Soteria: Provable Defense against Privacy Leakage in Federated Learning from Representation Perspective

Jingwei Sun, Ang Li, Binghui Wang, Huanrui Yang, Hai Li, Yiran Chen
Department of Electrical and Computer Engineering, Duke University
{jingwei.sun, ang.li630, binghui.wang, huanrui.yang, hai.li, yiran.chen}@duke.edu

Abstract

Federated learning (FL) is a popular distributed learning framework that can reduce privacy risks by not explicitly sharing private data. However, recent works have demonstrated that sharing model updates makes FL vulnerable to inference attack. In this work, we show our key observation that the data representation leakage from gradients is the essential cause of privacy leakage in FL. We also provide an analysis of this observation to explain how the data presentation is leaked. Based on this observation, we propose a defense called Soteria against model inversion attack in FL. The key idea of our defense is learning to perturb data representation such that the quality of the reconstructed data is severely degraded, while FL performance is maintained. In addition, we derive a certified robustness guarantee to FL and a convergence guarantee to FedAvg, after applying our defense. To evaluate our defense, we conduct experiments on MNIST and CIFAR10 for defending against the DLG attack and GS attack. Without sacrificing accuracy, the results demonstrate that our proposed defense can increase the mean squared error between the reconstructed data and the raw data by as much as 160× for both DLG attack and GS attack, compared with baseline defense methods. Therefore, the privacy of the FL system is significantly improved. Our code can be found at https://github.com/jeremy313/Soteria.

1. Introduction

Federated learning (FL) [16] is a popular distributed learning approach that enables a number of devices to train a shared model in a federated fashion without transferring their local data. A central server coordinates the FL process, where each participating device communicates only the model parameters on the central server while keeping local data private. Thus, FL becomes a natural choice for developing mobile deep learning applications, such as next-word prediction [10], emoji prediction [22], etc.

Privacy preservation is the major motivation for proposing FL. However, recent works demonstrated that sharing model updates or gradients also makes FL vulnerable to inference attack, e.g., *property inference attack* [18] and *model inversion attack* [5, 28, 7, 26]. Here property inference attack infers sensitive properties of training data using the model updates, and model inversion attack reconstructs training data using model gradients. However, the essential causes of such privacy leakages have not been thoroughly investigated or explained. Some defense strategies have been presented to prevent the privacy leakage and can be categorized into three types: *differential privacy* [21, 24, 9, 17, 8], *secure multi-party computation* [4, 19, 3, 18], and *data compression* [28]. But these defensive approaches incur either significant computational overheads or unignorable accuracy loss. The reason is that existing defenses are not specifically designed for the privacy leakage from the communicated local updates. The privacy issues seriously hinder the development and deployment of FL. There is an urgent necessity to unveil the essential cause of privacy leakage such that we can develop effective defenses to tackle the privacy issue of FL.

In this work, we assume that the server in FL is malicious and it aims to reconstruct the private training data from devices. Our key observation is: **the class-wise data representations of each device's data are embedded in shared local model updates, and such data representations can be inferred to perform model inversion attacks. Therefore, the information can be severely leaked through the model updates.** In particular, we provide an analysis to reveal how the data representations, e.g., in the fully connected (FC) layer, are embedded in the model updates. We then propose an algorithm to infer class-wise data representation to perform model inversion attacks. Our empirical study demonstrates that the correlation between the inferred data representations using our algorithm and the real data representations is as high as 0.99 during local training, and thus proves that the representations leakage is the essential cause behind existing attacks. Note that the data is often non-IID (identically and independently distributed) across the devices in FL. We also show that the non-IID character-

istic aggravates the representation leakage.

Based on our observation of the representation leakage from local updates, we design a defense called Soteria. Specifically, we present an algorithm to generate a perturbation added to the data representations, such that: 1) the perturbed data representations are as similar as possible to the true data representations to maintain the FL performance; and 2) the reconstructed data using the perturbed data representations are as dissimilar as possible to the raw data. Importantly, we also derive certified robustness guarantee to FL and convergence guarantee to FedAvg, a popular FL algorithm, when applying our defense. To evaluate the effectiveness of our defense, we conduct experiments on MNIST and CIFAR10 for defending against the DLG attack [28] and GS attack [7]. The results demonstrate that without sacrificing accuracy, our proposed defense can increase mean squared error (MSE) between the reconstructed data and the raw data for both DLG attack and GS attack by as much as 160×, compared with baseline defense methods. Therefore, the privacy of the FL system is significantly improved.

