DeepObfuscator
DeepObfuscator

Obfuscating Intermediate Representations with Privacy-Preserving Adversarial Learning on Smartphones

- Ang Li – Duke University
- Jiayi Guo – Tsinghua University
- Huanrui Yang – Duke University
- Flora D. Salim – RMIT University
- Yiran Chen – Duke University
Obfuscator?

**obfuscator**  noun

-ob-fus-ca-tor  |  \\æbˈfəskətər, æbˈ-; ˈæb(ˌ)foʊ-\

plural -s

**Definition of obfuscator**

: one that obfuscates
obfuscate  verb

Save Word

obfus·cate  \\ˈəb-fə-ˌskät\; əb-ˌfə-, skət, əb-

obfuscated; obfuscating

Definition of obfuscate

transitive verb

1  a  : to throw into shadow : DARKEN

b  : to make obscure
    // obfuscate the issue
    // officials who ... continue to obscure and obfuscate what happened
    — Mary Carroll

2  : CONFUSE
    // obfuscate the reader
DeepObfuscator

An adversarial training framework to learn an obfuscator that can hide sensitive information that can be exploited for reconstructing raw images and inferring private attributes and maintain useful features for image classifications.
Why?

- Hard to run deep learning models on mobile devices
  - Computational resource limitations
- Alternative solution
  - The cloud
Why?

- Large-sized deep-learning-based applications are deployed on cloud servers (MLaaS)
  - Amazon Rekognition
  - Microsoft Cognitive Services
- Require users to send data (images) to cloud provider
- Leads to privacy concern due to vulnerable data
  - Age
  - Gender
Propositions

- Send features extracted from raw data to the cloud
- Can still be exploited by attackers to recover raw images and to infer private attributes like age and gender
DeepObfuscator: Overview

The algorithm simulates the game between an attacker who makes efforts to reconstruct raw image and infer private attributes from the extracted features who aims to protect user privacy.

Can deploy the trained obfuscator on the smartphone.

Features can be locally extracted and then sent to the cloud.
Reconstruction Attacks

An example where the reconstruction attack and private attribute leakage occur in a MLaaS for facial attribute recognition.
Reconstructed Images

a) Raw images

b) Image reconstructed, defending against only private attribute leakage

c) Image reconstructed, defending against only reconstruction attack
Defense against Reconstruction Attacks: Related Works

Anonymization techniques

- Designed for protecting sensitive attributes in a static database
- Not suitable for obfuscating intermediate representations of data while retaining the utility for deep neural network inference.
Design
Obfuscator

The obfuscator is a typical encoder which consists of an input layer, multiple convolutional layers, max-pooling layers and batch-normalization layers.

Trained to hide privacy-related information while retaining useful information for intended classification tasks
Classifier

The performance of the classifier $C$ is measured using the cross-entropy loss function, which is expressed as:

$$\mathcal{L}(C) = -\sum_{j=1}^{N} \sum_{i=1}^{M} y_{ij} \log(y'_{ij}) + (1 - y_{ij}) \log(1 - y'_{ij}), \quad (1)$$

where $(y_{1j}, \ldots, y_{Mj})$ denote the ground truth labels for the $j$th data sample, and $(y'_{1j}, \ldots, y'_{Mj})$ are the corresponding predictions. Therefore, the obfuscator and the classifier can be optimized by minimizing the above loss function as:

$$\theta_o, \theta_c = \arg \min_{\theta_o, \theta_c} \mathcal{L}(C), \quad (2)$$

where $\theta_o$ and $\theta_c$ are the parameters of the obfuscator and classifier, respectively.

DeepObfuscator adopts VGG16
Adversary Classifier

Similar to the classifier, the performance of the adversary classifier $AC$ is also measured using the cross-entropy loss function as:

$$
\mathcal{L}(AC) = - \sum_{j=1}^{N} \sum_{i=1}^{K} z_{ij} \log(z'_{ij}) + (1 - z_{ij}) \log(1 - z'_{ij}),
$$

