University of Idaho

Department of Computer Science

CS 487/587 Adversarial Machine Learning

Dr. Alex Vakanski

Lecture 12

Defenses against Privacy Attacks

Lecture Outline

- Defenses against Privacy Attacks
  - Anonymization techniques
  - Encryption techniques
  - Differential privacy
  - Distributed learning
  - ML-specific techniques
- Elijah Darko presentation
  - Introduction to differential privacy
- Differentially private SGD
- Scalable private learning with PATE

Defenses against Privacy Attacks

Defenses against Privacy Attacks

- ***Data privacy*** techniques have the goal of allowing analysts to learn about *trends* in data, without revealing information specific to *individual data instances*
  - Therefore, privacy techniques involve an intentional release of information, and attempt to control what can be learned from the released information
- Related to data privacy is the ***Fundamental Law of Information Recovery***, which states that "*overly accurate estimates of too many statistics can completely destroy privacy*"
  - I.e., extracting useful information from a dataset (e.g., for training an ML model) poses a privacy risk to the data
- There is an inevitable trade-off between privacy and accuracy (i.e., utility)
  - Preferred privacy techniques should provide an estimate of how much privacy is lost by interacting with data

Defenses against Privacy Attacks

Defenses against Privacy Attacks

- Defense strategies against privacy attacks in ML can be broadly classified into:
  - Anonymization techniques
  - Encryption techniques
  - Differential privacy
  - Distributed learning
  - ML-specific techniques

Anonymization Techniques

Anonymization Techniques

- ***Anonymization*** techniques provide privacy protection by removing identifying information in the data
- E.g., remove personal identifiable information (PII)
  - In the example below, the Name and Address columns are removed

User ID	Name	Address	Account Type	Subscription Date
001	Alice	123 A St	Pro	01/02/20
002	Bob	234 B St	Free	02/03/21
003	Charlie	456 C St	Pro	03/04/18

Anonymization Techniques

Anonymization Techniques

- Anonymization is not an efficient defense method, since the remaining information in the data can be used for identifying the individual data instances
  - For example, based on health records (including diagnoses and prescriptions) with removed personal information released by an insurance group in 1997, a researcher extracted the information for the Governor of Massachusetts
    - This is referred to as ***de-anonymization***
  - The same researcher later showed that 87% of all Americans can be uniquely identified using 3 bits of information: ZIP code, birth date, and gender

Dataset 1: Users medical database

User ID	Name	Address	Zip Code	Birth date	Gender	Probable disease ID
001	Alice	123 A St	83401	01/02/1997	F	120
002	Bob	234 B St	83402	02/03/1995	M	35
003	Charlie	456 C St	83403	03/04/1999	M	240

Dataset 2: Users medical database with name and address removed

User ID	Zip Code	Birth date	Gender	Probable disease ID
001	83401	01/02/1997	F	120
002	83402	02/03/1995	M	35
003	83403	03/04/1999	M	240

Linkage Attack

Anonymization Techniques

- De-anonymization of data by using connections to external sources of information is referred to as ***linkage attack***
  - For example:
    - In 2006, Netflix published anonymized 10 million movie rankings by 500,000 customers
    - Two researchers showed later that by using movie recommendations on IMDb (Internet Movie Database) they could identify the customers in the Netflix data

Dataset 1: Annonymized dataset with removed personal information

User ID	Name	Address	Account Type	Subscription Date
001			Pro	01/15/20
002			Pro	02/03/21
003			Free	03/04/18

Dataset 2: External public dataset that reveals the users in Dataset 1

User ID	Product Name	Product Price	Purchase Date
001	TV	400	01/02/20
002	Iphone	1,199	02/02/21
003	Watch	130	02/22/18

k-anonymity

Anonymization Techniques

- ***k-anonymity*** is an approach for protecting data privacy by suppressing certain identifying data features
  - This approach removes fields of data for individuals who have unique characteristics
    - E.g., students at UI who are from Latvia and are enrolled in Architecture
- A dataset is *k*-anonymous if for any person's record, there are at least $k - 1$ other records that are indistinguishable
- Limitation: this approach is mostly applicable to large datasets with low-dimensional input features
  - The more input features there are for each record, the higher the possibility of unique records

Encryption Techniques

Encryption Techniques

- ***Encryption*** is a cryptography approach, which converts the original representation of information into an alternative form
  - The sender of encrypted information shares the decoding technique only with the intended recipients of the information
- Encrypting the training data has been applied in ML
  - Common techniques for data encryption include:
    - Homomorphic encryption (HE)
    - Secure multi-party computation (SMPC)
- Encrypting ML models is less common approach
  - Homomorphic encryption has been applied to the model gradients in collaborative DL setting to protect the model privacy

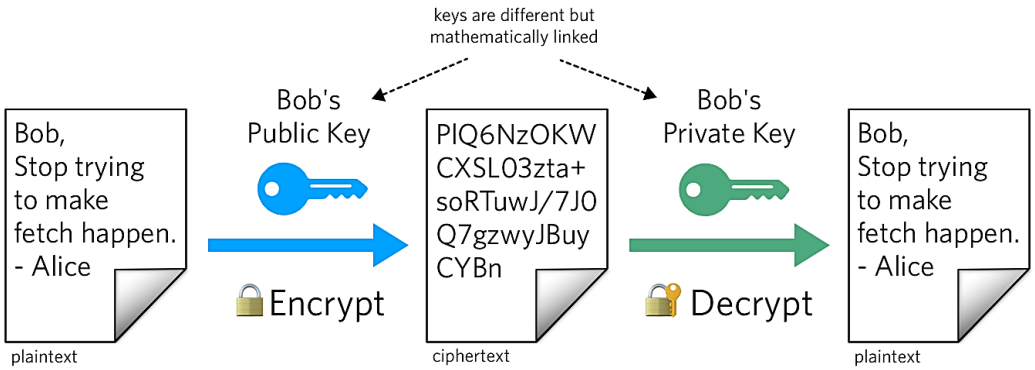


Figure form: What is Public Key Cryptography?

Homomorphic Encryption

Encryption Techniques

- ***Homomorphic encryption (HE)*** allows users to perform computations on encrypted data (without decrypting it)
  - Encrypted data can be analyzed and manipulated without revealing the original data
- HE uses a public key to encrypt the data, and applies an algebraic system (e.g., additions and multiplications) to allow computations while the data is still encrypted
  - Only the person who has a matching private key can access the decrypted results

Homomorphic Encryption

Encryption Techniques

- In ML, training data can be encrypted and send to a server for model training
  - Even if the server is untrusted or it is compromised, confidentiality of the data will remain preserved
  - One main limitation of HE is the slowing down of the training process
- HE has been applied to traditional ML approaches, such as Naïve Bayes, Decision Trees
  - Training DNNs over encrypted data is still challenging, due to the increased computational complexity

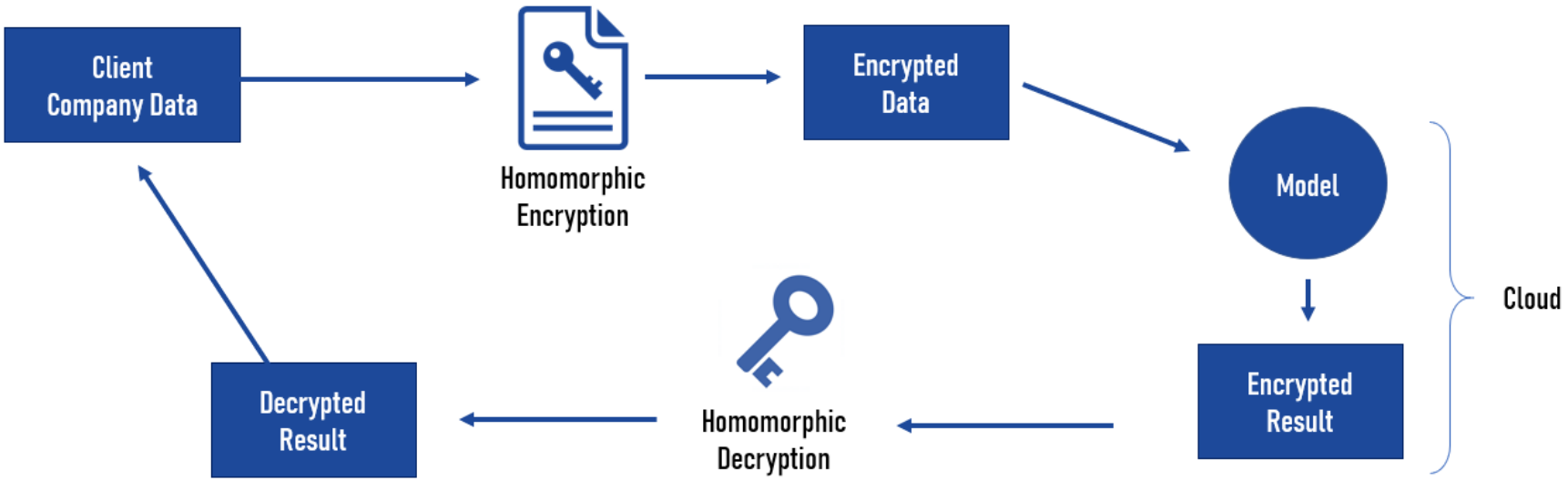


Figure form: Homomorphic Encryption & Machine Learning: New Business Models

Privacy versus Confidentiality

Encryption Techniques

- Encryption techniques in ML are mainly applied to protect the confidentiality of the data or model
- ***Confidentiality*** refers to keeping the information (training data, model parameters) hidden from the clients and the public
  - It is ensuring that only authorized parties have access to the information
  - E.g., a server has an ML model trained on private data and provides the model to a client for inference
    - It is preferred to preserve the confidentiality of the model parameters from the client
- ***Privacy*** refers to intentional release of information in a controlled manner to prevent unintended information leakage
  - It is ensuring that released data cannot uniquely identify individual inputs
  - E.g., a server applies Differential Privacy to a trained ML model to prevent information leakage about individual inputs
- Protecting privacy is more challenging than protecting confidentiality

Secure Multi-Party Computation

Encryption Techniques

- ***Secure Multi-Party Computation*** (SMPC) is an extension of encryption in multi-party setting
  - SMPC allows two or more parties to jointly perform computation over their private data, without sharing the data
  - E.g., two banks want to know if they have both flagged the same individuals and learn about the activities by those individuals
    - The banks can share encrypted tables of flagged individuals, and they can decrypt only the matched records, but not the information for individuals that are not in both tables

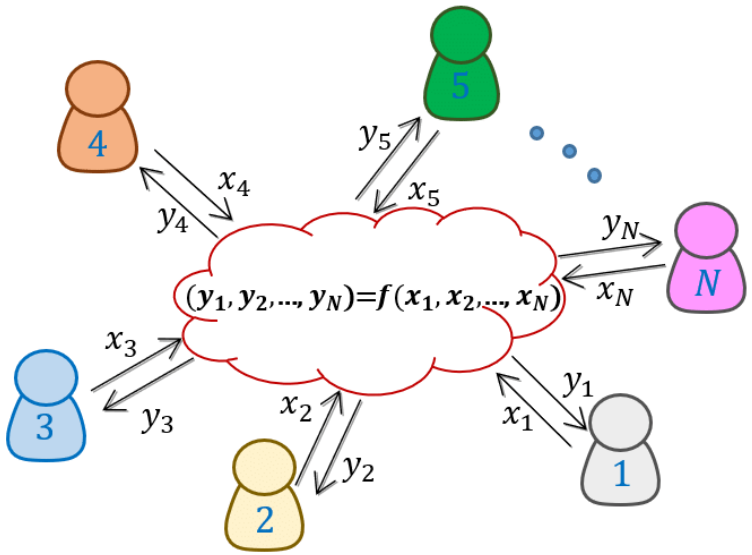


Figure form: Generation and Distribution of Quantum Oblivious Keys for Secure Multiparty Computation

Secure Multi-Party Computation

Encryption Techniques

- SMPC versus HE
  - SMPC protects the privacy of the data in collaborative learning
    - E.g., participants in collaborative learning do not trust the other participants or the central server
  - HE protects the confidentiality of the data from external adversaries
    - E.g., a data owner wants to use a MLaaS (Machine Learning as a Service), but does not trust the service provider: (1) the owner sends encrypted data, (2) the provider processes encrypted data and sends back encrypted results, (3) the owner decrypts the results
    - Or, a bank can store encrypted banking information in the cloud, and use HE to ensure that only the employees of the bank can access the data

