Trustworthy AI Systems

-- Privacy of AI

Instructor: Guangjing Wang

guangjingwang@usf.edu

Last Lecture

- Poisoning Attacks

- Poisoning Scenarios
  - Centralized
  - Distributed

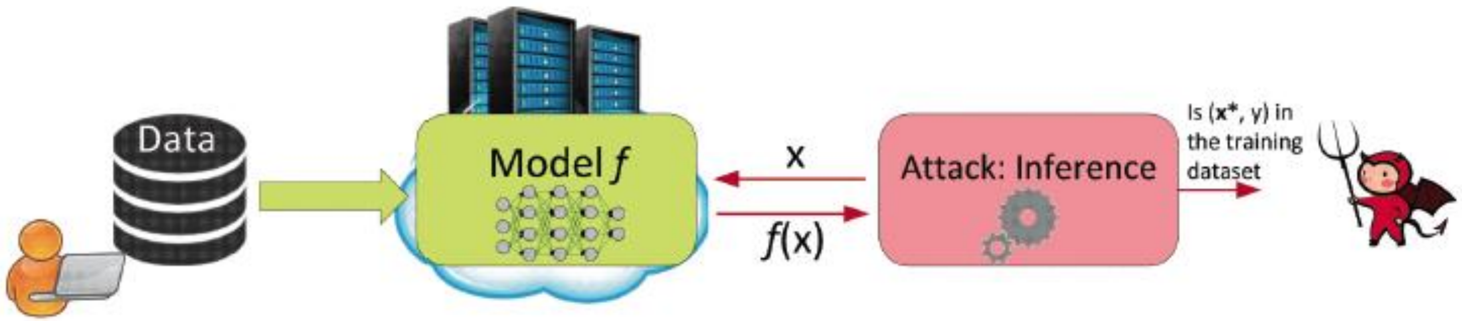
- Defense for Poisoning Attacks

This Lecture

- Membership Inference Attacks
- Model Inversion Attacks
- Model Stealing Attacks
- Privacy Protection Methods

Membership Inference Attacks

- Determine whether an individual data instance x^* is part of the training dataset $\mathcal{D}$ for a model.
- The membership inference attacks on both supervised classification models and generative models (GANs, VAEs) have been demonstrated.
- A common approach is to first train several ***shadow models*** that imitate the behavior of the target model and use the prediction vectors of the shadow models for training a binary classifier (that infers the membership).



https://dl.acm.org/doi/abs/10.1145/3436755

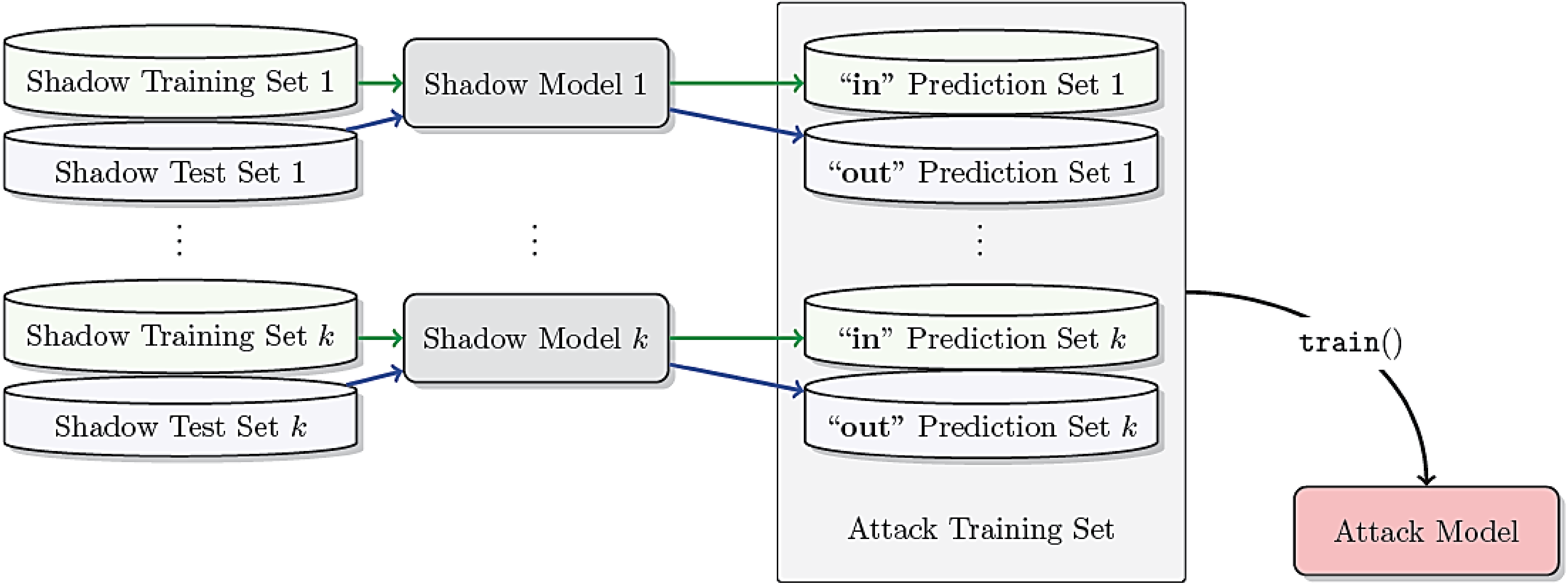
Shadow Training Attack (1)