(3)

where $(z_{1j}, \ldots, z_{Mj})$ denote the ground truth labels for the $j$th eavesdropped feature, and $(z'_{1j}, \ldots, z'_{Mj})$ stand for the corresponding predictions. When we simulate an attacker who tries to enhance the accuracy of the adversary classifier as high as possible, the adversary classifier needs to be optimized by minimizing the above loss function as:

$$
\theta_{ac} = \arg \min_{\theta_{ac}} \mathcal{L}(AC),
$$

(4)

where $\theta_{ac}$ is the parameter set of the adversary classifier. On the contrary, when defending against private attribute leakage, we train the obfuscator in our proposed adversarial training procedure that aims to degrade the performance of the adversary classifier while improving the accuracy of the classifier. Consequently, the obfuscator can be trained using Eq. 5 when simulating a defender:

$$
\theta_{o} = \arg \min_{\theta_{o}} \mathcal{L}(C) - \lambda_{1} \mathcal{L}(AC),
$$

(5)

where $\lambda_{1}$ is a tradeoff parameter.
Adversary Reconstructor

The adversary reconstructor (AR), which is trained to recover the raw image from the eavesdropped features, also plays an attacker role. The attacker can apply any neural network architecture in the adversary reconstructor design.

When playing as an attacker, the adversary reconstructor is trained to optimize the quality of the reconstructed image $I_r$ as close as the original image $I_0$. In DeepObfuscator, we leverage MS-SSIM [20, 38] to evaluate the performance of the adversary reconstructor, which is expressed as:

$$\mathcal{L}(AR_1) = 1 - \text{MS-SSIM}(I_0, I_r).$$  \hfill (6)

The MS-SSIM value ranges between 0 and 1. The higher the MS-SSIM value is, the more perceptual similarity can be found between the two compared images, indicating a better quality of the reconstructed images. Consequently, an attacker can optimize the adversary reconstructor as:

$$\theta_{ar} = \arg\min_{\theta_{ar}} \mathcal{L}(AR_1).$$  \hfill (7)

where $\theta_{ar}$ is the parameter set of the adversary reconstructor. On the contrary, a defender expects to degrade the quality of the reconstructed image as much as possible. To this end, we generate one additional Gaussian noise image $I_{noise}$. The adversary reconstructor is trained to make each reconstructed image similar to $I_{noise}$ but different from $I_0$, and the performance of the classifier should be maintained. When playing as a defender, the obfuscator can be trained as:

$$\mathcal{L}(AR_2) = 1 - \text{MS-SSIM}(I_{noise}, I_r)$$

$$\theta_0 = \arg\min_{\theta_0} \mathcal{L}(C) + \lambda_2 (\mathcal{L}(AR_2) - \mathcal{L}(AR_1)),$$

where $\lambda_2$ is a tradeoff parameter.
Algorithm 1 Adversarial Training Algorithm

Input: Dataset $D$
Output: $\theta_o, \theta_c, \theta_{ar}, \theta_{ac}$

1: Input: Dataset $D$
2: for every epoch do
3:   for every four batches do
4:     if batch idx mod 4 == 0 then
5:       Defend against AR:
6:       $\mathcal{L}(C) + \mathcal{L}(AR_2) - \mathcal{L}(AR_1)$ → update $O(\theta_o)$
7:     else if batch idx mod 4 == 1 then
8:       Defend against AC:
9:       $\mathcal{L}(C) - \mathcal{L}(AC)$ → update $O(\theta_o)$
10:    else if batch idx mod 4 == 2 then
11:       reconstruction attack:
12:       $\mathcal{L}(AR_1)$ → update $AR(\theta_{ar})$
13:    else
14:       Infer private attributes:
15:       $\mathcal{L}(AC)$ → update $AC(\theta_{ac})$
16:    end if
17:  end for
18: end for

First: jointly train obfuscator and classifier without privacy concern to obtain the optimal performance on the intended classification tasks. Also pre-train adversary classifier and adversary reconstructor for initialization.
Technical Objective

Apply adversarial training for maximizing the reconstruction error of the adversary reconstructor and the classification error of the adversary classifier, but minimize the classification error of the intended classifier.
Experiment Setup

- Implemented with PyTorch
- Trained on server with 4x NVIDIA TITAN RTX GPUs
- Adopt AdamOptimizer with an adaptive learning rate
Experiment Setup: Datasets

- CelebA
  - 200k face images
  - 40 binary facial attributes
  - 160k images for training
  - 40k images for testing
- LFW
  - 13k face images
  - 16 binary facial attributes
  - 10k images for training
  - 3k images for testing
Evaluation

Evaluate DeepObfuscator’s performance on two real-world datasets, with a focus on the utility-privacy tradeoff.

