Demo: Image Disguising for Scalable GPU-accelerated Confidential Deep Learning

Yuechun Gu
Marquette University
Trustworthy and Intelligent Computing Lab
Milwaukee, Wisconsin, USA
ethan.gu@marquette.edu

Sagar Sharma
Tiktok Inc.
Privacy Innovation Lab
Bellevue, Washington, USA
sagar.sharma@tiktok.com

Keke Chen
Marquette University
Trustworthy and Intelligent Computing Lab
Milwaukee, Wisconsin, USA
keke.chen@marquette.edu

ABSTRACT

Deep learning training involves large training data and expensive model tweaking, for which cloud GPU resources can be a popular option. However, outsourcing data often raises privacy concerns. The challenge is to preserve data and model confidentiality without sacrificing GPU-based scalable training and low-cost client-side preprocessing, which is difficult for conventional cryptographic solutions to achieve. This demonstration shows a new approach, *image disguising*, represented by recent work: DisguisedNets, NeuraCrypt, and InstaHide, which aim to securely transform training images while still enabling the desired scalability and efficiency. We present an interactive system for visually and comparatively exploring these methods. Users can view disguised images, note low client-side processing costs, and observe the maintained efficiency and model quality during server-side GPU-accelerated training. This demo aids researchers and practitioners in swiftly grasping the advantages and limitations of image-disguising methods.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**.

KEYWORDS

Instance encoding; Privacy-preserving machine learning; GPU-acceleration

ACM Reference Format:

Yuechun Gu, Sagar Sharma, and Keke Chen. 2023. Demo: Image Disguising for Scalable GPU-accelerated Confidential Deep Learning. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26–30, 2023, Copenhagen, Denmark.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3576915.3624364

1 INTRODUCTION

Deep neural network (DNN) training is resource-intensive and time-consuming, requiring large training data, careful model architecture selection, and exhaustive model parameter tweaking. As a result, data owners or model developers often utilize cloud GPUs or online model training services to lower their costs. However, outsourcing DNN learning to the cloud raises privacy and security concerns about the sensitive training data and trained models.

Various data and model protection methods were proposed for outsourced learning. (1) Encrypted DNN models on encrypted data face limitations due to extensive training data and costly deep learning processes, making cryptographic approaches impractical. A recent study [6] on small-scale neural networks (e.g., two layers, 128 neurons each) reveals substantial communication, computation, and storage costs. (2) Cloud-client federated learning suggests segmenting datasets and tasks into sensitive and non-sensitive parts. However, compromised clouds can reconstruct client-sensitive data using Generative Adversarial Network (GAN)-based attacks [4]. (3) Differential privacy (DP) is employed in distributed learning or with a trusted central training server [1]. While DP safeguards data and model sharing without exposing individual training examples, it isn't suitable for scenarios requiring data and model confidentiality protection.

The major challenges for scalable, confidential training in the cloud are: (1) Methods should enable the use of GPU resources, as they are essential to scalable deep learning; (2) Low-cost client-side data pre-processing, since many applications collect data from resource-strapped mobile clients; And (3) preferably we can reuse all existing DNN training methods without the need to develop specialized algorithms.

In recent few years, a new approach *image disguising* emerges, which securely transforms training data at a low cost on the client side and enables GPU-accelerated training with existing DNN learning methods on the cloud. (1) We proposed the DisguisedNets approach [7] that combines image block permutation, block-wise random projection or AES encoding, and noise addition to preserve data and model confidentiality. (2) NeuraCrypt [8] transforms training images with a randomly constructed neural network, of which the encoded images, together with the original labels, can still be used to train models. These methods were developed independently, and there is no study to analyze these works comparatively. Thus, we believe we should have an easier way to share these methods and explore potential attacks to leverage more research efforts. We develop an interactive demonstration system to help the audience fully understand the ideas behind these image-disguising methods. (1) Users can select image data, choose from disguising mechanisms, set their parameters, and browse the disguised images, to understand the client-side costs, (2) Users can try different attacks on the disguised images and observe the attack effectiveness under different parameter setting and adversarial knowledge, (3) Users

CCS '23, November 26–30, 2023, Copenhagen, Denmark

ACM ISBN 979-8-4007-0050-7/23/11.
https://doi.org/10.1145/3576915.3624364

can observe the impact of disguised images on the training process, e.g., convergence rate and model quality.

The main contributions of this demonstration include: (1) We implement and integrate the major algorithms in an open-source demonstration system, which can attract more researchers and facilitate follow-up research; (2) The GUI-based system will help users quickly understand the features of each method, compare different methods, and interactively explore potential problems.