Secure Multi-Party Computation

Encryption Techniques

- In ML, SMPC can be used to compute updates of the model parameters by multiple parties that have access to their private data
  - For examples, SMPC has been applied to federated learning, where participants encrypt their updates, and the central server can recover only the sum of the updates from all participants
  - Beside the data privacy, SMPC also offers protection against adversarial participants
    - Either all parties are honest and can jointly compute the correct output, or if a malicious party is dishonest the joint output will be incorrect
- SMPC has been applied to traditional ML models, such as decision trees, linear regression, logistic regression, Naïve Bayes, *k*-means clustering
  - Application of SMPC to deep NNs is challenging, due to increased computational costs

Differential Privacy

Differential Privacy

- ***Differential privacy*** is based on employing obfuscation mechanisms for privacy protection
  - A randomization mechanism $\mathcal{M}(D)$ applies noise ξ to the outputs of a function $f(D)$ to protect the privacy of individual data instances, i.e., $\mathcal{M}(D) = f(D) + \xi$
  - Commonly used randomization mechanisms include Laplacian, Gaussian, and Exponential mechanism
- DP is often implemented in practical applications
- Examples include:
  - 2015: Google, for sharing historical traffic statistics
  - 2016: Apple, for improving its Intelligent Personal Assistant technology
  - 2017: Microsoft, for telemetry in Windows
  - 2020: LinkedIn, for advertiser queries
  - 2020: U.S. Census Bureau, for demographic data

DP Example

Differentially Private SGD

- Consider two databases D_1 and D_2 that show if a person has diabetes or not
  - The only difference between the two databases is that D_2 does not include the last record in D_1 (for Bob)
- Let's assume that the databases are publicly available for making queries
  - To protect patient identities, it is not allowed to query the patient names
- However, an adversary can query the sum of the persons with diabetes in the first database (e.g., $f(D_1) = 64$), and the sum in the second database (e.g., $f(D_2) = 63$)
  - Based on the difference $f(D_1) - f(D_2) = 64 - 63 = 1$, the adversary can infer that Bob has diabetes
  - Alternatively, if $f(D_1) = 63$ and $f(D_2) = 63$, the adversary can infer that Bob does not have diabetes

D_1 (includes Bob)

Name	Has Diabetes
Don	1
Monica	0
…	
…	
Chris	1
Bob	1

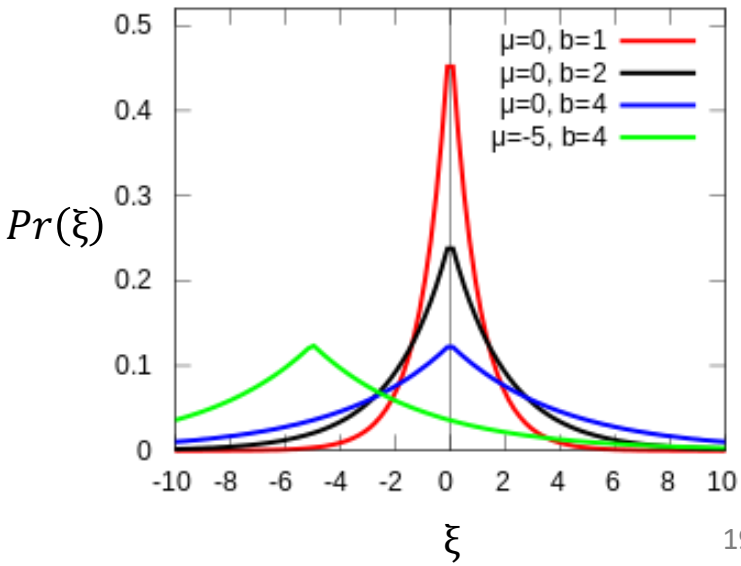
D_2 (without Bob)

Name	Has Diabetes
Don	1
Monica	0
…	
…	
Chris	1

DP Example (cont'd)

Differentially Private SGD

- An algorithm that is ***differentially private*** adds noise to the answers for $f(D_1)$ and $f(D_2)$ to make it difficult to infer the information about Bob
  - I.e., a randomization mechanism $\mathcal{M}(D)$ is selected to add noise ξ to the output answers to queries $f(D)$, that is, $\mathcal{M}(D) = f(D) + \xi$
- Additive noise ξ from a Laplacian distribution (shown) is commonly applied
  - E.g., let's assume a privacy budget $\varepsilon = 0.5$ and let's sample noise from a Laplacian distribution with $\mu = 0$ and scale $b = 1/\varepsilon = 1/0.5 = 2$
  - 6 random noise samples are: $\xi \in \{-0.13, 2.06, -1.67, -2.49, -0.52, \ 0.37\}$
  - Consider 3 queries by the adversary having the outputs $f(D_1) = 64$ and $f(D_2) = 63$ with added Laplacian noise ξ:
    - $\mathcal{M}(D_1) - \mathcal{M}(D_2) = 63.87 - 65.06 = -1.19$
    - $\mathcal{M}(D_1) - \mathcal{M}(D_2) = 62.33 - 60.51 = 1.82$
    - $\mathcal{M}(D_1) - \mathcal{M}(D_2) = 63.48 - 63.37 = 0.11$
  - Based on the differences between the randomized outputs from the queries for D_1 and D_2, now it is impossible for the adversary to tell if Bob has diabetes



DP Mechanism

Differentially Private SGD

- The important question in DP is: how much noise to add?
  - The amount of noise ξ depends on the data, and it needs to be adjusted
    - E.g., a function $f_1(D)$ that provides the yearly income of people in thousands of dollars would require different level of noise than a function $f_2(D)$ that provides the height in feet
- The ***sensitivity*** of the function f determines how much the output $f(D)$ changes by adding a single data instance
  - Sensitivity is defined as $\Delta f = max\|f(D_1) - f(D_2)\|_1$ for all possible datasets D_1 and D_2 differing in one data instance, where $\|\cdot\|_1$ denotes ℓ_1-norm
    - E.g., for the example with medical diabetes records, the sensitivity is $\Delta f = 1$, since the sum of the people with diabetes can change only by 1 when a single input is added
- A Laplacian mechanism that is ε-differentially private adds a Laplacian noise with scale $b = \Delta f / \varepsilon$
- Note that if the privacy budget ε has smaller values, this will result in larger amount of Laplacian noise ξ added to $f(D)$
  - Thus, the noisy outputs $\mathcal{M}(D)$ will reveal less private information about the inputs (i.e., provide better privacy protection), but also the noisy answers to the queries $\mathcal{M}(D)$ will be less accurate

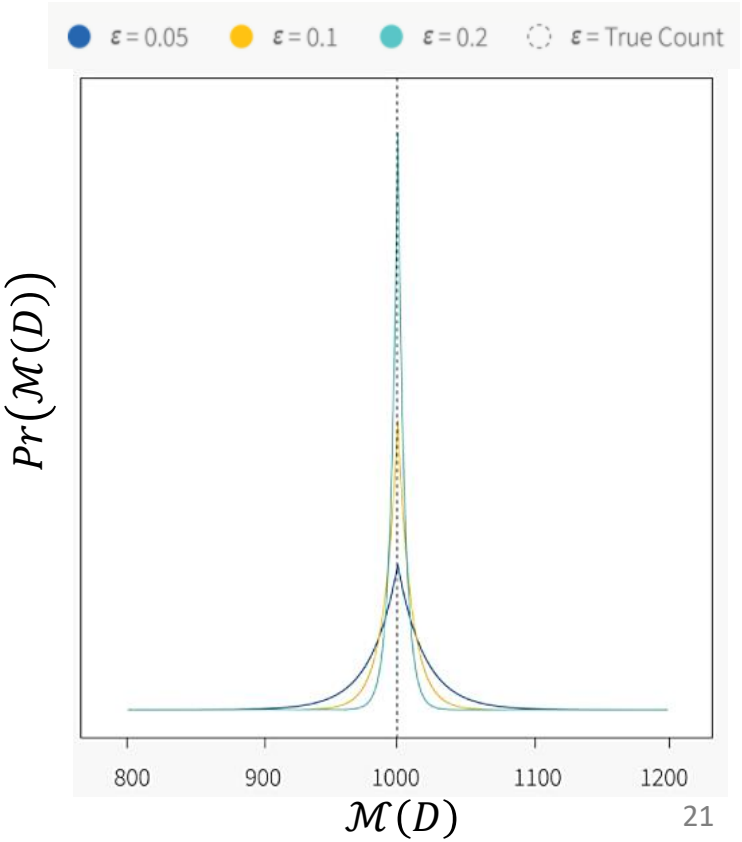
DP with Laplacian Randomization

Differentially Private SGD

- The figure shows the probability distributions of the outputs $\mathcal{M}(D)$ for three different levels of Laplacian noise with $\varepsilon \in \{0.05, 0.1, 0.2\}$
  - The true output value is $f(D) = 1{,}000$
  - Larger values of ε have distributions that are tighter around the true value of $f(D) = 1{,}000$ in the figure, and hence are more accurate, but leak more privacy
- A mechanism $\mathcal{M}(D)$ is ε-differentially private if for all databases D_1 and D_2 that differ by at most one instance, and for any subset of outputs S:

$$Pr(\mathcal{M}(D_1) \in S) \leq e^{\varepsilon}\, Pr(\mathcal{M}(D_2) \in S)$$

  - In other words, ε-differential privacy ensures that the probabilities of any two outputs $\mathcal{M}(D_1)$ and $\mathcal{M}(D_2)$ differ by at most e^{ε}
  - E.g., for $\varepsilon = 0.05$, $Pr(\mathcal{M}(D_1))/Pr(\mathcal{M}(D_2))$ is at most $e^{0.05} = 1.05$
  - Smaller ε ensures less similar outputs $\mathcal{M}(D_1)$ and $\mathcal{M}(D_2)$, and provides higher levels of privacy



DP with Gaussian Randomization

Differentially Private SGD

- There are other DP mechanisms besides the Laplacian mechanism, that are more suitable for some applications
- The ***Gaussian mechanism*** adds Gaussian noise instead of Laplacian noise, and the level of noise is based on the ℓ_2-norm sensitivity, instead of ℓ_1-norm
- A Gaussian mechanism is (ε, δ)-differentially private if for all databases D_1 and D_2 that differ by at most one instance, and for any subset of outputs S:

$$Pr(\mathcal{M}(D_1) \in S) \leq e^{\varepsilon}\ Pr(\mathcal{M}(D_2) \in S) + \delta$$

- The (ε, δ)-differential privacy that is provided by the Gaussian mechanism introduces the probability parameter δ
  - Informally, (ε, δ)-differential privacy is guaranteed with probability $1 - \delta$
  - E.g., for $\delta = 0.05$, the method is ε-differentially private with 95% probability
- The Gaussian mechanism is therefore weaker than the Laplacian mechanism, since it allows scenarios when the privacy cannot be guaranteed

DP in Machine Learning

Differentially Private SGD

- Training ML models can be considered an extension of the previous example on querying databases
  - I.e., ML models use data to learn a function, which is afterward used for prediction
- The datasets for training ML models often contain sensitive information (e.g., medical records, personal information), so it is important to provide privacy guarantees
  - On the other hand, we know that ML models can memorize the training data, which can be exploited by adversaries to recover information about the data from a trained model
- The challenge is: how to extract enough information from data to train accurate ML models without revealing the data

DP in Machine Learning

Differential Privacy

- In ML, DP is achieved by adding noise to:
  - ***Model parameters***
    - Several works applied DP to conventional ML methods
    - Differentially private SGD (Abadi, 2016) clips and adds noise to the gradients of deep NNs during training
      - This reduces the memorization of individual input instances by the model
    - The approaches that apply obfuscation to the model parameters via DP are also referred to as differentially private ML
  - ***Model outputs***
    - PATE (Private Aggregation of Teacher Ensembles) approach (Papernot, 2018) employs an ensemble of models trained on disjoint subsets of the training data, called teacher models
    - Noise is added to the outputs of the teacher models, and the aggregated outputs are used to train another model, called student model
  - ***Training data***
    - Obfuscation of training data in ML has been also investigated in several works

DP in Machine Learning

Differential Privacy

- DP is typically applied in a ***centralized learning setting***, where the data and model are at the same location
  - In this scenario, all data is gathered in one central location for model training
  - E.g., MLaaS typically requires that the users upload their data to a cloud-based server for training a model
- Recently, DP has also been applied in a ***distributed learning setting***, where the data are kept at separate locations from the model
  - DP-FedAvg (McMahan, 2018) is applied to federated learning
  - It introduced the Federated Averaging algorithm to limits the contributions by the data from individual users to the learning model

