Membership Inference Attacks against Machine Learning Models

Reza Shokri, Marco Stronati, Congzheng Song, Vitaly Shmatikov
Membership Inference Attack

Was this specific data record part of the training set?

Model

Prediction

Input data

Classification

Training

Data

airplane
automobile...
ship
truck
Membership Inference Attack on Summary Statistics

- Summary statistics (e.g., average) on each attribute
- Underlying distribution of data is known

[Homer et al. (2008)], [Dwork et al. (2015)], [Backes et al. (2016)]

on Machine Learning Models

Black-box setting:
- No knowledge about the models’ parameters
- No access to internal computations of the model
- No knowledge about the underlying distribution of data
Exploit Model’s Predictions

Main **insight**: ML models overfit to their training data
Exploit Model’s Predictions

Main **insight**: ML models overfit to their training data
Exploit Model’s Predictions

Main insight: ML models overfit to their training data
Exploit Model’s Predictions

Model

Prediction API

Input from the training set

Classification

Input NOT from the training set

Classification

Training API

DATA

Recognize the difference
ML against ML

Train a ML model to recognize the difference
Train Attack Model using Shadow Models

Train the attack model to predict if an input was a member of the training set (in) or a non-member (out)
Obtaining Data for Training
Shadow Models

- **Real**: similar to training data of the target model (i.e., drawn from same distribution)

- **Synthetic**: use a sampling algorithm to obtain data classified with high confidence by the target model
Constructing the Attack Model
Constructing the Attack Model

Using the Attack Model
Purchase Dataset — Classify Customers (100 classes)

- Shadows trained on real data: Overall accuracy: 0.93
- Shadows trained on synthetic data: Overall accuracy: 0.89
Privacy Learning

data universe

training set

Model

train
Does the model leak information about data in the training set?
Privacy

Does the model leak information about data in the training set?

Learning

Does the model generalize to data outside the training set?

data universe

Model
Overfitting is the common enemy!

Privacy

Does the model leak information about data in the training set?

Learning

Does the model generalize to data outside the training set?

data universe
Not in a Direct Conflict!

Privacy-preserving machine learning
DEEP LEARNING WITH DIFFERENTIAL PRIVACY

Martin Abadi, Andy Chu, Ian Goodfellow*, Brendan McMahan, Ilya Mironov, Kunal Talwar, Li Zhang

Google

* OpenAI
Differential Privacy

$(\varepsilon, \delta)$-Differential Privacy: The distribution of the output $M(D)$ on database $D$ is (nearly) the same as $M(D')$:

$$\forall S: \quad \Pr[M(D) \in S] \leq \exp(\varepsilon) \cdot \Pr[M(D') \in S] + \delta.$$ 

quantifies information leakage
allows for a small probability of failure
Interpreting Differential Privacy

Training Data $D_{D'}$

SGD

Model
Differential Privacy: Gaussian Mechanism

If $\ell_2$-sensitivity of $f: D \rightarrow \mathbb{R}^n$:

$$\max_{D, D'} \|f(D) - f(D')\|_2 < 1,$$

then the Gaussian mechanism

$$f(D) + N^n(0, \sigma^2)$$

offers $(\varepsilon, \delta)$-differential privacy, where $\delta \approx \exp(-{(\varepsilon \sigma)^2}/2)$.

Dwork, Kenthapadi, McSherry, Mironov, Naor, “Our Data, Ourselves”, Eurocrypt 2006
Basic Composition Theorem

If $f$ is $(\varepsilon_1, \delta_1)$-DP and $g$ is $(\varepsilon_2, \delta_2)$-DP, then

$f(D), g(D)$ is $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-DP
Simple Recipe for Composite Functions

To compute composite $f$ with differential privacy

1. Bound sensitivity of $f$'s components
2. Apply the Gaussian mechanism to each component
3. Compute total privacy via the composition theorem
Deep Learning with Differential Privacy
Differentially Private Deep Learning

1. Loss function  softmax loss
2. Training / Test data  MNIST and CIFAR-10
3. Topology  PCA + neural network
4. Training algorithm  Differentially private SGD
5. Hyperparameters  tune experimentally
Stochastic Gradient Descent

\[ \theta_2 := \theta_1 - \eta \nabla L(\theta_1) \]

\[ \theta_3 := \theta_2 - \eta \nabla L(\theta_2) \]
Stochastic Gradient Descent with Differential Privacy