Also compare DeepObfuscator with existing solutions proposed in the literature and visualize the results.
Comparison Baselines

- **Noisy method** perturbs the raw data $x$ by adding Gaussian noise $N(0, \sigma^2)$, where $\sigma$ is set to 40.

- **DP** approach injects Laplace noise the raw data $x$ with diverse privacy budgets ($0.1$, $0.2$, $0.5$, $0.9$).

- **Encoder** learns the latent representation of the raw data $x$ using a DNN-based encoder.

- **Hybrid method** further perturbs the above encoded features by performing principle components analysis (PCA) and adding Laplace noise with varying privacy budgets.
Effectiveness of Defending Against Reconstruction Attack

- MS-SSIM is used to evaluate quality of reconstructed images
- Smaller value of MS-SSIM implies less similarity

Table 3: MS-SSIM for different attack reconstructors.

<table>
<thead>
<tr>
<th>Training Reconstructor</th>
<th>Attack Reconstructor</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AR in DeepObfuscator</td>
<td>AR in DeepObfuscator</td>
<td>0.3175</td>
<td>0.3123</td>
<td>0.3095</td>
</tr>
</tbody>
</table>
Effectiveness of Defending Against Reconstruction Attack

- Peak Signal to Noise Ratio (PSNR) is a widely used metric of image quality is used to evaluated the quality of reconstructed images

<table>
<thead>
<tr>
<th>Metric</th>
<th>DeepObfuscator</th>
<th>Encoder</th>
<th>Noisy</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-SSIM</td>
<td>0.3175</td>
<td>0.9458</td>
<td>0.7263</td>
</tr>
<tr>
<td>PSNR</td>
<td>6.32</td>
<td>27.81</td>
<td>16.97</td>
</tr>
</tbody>
</table>

Table 4: Average PSNR and MS-SSIM for DeepObfuscator and two baseline models.
Comparison
Figure 5: An example question of the human perceptual study. (a) is the reconstructed image, and (b)-(e) are the four options.
Effectiveness of Defending Against Private Attribute Leakage
Effectiveness of Defending Against Private Attribute Leakage

Figure 7: The impact of the utility-privacy budget $\lambda_2$ on CelebA ($\lambda_2 = 1$)

Figure 8: The impact of the utility-privacy budget $\lambda_2$ on LFW ($\lambda_2 = 1$).
Table 5: Evaluate the transferability of DeepObfuscator with cross-dataset experiments.

<table>
<thead>
<tr>
<th>Test Dataset</th>
<th>Training Dataset</th>
<th>‘gender’</th>
<th>‘black hair’</th>
</tr>
</thead>
<tbody>
<tr>
<td>LFW</td>
<td>CelebA</td>
<td>53.74%</td>
<td>94.79%</td>
</tr>
<tr>
<td>LFW</td>
<td>LFW</td>
<td>57.23%</td>
<td>94.31%</td>
</tr>
<tr>
<td>CelebA</td>
<td>LFW</td>
<td>59.87%</td>
<td>93.57%</td>
</tr>
<tr>
<td>CelebA</td>
<td>CelebA</td>
<td>58.82%</td>
<td>94.88%</td>
</tr>
</tbody>
</table>
Results

Experimented on CelebA and LFW datasets. Results: quality is dramatically decreased from 0.9458 to 0.3175 in terms of multi-scale structural similarity, so person in image is hard to identify. Classification accuracy of private attributes is reduced to a random guessing level 97.36% to 58.85%. Cloud services only reduce by 2%. 
Table 6: Performance of running the learned obfuscator on Google Pixel 2 and Pixel 3.

<table>
<thead>
<tr>
<th>Smartphone</th>
<th>Latency (ms)</th>
<th>Storage (MB)</th>
<th>Energy (mJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Pixel 2</td>
<td>105</td>
<td>5.6</td>
<td>2.8</td>
</tr>
<tr>
<td>Google Pixel 3</td>
<td>101</td>
<td>5.6</td>
<td>2.7</td>
</tr>
</tbody>
</table>
Discussion

- Did not perform the model optimization for the obfuscator in terms of efficiency.
- DeepObfuscator can be extended to other data modalities such as sensor data (accelerometer, gyroscope)
- Even though DeepObfuscator attains a notably better privacy-utility tradeoff than existing works, it requires prior knowledge of primary learning tasks before training.
  - This requirement lay limit applicability and generalization in practice