Measuring Data Reconstruction Defenses in Collaborative Inference Systems

Mengda Yang[1], Ziang Li[1], Juan Wang[1 *], Hongxin Hu[2], Ao Ren[3], Xiaoyang Xu[1] and Wenzhe Yi[1]

[1]Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University
[2]Department of Computer Science and Engineering, University at Buffalo, SUNY
[3]College of Computer Science, Chongqing University

Abstract

The collaborative inference systems are designed to speed up the prediction processes in edge-cloud scenarios, where the local devices and the cloud system work together to run a complex deep-learning model. However, those edge-cloud collaborative inference systems are vulnerable to emerging Reconstruction Attacks, where malicious cloud service providers are able to recover the edge-side users' private data. To defend against such attacks, several countermeasures have been recently introduced. Unfortunately, little is known about the robustness of those defense countermeasures. In this paper, we take the first step towards measuring the robustness of those state-of-the-art defenses with respect to reconstruction attacks. Specifically, we show the latent privacy features still retain in the obfuscated representations. Motivated by such an observation, we propose a novel technology called Sensitive Feature Distillation (SFD) to restore sensitive information from the protected feature representations. Our experiments show that SFD can break through defense mechanisms in model partitioning scenarios, demonstrating the inadequacy of existing defense mechanisms as a privacy-preserving technique against reconstruction attacks. We hope our findings inspire further work in improving the robustness of defense mechanisms against reconstruction attacks for collaborative inference systems.

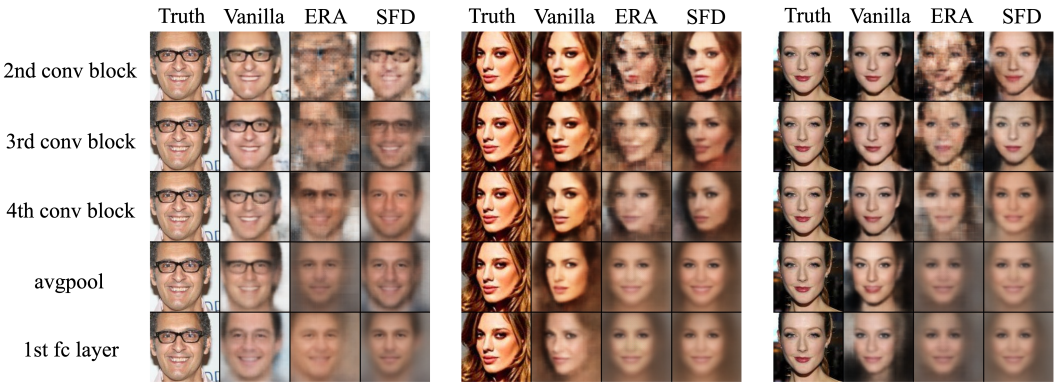
1 Introduction

With the rapid development of deep neural network technology, the deployment of deep learning (DL) applications on edge devices is becoming more widespread. However, on the one hand, resource-constrained edge devices are often difficult to accommodate computationally intensive DL models, and on the other hand, cloud-based DL inference applications require devices to upload users' private data to cloud service providers, which poses a serious privacy threat. Collaborative DL system has recently emerged as a good alternative to address these challenges, where the local devices and the remote server work together to run a complex neural network [Eshratifar et al., 2019; Banitalebi-Dehkordi et al., 2021; Li et al., 2021b]. The DL models can be split such that only the first few layers are executed on the local edge device while the remaining layers are executed on the remote cloud. And the end user does not need to upload her raw data to the cloud, so that privacy concerns can be addressed to some extent.

This paper focuses on *Reconstruction Attacks*, where existing research has shown that adversaries can easily recover sensitive data with the output of the middle layers [He et al., 2019; Singh et al.,

*Corresponding author: jwang@whu.edu.cn

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



a) Adversarial Learning b) Noise Mask c) Dropout Defense

Figure 1: An example showing the superiority of Sensitive Feature Distillation (SFD) compared to Existing Reconstruction Attack (ERA).

2021]. To address this problem, many defense approaches have been proposed for collaborative inference systems. [Liu et al., 2019; Xiao et al., 2020; Li et al., 2021a] demonstrates that adversarial learning improves the robustness of the model against reconstruction attacks. [Titcombe et al., 2021; Mireshghallah et al., 2020] proposes learning additive Laplace noise to protect the privacy of inference data. [He et al., 2020] uses dropout to defend against reconstruction attacks. However, none of the existing work provides a systematic security measurement for various defenses. This leads to our main research question: *Are those privacy protection techniques against reconstruction attacks for collaborative inference systems really robust enough?*

In this work, we summarise the various defenses, systematically evaluate their performance in defending against reconstruction attacks, and present their trade-offs regarding data utility and privacy leakage. We propose a novel technique to break down existing defenses, which can achieve improved reconstruction performance for a variety of datasets, target models, and defense algorithms. We focus on the privacy of the inference data [He et al., 2019; Singh et al., 2021]. Since protectors obfuscate the intermediate feature representation to limit the redundant information about the model input, our idea is to learn to distill obfuscated features into sanitized features for better reconstruction. Specifically, we introduce Sensitive Feature Distillation (SFD) to extract the image structure knowledge of sensitive features and find that the proper formulation of prior knowledge will be helpful to solve the ill-posed reconstruction problem. We show that this vulnerability is inevitable even when different defense countermeasures are utilized. Figure 1 shows the reconstruction results using our proposed method and the existing one.