Distributed Learning

Distributed Learning

- ***Distributed learning*** allows multiple parties to train a global model without releasing their private data
- Some form of aggregation is applied to the local updates of the model parameters by the users in distributed learning to create a global model
  - E.g., averaging is one common form of aggregation
- Federated learning is the most popular distributed learning scheme

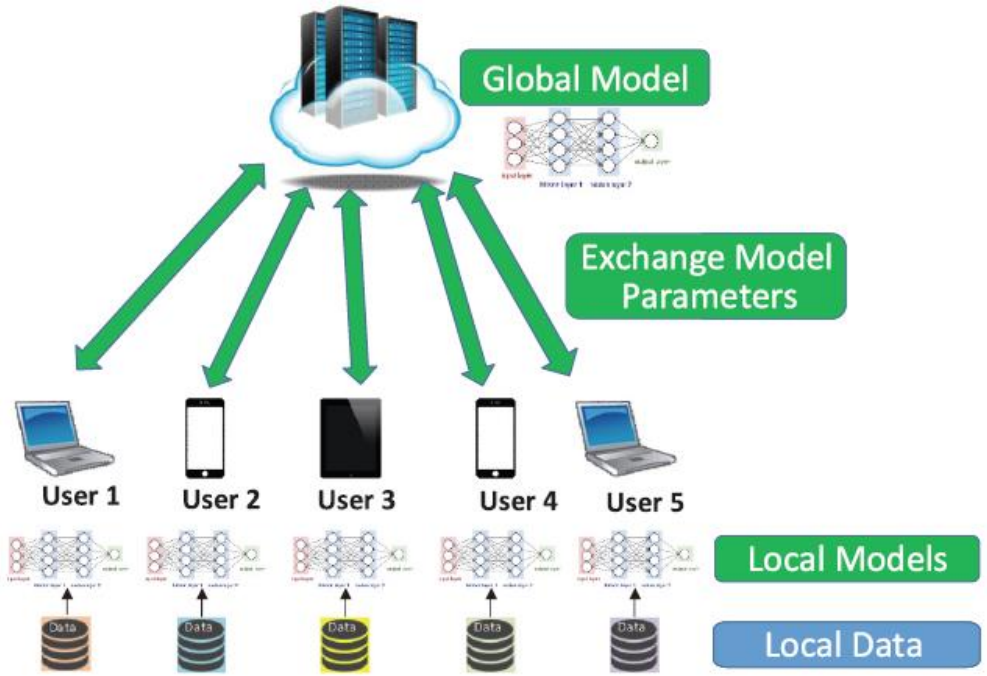


Figure form: Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook

Distributed Learning

Distributed Learning

- ***Federated learning*** or ***collaborative learning*** – learn one global model using data stored at multiple locations (e.g., remote devices)
  - The data are processed locally, and used to update the model
    - The data do not leave the remote devices, remains private
  - The central server aggregates the updates and creates the global model
- ***Decentralized Peer-to-Peer (P2P) learning*** – the remote devices communicate and exchange the updates directly, without a central server
  - Removes the need to send updates to a potentially untrusted central server
- ***Split learning*** – each remote device is used to train several layers of the global model, and send the outputs to a central server
  - The remote devices can train the initial layers of a DNN, and the central server can train the final layers
    - The gradient is back-propagated from the central server to each user to sequentially complete the back-propagation through all layers of the model
  - The devices send the intermediate layers outputs, rather than model parameters
  - Split learning is more common for IoT devices with limited computational resources

ML-Specific Techniques

ML-Specific Techniques

- In the lecture on privacy attacks in ML, we mentioned that overfitting is one of the reasons for information leakage
- ***Regularization techniques*** in ML can therefore be used to reduce overfitting, as well as a defense strategy
  - Different regularization techniques in NNs include:
    - Explicit regularization: dropout, early stopping, weight decay
    - Implicit regularization: batch normalization
- Other ML-specific techniques include:
  - Dimensionality reduction – removing inputs with features that occur rarely in the training set
  - Weight-normalization – rescaling the weights of the model during training
  - Selective gradient sharing – in federated learning, the users share a fraction of the gradient at each update

Introduction to Differential Privacy

Elijah Danquah Darko

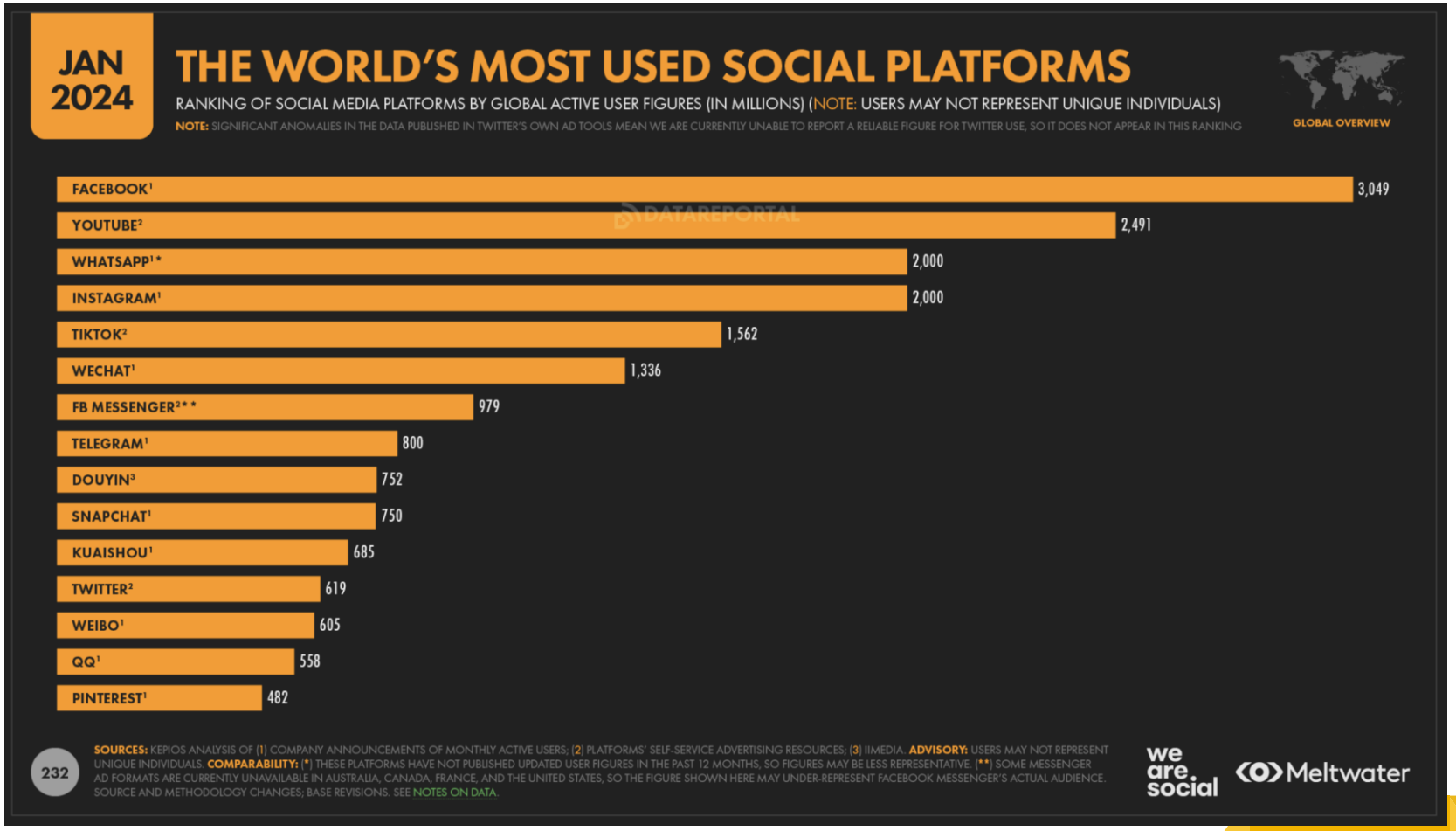
University of Idaho

Outline

- Privacy in information age
- Traditional Privacy Models
- Differential Privacy
- Differential Privacy Mechanism
- Applications in Machine Learning

University of Idaho

Privacy in information age



Source: https://datareportal.com/social-media-users

Traditional Privacy Models

- Anonymization Techniques
  - ❑ Data Anonymization / De-identification
    - Removing identifiable information from the dataset column-wise
    - Methods consist of stripping, masking, swapping, perturbing, some columns in the dataset

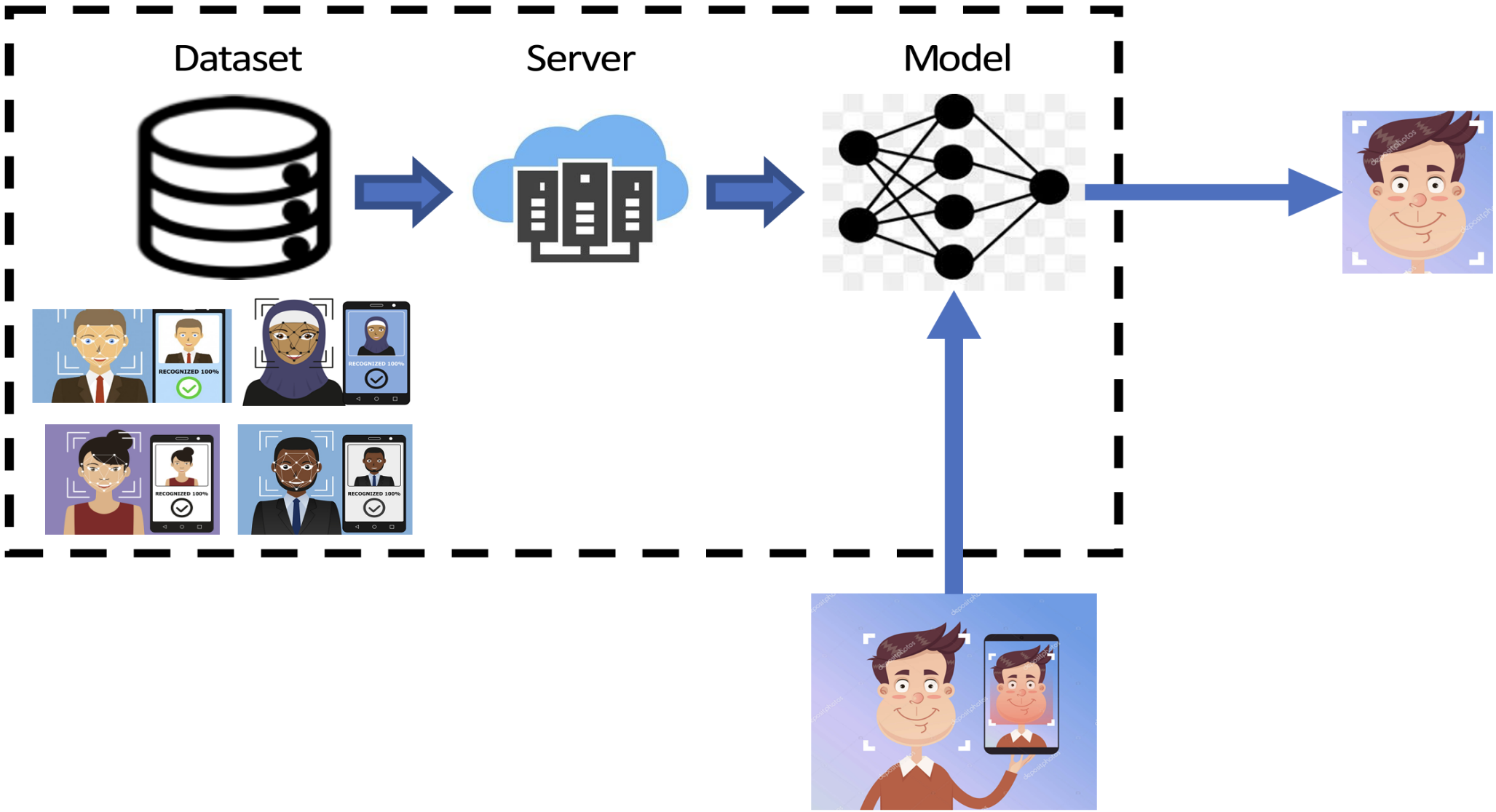
  - k-Anonymity [Emam et al.]

