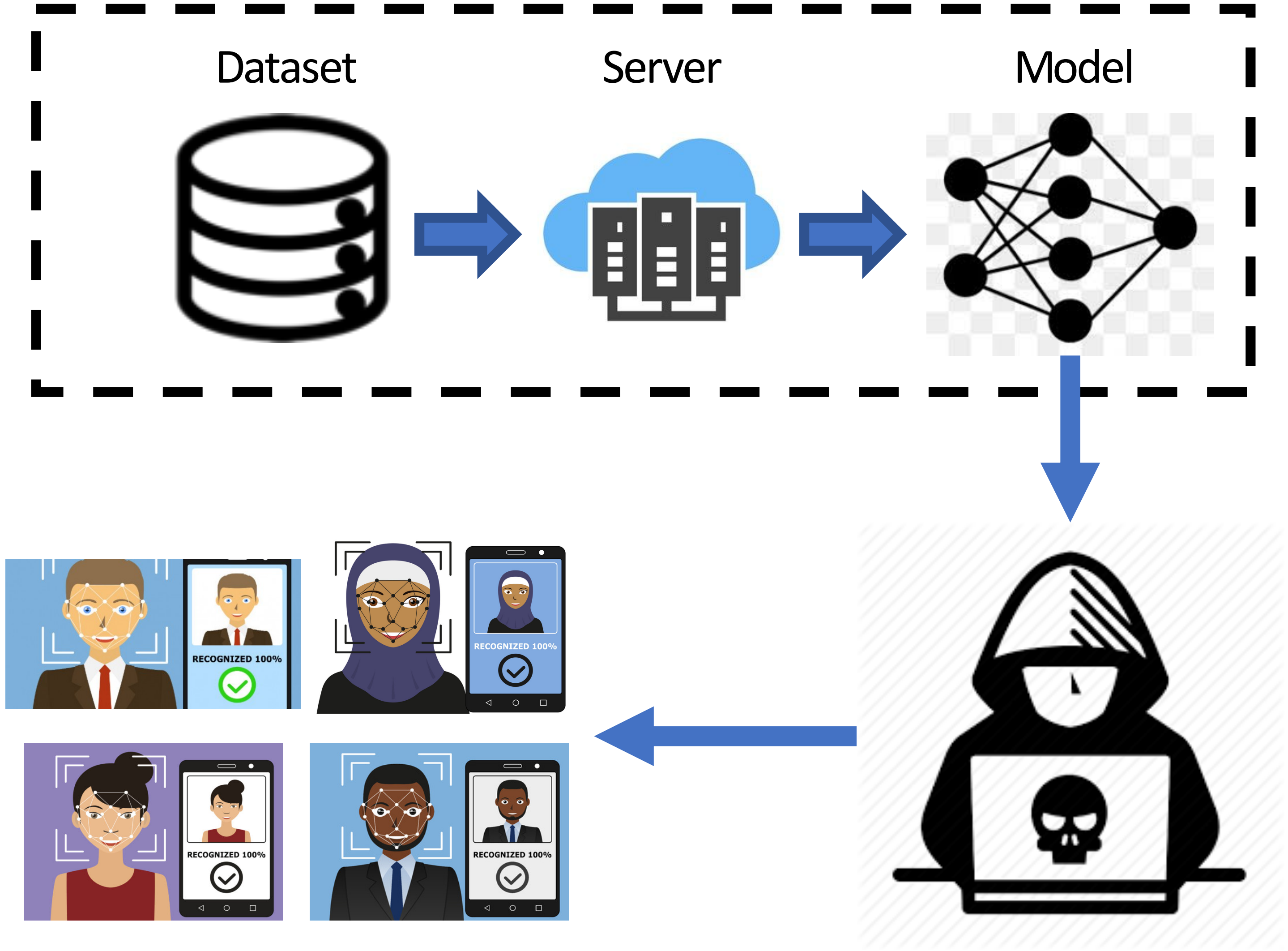
Deep Learning Framework



Privacy Issues of Training Data



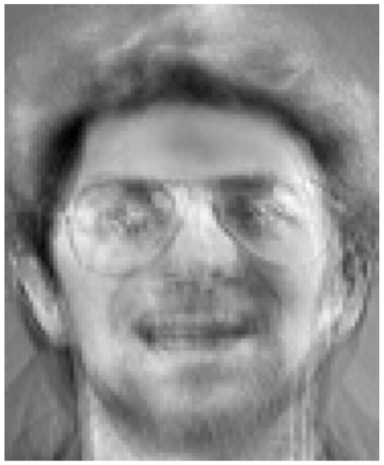
What information will be leaked from the deep learning model?