The contributions of this work are summarized as follows:

- We are the first to experimentally verify the robustness of reconstruction defenses for inference data privacy in collaborative systems.
- We reveal the presence of latent privacy features in protected representations and validate the vulnerability of existing defense methods.
- We propose a technique called SFD against the exisitng defense mechanisms. SFD can achieve better reconstruction by carefully extracting sensitive information to transfer the obfuscated features into the sanitized features.
- We conduct a comprehensive measurement of the robustness of various state-of-the-art defenses leveraging our proposed technique. Our experiments show that SFD can achieve 74.2% performance improvement compared with the baseline attack, making the existing defenses significantly less effective.

2 Background and Related Work

This paper focuses on the most common DL task: image classification. A Deep Neural Network (DNN) model can be characterized as $F : \mathcal{X} \mapsto \mathcal{Y}$ which maps an input data $x \in \mathcal{X}$ into an output result $y \in \mathcal{Y}$.

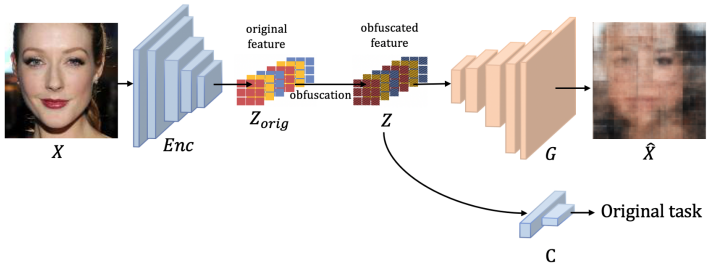
Collaborative Inference Systems. Running the entire deep learning inference application on the edge device can be time- and resource-consuming. A common scaling approach is to partition the DNN model where the first few layers of the model are executed within the local edge device while the remaining layers are offloaded to the remote cloud [Kang et al., 2017; Eshratifar et al., 2019; Banitalebi-Dehkordi et al., 2021; Li et al., 2021b]. This scheme can reduce the computational burden of edge devices on the one hand, and can mitigate privacy leaks of inference raw data on the other [Wang and Gong, 2018; Osia et al., 2020]. Collaborative inference makes it feasible and efficient to deploy large-scale intelligent workloads on today's edge platforms.

In the collaborative inference system, a model is split into two parts: an on-device part and a remote cloud-based part. Each part will contain several layers of the model. From now on, we reformulate the model $F = C \circ Enc$ as the Encoder module Enc and the Classifier module C. The edge device deploys the first part of the model Enc, it feeds the users' inference data x to the feature extractor to get an intermediate feature representation $z = Enc(x)$, and then sends z to the cloud. The cloud hosts the second part C, which calculates the prediction output $y = C(z)$ when the intermediate value z is received from the edge device, then it returns the expected result y to the edge side.

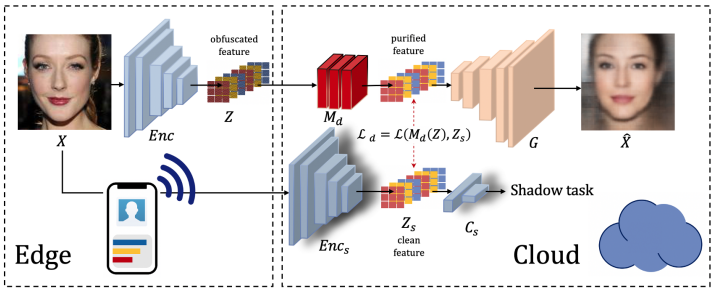
Machine Learning Privacy and Reconstruction Attacks. Recent research presented a variety of privacy threats against machine learning model, such as membership inference attacks [Shokri et al., 2017a; Song and Mittal, 2021], model extraction attacks [Tramèr et al., 2016], property inference attacks [Ganju et al., 2018] and model inversion attacks [Yang et al., 2019; He et al., 2019]. These threats are also important in the context of collaborative DL systems.

This paper focuses on the *Reconstruction Attacks* which are closely related to much previous work on model inversion attacks. They all aim to obtain sensitive information about the original data given access to the target machine learning model. In the model inversion attack, the adversary is asked to recover sensitive features of training data by taking advantage of their correlation with the model output. Although the original model inversion attack [Fredrikson et al., 2015] has limited performance [Hitaj et al., 2017], the inversion performance is significantly improved by training transposed CNNs on an auxiliary dataset [Yang et al., 2019; Zhang et al., 2020; Zhao et al., 2021; Wang et al., 2021].