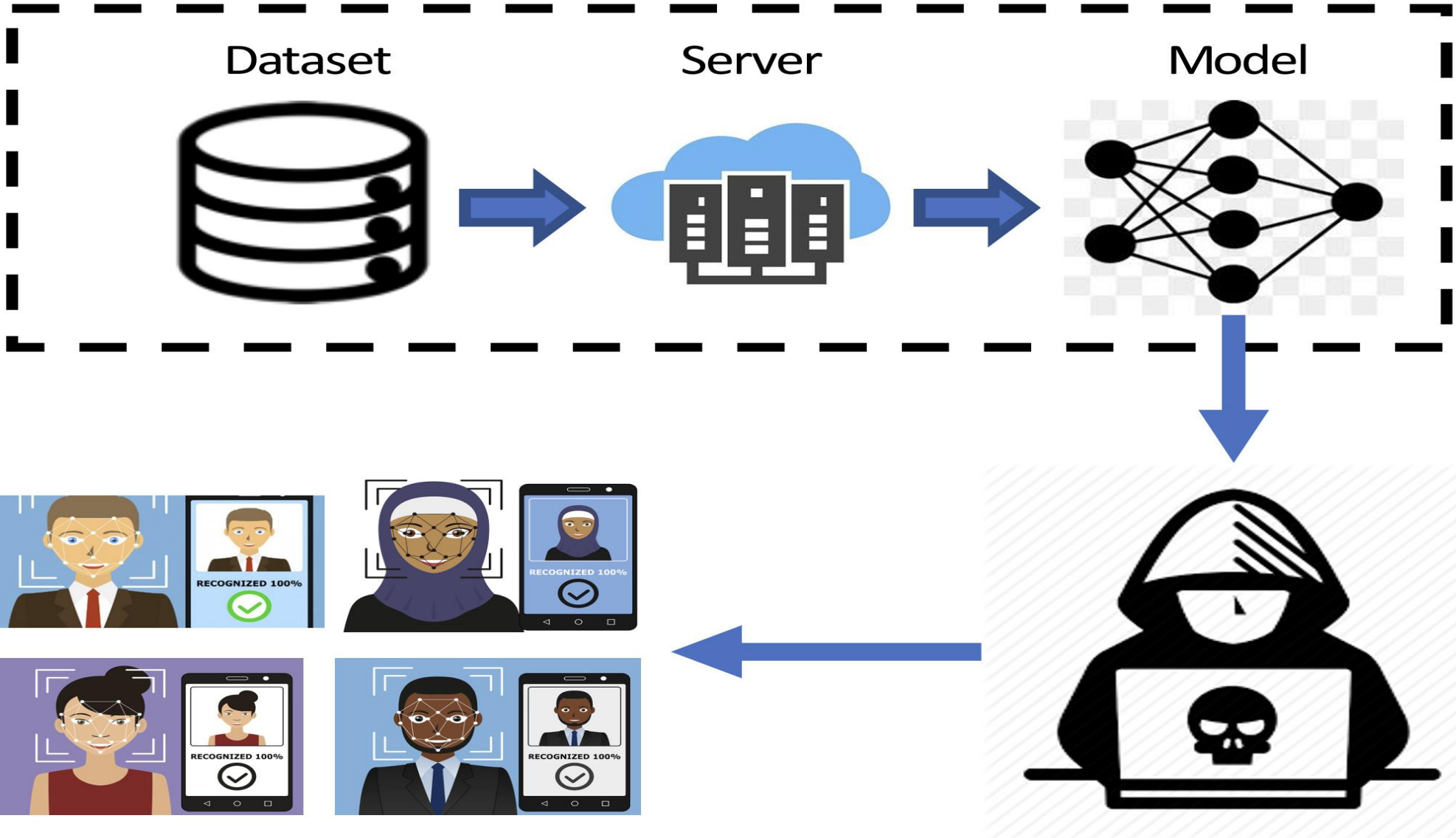
- Encryption Techniques
  - Homomorphic encryption

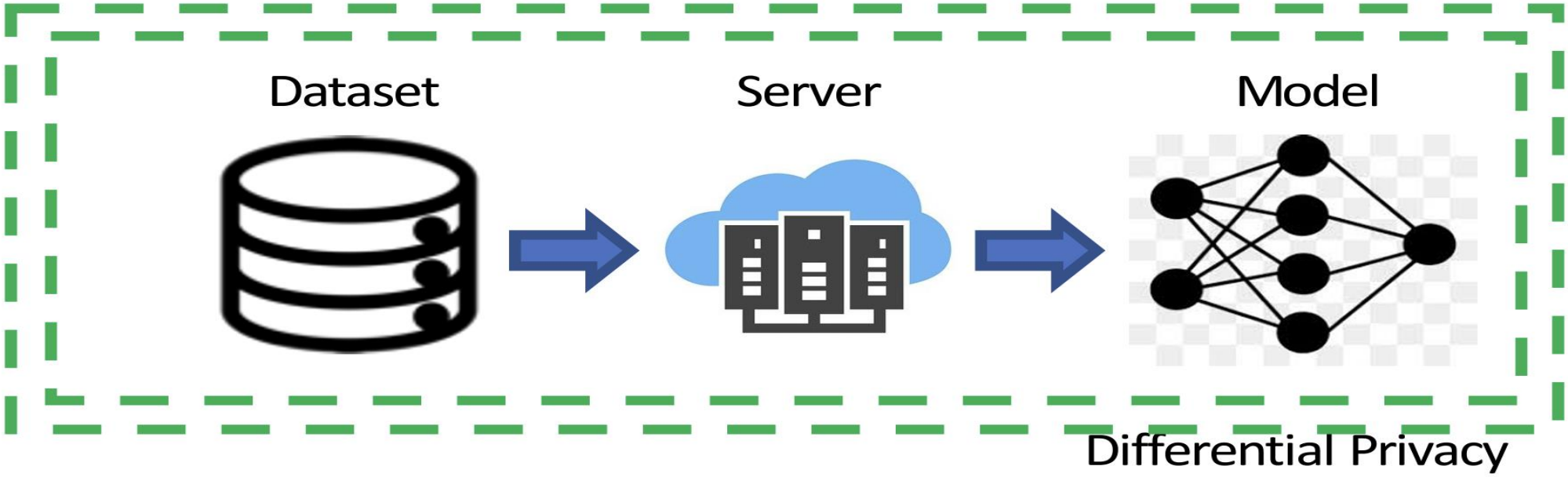
  - Secure multi-party computation

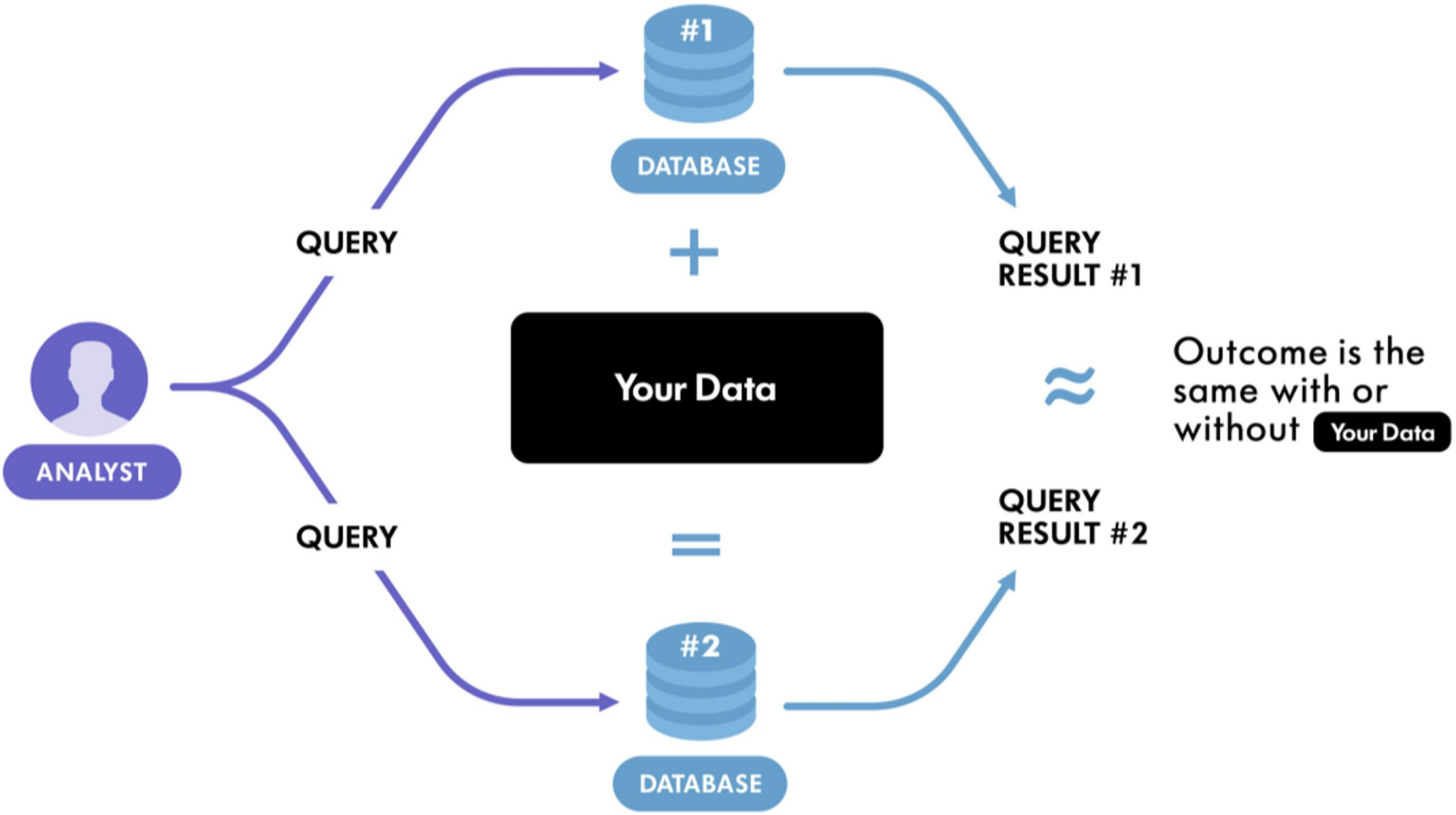
Differential Privacy

- Differential privacy is a privacy framework that aims to protect sensitive information while still allowing for useful data analysis.
- Differential privacy mathematically guarantees that anyone seeing the result of a differentially private analysis will essentially make the same inference about any individual's private information, whether or not that individual's private information is included in the input to the analysis.
- In practice, differential privacy involves adding noise to data before it is analyzed, so that the true values of individual data points are obscured. One of the key challenges in implementing differential privacy is balancing privacy with the usefulness of the data.
- Deep neural networks are highly expressive models that can potentially memorize individual training examples. Deep learning with differential privacy is an emerging field that combines the power of deep neural networks with the privacy guarantees of differential privacy.
- The goal is to develop machine learning algorithms that can analyze sensitive data while preserving the privacy of individuals in the data.


Dataset
Server
Model
RECOGNIZED 100%
RECOGNIZED 100%
RECOGNIZED 100%
RECOGNIZED 100%


Dataset
Server
Model
RECOGNIZED 100%
RECOGNIZED 100%
RECOGNIZED 100%
RECOGNIZED 100%


Dataset
Server
Model
Differential Privacy

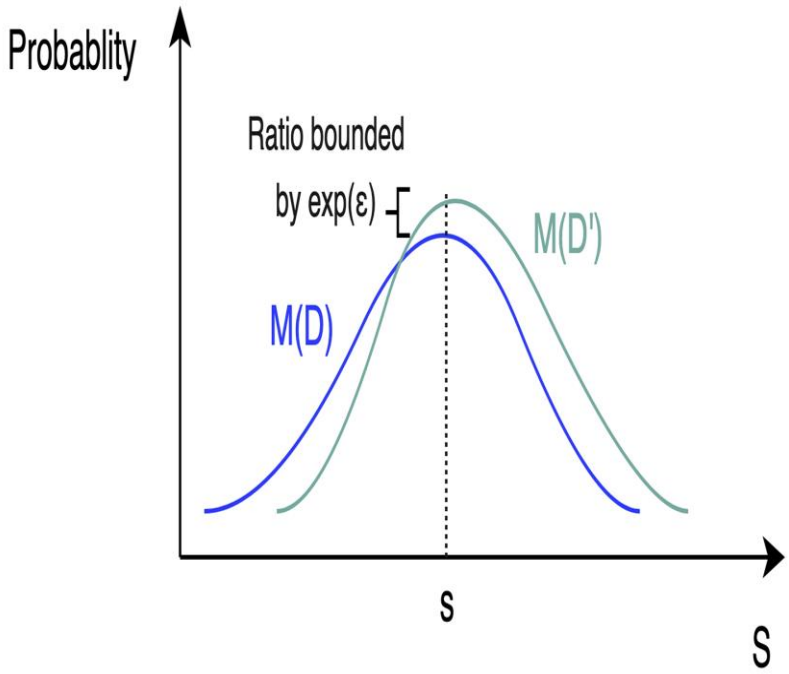

#1
DATABASE
QUERY
+
QUERY
RESULT #1
Your Data
≈
Outcome is the same with or without Your Data
ANALYST
QUERY
=
QUERY
RESULT #2
#2
DATABASE

Differential Privacy

- The formal definition of (ε, δ)-differential privacy:
  - A randomized mechanism M: D → R satisfies ε-differential privacy if for any two adjacent inputs d, d' ∈ D and for any subset of outputs S ⊆ R, it holds that:

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

  - ε is a parameter that determines the strength of the privacy guarantee provided by a differentially private mechanism. A smaller value of ε corresponds to a stronger privacy guarantee.
  - δ is the probability that allows for plain ε-differential privacy broken.