Our key contributions are summarized as follows:

- To the best of our knowledge, this is the first work to explicitly reveal that data representations embedded in the model updates are the essential cause of leaking private information from the communicated local updates in FL. In addition, we develop an algorithm to effectively reconstruct the data from the local updates.
- We develop an effective defense by perturbing data representations. We also derive certified robustness guarantee to FL and convergence guarantee to FedAvg when applying our defense.
- We empirically evaluate our defense on MNIST and CIFAR10 against DLG and GS attacks. The results show our defense can offer a significantly stronger privacy guarantee without sacrificing accuracy.

2. Related work

Privacy Leakage in Distributed Learning. There exist several adversarial goals to infer private information: *data reconstruction*, *class representative inference*, *membership inference*, and *attribute inference*. Data reconstruction aims to recover training samples that are used by participating clients. The quality of the reconstructed samples can be assessed by comparing the similarity with the original data. Recently, Zhu *et al*. [28] present an algorithm named *DLG* to reconstruct training samples by optimizing the input to generate the same gradients for a particular client. Following up DLG, *iDLG* [27] is proposed to improve the efficiency and accuracy of DLG. Aono *et al*. [1] also show that an honest-but-curious server can partially reconstruct clients' training inputs using their local updates. However, such an attack is applicable only when the batch consists of a single sample. Wang *et al*. [26] present a reconstruction attack by incorporating a generative adversarial network (GAN) with a multi-task discriminator. But this method is only applicable to scenarios where data is mostly homogeneous across clients and auxiliary dataset is available. Several approaches have been proposed to infer class features or class representatives. Hitaj *et al*. [11] demonstrate that an adversarial participant in the collaborative learning can utilize GANs to construct class representatives. However, this technique is evaluated only when all samples of the same class are virtually similar (e.g., handwritten digits, faces, etc.). Membership inference attack (MIA) is performed to accurately determine whether a given sample has been used for the training. This type of attack is first proposed by Shokri*et al*. [25], and it can be applied to any types of machine learning models even under black-box settings. Sablayrolles *et al*. [23] propose an optimal strategy for MIA under the assumption that model parameters conform to certain distributions. Nasr *et al*. [20] extend MIA to federated learning for quantifying the privacy leakage in the distributed setting. Attribute inference attack [2, 6, 5, 11] tries to identify some sensitive attributes of training data. Fredrikson *et al*. [6] proposes a method to reveal genomic information of patients using model outputs and other non-sensitive attributes. More recently, Melis *et al*. [18] demonstrate that an adversarial client can infer attributes that hold only for a subset of the training data based on the exchanged model updates in federated learning.

Privacy-preserving Distributed Learning. Existing privacy-preserving distributed learning methods can be categorized into three types: *differential privacy* (DP), *secure multi-party computation* (MPC), and *data compression*. Pathak *et al*. [21] present a distributed learning method to compose a deferentially private global model by aggregating locally trained models. Shokri *et al*. [24] propose a collaborative learning method where the sparse vector is adopted to achieve DP. Hamm *et al*. [9] design a distributed learning approach to train a deferentially private global model via transferring the knowledge of the local model ensemble. Recently, participant-level deferentially private federated learning are proposed [17, 8] via injecting Gaussian noise to local updates. However, these DP-based methods require a large number of participants in the training to converge and realize a desirable privacy-performance tradeoff. In addition, MPC has also been applied to develop privacy-preserving machine learning in a distributed fashion. For example, Danner *et al*. [4] propose a secure sum protocol using a tree topology. Another example of the MPC-based approach is SecureML [19], where participants distribute their private data among two non-colluding servers, and then the two servers use MPC to train a global model using the participants' encrypted joint data. Bonawitz *et al*. [3] propose a secure multi-party