Unlike recovering training data from the target ML models, a reconstruction attack is a class of techniques that aims to recreate the input data fed through a predictive model at inference time. [He et al., 2019; Singh et al., 2021] formalized diverse attack schemes on sensitive inputs in the context of edge-cloud collaborative inference systems. In the white-box attack setting, they developed a regularized likelihood maximization algorithm, which intended to find the optimal input data for the given deep neural representations and auxiliary information. The basic idea of the algorithm is to formulate the reconstruction attack as a gradient-based optimization problem in the data space $\mathcal{X}$ to find an image $\hat{x}$ whose prediction intermediate feature representation $Enc(\hat{x})$ approximates a given $Enc(x)$. However, it requires white-box access to the target model to compute the gradients to minimize the loss function. Furthermore, this method involves optimization for each natural sample at inference time which makes it relatively inefficient. The black-box reconstruction attack trains another neural network G to directly identify the inversed mapping from $Enc(x)$ to x. Specifically, without having the knowledge of the parameters of Enc, the adversary can collect some pairs (z, x) by querying the model, and then she trains a Decoder module G to learn to well approximate the mapping between intermediate representation z and corresponding input x. In contrast to the white-box optimization-based attack, this approach is more efficient: (1) calculating the reconstructed input is parameter-free; (2) for each target sample, the adversary only requires just one single forward pass through the Decoder module G. In this work, we focus on the black-box reconstruction attack, which is a more practical and realistic scenario.



(a) Defense against reconstruction attack



(b) Reconstruction attack based on SFD

Figure 2: Reconstruction defense and attack in edge-cloud collaborative inference systems. (a) shows the main idea of defenses. (b) shows the proposed reconstruction attack based on Sensitive Feature Distillation (SFD).

3 Problem Setup: Defenses Against Reconstruction Attacks

3.1 Adversary: Learn to Reconstruct

The goal of the adversary is to reconstruct the private input data sent by the user to the edge-cloud DL inference services. Following existing works on reconstruction attacks [He et al., 2019; Singh et al., 2021], we consider a collaborative inference system between a local, edge-device part $\mathbb{E}$ and a remote, cloud-based part $\mathbb{C}$. For a partitioned model $F = C \circ Enc$, $\mathbb{E}$ performs the earlier layers Enc and $\mathbb{C}$ performs C. We consider the $\mathbb{E}$ is trusted, when input is fed into Enc, $\mathbb{E}$ correctly processes it and computes a representation. $\mathbb{E}$ also attempts to prevent the input data from being reconstructed for privacy, so $\mathbb{E}$ applies the defenses to the representation and sends it to the cloud. The cloud $\mathbb{C}$, i.e. the adversary, is untrusted to try to steal the input.

3.2 Protector: Learn "NOT" to Reconstruct

For privacy, the edge part of the model typically obfuscates the intermediate feature representations before sending them to the cloud. The goal of the specially designed obfuscation is to encode the input x into a representation z that does not leak the characteristics of x, but whose expressive power is sufficient to predict the task label y. Several defense mechanisms have been proposed to mitigate the risks of reconstruction attacks. Figure 2 (a) shows the main idea of the defenses.

Adversarial Learning. The most straightforward defense against reconstruction attack is to apply adversarial training [Goodfellow et al., 2014] to improve the robustness of the model's intermediate feature representation. [Liu et al., 2019] presents the training objective of the target model, which aims at learning a privacy-preserving and task-oriented representation.

The protector formalizes the joint privacy and classification objectives in the following min-max optimization problem:

$$\underbrace{\min_{Enc,C}(\mathcal{L}_{task} + \lambda \underbrace{\max_{Dec} \mathcal{G}_{adv}}_{\text{optimal inversion}})}_{\text{optimal privacy-preserving classification}}. \tag{1}$$

, where the Enc aims to fool the adversary's reconstructor model by minimizing the reconstruction gain $\mathcal{G}_{adv}$, and the performance of C should be maintained. This results in designing the optimal privacy mechanism, which is also utility-maximizing.

Noise Mask. Noisy method has been introduced to protect model inference [Titcombe et al., 2021; Mireshghallah et al., 2020] by adding random noise to the inference data. [Titcombe et al., 2021] suggests defending against reconstruction attacks by adding additive Laplacian noise to the intermediate feature representation. Formally, it calculates $z^{noise} = Enc(x) + \epsilon$, the random noise ϵ is sampled from a Laplace distribution parameterized by location μ and scale b each time the model is used. The partitioned model, no longer a one-to-one function, obscures the data communicated between the edge and the cloud.

Dropout Defense. [He et al., 2020] introduces dropout defense to defeat reconstruction attacks. Specifically, the protector generates random masks M to protect the intermediate data during the inference phase of the model:

$$z^{dropout} = Enc(x) \otimes M \tag{2}$$

, where M denotes a randomly generated mask, $\otimes$ denotes the element-wise multiplication. To allocate each element of M to two values at random: one of them is 0 with probability r and the other is 1 with probability $1 - r$. Note that the dropout defense differs from noise obfuscation in that dropout hides partial sensitive information in the feature representation, whereas noise disturbs all the information of the intermediate data.

Applying noise mask or dropout defense directly to the feature representation in prediction can introduce a significant performance loss. As a result, the protector would prefer to perform training with the noise mask or dropout defense imposed, enhancing the robustness of the model to randomness.

3.3 Latent Privacy Features

We refer to [Singh et al., 2021] and visualize the original face image and the learned intermediate feature representations using adversarial learning [Liu et al., 2019], as depicted in Appendix A. We can note that after obfuscation by the defense mechanism, the learned individual feature mapping retains the latent sensitive features of the input image, especially the redundant spatial feature information. Consequently, we posit that extracting the potential image structure knowledge of sensitive features can help achieve better reconstruction results.