2 ARCHITECTURE AND METHODS

Figure 1 depicts the overall architecture framework for confidential deep learning with the image disguising methods. A data owner transforms her private images using one of the disguising techniques before outsourcing them to the cloud. She holds the secret transformation keys for the selected method.

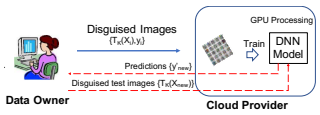

Figure 1: Disguising Images for DNN learning.

Specifically, assume the data owner owns a set of images for training, notated as pairs $\{(X_i, y_i)\}$, where X_i is the image pixel matrix (or a tensor for RGB images), and y_i the corresponding label. The disguising process can be defined as follows. Let the disguising mechanism be a transformation T_K, where K is the secret key that depends on the selected disguising method. By applying image disguising, the training data is transformed to $\{(T(X_i), c_i)\}$ with $c_i = y_i$, while some methods might also hide y_i [5]. The converted images and class labels are used to train a DNN, denoted as a function $\mathbf{D}_{T_K}$, that takes disguised images $T(X)$ and outputs a predicted label $\hat{c}$.

Threat Modeling. (1) The cloud provider is an honest-yet-curious adversary, who will honestly run the training program but be curious about the training data. (2) The adversary can observe training data, training processes, and models, including DNN structure and training parameters, enabling probing the observed items via methods like image reconstruction, re-identification, and model-based attacks. (3) The adversary may have different levels of prior knowledge. Level-1: they may know what the model is used for (e.g., for face recognition) but do not know the disguising parameters. Level-2: additionally, they may know a few pairs of images and their disguised versions. (4) The client infrastructure and communication channels are secure. Since InstaHide is vulnerable to Level-1 attacks [2], we focus on the other two methods in the following.

2.1 DisguisedNets

DisguisedNets [7] incorporates pixel-block partitioning, random block permutation, block-wise transformations of images, and noise additions. This amalgam of multiple transformations provides ample parameter space to protect data from attacks with Level-1 knowledge.

An image $X_{l\times m}$ is first partitioned into t blocks of uniform size $r \times s$. If we label the blocks sequentially as $v =< 1, 2, \ldots, t >$, a pseudorandom permutation of the image, $\pi(X)$, shuffles the blocks and reassembles the corresponding image accordingly. Next, the frameworks establish pixel-block-level protection mechanisms – including AES encryption and random multidimensional transformation (RMT). The encoding parameters serve as a secret key for the whole dataset: the block size, the block permutation, and block-level transformation parameters (i.e., the AES key or random projection matrix) are shared among all images.

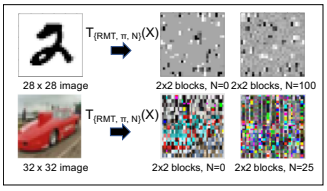

Figure 2: Block-wise RMT+Noise on MNIST and CIFAR10 images.

Block-level Encoding. (1) The AES method uses the AES Electronic Code Book (ECB) mode, which is a fixed mapping function between 16-byte original data to 16-byte encrypted data, to preserve the neighboring information between encoded blocks as possible. (2) The RMT method uses a noise-added transformation: $G(X) = R(X + D)$, where X is an image block, $R_{m\times m}$ is a random invertible matrix shared between images, and D is a random noise matrix independently generated per image. When an image is partitioned into t blocks for random permutation, we prepare a list of random matrices $\{R_i, i = 1..t\}$, one for each image-block X_i, and share this list for each image. Figure 2 shows the effect of RMT disguised image.

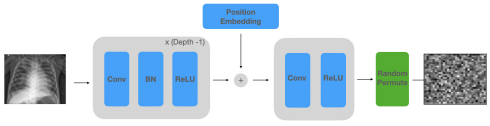

Figure 3: Neuracrypt uses a specific CNN with random weights to transform the original images (Source [8]).

2.2 NeuraCrypt

Yala et al. [8] utilize a random convolutional neural network (CNN) as an encoder, transforming original training images into disguised ones. The encoder is a flexible DNN that maps images to matrices. They provide a specific encoder network (Figure 3) with convolutional layers, batch normalization, and ReLU non-linearities. A random positional embedding conceals spatial structure, preserving positions; patches are permuted before output. They have shown that these transformations don't notably degrade model quality.

2.3 Potential Attacks

Due to the utility-preserving nature of the image disguising methods, we expect that the preserved privacy is weaker than well-known notions, such as indistinguishability in encryption and differential privacy. Related studies have started exploring possible

attacks and shown some specific attacks under certain prior adversarial knowledge. While both methods are resilient to attacks under Level-1 adversarial knowledge, they can be vulnerable to Level-2 attacks [3, 7]. It's thus interesting to develop new disguising methods resilient to Level-2 attacks.