- Threat model:
  - The adversary has back-box query access to the target model
  - The goal is to infer whether input samples were part of its private training set
- Shadow training approach:
  - Create several shadow models to substitute the target model
  - Each shadow model is trained on a dataset that has a similar distribution as the private training dataset of the target model
    - E.g., if the target model performs celebrity face recognition, the attacker can collect images of celebrities from the Internet

Shadow Training Attack (2)

- The output probability vectors from the shadow models are next used as inputs to train attack models (as binary classifiers) for each class
  - E.g., the probability vectors for all input images of Alice from all shadow **training sets** are labeled with 1 (meaning 'in' the training set)
  - The probability vectors for all input images of Alice from all shadow **test sets** are labeled with 0 (meaning 'out' or not in the training set)
  - An attack model is trained on these inputs to perform binary classification (in or out)
  - A separate attack model is trained for each celebrity person in the shadow training sets

Shadow Training Attack (3)



https://arxiv.org/abs/1610.05820

Shadow Training Attack (4)

- The attack models for each class are afterward used to predict whether individual input instances were members of the private training set of the target model.
- The assumption in this attack is that the output probability vectors for samples that are members of the training sets are different from samples out of the training sets.
- Experiments showed that increasing the number of shadow models improves the accuracy of membership inference, but it also increases the computational recourses.

This Lecture

- Membership Inference Attacks
- Model Inversion Attacks
- Model Stealing Attacks
- Privacy Protection Methods

Model Inversion Attack (1)

- *Model inversion attack* creates prototype examples for the classes in the dataset
  - The authors demonstrated an attack against a DNN model for face recognition.
  - Given a person's name and white-box access to the model, the attack reverse-engineered the model and produced an averaged image of that person.
    - The obtained averaged image (left image below) makes the person recognizable.
  - This attack is limited to classification models where the classes pertain to one type of object (such as the faces of the same person).

Recovered Image

Training Image Data

https://dl.acm.org/doi/10.1145/2810103.2813677

Model Inversion Attack (2)

- The model inversion attack applies gradient descent to start from a given label and follows the gradient in a trained network to recreate an image for that label
  - In the algorithm, *c* denotes the cost function, whereas the PROCESS function applies image denoising and sharpening operations to improve the reconstructed image

Algorithm 1 Inversion attack for facial recognition models.

function MI-FACE($label, \alpha, \beta, \gamma, \lambda$)
$\quad c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \boxed{\tilde{f}_{label}(\mathbf{x})} + \text{AUXTERM}(\mathbf{x})$
$\quad \mathbf{x}_0 \leftarrow \mathbf{0}$
$\quad$ **for** $i \leftarrow 1 \ldots \alpha$ **do**
$\quad\quad \mathbf{x}_i \leftarrow \text{PROCESS}(\mathbf{x}_{i-1} - \lambda \cdot \nabla c(\mathbf{x}_{i-1}))$
$\quad\quad$ **if** $c(\mathbf{x}_i) \geq \max(c(\mathbf{x}_{i-1}), \ldots, c(\mathbf{x}_{i-\beta}))$ **then**
$\quad\quad\quad$ **break**
$\quad\quad$ **if** $c(\mathbf{x}_i) \leq \gamma$ **then**
$\quad\quad\quad$ **break**
$\quad$ **return** $[\arg\min_{\mathbf{x}_i}(c(\mathbf{x}_i)), \min_{\mathbf{x}_i}(c(\mathbf{x}_i))]$