4 Methodology

The main goal of this paper is to study the potential privacy leakage points of existing defenses in collaborative inference systems. Specifically, we propose an anti-defense method and describe how to exploit Sensitive Feature Distillation (SFD) for a more aggressive reconstruction attack through the carefully designed algorithm.

4.1 Adversary's Capability

We consider a black-box scenario where an adversary is allowed to query the target model's encoder module Enc to make inferences from their collected data samples and obtain the corresponding intermediate features. For adversaries, this is the most challenging attack setting [Salem et al., 2018]. The Feature Distiller module of our attack model is defined as a supervised DL model, which needs ground truth data for training. So we adopt shadow training techniques following previous works [Shokri et al., 2017b; Salem et al., 2020; Zhou et al., 2021]. We consider the adversary trains a shadow model M_s, which is designed to imitate the behavior of the target model. We also assume that the shadow model has the same model architecture as the target model. In order to achieve this in practice, the adversary can perform model extraction attacks [Pal et al., 2020; Sha et al., 2022]. Moreover, to train the shadow model, we further assume that the adversary has a local dataset, denoted by D_s, which comes from the same distribution as the target model's training set. Once the shadow model has been trained, the adversary can rely on the shadow model to derive the ground truth clean feature to train her attack model. Note that most of the experiments related to shadow models' training by previous works make the same assumptions. We later show in Section 6 that the assumptions about the knowledge of model architecture can be relaxed as well.

4.2 Anti-defense Reconstruction Algorithm

Our general attack pipeline contains three phases. In the first phase, the adversary constructs a shadow model locally, which is used to generate training data for the next stage. In the second phase, the Feature Distiller module of our attack model learns to distill obfuscated features into sanitized features. In the last phase, the decoder module of our attack model takes the sanitized features to produce reconstructed images. The second and third steps are performed alternately in the concrete attack flow. Figure 2 (b) provides a schematic view of our attack pipeline.

Shadow Model. The Feature Distiller module of our attack model needs to be trained in a supervised manner. Due to our minimal assumptions in Section 4.1, the adversary can only obtain the obfuscated intermediate features by querying the target model. Nevertheless, we also need ground truth data, i.e. clean intermediate features, to train the Feature Distiller. To overcome this problem, we rely on shadow training, the shadow model is applied to mimic an unshielded target model. The adversary first establishes a shadow model $M_s = C_s \circ Enc_s$ with the same structure as the target model and then uses its own dataset to train M_s. She can use SGD or ADAM to optimize the cross-entropy loss function of the shadow model. If the adversary's goal is to find the inversion of the DNN for different tasks, then other more appropriate loss functions can be utilized. Once the shadow model M_s is obtained, the adversary can use it to generate pure intermediate features that are not obfuscated.

Sensitive Feature Distillation. A straightforward inversion attack may not succeed if the intermediate data z is obfuscated to reduce the amount of information it reveals about input x. To address this issue, we borrow the idea from the feature-based knowledge distillation [Romero et al., 2014; Zagoruyko and Komodakis, 2016]. But unlike these efforts, which use the intermediate feature mappings of the teacher model to help the student model achieve better results in the original classification task, we use the teacher model (shadow model) to help the student model (Feature Distiller) distill sensitive information to improve the effectiveness of the reconstruction attack.

To successfully reconstruct a user's private image x, our approach utilizes a convolutional neural network, termed Feature Distiller, to extract missing sensitive regions in the obfuscated intermediate features. In our settings, the Feature Distiller needs to use clean feature data as the training targets. Specifically, the adversary first uses her local training set X (X are sampled from the adversary's datasets $\mathcal{D}_s$) to query both the target model M and the shadow model M_s, and observes the corresponding intermediate outputs $Z = Enc(X)$ and $Z_s = Enc_s(X)$. Next, the adversary can directly train her Feature Distiller using Z as the training inputs and Z_s as the training targets. We take Sensitive Feature Distillation as an optimization problem with feature space L_2 loss:

$$\mathcal{L}_d = \sum_{z \in Z, z_s \in Z_s} \|M_d(z) - z_s\|_2^2 \tag{3}$$

, where M_d is the Feature Distiller that the adversary wants to learn. The possibility of training with a feature-space loss has been demonstrated in [Nguyen et al., 2016; Shen et al., 2019]. We match obfuscated features Z and clean features Z_s in order to encourage the Feature Distiller to identify the critical mappings in sensitive spatial information. After training, we can use the Feature Distiller to distill the latent vector $\hat{z}$ that is most similar to the clean feature from the target obfuscated feature z.

Reconstruction Attack. At this phase, we use the same black-box attack scheme proposed in [He et al., 2019; Singh et al., 2021]: the Inverse-Network (Supervised Decoder). Conceptually, the Inverse-Network is the approximated inverse function of Enc. However, our approach differs from the previous one in that the training input to the Inverse-Network is no longer observed data $Enc(x)$, but purified data $M_d(Enc(x))$ produced by the Feature Distiller. Meanwhile, the Inverse-Network G and the Feature Distiller M_d are jointly trained, which leads to better reconstruction quality. We show the detailed description of our proposed reconstruction attack using sensitive feature distillation in Appendix D.