3 DEMONSTRATION

The audience will experience the demonstration through the following components. (1) **Introduction.** The first part of the demonstration uses a poster to outline the problem, the architecture, the image disguising methods, and potential attack methods, and then introduces the demonstration system. (2) **Live System.** The user will be able to use the system to try image disguising methods, train with disguised images, and examine attacks.

3.1 Implemented Functionality

We introduce the major components of the demonstration system: disguising methods, model training, and attacks. We have implemented most core components and designed the interactive user interface. We will integrate, test, and touch up the whole system in the coming months.

Disguising Methods. We have implemented all the DisguisedNets methods [7] and will integrate the open-source code of NeuraCrypt in the demo system.

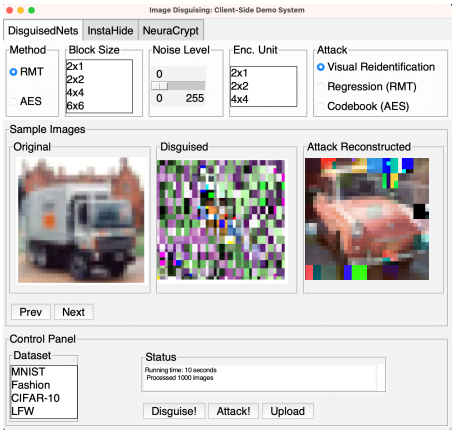

Figure 4: Sample user interface design (client side).

Attacks. We have implemented the main Level-2 attacks. The regression attack on DisguisedNets-RMT, and the codebook attack on DisguisedNets-AES are relatively fast – users can interactively check the result. Carlini et al. [3] have also provided the source code for their Level-2 attack on NeuraCrypt. However, they reported that the attack is expensive (e.g., about 10 GPU server hours). Therefore, we will run the attack offline in advance and visualize sample attack results.

Training. Users can run DNN training with a specific disguised image dataset. Due to the time cost of training, we only choose MNIST for the interactive demonstration. Training on other disguised image datasets will be done offline, with some figures shown in the demo.

3.2 Interactive Demo Workflow

We aim to give users a hands-on experience with different disguising mechanisms and compare them regarding cost, confidentiality guarantee, utility, and trade-offs under attacks. For simplicity, the demo system will handle only a few small well-known datasets: MNIST, CIFAR10, Fashion, and LFW. The demo system contains the client-side and cloud-side subsystems.

We describe the main demo scenarios. (1) In the client-side system, users can select from the image datasets, pick one of the disguising options, tune the transformation parameters, and observe the encoding results and costs. The user can also observe the effectiveness of known attacks for some disguising methods. The expensive attacks will be precomputed and presented interactively. (2) The user can select one of the small DNN architectures for server-side Training and observe the training result, e.g., the convergence speed and model quality. To improve the interactivity, we will pre-train and store most models and results on a GPU server for users to explore.

4 SUMMARY

The demonstration showcases representative image disguising mechanisms proposed during the past few years that aim to enable confidential cloud-GPU-based model training. The audience can interactively explore the demo, which, we believe, will help researchers and practitioners better understand these methods' unique benefits and limitations and inspire the development of more effective and attack-resilient image-disguising methods.

ACKNOWLEDGMENT

This research was partially supported by the National Science Foundation (Award# 2232824).

REFERENCES

[1] Martin Abadi and et al. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*.

[2] Nicholas Carlini, Samuel Deng, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, Shuang Song, Abhradeep Thakurta, and Florian Tramèr. 2021. Is Private Learning Possible with Instance Encoding?. In *IEEE Symposium on Security and Privacy (S&P)*.

[3] Nicholas Carlini, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Florian Tramèr. 2021. NeuraCrypt is not private. *CoRR* abs/2108.07256 (2021). arXiv:2108.07256 https://arxiv.org/abs/2108.07256

[4] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM.

[5] Yangsibo Huang, Zhao Song, Kai Li, and Sanjeev Arora. 2020. InstaHide: Instance-hiding Schemes for Private Distributed Learning. In *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119. PMLR, 4507–4518.

[6] Payman Mohassel and Yupeng Zhang. 2017. SecureML: A System for Scalable Privacy-Preserving Machine Learning. In *2017 IEEE Symposium on Security and Privacy (SP)*. 19–38.

[7] Sagar Sharma, AKM Mubashwir Alam, and Keke Chen. 2021. Image Disguising for Protecting Data and Model Confidentiality in Outsourced Deep Learning. In *IEEE Conference on Cloud Computing*.

[8] Adam Yala and et al. 2021. NeuraCrypt: Hiding Private Health Data via Random Neural Networks for Public Training. *CoRR* abs/2106.02484 (2021). arXiv:2106.02484 https://arxiv.org/abs/2106.02484