Training Privacy Leakage

- Model Inversion Attack

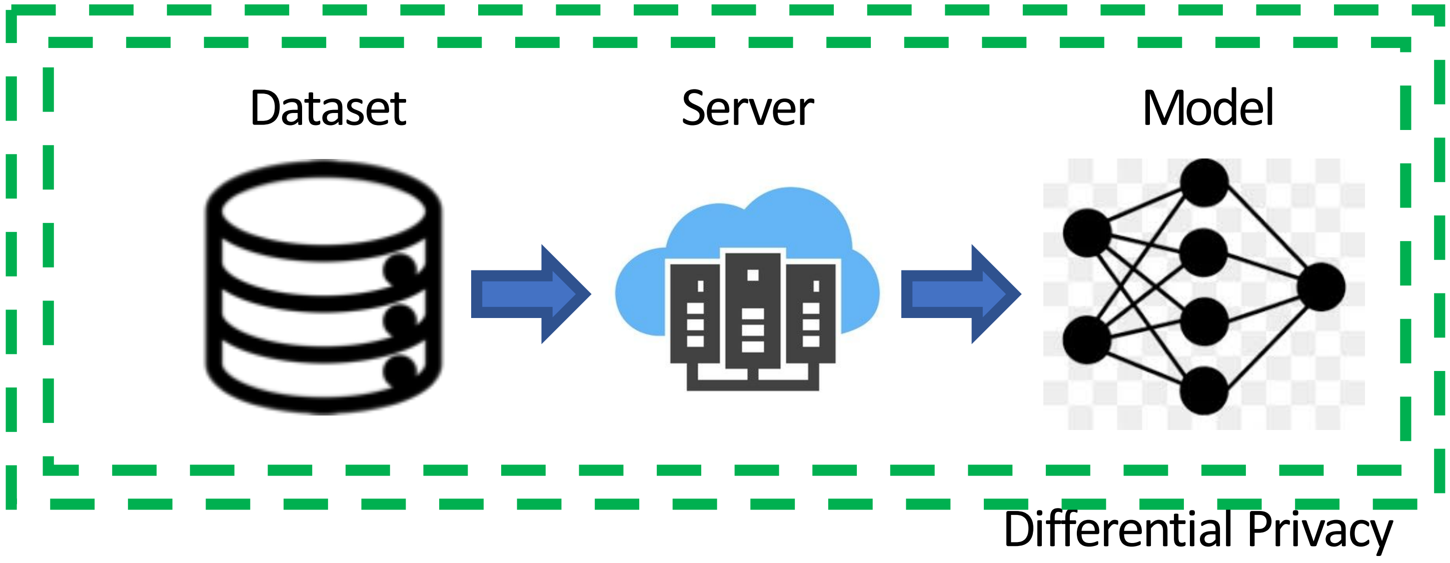
Model inversion attacks that exploit confidence information and basic countermeasures (CCS'15)



- Membership Inference attack
  - Infer whether or not a data case is in the training set.

Membership inference attacks against machine learning models (Oakland'17)

Protecting Privacy of Training Data



Differential Privacy(DP)

- Protect the privacy of individuals while allowing global query.

- e.g.: how many individuals in the database have property P?

Output of D and D’ should be similar!

Individual	Property P?
…	…
Alice	Yes
Victim	Yes
Bob	No
…	…

Database D

Individual	Property P?
…	…
Alice	Yes
Bob	No
…	…

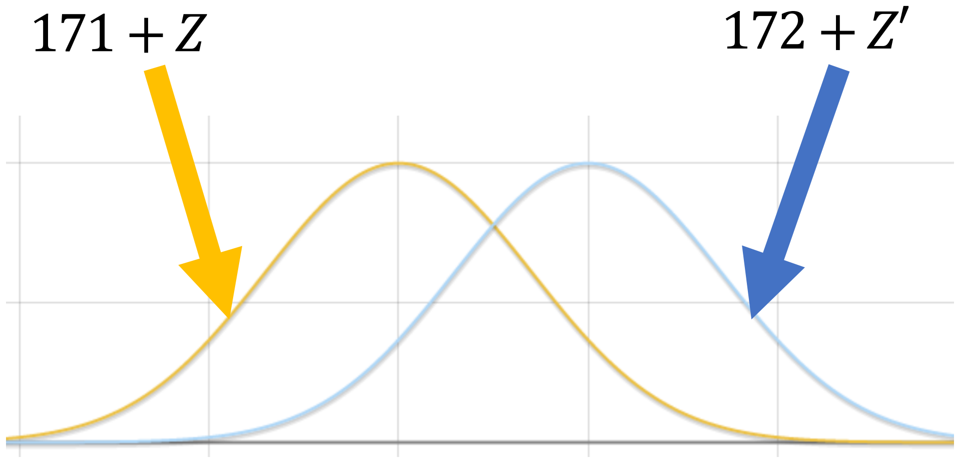
Database D’

Differential Privacy(DP)

- Solution: randomize the query answer (e.g. by adding random noise Z.)

Individual	Property P?
…	…
Alice	Yes
Bob	No
…	…

Individual	Property P?
…	…
Alice	Yes
Victim	Yes
Bob	No
…	…



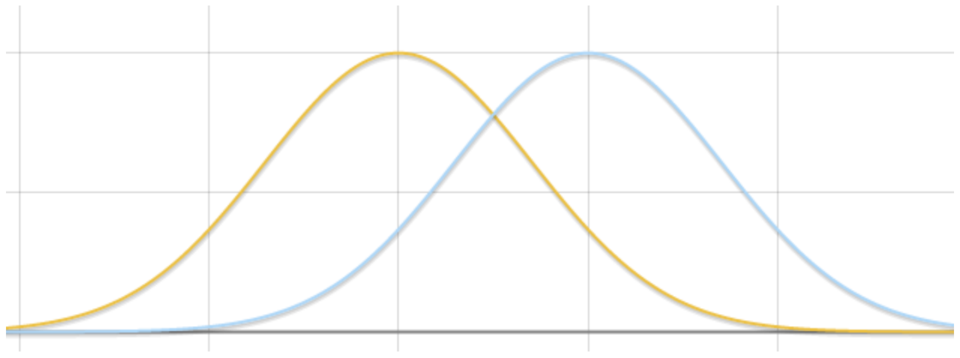
Differential Privacy

- Definition: Let $\mathrm{A}: D \to Y$ be a randomized function database domain D to output domain Y. Then A is (ε, δ)-differentially private if for any $\mathrm{S} \subseteq Y$ and any two databases d, d' which differs in only one element:

$$\Pr[A(d) \in S] \leq \exp(\varepsilon) \Pr[A(d') \in S] + \delta$$

Differentially Private Mechanism

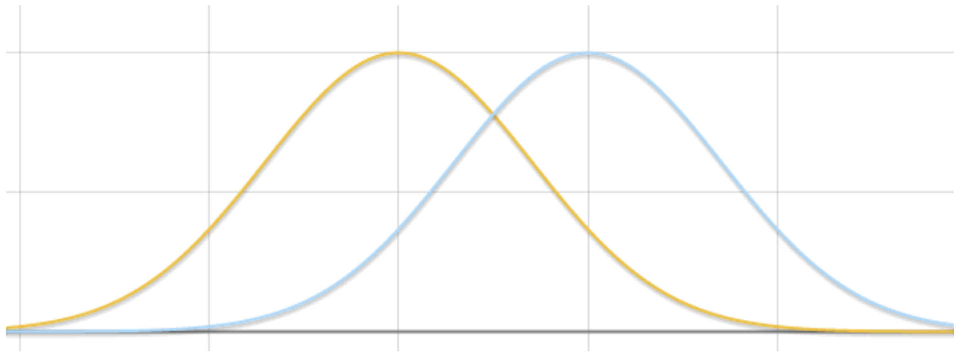
- Take function $f(d)$, add noise $A(d) = f(d) + Z$