Similar to the previous phase, the adversary generates a stack of training samples of labeled (z, x) pairs. She can then use $M_d(z)$ as the sanitized latent vectors of z to train the Inverse-Network. We leverage the L2 norm in the pixel space as the loss function of G:

$$\mathcal{L}_G = \sum_{z \in Z, x \in X} \|G(M_d(z)) - x\|_2^2 \tag{4}$$

Table 1: Experiment Configurations. We generate intermediate features from the different CNNs where several set of convolution, batch normalization, and activation layers are grouped under a conv block.

Dataset	MNIST	CIFAR10	CelebA
Target Model	ConvNet	VGG-16	ResNet-50
Split point	1st conv layer (8*8) 2nd conv layer(4*4) 1st fc layer(64)	1st conv block (16*16) 2nd conv block (8*8) 3rd conv block (4*4)	2nd conv block (16*16) 3rd conv block (8*8) 4th conv block (4*4) avgpool (1*1) 1st fc layer (512)

, G is optimized to reconstruct data from sanitized latent vectors. It is worth noting that the structure of G is not necessarily correlated with the target Encoder module Enc. After G and M_d are trained, the adversary can feed any observed intermediate-level value $z^\star = Enc(x^\star)$ to the attack model and obtain the reconstructed sensitive input $\hat{x}^\star = G(M_d(z^\star))$.

5 Experiment

We systematically measure the effectiveness of the existing defenses against reconstruction attacks and our proposed method. We conduct extensive experiments on three representative image classification tasks, each of which uses a different architecture of the target DNN model.

5.1 Experiment Setup

Datasets. To perform experimental evaluation, we use three standard benchmark image recognition datasets: MNIST [LeCun, 1998], CIFAR10 [Krizhevsky et al., 2009] and CelebA [Liu et al., 2015]. These datasets are also benchmark datasets for various ML security and privacy tasks. In most of our experiments, we split the dataset into three parts, target models' private dataset $\mathcal{D}_p$, adversary's shadow dataset $\mathcal{D}_s$ and the test set $\mathcal{D}_t$ with the rest. We detail each dataset in the following: MNIST contains 70,000 grey-scale images. We set $\mathcal{D}_p$, $\mathcal{D}_s$ and $\mathcal{D}_t$ to 50,000, 10,000 and 10,000 respectively; CIFAR10 consists of 60,000 color images. We separate $\mathcal{D}_p$, $\mathcal{D}_s$ and $\mathcal{D}_t$ to 40,000, 10,000 and 10,000 respectively; CelebA is a dataset composed of 202,599 images of celebrities. For simplicity, the task is to predict gender. The $\mathcal{D}_p$, $\mathcal{D}_s$ and $\mathcal{D}_t$ are set to 101,299, 81,040 and 20,260 accordingly.

Models. For MNIST, We experiment with ConvNet which is similar to the LeNet [LeCun et al., 1998] architecture, only making the network layer wider. For CIFAR10, we apply VGG-16 [Simonyan and Zisserman, 2014]. For CelebA, we adopt ResNet-50 [He et al., 2016]. We split each target model into different layers. For Inverse-Network G, following [He et al., 2019; Yang et al., 2019], we use transposed CNN blocks[2] to decode target intermediate feature representations to pixel images, where each block has a transposed convolutional layer succeeded by a batch normalization layer and a activation layer. The number of transposed CNN blocks increases with the deepening of the split point. For Feature Distiller M_d, we choose a neural network using multiple convolutional layers (when split layer is conv block) or full connection layers (when split layer is fc layer), with the number of hidden units set to the dimension of the feature data.

Compared Baselines. In the collaborative inference system scenario, we evaluate the effectiveness of the existing defenses by comparing the proposed attack with Existing Reconstruction Attack (ERA) [He et al., 2019] against all three defense methods.

Experiment Details and Evaluation Metrics. Table 1 gives details of the experimental configurations. To quantify the privacy risks of reconstruction attacks, we follow the setting in most of the previous work [Xiao et al., 2020; Singh et al., 2021]. We adopt two metrics, Mean Squared Error (MSE) and Structural Similarity Index (SSIM) [Wang et al., 2004]. We use different evaluation

[2]The official implementation of [He et al., 2019]: `https://github.com/zechenghe/Inverse_Collaborative_Inference`