Differential Privacy Mechanisms

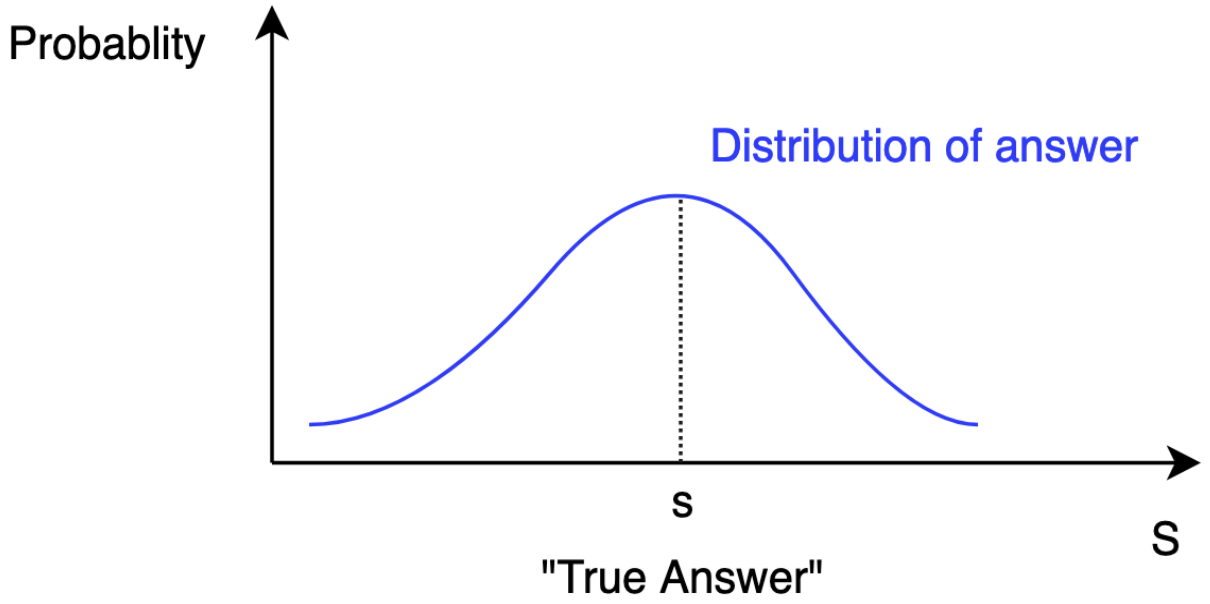
Randomized Response: A first attempt at Differential Privacy [Warner, 1965]

- **Problem**: Given a survey which contains a sensitive or embarrassing question (**query**), how to protect the privacy of participants' responses while performing statistics on the **answers.**
- **Approach:** Use coin flips
  - When participants are asked a sensitive question, they flip a coin before answering
  - If they get heads, they are asked to answer YES to the question regardless of their experience.
  - When they get tails, they are asked respond truthfully according to their experience.
- **Results:** Double the resulting statistics
  - At the end of the survey, half of the participant would have answer correctly, while the other half answer falsely.
  - To account for the half false answers they got, the organizers double the statistics at the end of the survey,
- **Alternative**: [Greenberg, 1969]
  - When participants get a tail, they are asked to flip a coin again to answer YES when it is heads, and NO when tails.

Differential Privacy Mechanisms

Randomized Response: A first attempt at Differential Privacy [Warner, 1965]

- The query answer is no longer a deterministic value, but a sampling from a distributio

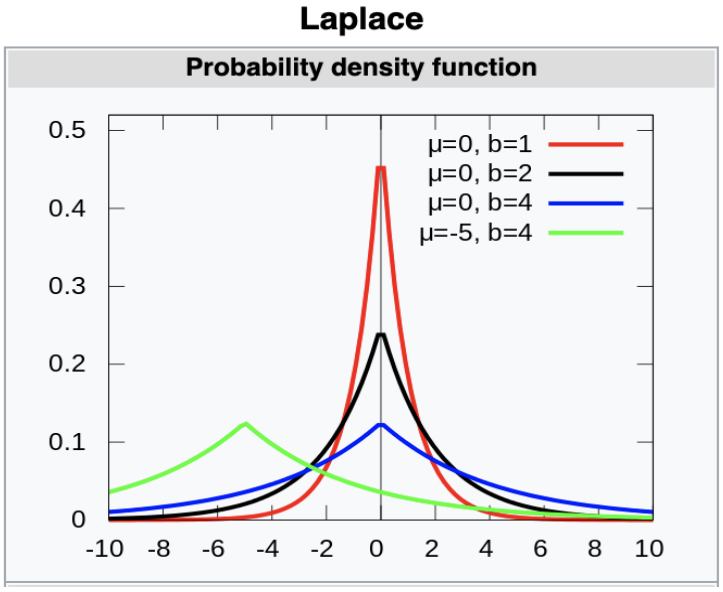


Differential Privacy Mechanisms

Laplace Mechanism: Achieving differential privacy by adding Laplace noise

- Privacy mechanism which outputs **q(x) + (v ~ Lap(Δq, ε))**
- Laplace distribution is centered at 0 with a std of $b\sqrt{2}$.
- Its probability density function is given by:

$$f(x \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$



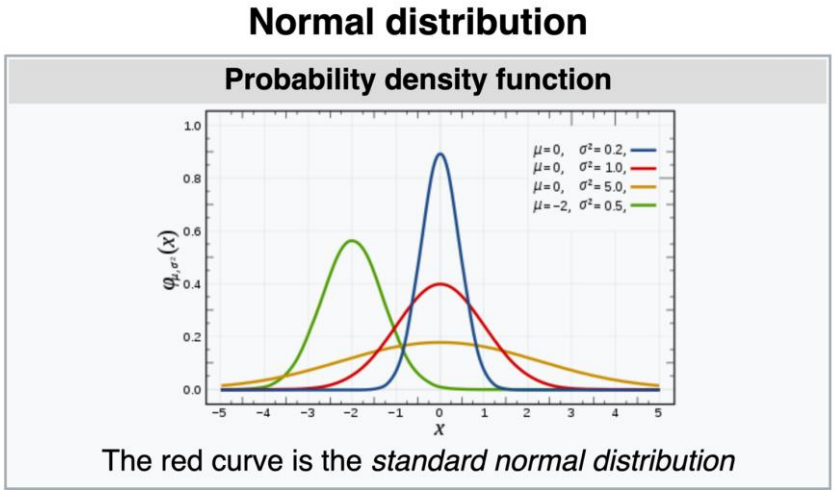
Source: Wikipedia

Differential Privacy Mechanisms

Gaussian Mechanism: Achieving differential privacy by adding Gaussian noise

- Privacy mechanism which outputs **q(x) + (v ~ N(Δq, ε))**
- The Gaussian or Normal Distribution has a probability density function given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



Source: Wikipedia

Differential Privacy Mechanisms

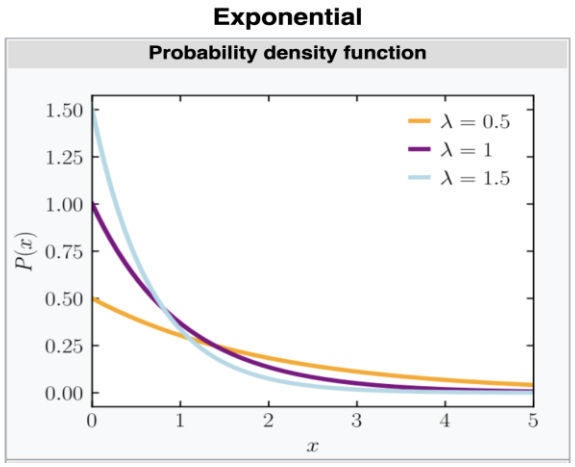
Exponential Mechanism: Achieving differential privacy by selecting lowest sensitivity score

- Given a sensitivity score function **H**, selects answer **a** with the lowest sensitivity score such that $P(a \in A \text{ is selected}) \propto e^{\epsilon H(D,a)/2s(H,\|.\|)}$

$$s(H, \|.\|) = \max_{d(D,D')=1, a \in A} \|H(D,a) - H(D',a)\|$$

- The Exponential distribution is parameterized by with a probability density function given by:

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$



Source: Wikipedia

Applications in Machine Learning

Issues with Deep Learning without privacy mechanisms

- Neural network models memorize training examples [Carlini et al., 2019]
  - **Memorization violates privacy**

- Given a public model, it is possible to a transformed (blurred) image through Generative Model-Inversion (GMI) attacks [Zhang et al.]

Target

GMI

Applications in Machine Learning

- In Machine/Deep Learning, Differential Privacy can be achieved by adding noise to or perturbing :
  - *Training data*
  - *Model outputs*
  - *Model parameters*
    - Differentially private SGD (Abadi, 2016) clips and adds noise to the gradients of deep NNs during training
      - Aims to reduce memorizations of training data of by the model

Applications in Machine Learning

Differentially Private Deep Learning

Differential Privacy in Gradient Descent

- Update rule [Xie et al.]

$$w_{t+1} = w_t - \eta_t(\nabla f(w_t; D_{m(t)}) + N_t(0,\ \lambda))$$

- Due to sequential composition, the privacy loss in unbounded when we perform many iterations of the SGD algorithm
- More iterations leads to a larger privacy cost. However, in practical Deep Learning, more iterations generally result in a better model.
- But, in differentially private SGD version, more iterations can make the model worse, i.e., the noise increases with each iterations.
- Tradeoff: Find the right balance between the number of iterations and the scale of the noise added.
- Some Techniques: **Gradient clipping** before adding the noise [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.

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.

Applications in Machine Learning

Differentially Private Deep Learning with Pytorch and Opacus

```
model = Net()
optimizer = SGD(model.parameters(), lr=0.05)
privacy_engine = PrivacyEngine(
    model,
    sample_rate=0.01,
    alphas=[1, 10, 100],
    noise_multiplier=1.3,
    max_grad_norm=1.0,
)
privacy_engine.attach(optimizer)
# Now it's business as usual
```

Source: https://opacus.ai/

Conclusion

- Traditional data analysis is done without any privacy mechanism which leads to privacy violation
- Traditional privacy models are vulnerable to linking attacks
- Randomized Response was the first attempt to offer differential privacy.
- The fixed privacy loss in Randomized Response makes it unsuitable for some application
- Methods based on sampling from a distribution (Laplace, Gaussian, Exponential, etc) offer better privacy guarantees.
- Differential Privacy can be applied to ML to achieve Privacy Preserving ML.

Thank you
Questions?

University of Idaho

Differentially Private SGD

Differentially Private SGD

- Abadi (2016) Deep Learning with Differential Privacy
- This work introduced differential privacy (DP) for training ML models for protecting the privacy of the training data
  - ***Differential privacy (DP)*** is applied to ***Stochastic Gradient Descent (SGD)*** during model training
  - DP-SGD clips the gradients and adds Gaussian noise to the gradients with respect to the model parameters
  - This approach controls the amount of information from the training data that is memorized by the model during training
  - The goal is to produce ML models which provide approximately the same privacy when an individual input instance is removed from the training dataset
- The paper also introduces a method for calculating the privacy loss, called ***moments accountant***

Differentially Private SGD

Differentially Private SGD

- Differentially Private Stochastic Gradient Descent (DP-SGD)
  - DP-SGD adds two additional steps to SGD: Clip gradient, and Add noise

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.