- Noise level is related with:
  - ε and δ.
  - The maximal possible difference between $f(d)$ and $f(d')$

Differentially Private Mechanism

- Take function $f(d)$, add noise $A(d) = f(d) + Z$

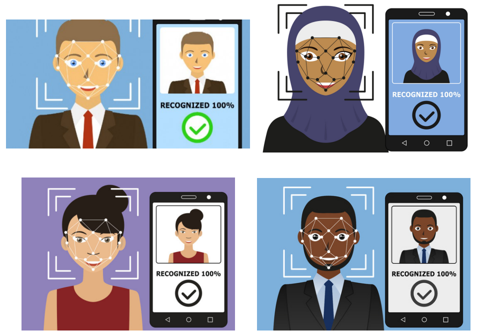


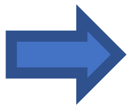
- When adding Gaussian noise, the scale should be:

$$\sigma = \sqrt{2\ln(\frac{1.25}{\delta})}\,\Delta_2/\varepsilon$$

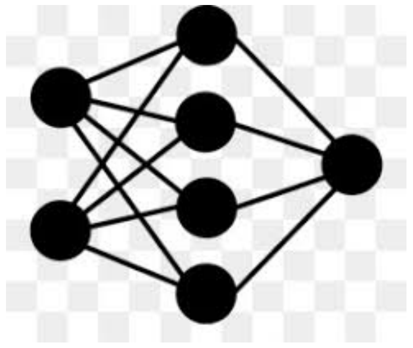
Deep Learning with DP

Dataset1

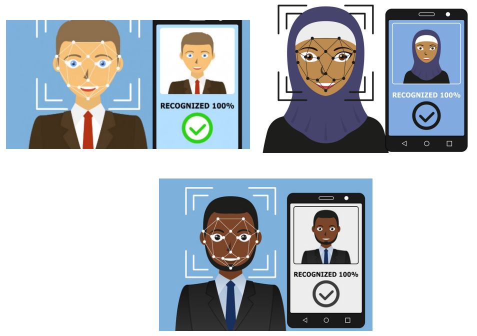


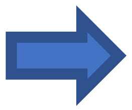


f_{θ_1}

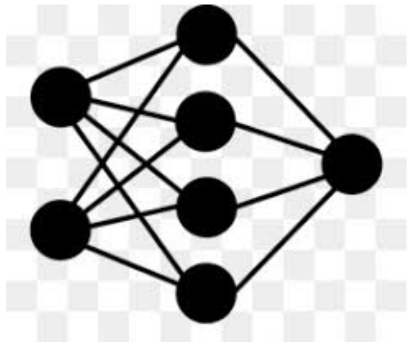


Dataset2





f_{θ_2}



$$\Pr[\theta_1 \in S] \leq \exp(\varepsilon) \Pr[\theta_2 \in S] + \delta$$

Achieving DP Deep Learning

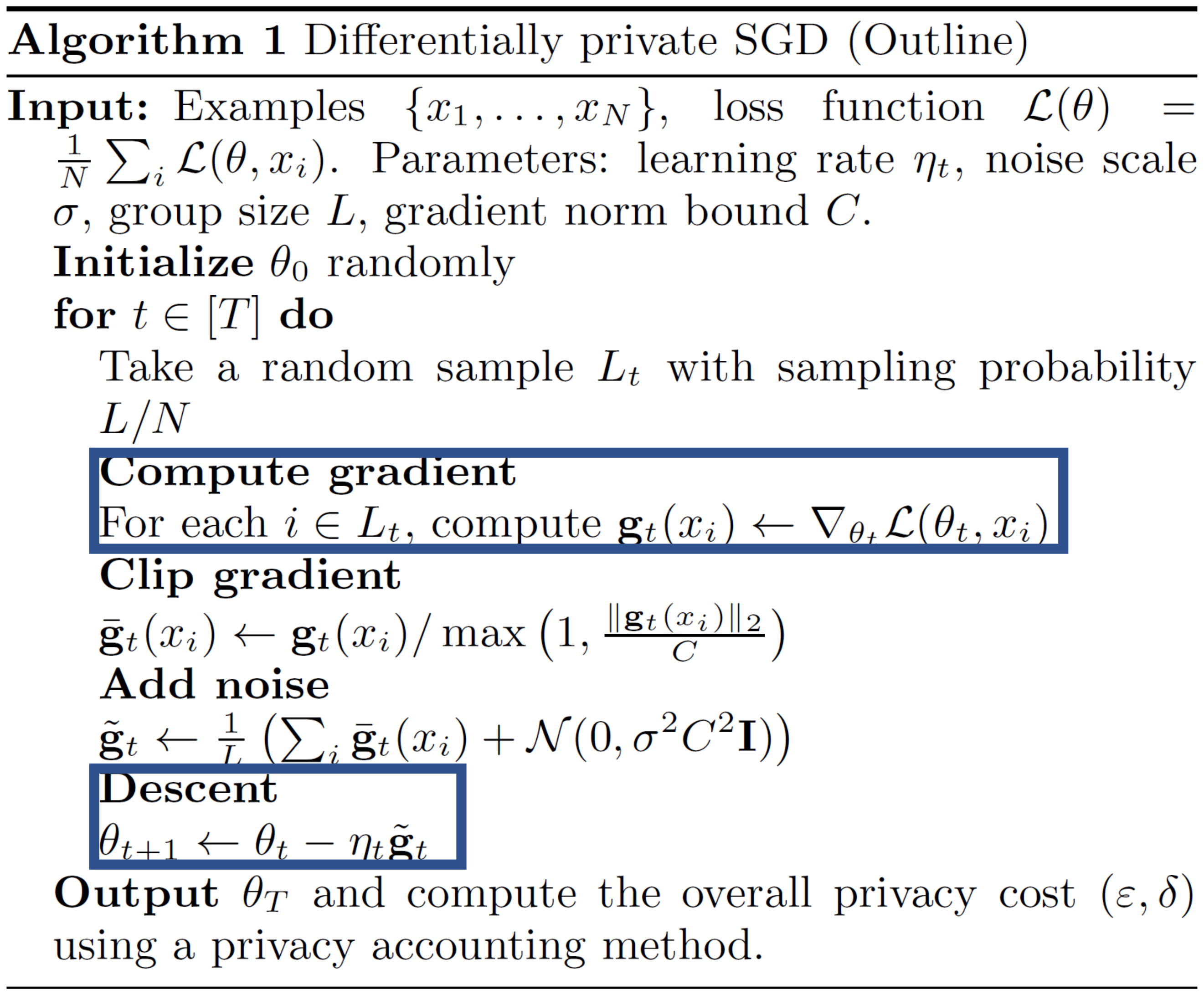
- How to achieve DP for deep learning model?
  - Directly adding noise to θ.
  - Not releasing model parameters, and during application, adding noise to the model output.
  - Adding noise in the process training.