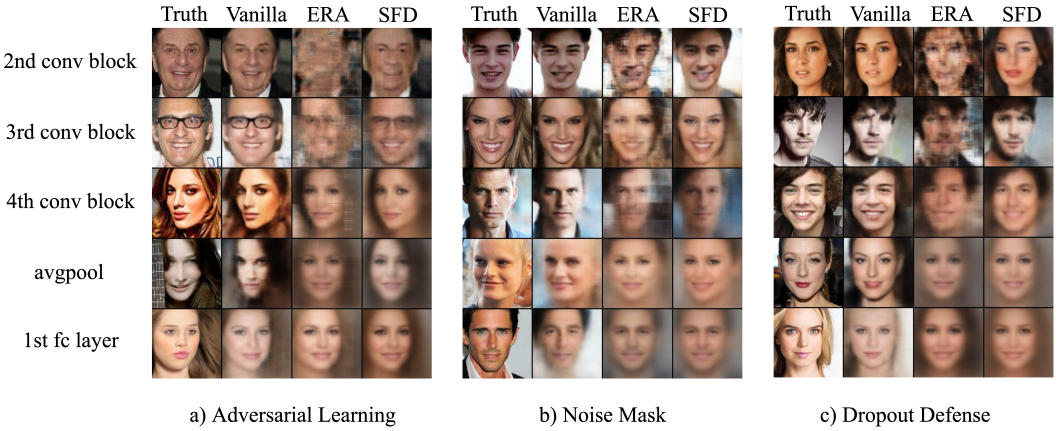


Figure 3: Reconstruction attack qualitative evaluation: We show the reconstructed images on CelebA. Each row corresponds to different split points of ResNet-50. In both (a), (b) and (c), the first column depicts the original images, and the second column depicts the result derived from attacking unprotected features, which is the upper bound for our approach. The third column shows the reconstruction results of ERA, while the fourth column shows the results of SFD.

metrics between the reconstructed images and the ground truth to gather multiple pieces of evidence of how well the attack reconstructs the private target image.

5.2 Empirical Results and Performance Analysis

Reconstruction Performance. Table 2 shows the performance of the proposed approach. We report the difference values (values in brackets) between the MSE loss of reconstruction attack based on SFD and the baseline ERA, and we also report the difference values between the SSIM of the reconstructed image generated from our method and from the baseline, both of which represent performance gains. Averagely, the proposed method provides a great improvement in reconstruction results against all three defenses. We owe this to SFD used in our method, which can capture sensitive spatial information for more aggressive attacks. Note that the vanilla results show the reconstruction effect of using ERA when the target model is without defense. Here, we just use vanilla results to show the best possible results for the reconstruction. The observation is that our approach still provides a competitive performance. Specifically, on the CelebA dataset, we improve on decreasing the MSE loss from **0.124** to **0.032** and increasing the SSIM from **0.370** to **0.618** compared to ERA when the split point is set to the 2nd conv block and against adversarial learning, Which brings a **74.2%** performance improvement (in terms of MSE loss values) while being close to the vanilla SSIM results **0.785**. For noise mask and dropout defense, we can achieve a **47.7%** and **40.9%** performance improvement, respectively.

Comparison for Different Split Points. We perform both quantitative (Table 2) and qualitative analysis (Figure 3) for different split points of the target model. The visual results highlight that SFD achieves significantly better reconstruction performance than ERA, and it works better at early layers than at later layers. This is confirmed by the quantitative results where SFD obtained MSE improvement of **0.092** (**74.2%**) and SSIM improvement of **0.248** in the 2nd conv block, compared to **0.003** (**1.6%**) and **0.015** in the avgpool. It is consistent with the intuition that features are becoming more and more abstract during the forward propagation process of the neural network, so the quality of reconstructed images of both SFD and ERA deteriorates as the layer goes deeper. See Appendix C for more examples and quantitative results.

Besides, we also observe a significant deterioration in reconstructed image quality on deeper layers (especially the fully connected layer). It is because the size of feature mapping on deeper layers is typically smaller than on shallower layers, and the fully connected layer even mixes up previous visual information, keeping only the semantic information. So it is relatively harder for SFD to distill sensitive features from small obfuscated representations. When selecting the split point in

Table 2: Reconstruction results against all three defenses, the data in brackets represents the attack performance improvement of reconstruction attacks based on SFD compared to the baseline ERA. Vanilla results show the reconstruction effect of using ERA when the target model is without defense. Defense hyper-parameters are set $\lambda = 2.0$ for adversarial learning (For MNIST and CelebA, $\lambda = 5.0$ for better privacy tradeoff), $b = 10.0$ for noise mask and $r = 0.9$ for dropout defense. The smaller the MSE and the larger the SSIM indicates the better the reconstruction results.

Dataset	**Split Point**	Vanilla		Adversarial		Noise		Dropout	
		MSE	SSIM	MSE ↓	SSIM ↑	MSE ↓	SSIM ↑	MSE ↓	SSIM ↑
MNIST	1st conv layer	0.002	0.994	0.442-(0.310)	0.123+(0.451)	0.323-(0.070)	0.138+(0.082)	0.022-(0.006)	0.943+(0.013)
	2nd conv layer	0.057	0.838	0.324-(0.191)	0.109+(0.494)	0.255-(0.089)	0.146+(0.319)	0.231-(0.110)	0.299+(0.358)
	1st fc layer	0.103	0.709	0.199+(0.016)	0.369-(0.066)	0.140+(0.007)	0.585-(0.036)	0.235-(0.002)	0.246-(0.006)
CIFAR10	1st conv block	0.006	0.942	0.220-(0.136)	0.457+(0.221)	0.045-(0.022)	0.723+(0.116)	0.075-(0.030)	0.537+(0.116)
	2nd conv block	0.026	0.762	0.221-(0.069)	0.167+(0.164)	0.112-(0.024)	0.409+(0.077)	0.173-(0.044)	0.217+(0.106)
	3rd conv block	0.100	0.397	0.219-(0.016)	0.156+(0.017)	0.212-(0.005)	0.130+(0.019)	0.223-(0.016)	0.132+(0.035)
CelebA	2nd conv block	0.010	0.785	0.124-(0.092)	0.370+(0.248)	0.172-(0.082)	0.289+(0.211)	0.088-(0.036)	0.416+(0.175)
	3rd conv block	0.020	0.698	0.189-(0.045)	0.331+(0.121)	0.206-(0.038)	0.332+(0.083)	0.155-(0.021)	0.370+(0.090)
	4th conv block	0.038	0.625	0.216-(0.023)	0.369+(0.048)	0.219-(0.010)	0.364+(0.034)	0.202-(0.005)	0.375+(0.027)
	avgpool	0.067	0.551	0.187-(0.003)	0.405+(0.015)	0.231-(0.002)	0.372+(0.015)	0.230-(0.008)	0.371+(0.025)
	1st fc layer	0.124	0.452	0.216+(0.005)	0.399-(0.002)	0.232+(0.002)	0.376+(0.010)	0.231-(0.003)	0.382+(0.007)