AuxTerm: case-specific function, any available auxiliary information to inform the cost function.

f_{label}: facial recognition model

GAN-based Model Inversion Attack

Inferring sensitive features (e.g., face) in the training data:

Rather than reconstructing private training data from scratch, we leverage partial public information, to **learn a distributional prior** via generative adversarial networks (GANs) and use it to guide the inversion process.

Blurred or corrupted version of the private image

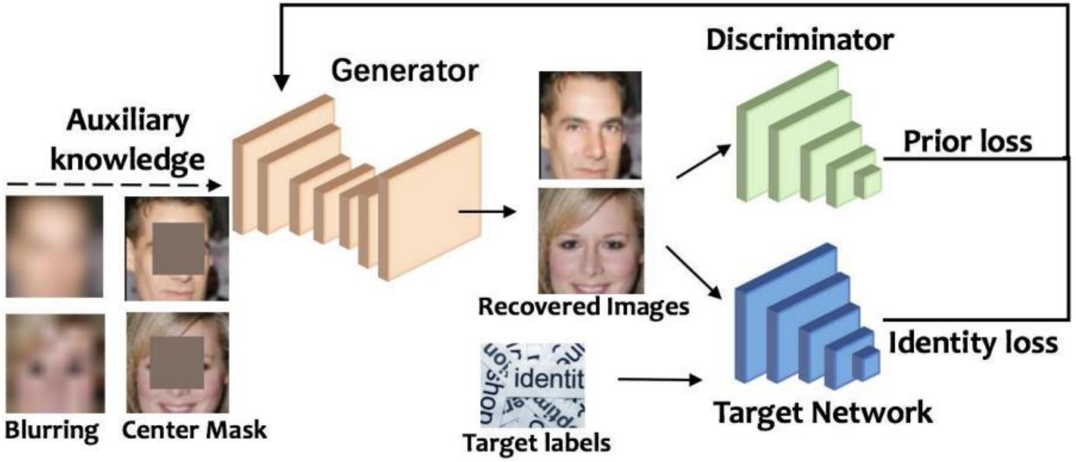


Figure 1: Overview of the proposed GMI attack method.

The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. CVPR'20 Yuheng Zhang, Ruoxi Jia , Hengzhi Pei1, Wenxiao Wang , Bo Li , and Dawn Song

GAN-based Model Inversion Attack

Stage 1: Train the generator and the discriminators on public datasets in order to encourage the generator to generate realistic-looking images.

$$\min_G \max_D L_{\text{wgan}}(G, D) = E_x[D(x)] - E_z[D(G(z))]$$

$$\max_G L_{\text{div}}(G) = E_{\mathbf{z_1},\mathbf{z_2}}\left[\frac{\|F(G(\mathbf{z_1})) - F(G(\mathbf{z_2}))\|}{\|\mathbf{z_1} - \mathbf{z_2}\|}\right]$$

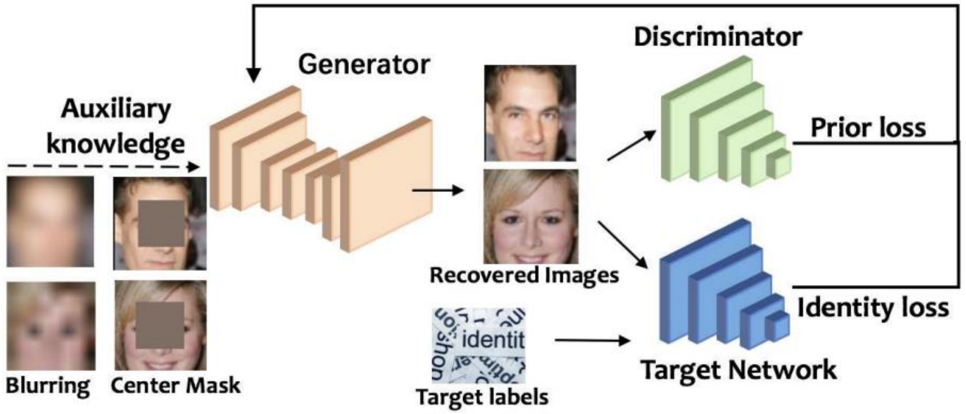


Figure 1: Overview of the proposed GMI attack method.