Training Deep Learning Models

- Model function f_θ
- Training dataset $D = \{(x_1, y_1), \ldots, (x_{|D|}, y_{|D|})\}$

- Repeat:
  - Sample a batch from D.
  - Learn from the batch by calculating update $\Delta\theta$.
  - $\theta \coloneqq \theta + \Delta\theta$

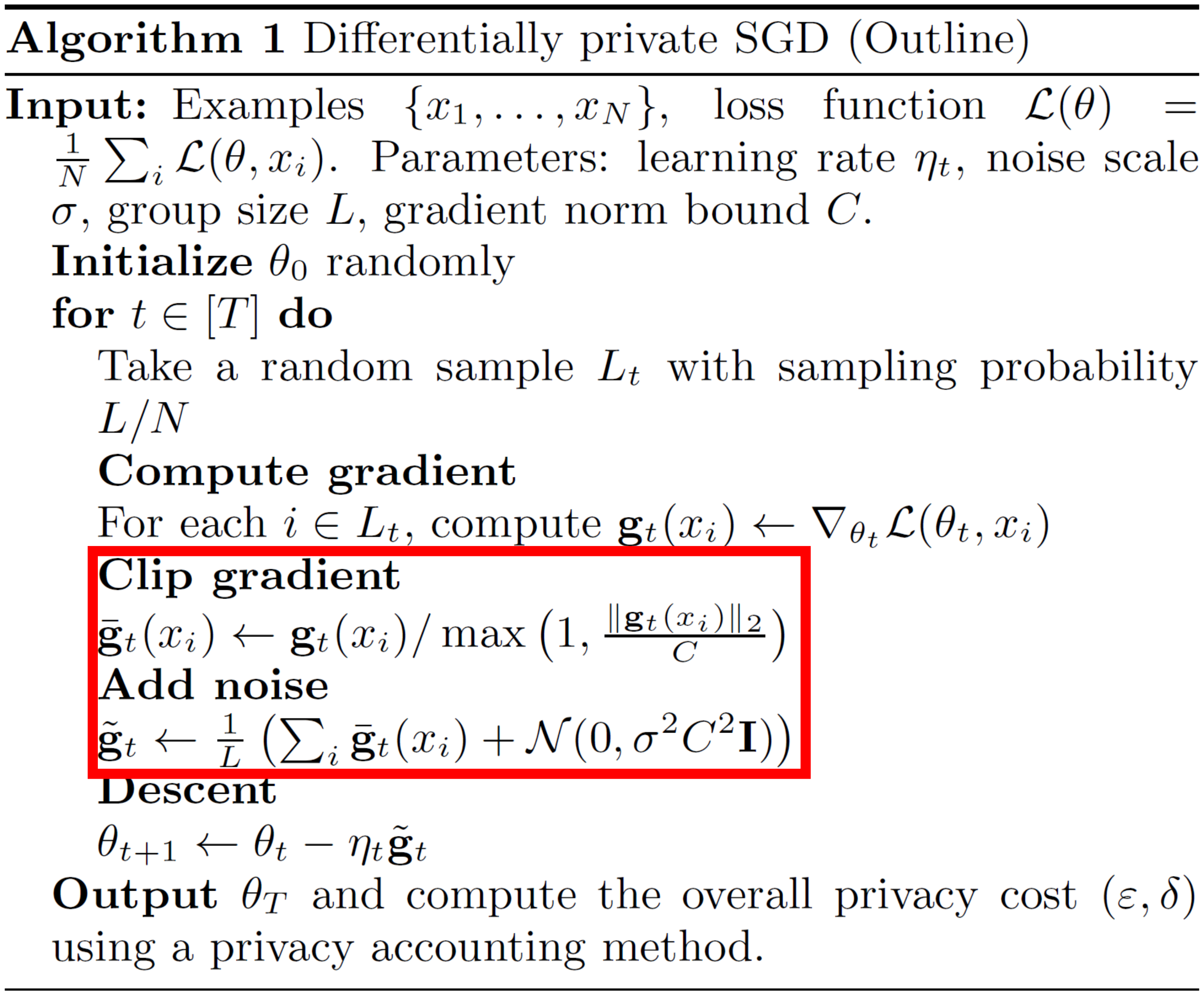
Deep Learning With DP



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 $\eta_t$, noise scale $\sigma$, group size $L$, gradient norm bound $C$.
  Initialize $\theta_0$ 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 $\theta_T$ and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method.
```

Deep Learning With DP