Introduced DP steps

Gradient Clipping

Differentially Private SGD

- ML models tend to memorize more information about some input samples than others
  - Input samples that produce large gradients are prone to be memorized by the model, and violate privacy
- DP-SGD approach proposes to clip the ℓ_2 norm of the gradient to a threshold C, in order to limit the influence by the individual input samples
  - Also, since the values of the gradients cannot be estimated ahead of time, the clipping operation controls the sensitivity of the DP randomization mechanism
- If the gradient at step t by an input sample x_i is $\mathbf{g}_t(x_i) = \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$, the clipped gradient $\bar{\mathbf{g}}_t(x_i)$ is:

$$\bar{\mathbf{g}}_t(x_i) = \begin{cases} \mathbf{g}_t(x_i) & \text{if } \| \mathbf{g}_t(x_i)\|_2 \leq C \\ \dfrac{\mathbf{g}_t(x_i)}{\| \mathbf{g}_t(x_i)\|_2 / C} & \text{if } \| \mathbf{g}_t(x_i)\|_2 > C \end{cases}$$

- That is, if the norm $\| \mathbf{g}_t(x_i)\|_2$ is greater than C, the gradient is scaled down to have a norm equal to C

Adding Noise

Differentially Private SGD

- GP-SGD approach employs a Gaussian randomization mechanism
- Gaussian noise is added to the gradients at each training step t, according to:

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

- At each step t, the average of the clipped gradient for a batch of inputs (with a batch size L) is first calculated as $\frac{1}{L}\sum_i \bar{\mathbf{g}}_t(x_i)$
- Gaussian noise $\mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$ with mean 0 and diagonal co-variance $\sigma^2 C^2$ is afterward added to the batch-averaged gradient
  - Note that the co-variance is a function of the clipping threshold C
  - E.g., larger value of C does less clipping, but requires more noise to achieve the same level of privacy

Moments Accountant

Differentially Private SGD

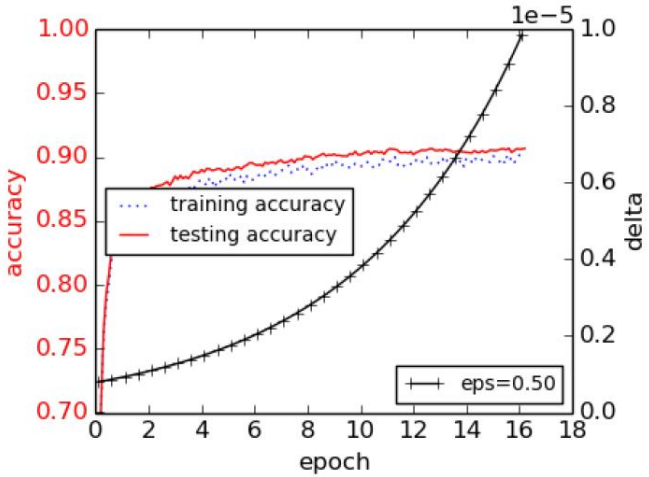
- The composition property in DP states that if the privacy loss (budget) for one interaction with the data is ε_1 and for another interaction with the data is ε_2, the combined privacy loss is $\varepsilon_1 + \varepsilon_2$
  - Therefore, by accumulating the privacy loss for each mini-batch when training an ML model, it is possible to calculate the overall privacy loss during training
- ***Moments accountant*** is an introduced approach in the paper that evaluates the privacy loss of a model training with DP-SGD
  - The privacy loss is estimated at each training step, and it is used to calculate the cumulative privacy loss over all training epochs
  - Note that increasing the number of training epochs increases the privacy loss
    - E.g., training a model for 100 epochs that achieved a privacy loss of $\varepsilon = 1.26$, when training for 400 epochs the privacy loss increased to $\varepsilon = 2.55$
- Moments accountant employs the moments of mixtures of Gaussian distributions to calculate the upper bound of the cumulative privacy loss
  - The approach is described in more detail in the paper

Experimental Evaluation

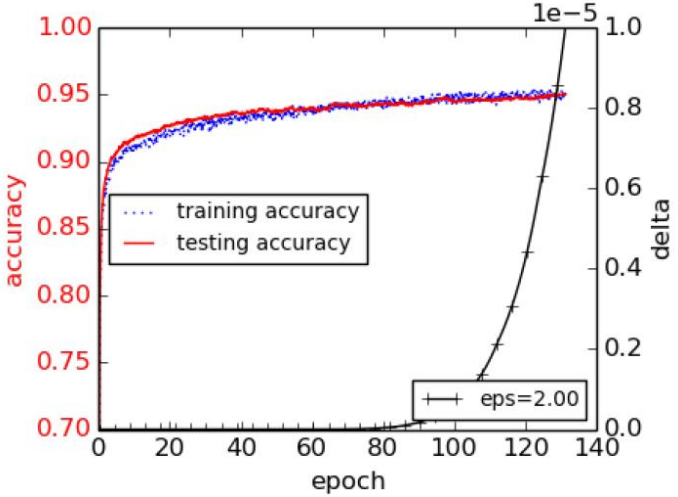
Differentially Private SGD

- MNIST dataset
  - Training and testing accuracies for three different levels of Gaussian noise with variances $\sigma \in \{8, 4, 2\}$, and clipping threshold $C = 4$
  - The corresponding privacy budget values are set to $\varepsilon \in \{0.5, 2, 8\}$
    - The upper bound for the privacy probability parameter is set to $\delta = 10^{-5}$
      - Thus, the (ε, δ)-differential privacies are $(0.5, 10^{-5})$, $(2, 10^{-5})$, $(8, 10^{-5})$
      - Moment Accountant allows to obtain a δ value for a fixed privacy budget ε
  - The obtained test set accuracies are 90%, 95%, and 97%, respectively
  - Larger noise achieves lower test accuracy, but provides increased privacy protection

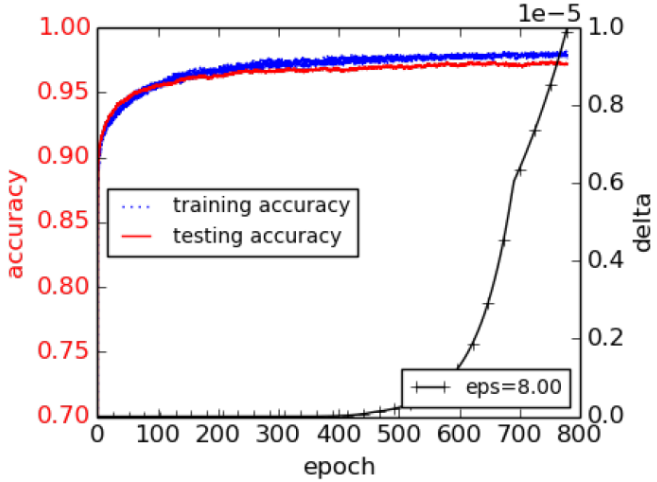
Large noise, $\varepsilon = 0.5$



Medium noise, $\varepsilon = 2$



Small noise, $\varepsilon = 8$

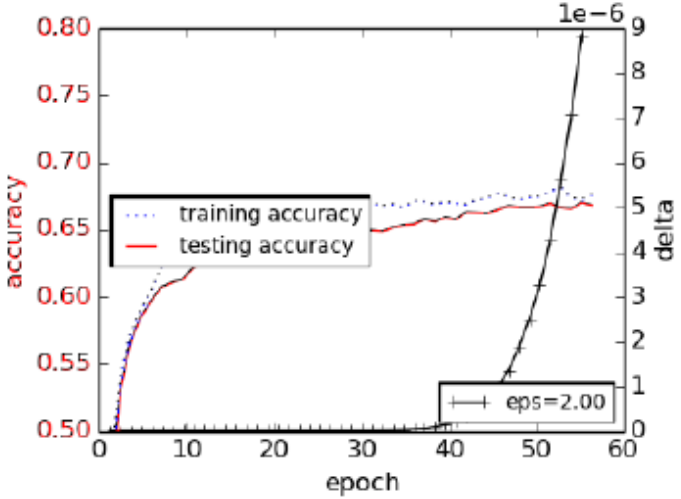


Experimental Evaluation

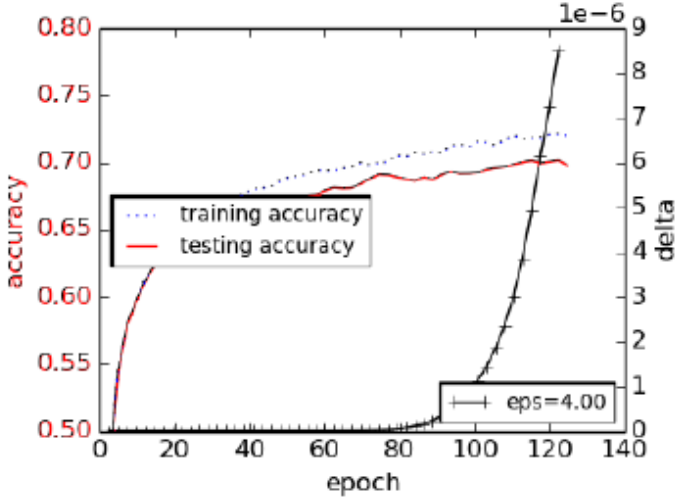
Differentially Private SGD

- CIFAR-10 dataset
  - The results are similar to the obtained performance for MNIST
  - Training and testing accuracies for three different levels of privacy budget $\varepsilon \in \{2, 4, 8\}$
    - The target probability parameter is set to $\delta = \{10^{-5}, 10^{-6}\}$
  - The Gaussian noise variance is fixed to $\sigma = 6$ for all experiments, the clipping threshold is $C = 3$
  - The achieved test set accuracies are 67%, 70%, and 73%, respectively

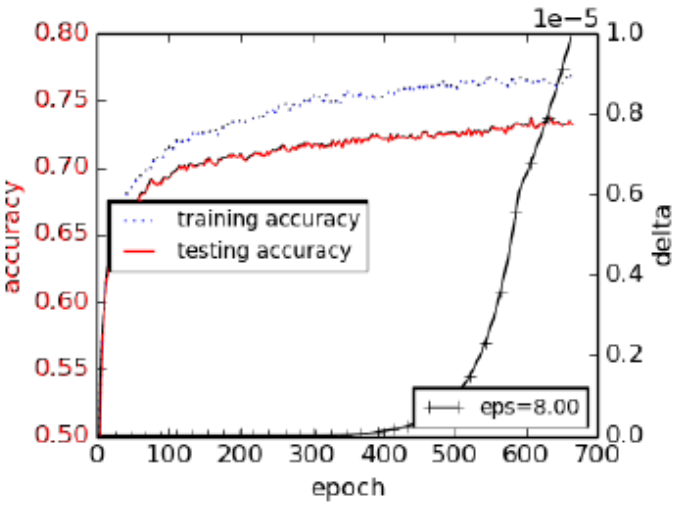
Privacy $\varepsilon = 2$



Privacy $\varepsilon = 4$



Privacy $\varepsilon = 8$



Privacy versus Accuracy Trade-Off

Differentially Private SGD

- Perfect privacy in ML models is not possible
  - Adding too much noise to the model parameters would diminish the accuracy and the usefulness of the model
  - There is a trade-off between privacy protection and accuracy
- DP-SGD achieves privacy protection for deep NNs with a small decrease in the model accuracy and small increase in the training complexity
  - The approach adds Gaussian noise to the gradients in SGD to reduce the possibility for memorization of individual input instances by the model
  - This work also developed the Moments Accountant approach to calculate the cumulative privacy loss for the combination of the model and dataset