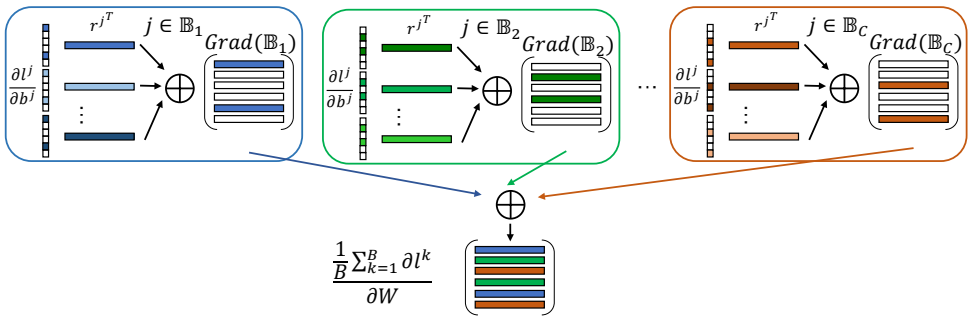

Figure 1. Illustration of the gradient updates of class-wise data in a batch.

aggregation method for FL, where participants are required to encrypt their local updates such that the central server can only recover the aggregation of the updates. However, these MPC-based approaches will incur unneglectable computational overhead. It is even worse that attackers can still successfully infer private information even if the adversary only observes the aggregated updates [18]. Furthermore, Zhu *et al*. [28] show applying gradient compression and sparsification can help defend against privacy leakage from shared local updates. However, such approaches require a high compression rate to achieve a desirable defensive performance. In Section 6, given the same compression rate, we show that our proposed method can achieve better defense and inference performance than that of the gradient compression approach.

3. Essential Cause of Privacy Leakage in FL

Existing works [28, 27, 1, 26] demonstrate that information leakage is from communicated model updates between the devices and server during FL training. However, they do not provide a thorough explanation. To understand the essential cause of information leakage in FL, we analyze the privacy leakage in FL. Our key observation is that privacy leakage is essentially caused by the data representations embedded in the model updates.

3.1. Representation Leakage in FL

Problem setup. In FL, there are multiple devices and a central server. The server coordinates the FL process, where each participating device communicates only the local model parameters with the server while keeping their local data private. We assume the server is malicious and it only has access to the devices' model parameters. The server's purpose is to infer the devices' data through the devices' model parameters.

Key observations on representation leakage in FL: Data representations are less entangled. For simplicity, we use the fully connected (FC) layer as an instance and analyze how data representation is leaked in FL. We note that such an analysis can be naturally extended to other types of layers. Specifically, we denote a FC layer as $\boldsymbol{b} = \boldsymbol{W}\boldsymbol{r}$, where $\boldsymbol{r}$ is the input to the FC layer (i.e., the learnt data representation by previous layers), $\boldsymbol{W}$ is the weight matrix, and $\boldsymbol{b}$ is the output. Then, given a training batch $\mathbb{B}$, the gradient of the loss l with respect to $\boldsymbol{W}$ is:

$$\frac{\frac{1}{|\mathbb{B}|}\sum_{i=1}^{|\mathbb{B}|}\partial l^i}{\partial \boldsymbol{W}} = \frac{1}{|\mathbb{B}|}\sum_{i=1}^{|\mathbb{B}|}\frac{\partial l^i}{\partial \boldsymbol{b}^i}\frac{\partial \boldsymbol{b}}{\partial \boldsymbol{W}} = \frac{1}{|\mathbb{B}|}\sum_{i=1}^{|\mathbb{B}|}\frac{\partial l^i}{\partial \boldsymbol{b}^i}(\boldsymbol{r}^i)^{\mathrm{T}}, \quad (1)$$

where l^i, $\boldsymbol{r}^i$, and $\boldsymbol{b}^i$ are the loss corresponding to the i^{th} sample, the input, and the output of the FC layer in this batch, respectively. We observe that the gradient for a particular sample is the product of a column vector $\frac{\partial l^i}{\partial \boldsymbol{b}^i}$ and a row vector $(\boldsymbol{r}^i)^{\mathrm{T}}$. Suppose the training data has C labels. We can split the batch $\mathbb{B}$ into C sets, i.e., $\mathbb{B} = \{\mathbb{B}_0, \mathbb{B}_1, \ldots, \mathbb{B}_C\}$, where $\mathbb{B}_k$ denotes the data samples with the k-th label. Then, Equ. (1) can be rewritten as:

$$\frac{\frac{1}{|\mathbb{B}|}\sum_{k=1}^{|\mathbb{B}|}\partial l^k}{\partial \boldsymbol{W}} = \sum_{i=1}^{C}\left(\frac{1}{|\mathbb{B}_i|}\sum_{j\in\mathbb{B}_i}\frac{\partial l^j}{\partial \boldsymbol{b}^j}(\boldsymbol{r}^j)^{\mathrm{T}}\right) \triangleq \sum_{i=1}^{C} Grad(\mathbb{B}_i), \quad (2)$$

where $Grad(\mathbb{B}_i)$ represents the gradient with respect to the data samples in $\mathbb{B}_i$. Figure 1 illustrates the gradient updates for a batch data in a FC layer. We observe that for data coming from different classes, the corresponding data representations tend to be embedded in different rows of gradients. If the number of classes is large in a batch, which is common in centralized training, the representations of different classes will be entangled in the gradients of this whole batch. In contrast to centralized training, the local data often covers a small number of tasks on a participating device in FL. Thus, the number of data classes C within one training batch may be very small compared to that of the centralized training. In this case, the entanglement of data representations from different classes can be significantly reduced. Such a low entanglement of data representations allows us to explicitly reconstruct the input data of each class from the gradients, because we can (almost) precisely locate the rows of data representations in the gradients.

Note that in the above analysis, we only consider a single batch during the FL training. In practice, FL is often trained with multiple batches. In this case, the data representations of different classes could be entangled, especially when the number of batches is large. However, in practical FL applications, the devices often have insufficient data. During FL

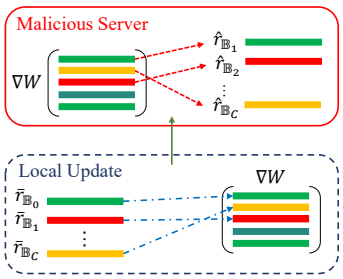

Figure 2. Illustration of our representation inference algorithm.

training, the numbers of batches and local training epochs of each device are both small. In this case, the data representations could still be less entangled across classes through inspecting the gradient updates in Equ. (2).

3.2. Inferring Class-wise Data Representations

We develop an algorithm to identify the training classes and infer the data representations of each class embedded in each FC layer from the model updates. The representation inference algorithm is in a back propagation fashion. Specifically, we first identify the classes using the gradients of the last $(L-th)$ layer. We denote the gradients of the L-th layer as $\nabla \boldsymbol{W}^L$. We notice that the gradient vector $\nabla \boldsymbol{W}_i^L$, which is the ith row in $\nabla \boldsymbol{W}^L$, shows significantly larger magnitudes than gradient vectors in other rows if the data from i-th class are involved in training. Then, we can infer the data representation of the i-th class in the last layer, because it linearly scales $\nabla \boldsymbol{W}_i^L$. If data representation of the i-th class in layer $\boldsymbol{W}^L$ is inferred, we can use their element values to identify the corresponding row from the $(L-1)$-th layer's gradients, i.e., $\boldsymbol{W}^{L-1}$, which embeds the data representation of the i-th class in the $(L-1)$-th layer. In this way, we can iteratively infer data representations of the i-th class in all FC layers. The inference process for one FC layer is illustrated in figure 2 and the details of our representation inference algorithm are described in Appendix A.

We conduct experiments on CIFAR10 [13] to evaluate the effectiveness of our algorithm. We consider the practical non-IID settings in FL, and follow the **2-class & balanced** configuration in [14] to construct non-IID datasets: 100 devices in total and 10 devices are randomly sampled to participate in training in each communication round. Each device holds 2 classes of data and each class has 20 samples.