```
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 $\eta_t$, noise scale $\sigma$, group size $L$, gradient norm bound $C$.
  Initialize $\theta_0$ 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 $\theta_T$ and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method.
```

What is the bound?

- At each step the gradient is (ε, δ)-DP w.r.t. the group(batch).
  - What is the DP guarantee after many steps for the gradient w.r.t. the dataset?

One step,
Within the group

ε, δ

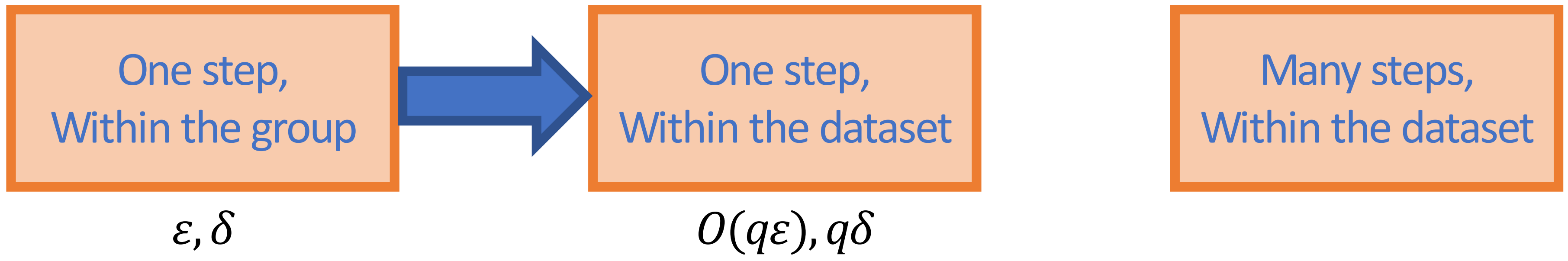
One step,
Within the dataset

Many steps,
Within the dataset

Amplification Theorem

- N: dataset size; L: size of each group
- $q = L/N$

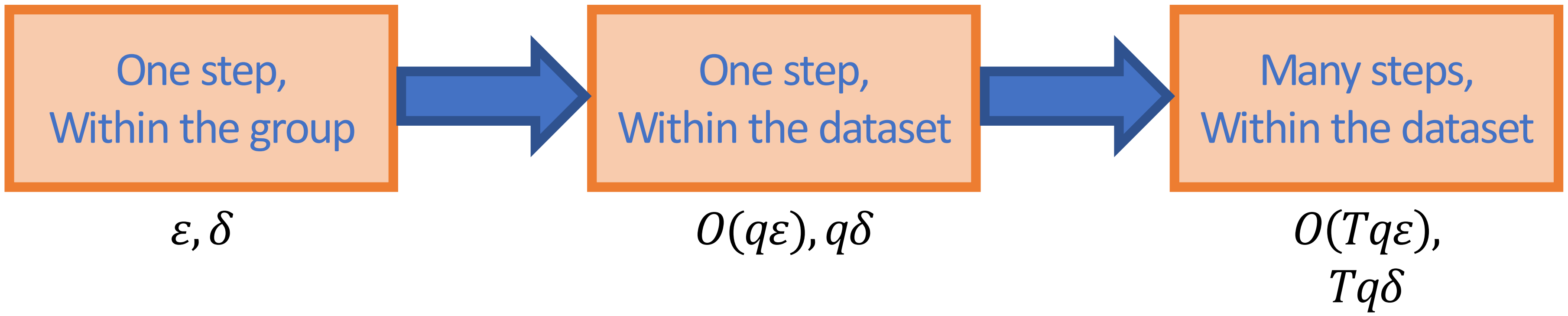
- Amplification Theorem: if gradient is (ε, δ)-DP within the group, then it's $(O(q\varepsilon), q\delta)$-DP within the dataset.


One step, Within the group
ε, δ
One step, Within the dataset
$O(q\varepsilon), q\delta$
Many steps, Within the dataset

Basic Composition Theorem

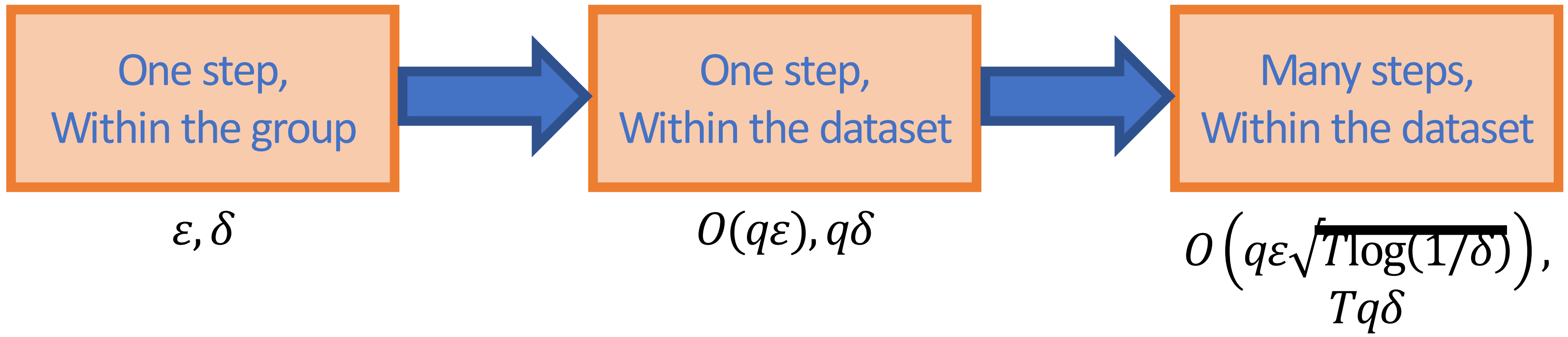
- Applying an $(\varepsilon_1, \delta_1)$-DP algorithm with an $(\varepsilon_2, \delta_2)$-DP algorithm together will give an $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-DP algorithm.

- So after T steps an (ε, δ)-DP algorithm is $(T\varepsilon, T\delta)$-DP.