Scalable Private Learning with PATE

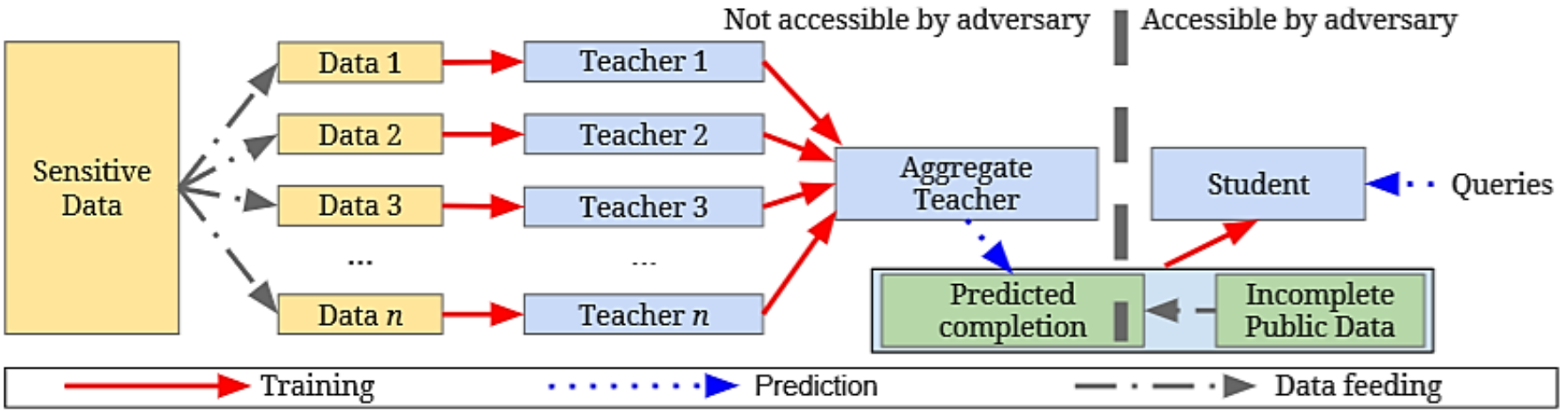
Scalable Private Learning with PATE

- ***Scalable Private Learning with PATE***
  - Papernot (2018) Scalable Private Learning with PATE
- PATE (Private Aggregation of Teacher Ensembles) is a privacy-preserving ML framework that enables model-agnostic training while providing differential privacy guarantees for the training dataset
  - It provides privacy without significantly compromising performance, making it a valuable approach for various applications where data privacy is a primary concern
- PATE uses a collection of teacher models to train a student model that answers user queries
  - PATE injects noise into the predictions by the teacher models to provide DP guarantees
  - The users cannot determine any PII from the predicted outputs or identify whose data the models were trained on

PATE Framework Components

Scalable Private Learning with PATE

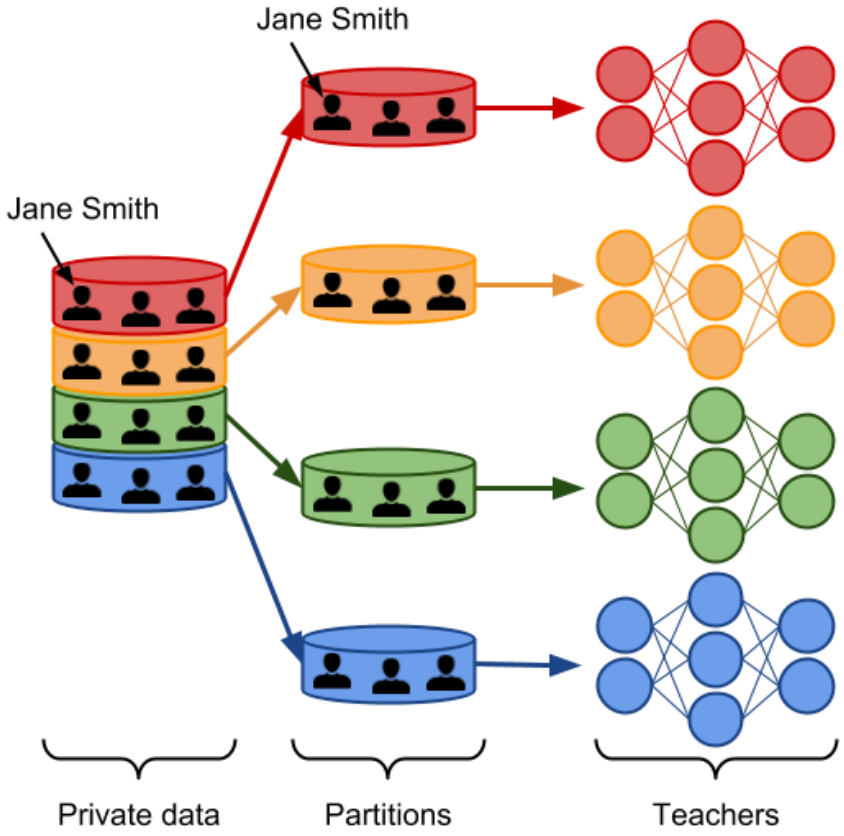
- In the PATE framework, multiple teacher models are trained independently on disjoint subsets of the sensitive training dataset
  - Any ML technique can be used for the teacher models
  - The predictions by the teacher models are aggregated where DP with controlled Gaussian noise is introduced to obtain the aggregated prediction
- A student model learns from the teacher models by querying them through the aggregator
  - The student model is trained on insensitive public data, allowing to generalize the knowledge acquired from the teacher models without directly accessing sensitive data



PATE Framework

Scalable Private Learning with PATE

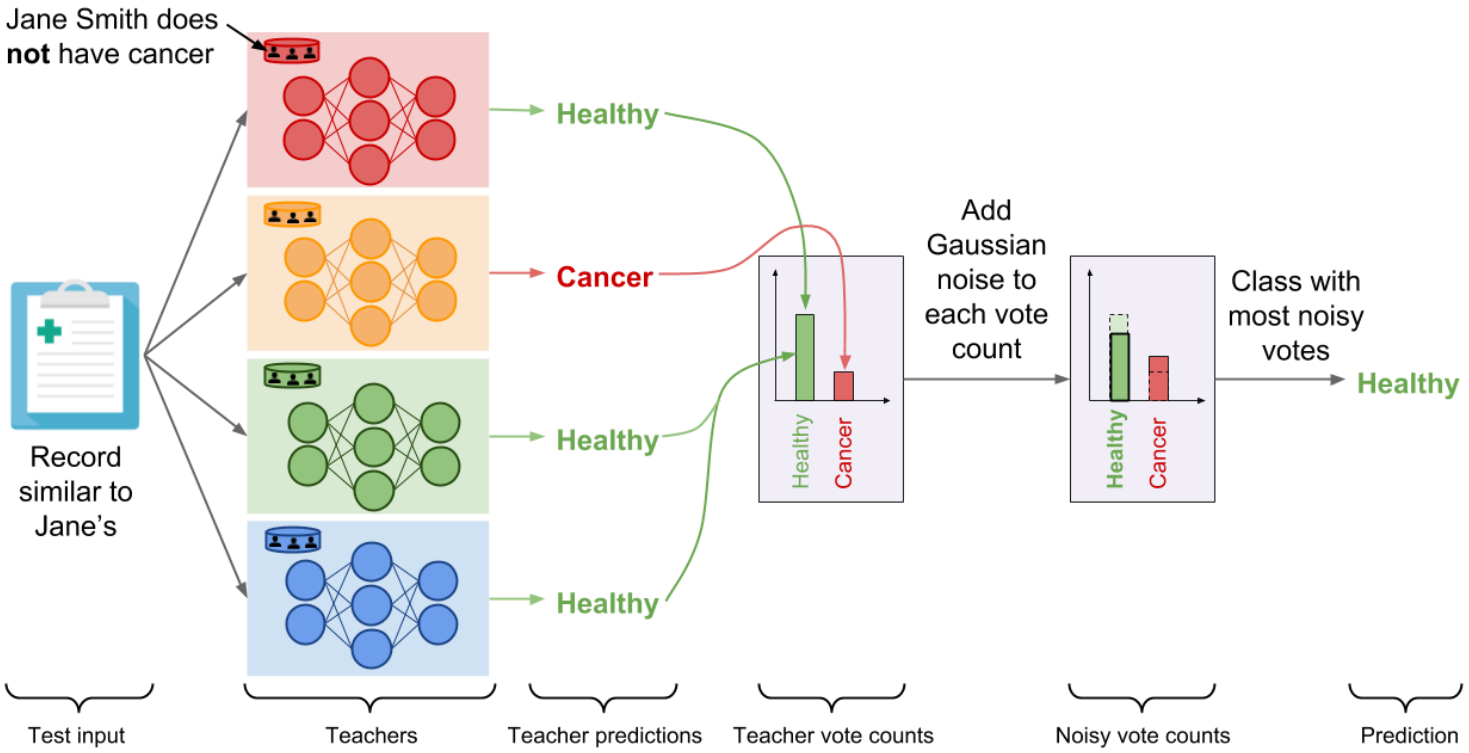
- The PATE approach is based on a simple intuition: if two different classifiers (teacher models), trained on two different datasets with no training examples in common agree on how to classify a new input example, then that decision does not reveal information about any single training example



PATE Framework

Scalable Private Learning with PATE

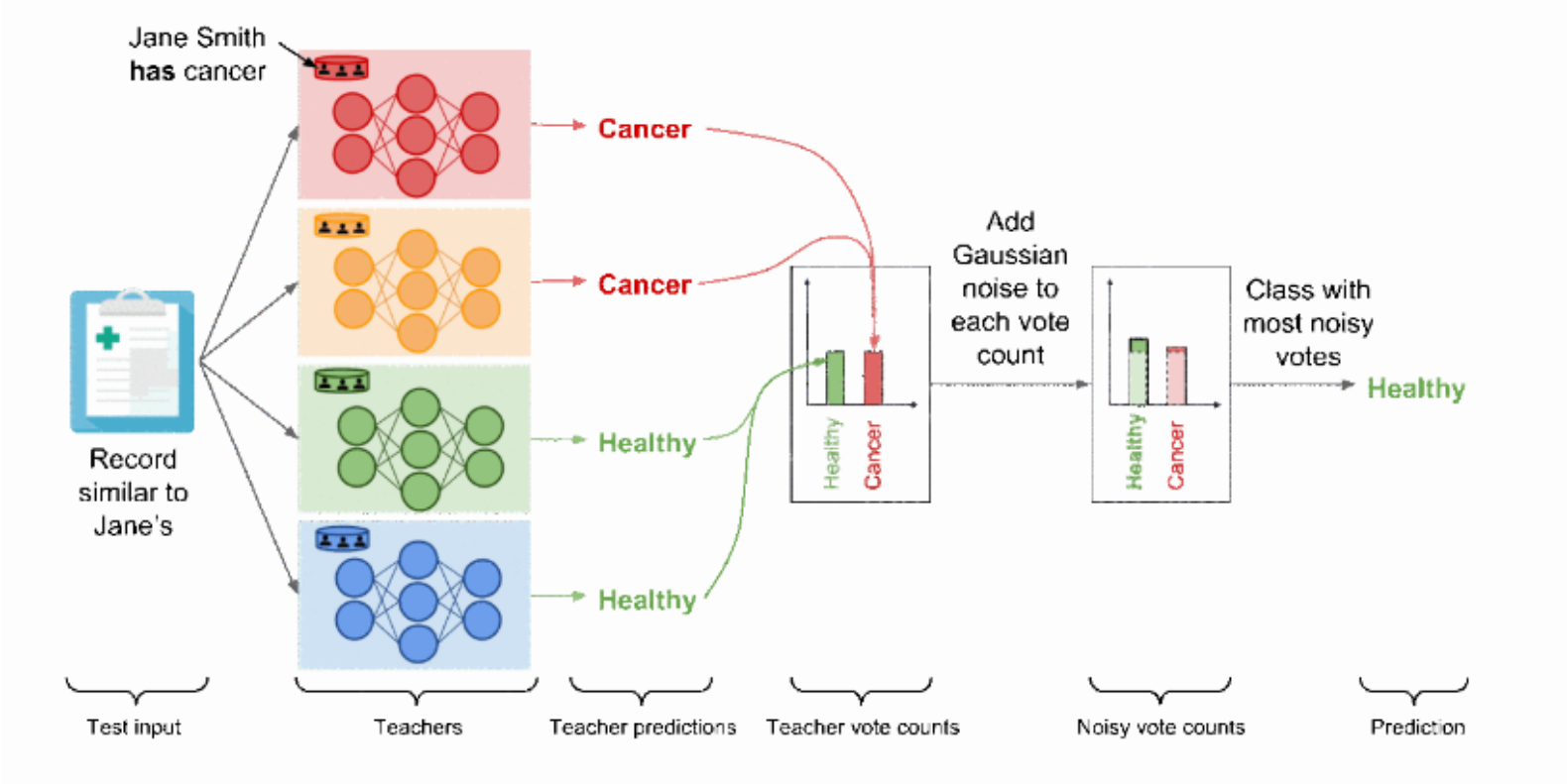
- PATE aggregates the votes of the teacher models for each class, and perturbs the vote counts by adding random DP noise sampled from Gaussian distribution
  - If most teachers agree on the same class, adding noise will not change the final predicted class ("Healthy" in this example)