1. Compute $\nabla L(\theta_1)$ on random sample
2. $\theta_2 := \theta_1 - \eta \nabla L(\theta_1)$
3. Compute $\nabla L(\theta_2)$ on random sample
4. $\theta_3 := \theta_2 - \eta \nabla L(\theta_2)$

Clip
Add noise

Clip
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 $\eta_t$, 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 $g_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$

Clip gradient

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

Add noise

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

Descent

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

Output $\theta_T$ and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method.
Naïve Privacy Analysis

1. Choose $\sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon}$

2. Each step is $(\varepsilon, \delta)$-DP

3. Number of steps $T$

4. Composition: $(T\varepsilon, T\delta)$-DP

$\varepsilon = 4$

$(1.2, 10^{-5})$-DP

10,000

$(12,000, .1)$-DP
Advanced Composition Theorems
Composition theorem

+ε for Blue

+.2ε for Blue

+ε for Red
Strong Composition Theorem

1. Choose $\sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon} = 4$
2. Each step is $(\varepsilon, \delta)$-DP $(1.2, 10^{-5})$-DP
3. Number of steps $T$ 10,000
4. Strong comp: $(\varepsilon \sqrt{T \log 1/\delta}, T\delta)$-DP $(360, .1)$-DP

Dwork, Rothblum, Vadhan, “Boosting and Differential Privacy”, FOCS 2010
Amplification by Sampling

1. Choose \( \sigma = \frac{\sqrt{2 \log 1/\delta}}{\epsilon} \)
   
2. Each batch is \( q \) fraction of data
   
3. Each step is \((2q\epsilon, q\delta)\)-DP
   
4. Number of steps \( T \)

5. Strong comp: \((2q\epsilon \sqrt{T \log 1/\delta}, qT\delta)\)-DP

\[ (10, .001)\text{-DP} \]

Moments Accountant

1. Choose \( \sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon} \) = 4
2. Each batch is a fraction of data \( q \) 1%
3. Keeping track of privacy loss’s moments
4. Number of steps \( T \) 10,000
5. Moments: \( 2q\varepsilon\sqrt{T}, \delta \)-DP \( (1.25, 10^{-5}) \)-DP
Results
Our Datasets: “Fruit Flies of Machine Learning”

MNIST dataset:
70,000 images
28×28 pixels each

CIFAR-10 dataset:
60,000 color images
32×32 pixels each
# Summary of Results

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>no privacy</td>
<td></td>
</tr>
<tr>
<td><strong>MNIST</strong></td>
<td>98.3%</td>
</tr>
<tr>
<td><strong>CIFAR-10</strong></td>
<td>80%</td>
</tr>
</tbody>
</table>
### Summary of Results

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>[SS15]</th>
<th>[WKC+16]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>no privacy</strong></td>
<td></td>
<td>reports $\varepsilon$ per parameter</td>
<td>$\varepsilon = 2$</td>
</tr>
<tr>
<td><strong>MNIST</strong></td>
<td>98.3%</td>
<td>98%</td>
<td>80%</td>
</tr>
<tr>
<td><strong>CIFAR-10</strong></td>
<td>80%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Summary of Results

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>[SS15]</th>
<th>[WKC+16]</th>
<th>this work</th>
</tr>
</thead>
<tbody>
<tr>
<td>no privacy</td>
<td></td>
<td>reports $\varepsilon$ per parameter</td>
<td>$\varepsilon = 2$</td>
<td>$\varepsilon = 8$</td>
</tr>
<tr>
<td>$\delta = 10^{-5}$</td>
<td>$\delta = 10^{-5}$</td>
<td>$\delta = 10^{-5}$</td>
<td>$\delta = 10^{-5}$</td>
<td></td>
</tr>
<tr>
<td>MNIST</td>
<td>98.3%</td>
<td>98%</td>
<td>80%</td>
<td>97%</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>80%</td>
<td></td>
<td>73%</td>
<td>67%</td>
</tr>
</tbody>
</table>
Contributions