Stage 2: Find the latent vector that generates an image achieving the maximum likelihood under the target network while remaining realistic.

$$\hat{z} = \arg\min_z L_{\text{prior}}(z) + \lambda_i L_{\text{id}}(z)$$

$$L_{\text{prior}}(z) = -D(G(z)) \quad L_{\text{id}}(z) = -\log[C(G(z))]$$

The Secret Revealer: Generative Model-Inversion Attacks Against Deep Neural Networks. CVPR’20

This Lecture

- Membership Inference Attacks
- Model Inversion Attacks
- Model Stealing Attacks
- Privacy Protection Methods

Model Stealing Attack

- Adversarial goal: reconstruct an approximated model $f'(x)$ of the target model $f(x)$.
- The approximated function $f'(x)$ will act as a substitute model and produce similar outputs as the target model.
  - The adversary has black-box query access to the model
  - The goal is to "steal" the model and use the substitute model for launching other attacks, such as synthesis of adversarial examples, or membership inference attacks
- Besides creating a substitute model, several works focused on recovering the hyperparameters of the model, such as the number of layers, optimization algorithm, activation types used, etc.

This Lecture

- Membership Inference Attacks
- Model Inversion Attacks
- Model Stealing Attacks
- Privacy Protection Methods

Causes of Privacy Leaks in Machine Learning

- Overfitting
  - It leads to poor generalization and memorization of the training data
  - Although adversarial training is often applied for increasing to model robustness, it reduces the accuracy of model on clean data, due to the trade-off between the model accuracy and robustness
    - The reduced accuracy can lead to increased sensitivity to data leakage
- Datasets that are more diverse and with a larger number of categories are more susceptible to attacks
  - Binary classifiers are safer than multiclass models
  - Input samples that are out-of-distribution (i.e., are considered outliers with respect to the distribution of the training data) are more susceptible to privacy leakage
- Model complexity
  - Complex models with a large number of parameters memorize more sensitive information about the training data

Defenses against Privacy Attacks

- Anonymization techniques
- Encryption techniques
- Differential privacy
- Distributed learning
- ML-specific techniques

Data Privacy

- 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 an attempt to control what can be learned from the released information
- The Fundamental Law of Information Recovery 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

Anonymization Techniques

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

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

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

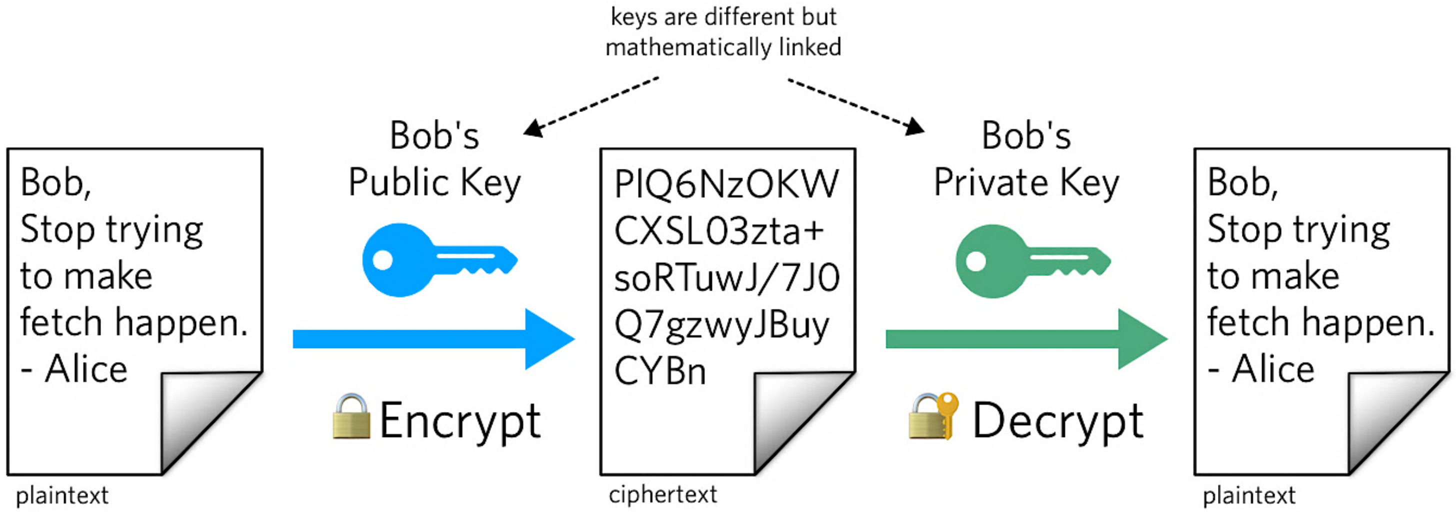
- Drawback: The remaining information in the data can be used for identifying the individual data instances
  - For example, based on health records (including diagnoses and prescriptions) with removed personal information released by an insurance group in 1997, a researcher extracted the information 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