PATE Framework

Scalable Private Learning with PATE

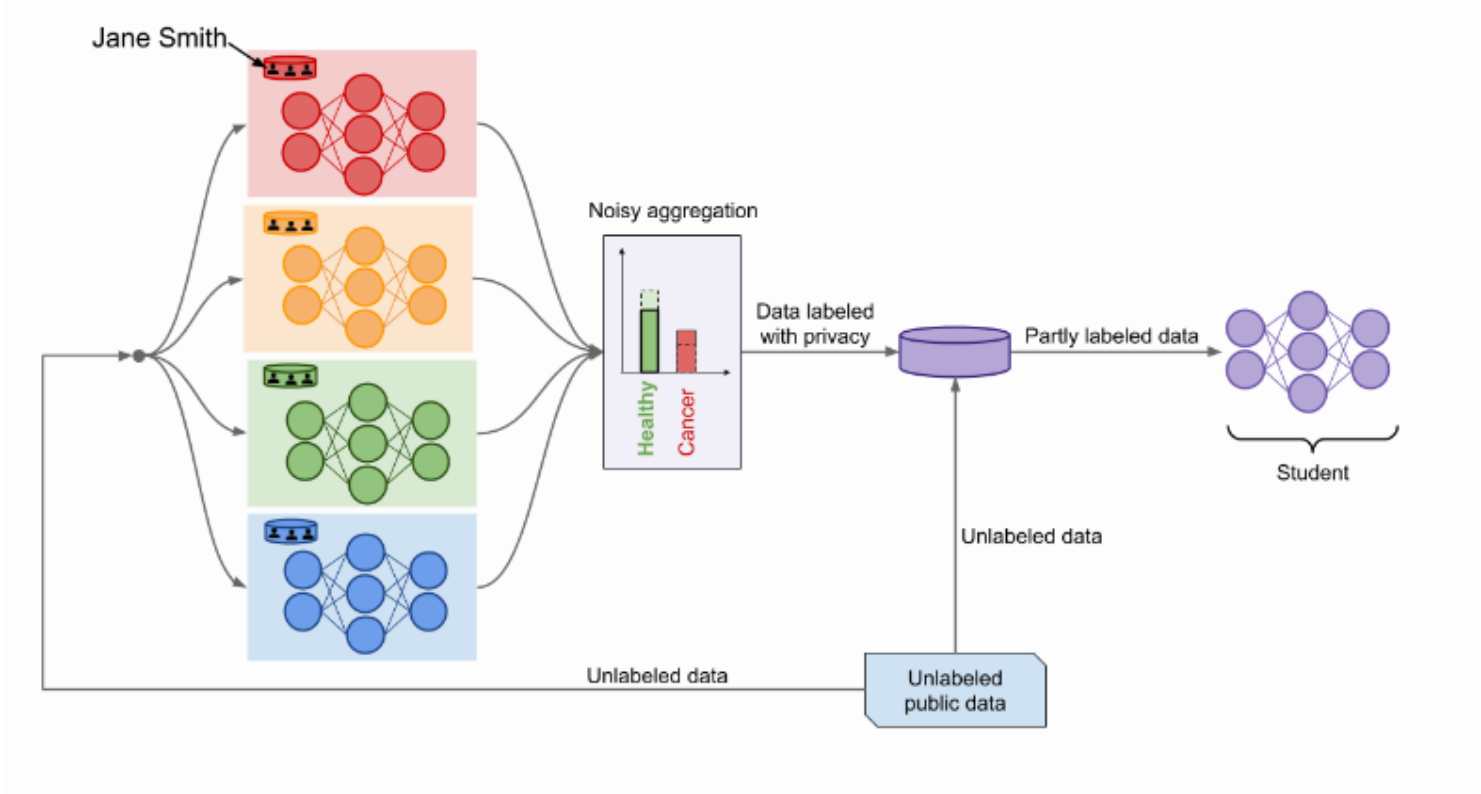
- If the aggregated votes by the teacher models for both classes are similar, adding DP noise will prevent revealing the votes of the individual teachers and protect the privacy
  - I.e., the noisy aggregated outcome is equally likely for "Healthy" and "Cancer" classes



PATE Framework

Scalable Private Learning with PATE

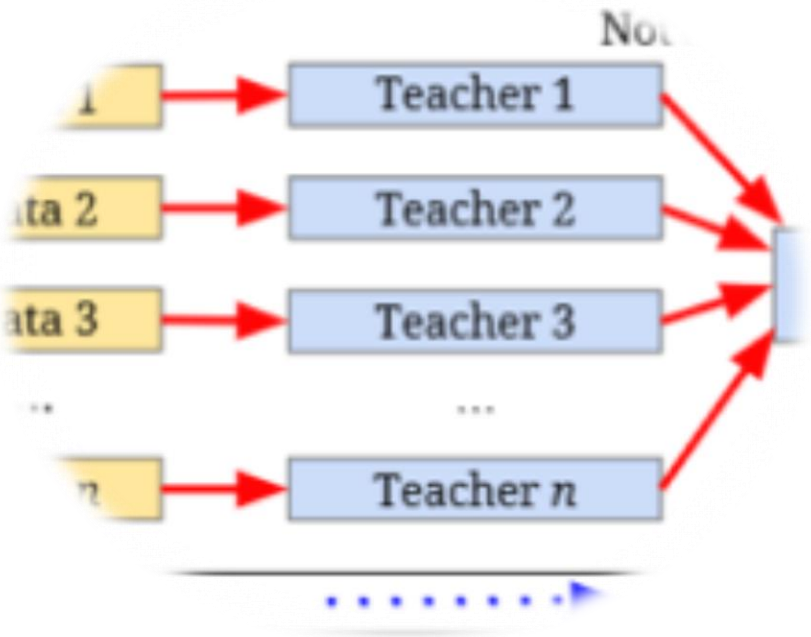
- After the teacher models are trained, the student model selects inputs from unlabeled public data and submit the inputs to the teacher ensemble to predict the class labels
  - The noisy labels from the aggregation mechanism are used to train the student model
  - Finaly, the trained student model is deployed for access to end-users



Teacher Models

Scalable Private Learning with PATE

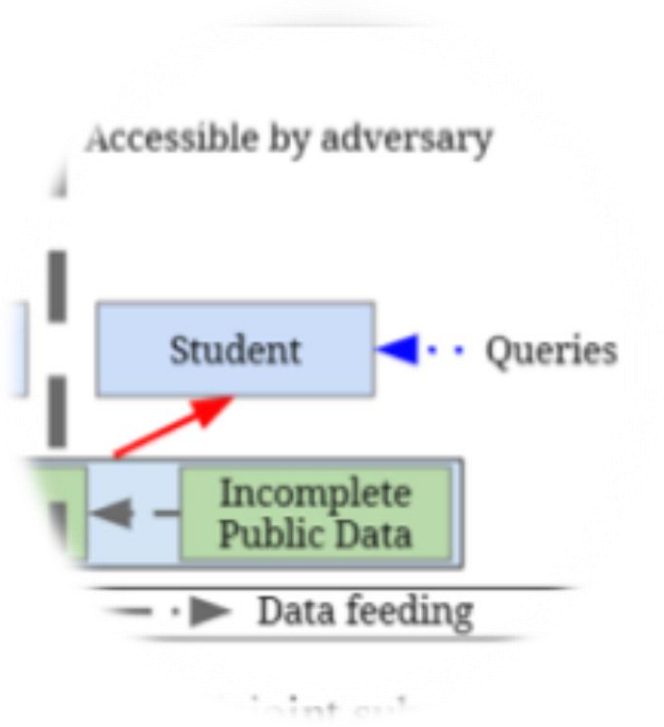
- ***Teacher models*** are trained on disjoint subsets of the sensitive training dataset, each with its own portion of the data
  - The models provide expertise on the data they were trained on while maintaining privacy
  - By using multiple teachers, the PATE framework leverages the wisdom of the ensemble, which helps increase the overall accuracy and generalization capabilities of the student model



Student Model

Scalable Private Learning with PATE

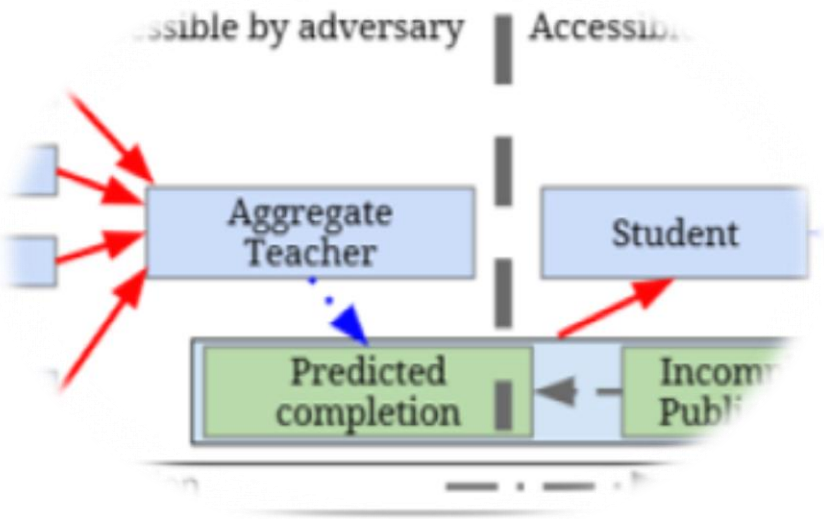
- ***Student model*** learns from the teacher models without having direct access to the sensitive data
  - It is trained on a public unlabeled dataset (i.e., without sensitive private data), which is labeled by interacting with the ensemble of teachers via the aggregator
  - The student model aims to replicate the performance of the teachers while ensuring privacy protection for the sensitive training data



Aggregators

Scalable Private Learning with PATE

- PATE framework proposed two new methods for aggregating teacher/student answers
  1. Confident Aggregator
     - Check if the vote by the teacher models is above a threshold (i.e., teacher reach consensus regarding the correct classification), and add smaller levels of noise
     - Eliminate queries where teachers don't reach strong consensus regarding the predicted class
  2. Interactive Aggregator
     - Check student confidence in class prediction, and if the student confidently predicts the same class as the teachers, eliminate those queries



Experimental Setup

Scalable Private Learning with PATE

- Datasets (contain private personal attributes)
  - MNIST
  - Street View House Numbers (SVHN)
  - US Census Income Adult (CI Adult)
  - Glyph: synthetically generated computer font symbols with 150 different classes
- Teacher ensembles: 250 number of teachers & partitions of data for MNIST, SVHN, and Adult, and 5,000 for Glyph
- Privacy probability parameter is set to $\delta = 10^{-5}$ for MNIST and ADULT, $\delta = 10^{-6}$ for SVHN, $\delta = 10^{-8}$ for Glyph (probability privacy will not be held)
- The total privacy loss ε is calculated on a subset from 100 to 12,000 queries depending on dataset

Figure 3: **Some example inputs from the Glyph dataset along with the class they are labeled as.**

Experimental Results

Scalable Private Learning with PATE

- Comparison of the proposed Confident-GNMax aggregation approach (based on Gaussian noise) to LNMax aggregation approach (based on Laplacian noise)
  - GNMax approach reduced the privacy bound/loss ε for all datasets, and increased the accuracy in student predictions for all datasets, in comparison to LNMax

Dataset	Aggregator	Queries answered	Privacy bound ε	Accuracy	
				Student	Baseline
MNIST	LNMax (Papernot et al., 2017)	100	2.04	98.0%	99.2%
	LNMax (Papernot et al., 2017)	1,000	8.03	98.1%	
	Confident-GNMax (T=200, σ_1=150, σ_2=40)	286	**1.97**	**98.5%**	
SVHN	LNMax (Papernot et al., 2017)	500	5.04	82.7%	92.8%
	LNMax (Papernot et al., 2017)	1,000	8.19	90.7%	
	Confident-GNMax (T=300, σ_1=200, σ_2=40)	3,098	**4.96**	**91.6%**	
Adult	LNMax (Papernot et al., 2017)	500	2.66	83.0%	85.0%
	Confident-GNMax (T=300, σ_1=200, σ_2=40)	524	**1.90**	**83.7%**	
Glyph	LNMax	4,000	4.3	72.4%	82.2%
	Confident-GNMax (T=1000, σ_1=500, σ_2=100)	10,762	2.03	**75.5%**	
	Interactive-GNMax, two rounds	4,341	**0.837**	73.2%	

References

1. Liu et al. (2020) When Machine Learning Meets Privacy: A Survey and Outlook (link)
2. Rigaki and Carcia (2021) A Survey of Privacy Attacks in Machine Learning (link)
3. Cristofaro (2020) An Overview of Privacy in Machine Learning (link)
4. Borealis AI Tutorial #13: Differential Privacy II: Machine Learning and Data Generation (link)
5. Davide Testuggine and Ilya Mironov blog: Differential Privacy Series Part 1- DP-SGD Algorithm Explained (link)
6. Joseph P. Near and Chiké Abuah, Programming Differential Privacy – Chapter III: Differential Privacy (link)