One step, Within the group
ε, δ
One step, Within the dataset
$O(q\varepsilon), q\delta$
Many steps, Within the dataset
$O(Tq\varepsilon), Tq\delta$

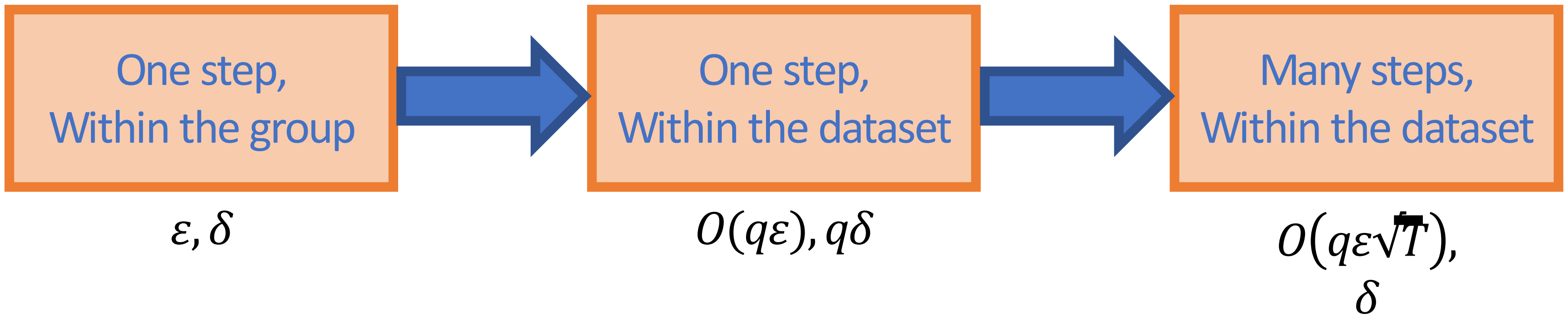
Strong Composition Theorem

- Applying the same (ε, δ)-DP algorithm T times will give an $\left(O\left(\varepsilon\sqrt{T\log\left(\frac{1}{\delta}\right)}\right), T\delta\right)$-DP algorithm.


One step, Within the group
ε, δ
One step, Within the dataset
$O(q\varepsilon), q\delta$
Many steps, Within the dataset
$O\left(q\varepsilon\sqrt{T\log(1/\delta)}\right)$, $Tq\delta$

Moments Accountant

- One major contribution of the paper!
- The noise at each step is Gaussian noise.


One step, Within the group
One step, Within the dataset
Many steps, Within the dataset
ε, δ
$O(q\varepsilon), q\delta$
$O(q\varepsilon\sqrt{T})$, δ

Comparison

Approach	Overall epsilon	Overall delta
Basic Composition	$O(Tq\varepsilon)$	$Tq\delta$
Advanced Composition	$O\left(q\varepsilon\sqrt{T\log(1/\delta)}\right)$	$Tq\delta$
Moments Accountant	$O(q\varepsilon\sqrt{T})$	δ

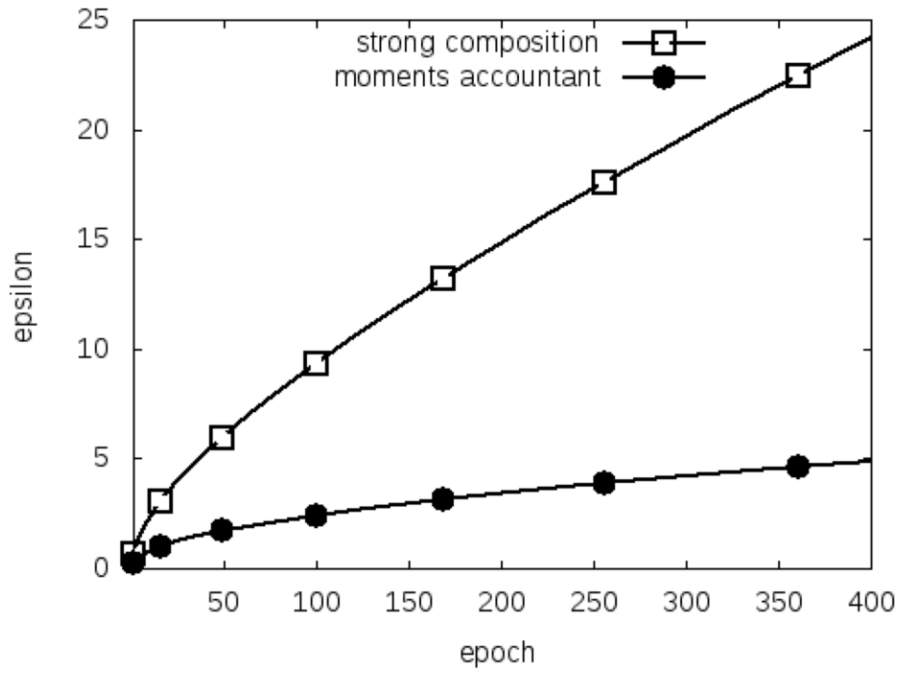
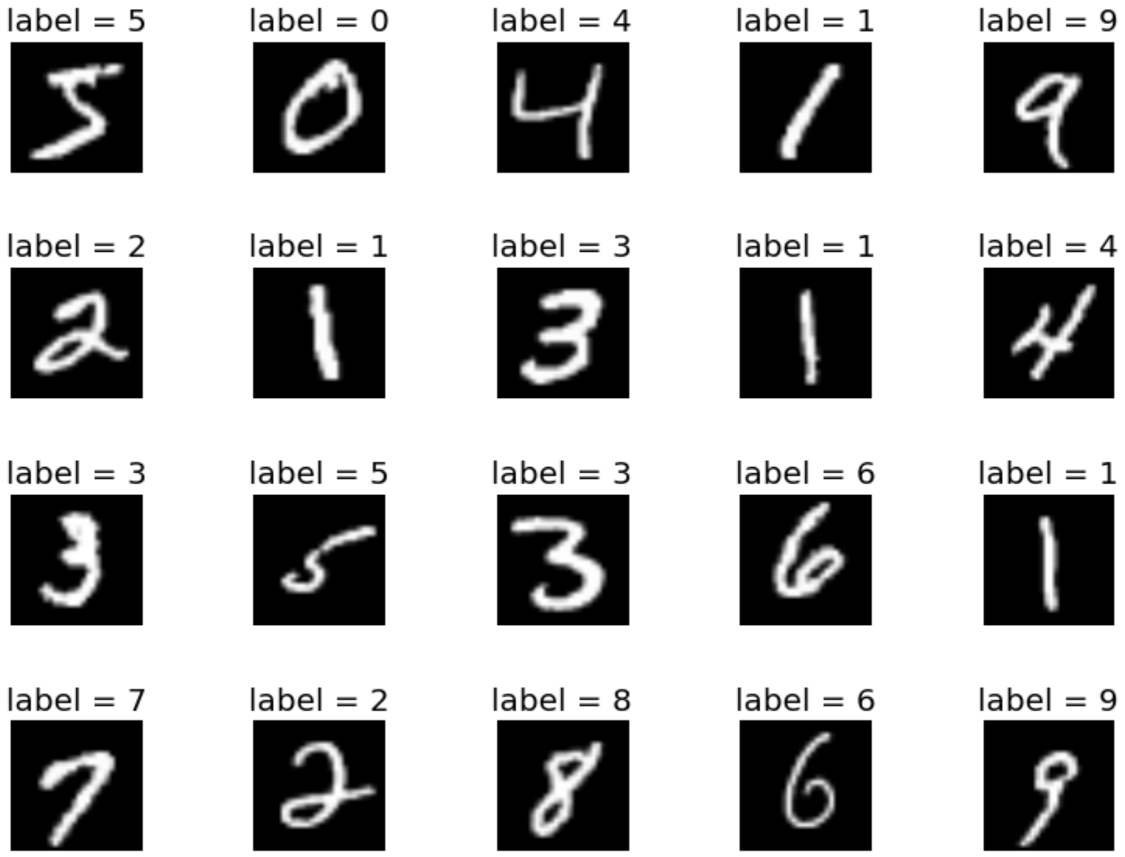


Figure 2: The ε value as a function of epoch E for $q = 0.01$, $\sigma = 4$, $\delta = 10^{-5}$, using the strong composition theorem and the moments accountant respectively.