K-anonymity

- *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
  - Therefore, a linkage attack will result in a group of k records that can belong to a person of interest
- Limitation: this approach is mostly applicable to large datasets with low-dimensional input features
  - The more input features for each record, the higher the possibility of unique records

Encryption Techniques

- Encryption is a cryptography approach, which converts the original representation of information (plaintext) into an alternative form (Ciphertext)
  - The sender of encrypted information shares the decoding technique only with the intended recipients of the information



https://www.twilio.com/en-us/blog/what-is-public-key-cryptography

Encryption Techniques

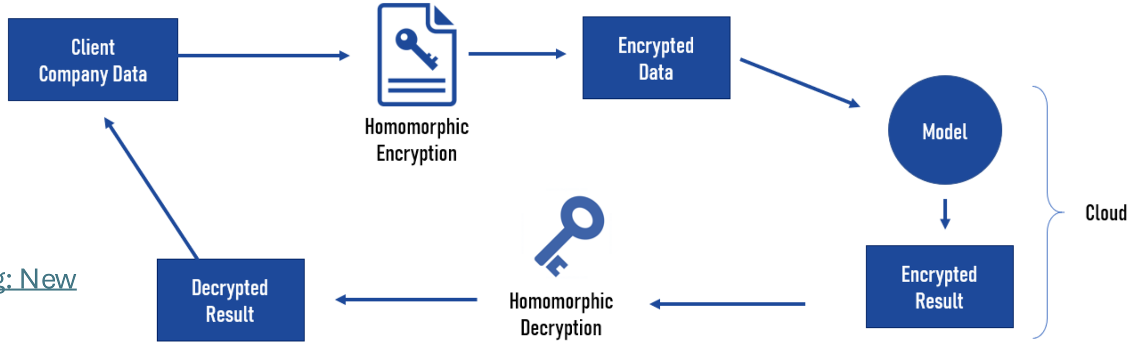
- Encrypting the training data has been applied in ML
  - Common techniques for data encryption include:
    - Homomorphic encryption (HE)
    - Secure multi-party computation (SMPC)
- Encrypting ML models is a less common approach
  - Homomorphic encryption has been applied to the model gradients in a collaborative deep learning setting to protect the model privacy

Homomorphic Encryption

- 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

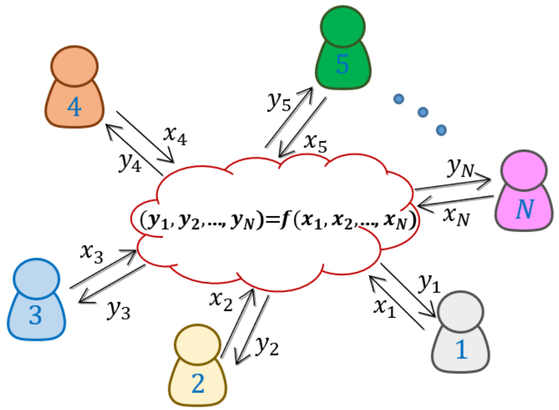
- In ML, training data can be encrypted and sent to a server for model training.
  - Even if the server is untrusted or compromised, the confidentiality of the data will remain preserved.
  - One main limitation of HE is the slowing down of the training process.
- HE has been applied to traditional ML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