Table 3: Evaluation for the impact of different defense hyper-parameters on CIFAR10 dataset.

Defense	**Param**	1st conv block		2nd conv block		3rd conv block	
		Acc	SSIM	Acc	SSIM	Acc	SSIM
Adversarial	1.0	83.1	0.945+(0.005)	83.5	0.380+(0.117)	83.4	0.153+(0.014)
	2.0	79.8	0.457+(0.221)	81.8	0.167+(0.164)	81.5	0.156+(0.017)
	5.0	76.5	0.309+(0.181)	79.0	0.149+(0.123)	77.3	0.143+(0.023)
Noise	1.0	85.5	0.876+(0.053)	86.9	0.729+(0.019)	86.0	0.283+(0.029)
	5.0	83.7	0.796+(0.089)	85.5	0.526+(0.062)	85.9	0.144+(0.015)
	10.0	82.3	0.723+(0.116)	84.3	0.409+(0.077)	84.9	0.130+(0.019)
Dropout	0.5	84.9	0.818+(0.058)	85.4	0.594+(0.022)	84.9	0.256-(0.015)
	0.7	83.7	0.732+(0.089)	84.2	0.461+(0.061)	85.8	0.169+(0.027)
	0.9	79.9	0.537+(0.116)	81.9	0.217+(0.106)	84.6	0.132+(0.035)

a Privacy-Aware collaborative inference system, we suggest that the split layer on the edge device should be set deep enough (e.g., at least one fully connected layer) to hide the information of sensitive input samples.

Effect of Defense Hyper-parameters. Table 3 presents the reconstruction performance for varying defense hyper-parameters. The results indicate that even if the hyper-parameters of adversarial learning, noise mask, and dropout defense are set to 5.0, 10.0, and 0.9 (These parameter settings have incurred considerable utility loss), SFD can achieve SSIM improvements of 0.181, 0.116 and 0.116 respectively. This demonstrates that SFD can still effectively extract sensitive information from feature spaces that are protected by stronger defenses (more heavily obfuscated features).

In addition, we note that the attack performance of ERA decreases (lower SSIM) with increasing hyper-parameters, which will negatively affect the reconstruction results of SFD (more generic reconstructed images). Thus, even against SFD attacks, larger defense hyper-parameters may also provide a stronger level of obfuscation in reconstruction, but as a trade-off, this will severely degrade the utility of the target model. For more discussion about the comparison of defense methods, see Appendix B.

6 Discussion

Relaxing Attacker Model Assumption. To further relax the attacker's knowledge of the target model architecture, and for simplicity, we use a shadow model with the same size intermediate feature representation as the target model, but with a different architecture. Table 4 depicts the evaluation results. In our experiments on CIFAR10, we use VGG-16 mentioned previously in Section 5.1 for the target model, and use VGG-16, VGG-13, and ResNet-18 for the shadow models respectively. The results of the reconstruction of the intermediate features of different sizes show both worse and better performance, but the variations are small, indicating that SFD is robust against such changes in the architecture of the shadow model.

Table 4: Performance of shadow models for a different structure. The accuracy of shadow models using VGG-16, VGG-13, and ResNet-18 is 72%, 65%, and 61%, respectively.

Shadow Model	1st conv block		2nd conv block		3rd conv block	
	MSE	SSIM	MSE	SSIM	MSE	SSIM
None	0.220	0.457	0.221	0.167	0.219	0.156
VGG-16	0.084	0.678	0.152	0.331	0.203	0.173
VGG-13	0.082	0.682	0.154	0.330	0.206	0.202
ResNet-18	0.078	0.671	0.153	0.302	0.194	0.207

Possible Defenses. As shown in Table 2 and Figure 3, both quantitative and qualitative measurements of the reconstruction images become more generic as the split point becomes deeper. Meanwhile, increasing the defense hyperparameters can help mitigate the attacks to a degree, as illustrated in Table 3. Combining these two strategies may provide better privacy protection for inference data. However, this may cause a heavy computational burden on the client devices and reduce model accuracy. We have made it our future work to explore mechanisms to effectively defend against reconstruction attacks in collaborative inference systems.