Experiments

- MNIST: 70000 gray-level images for hand written digits with size 28×28.



Experiments

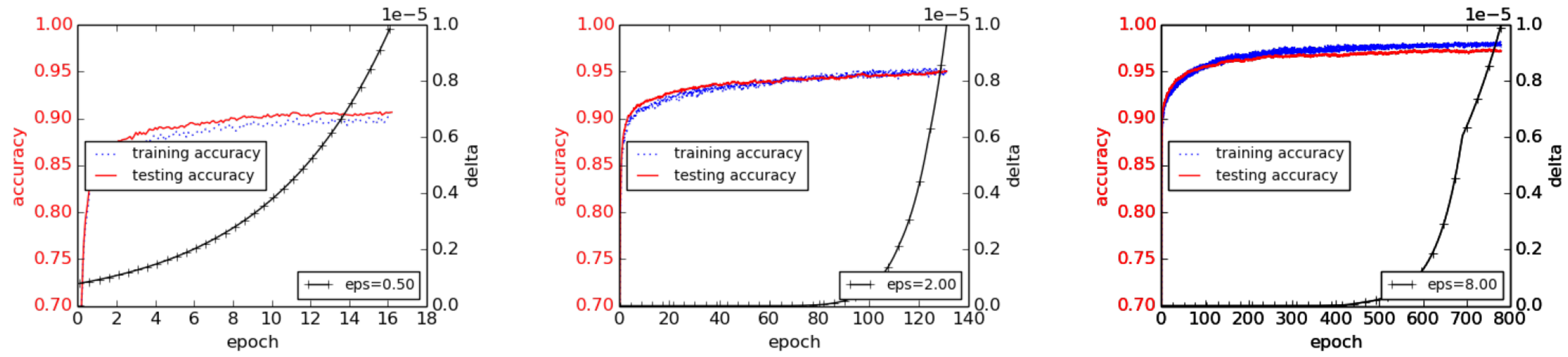
- CIFAR10: 60000 colored images of 10 classes with size 32×32.

Experiments

- MNIST: DP-PCA + DP-NeuralNetwork
- CIFAR-10: Pretrained Conv Layer + DP-NeuralNetwork

Experiment Results - MNIST

- Acc without DP: 98.3%

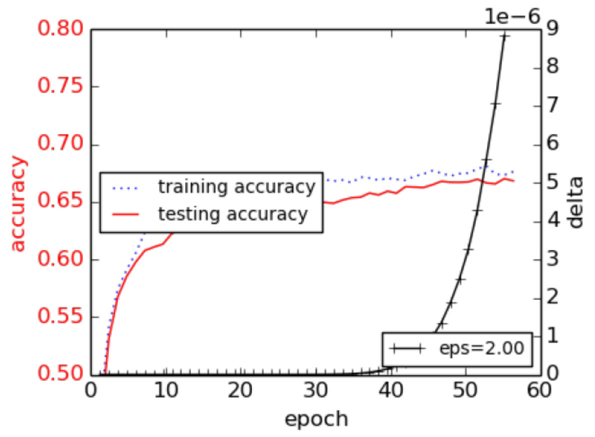


(1) Large noise (2) Medium noise (3) Small noise

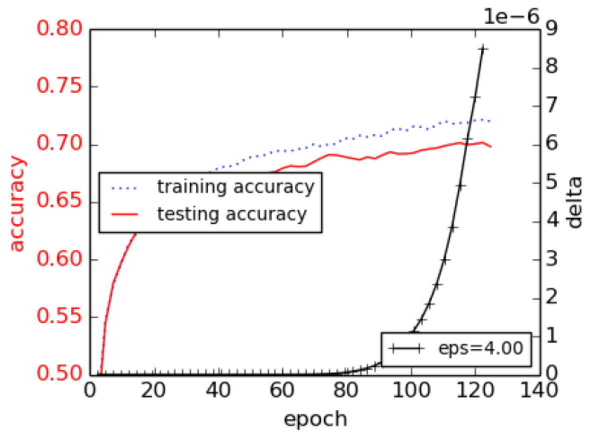
Figure 3: Results on the accuracy for different noise levels on the MNIST dataset. In all the experiments, the network uses 60 dimension PCA projection, 1,000 hidden units, and is trained using lot size 600 and clipping threshold 4. The noise levels (σ, σ_p) for training the neural network and for PCA projection are set at (8, 16), (4, 7), and (2, 4), respectively, for the three experiments.

Experiment Results – CIFAR10

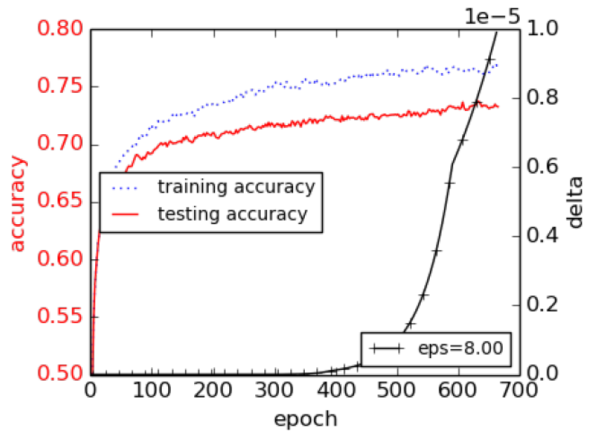
- Acc without DP: 80%



(1) $\varepsilon = 2$



(2) $\varepsilon = 4$



(3) $\varepsilon = 8$