Homomorphic Encryption & Machine Learning: New Business Models

Secure Multi-Party Computation

- 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 of 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.



https://arxiv.org/abs/1909.11701

Secure Multi-Party Computation

- In ML, SMPC can be used to compute updates of the model parameters by multiple parties without sharing their private data
  - For example, 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
  - Besides 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 DNNs is also challenging, due to increased computational costs

SMPC and 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 an MLaaS (Machine Learning as a Service) , but does not trust the service provider: the owner sends encrypted data, the provider processes encrypted data and sends back encrypted results, 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

Differential Privacy

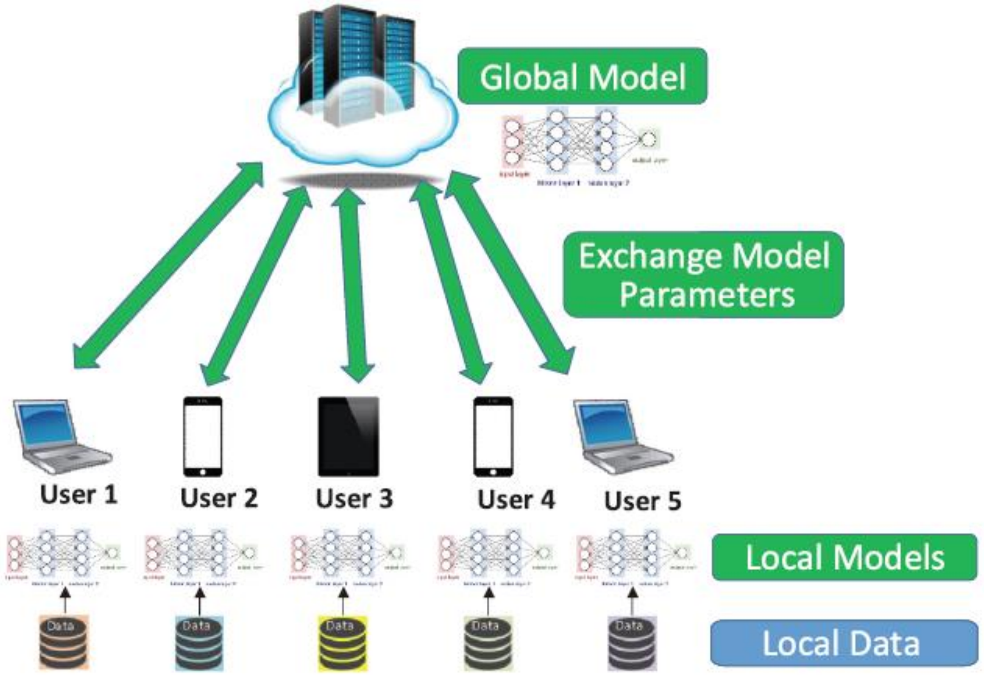
- ***Differential privacy*** is based on employing obfuscation mechanisms for privacy protection
  - A randomization mechanism $\mathcal{M}(D)$ applies noise ξ to the outputs of a function $f(D)$ to protect the privacy of individual data instances, i.e., $\mathcal{M}(D) = f(D) + \xi$
  - Commonly used randomization mechanisms include Laplacian, Gaussian, and Exponential mechanism
- DP is often implemented in practical applications, and examples include:
  - 2014: Google's RAPPOR, for statistics on unwanted software hijacking users' settings
  - 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

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 the student model.
  - *Training data*
    - Obfuscation of training data in ML has been also investigated in several works.

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



https://arxiv.org/abs/2011.11819

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 does not leave the remote devices and 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 outputs of intermediate layers, rather than model parameters
  - Split learning is more common for IoT devices with limited computational resources

ML-Specific Techniques

- 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

References

- When Machine Learning Meets Privacy: A Survey and Outlook (https://dl.acm.org/doi/abs/10.1145/3436755)
- Beyond Boundaries: A Comprehensive Survey of Transferable Attacks on AI Systems (https://arxiv.org/abs/2311.11796)
- A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection (https://ieeexplore.ieee.org/abstract/document/9599369)
- A Survey on Differential Privacy for Unstructured Data Content (https://dl.acm.org/doi/full/10.1145/3490237)
- https://differentialprivacy.org