- Differentially private deep learning applied to publicly available datasets and implemented in TensorFlow
  - https://github.com/tensorflow/models
- Innovations
  - Bounding sensitivity of updates
  - Moments accountant to keep tracking of privacy loss
- Lessons
  - Recommendations for selection of hyperparameters
- Full version: https://arxiv.org/abs/1607.00133
In their work, the threat model assumes:

- Adversary can make a potentially unbounded number of queries
- Adversary has access to model internals
Intuitive privacy analysis:
• If most teachers agree on the label, it does not depend on specific partitions, so the privacy cost is small.
• If two classes have close vote counts, the disagreement may reveal private information.
Noisy aggregation

\[ n_j(\vec{x}) = |\{ i : i \in [n], f_i(\vec{x}) = j \}| \quad \text{Count votes} \]

\[ \text{Lap} \left( \frac{1}{\gamma} \right) \quad \text{Add Laplacian noise} \]

\[ f(x) = \arg \max_j \left\{ n_j(\vec{x}) + \text{Lap} \left( \frac{1}{\gamma} \right) \right\} \quad \text{Take maximum} \]
The aggregated teacher violates the threat model:

- **Each prediction increases total privacy loss.**
  
  privacy budgets create a tension between the accuracy and number of predictions

- **Inspection of internals may reveal private data.**
  
  Privacy guarantees should hold in the face of white-box adversaries
Private Aggregation of Teacher Ensembles (PATE)

Privacy Analysis:
• Privacy loss is fixed after the student model is done training.
• Even if white-box adversary can inspect the model parameters, the information can be revealed from student model is unlabeled public data and labels from aggregate teacher which is protected with privacy.
**GANs**

IJ Goodfellow et al. (2014) *Generative Adversarial Networks*

2 computing models

**Generator:**
- **Input:** noise sampled from random distribution
- **Output:** synthetic input close to the expected training distribution

**Discriminator:**
- **Input:** output from generator OR example from real training distribution
- **Output:** in distribution OR fake

\[
P(\text{real}) = \ldots \quad P(\text{fake}) = \ldots
\]
Improved Training of GANs
T Salimans et al. (2016) *Improved Techniques for Training GANs*

**Generator:**
**Input:** noise sampled from random distribution

**Output:** synthetic input close to the expected training distribution

**Discriminator:**
**Input:** output from generator OR example from real training distribution

**Output:** in distribution *(which class)* OR fake

Sample

\[
P(\text{real0}) = ... \\
P(\text{real1}) = ... \\
... \\
P(\text{realN}) = ... \\
P(\text{fake}) = ... \\]
Private Aggregation of Teacher Ensembles using GANs (PATE-G)

Sensitivity Data

Data 1 → Teacher 1
Data 2 → Teacher 2
Data 3 → Teacher 3
... → ...
Data n → Teacher n

Aggregate Teacher

Not available to the adversary
Available to the adversary

Generator
Discriminator
Public Data

Queries
Aggregated Teacher Accuracy Before the Student Model is Trained
Evaluation

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$\varepsilon$</th>
<th>$\delta$</th>
<th>Queries</th>
<th>Non-Private Baseline</th>
<th>Student Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>2.04</td>
<td>$10^{-5}$</td>
<td>100</td>
<td>99.18%</td>
<td>98.00%</td>
</tr>
<tr>
<td>MNIST</td>
<td>8.03</td>
<td>$10^{-5}$</td>
<td>1000</td>
<td>99.18%</td>
<td>98.10%</td>
</tr>
<tr>
<td>SVHN</td>
<td>5.04</td>
<td>$10^{-6}$</td>
<td>500</td>
<td>92.80%</td>
<td>82.72%</td>
</tr>
<tr>
<td>SVHN</td>
<td>8.19</td>
<td>$10^{-6}$</td>
<td>1000</td>
<td>92.80%</td>
<td>90.66%</td>
</tr>
</tbody>
</table>

M Abadi et al. (2016) *Deep Learning with Differential Privacy*

$(0.5, 10^{-5})$ 90%
$(2, 10^{-5})$ 95%
$(8, 10^{-5})$ 97%

increase # teachers will increase privacy guarantee, but decrease model accuracy
# teachers is constrained by task’s complexity and the available data