7 Conclusion

In this work, we track the potential privacy leakage points of defense mechanisms against reconstruction attacks in collaborative inference systems. We posit that extracting the latent image structure knowledge of sensitive features can help achieve better reconstruction. Therefore, we propose a technique called SFD to bypass the existing defense mechanisms. SFD is designed to purify feature representations that are obfuscated by the defense mechanisms, so as to better extract sensitive information. We systematically measure the effectiveness of SFD against three state-of-the-art defenses. Experimental results show that SFD achieves significant performance gain over existing reconstruction techniques on multiple datasets, which demonstrates that current defense schemes are not robust enough. We hope our findings encourage critical exploration of defense mechanisms against reconstruction attacks for private collaborative inference.

References

Amin Banitalebi-Dehkordi, Naveen Vedula, Jian Pei, Fei Xia, Lanjun Wang, and Yong Zhang. Auto-split: a general framework of collaborative edge-cloud ai. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2543–2553, 2021.

Amir Erfan Eshratifar, Mohammad Saeed Abrishami, and Massoud Pedram. Jointdnn: an efficient training and inference engine for intelligent mobile cloud computing services. *IEEE Transactions on Mobile Computing*, 20(2):565–576, 2019.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 619–633, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Zecheng He, Tianwei Zhang, and Ruby B Lee. Model inversion attacks against collaborative inference. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 148–162, 2019.

Zecheng He, Tianwei Zhang, and Ruby B Lee. Attacking and protecting data privacy in edge–cloud collaborative inference systems. *IEEE Internet of Things Journal*, 8(12):9706–9716, 2020.

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 603–618, 2017.

Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '17, page 615–629, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450344654. doi: 10.1145/3037697.3037698. URL https://doi.org/10.1145/3037697.3037698.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ang Li, Jiayi Guo, Huanrui Yang, Flora D Salim, and Yiran Chen. Deepobfuscator: Obfuscating intermediate representations with privacy-preserving adversarial learning on smartphones. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 28–39, 2021a.

Min Li, Yu Li, Ye Tian, Li Jiang, and Qiang Xu. Appealnet: An efficient and highly-accurate edge/cloud collaborative architecture for dnn inference. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 409–414. IEEE, 2021b.

Sicong Liu, Junzhao Du, Anshumali Shrivastava, and Lin Zhong. Privacy adversarial network: representation learning for mobile data privacy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(4):1–18, 2019.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

Fatemehsadat Mireshghallah, Mohammadkazem Taram, Prakash Ramrakhyani, Ali Jalali, Dean Tullsen, and Hadi Esmaeilzadeh. Shredder: Learning noise distributions to protect inference privacy. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 3–18, 2020.

Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in neural information processing systems*, 29, 2016.

Seyed Ali Osia, Ali Shahin Shamsabadi, Sina Sajadmanesh, Ali Taheri, Kleomenis Katevas, Hamid R Rabiee, Nicholas D Lane, and Hamed Haddadi. A hybrid deep learning architecture for privacy-preserving mobile analytics. *IEEE Internet of Things Journal*, 7(5):4505–4518, 2020.

Soham Pal, Yash Gupta, Aditya Shukla, Aditya Kanade, Shirish Shevade, and Vinod Ganapathy. Activethief: Model extraction using active learning and unannotated public data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 865–872, 2020.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. {Updates-Leak}: Data set inference and reconstruction attacks in online learning. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1291–1308, 2020.

Zeyang Sha, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. Can't steal? cont-steal! contrastive stealing attacks against image encoders. *arXiv preprint arXiv:2201.07513*, 2022.

Zhiqiang Shen, Zhankui He, and Xiangyang Xue. Meal: Multi-model ensemble via adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4886–4893, 2019.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017a.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017b.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Abhishek Singh, Ayush Chopra, Ethan Garza, Emily Zhang, Praneeth Vepakomma, Vivek Sharma, and Ramesh Raskar. Disco: Dynamic and invariant sensitive channel obfuscation for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12125–12135, 2021.

Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2615–2632, 2021.

Tom Titcombe, Adam J Hall, Pavlos Papadopoulos, and Daniele Romanini. Practical defences against model inversion attacks for split neural networks. *arXiv preprint arXiv:2104.05743*, 2021.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pages 601–618, 2016.

Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 36–52. IEEE, 2018.

Kuan-Chieh Wang, Yan Fu, Ke Li, Ashish Khisti, Richard Zemel, and Alireza Makhzani. Variational model inversion attacks. *Advances in Neural Information Processing Systems*, 34:9706–9719, 2021.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Taihong Xiao, Yi-Hsuan Tsai, Kihyuk Sohn, Manmohan Chandraker, and Ming-Hsuan Yang. Adversarial learning of privacy-preserving and task-oriented representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12434–12441, 2020.

Ziqi Yang, Jiyi Zhang, Ee-Chien Chang, and Zhenkai Liang. Neural network inversion in adversarial setting via background knowledge alignment. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 225–240, 2019.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 253–261, 2020.

Xuejun Zhao, Wencan Zhang, Xiaokui Xiao, and Brian Lim. Exploiting explanations for model inversion attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 682–692, 2021.

Junhao Zhou, Yufei Chen, Chao Shen, and Yang Zhang. Property inference attacks against gans. *arXiv preprint arXiv:2111.07608*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of computing and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]