Figure 6: Results on accuracy for different noise levels on CIFAR-10. With δ set to 10^{-5}, we achieve accuracy 67%, 70%, and 73%, with ε being 2, 4, and 8, respectively. The first graph uses a lot size of 2,000, (2) and (3) use a lot size of 4,000. In all cases, σ is set to 6, and clipping is set to 3.

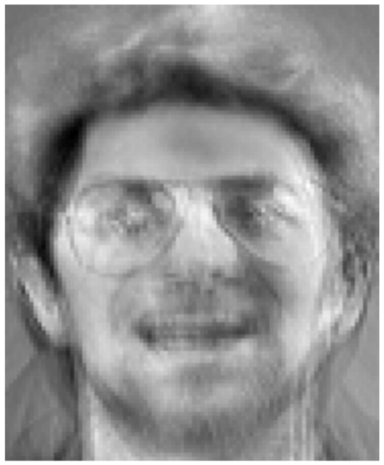
Effectiveness

- What can DP defend?
- What cannot DP defend?

Training Privacy Leakage

- Model Inversion Attack

  Model inversion attacks that exploit confidence information and basic countermeasures (CCS'15)



- Membership Inference attack
  - Infer whether or not a data case is in the training set.

  Membership inference attacks against machine learning models (Oakland'17)

What can DP protect?

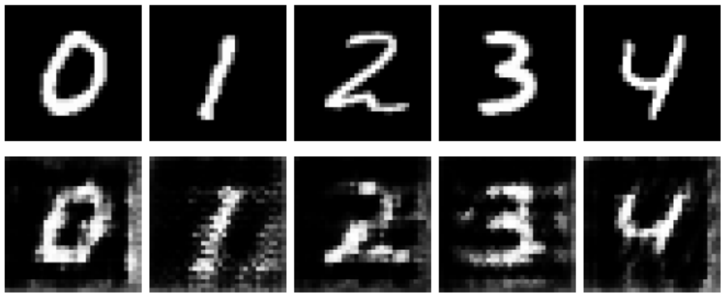
- Privacy of individual data in the dataset.
  - Membership Inference Attack

  - Extracting secrets from language models
    - My SSN is “xxx-xx-xxxx”

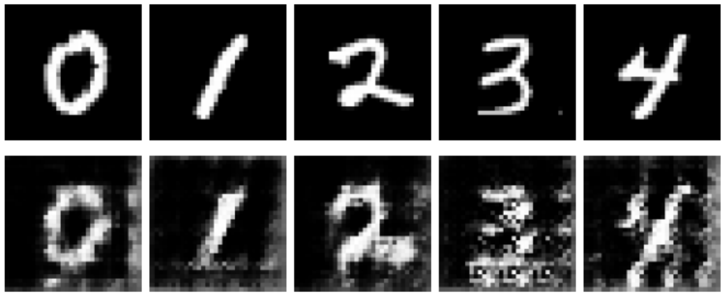
The Secret Sharer: Measuring Unintended Neural Network Memorization & Extracting Secrets (arXiv preprint)

What can't DP protect?

- Privacy leakage because of global information of the dataset.



(c) $\frac{\epsilon}{c} = 10, \theta_u = 1, \theta_d = 1$



(d) $\frac{\epsilon}{c} = 10, \theta_u = 0.1, \theta_d = 1$

Deep models under the GAN: information leakage from collaborative deep learning (CCS'17)

Q&A