As local training configurations can affect the performance of inferred representation. In this experiment, we vary the number of local training epochs $E \in \{1, 5, 10\}$ and local batch size $B \in \{8, 16, 32\}$. We adopt SGD as the optimizer with a learning rate $\eta = 0.01$. The model architecture is shown in Appendix B. We also consider a baseline in the IID setting, where we set E to be 1 and B to be 32.

We use the correlation coefficient *cor* between the true representation $\bar{\boldsymbol{r}}_{\mathbb{B}_i}^{\mathrm{T}}$ and our inferred $\hat{\boldsymbol{r}}_{\mathbb{B}_i}^{\mathrm{T}}$ to quantify the effectiveness of our proposed algorithm. We calculate *cor* for each class on each participating device. We extract data representations from all the FC layers in each of 200 communication rounds between the devices and the server, and the average *cor* across all communication rounds and devices is shown in Table 2. As Table 2 presents, the correlation *cor* is as high as 0.99, indicating a serious representation leakage in FL. *cor* decreases with B or a larger number of batches in one epoch and increases as E goes lower, which validate our claim in Section 3.1. However, *cor* is still higher than 0.8 in almost all cases. We note that *cor* is much lower in the IID-setting. This is because each device has more classes of data for training than those in non-IID setting, making the representations entangled. Our results validate that the practical non-IID setting in FL dramatically worsens the representation leakage.

Table 1. Average *cor* across 200 communication rounds for different layers under different settings.

Local Training Configurations	FC1	FC2	FC3
E=1, B=32	0.98	0.99	0.99
E=5, B=32	0.82	0.90	0.92
E=10, B=32	0.70	0.78	0.82
E=1, B=16	0.82	0.93	0.99
E=1, B=8	0.85	0.89	0.92
E=1, B=32 (IID)	0.48	0.31	0.18

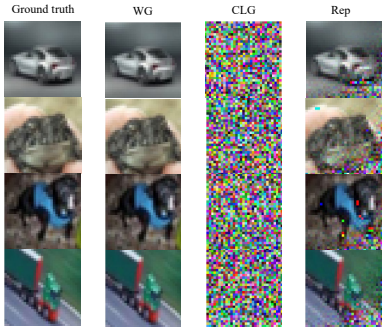

Figure 3. DLG attack results utilizing different parts of gradients.

3.3. Unveiling Representation Leakage

In this section, we investigate whether the representation leakage is the essential cause of information leakage in FL. Particularly, we conduct experiments on CIFAR10 to reconstruct the input based on the existing DLG attack [28]. DLG attack requires the gradient information, and we consider three different portions of the gradients: *the whole model gradients (WG)*, *the gradients of convolutional layers only (CLG)*, and *inferred representations using our method (Rep)*. The experiment settings are presented in Appendix B. As figure 3 shows, only utilizing gradients of convolutional layers cannot successfully reconstruct the input data, but using the representation inferred by our method can reconstruct the input data as effectively as utilizing the whole gradients in terms of visual quality. This result validates that representation leakage is the essential cause of privacy leakage in FL.

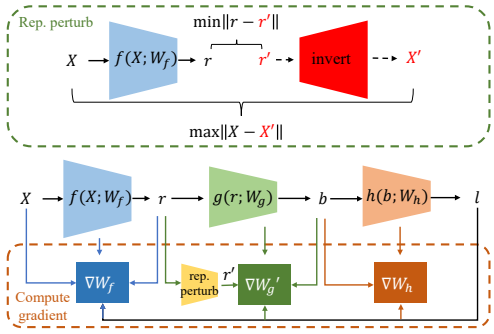

Figure 4. Illustration of our representation perturbation defense.

4. Defense Design

4.1. Defense Formulation

Our aforementioned observation shows that the privacy leakage in FL mainly comes from the representation leakage (e.g., in the FC layer). In this section, we propose a defense against such privacy leakage called Soteria. In particular, we propose to perturb the data representation in a single layer (e.g., a FC layer), which we call *the defended layer*, to satisfy the following two goals:

- Goal 1: To reduce the privacy information leakage, the reconstructed input through the perturbed data representations and the raw input should be dissimilar.
- Goal 2: To maintain the FL performance, the perturbed data representation and the true data representations without perturbation should be similar.

Let r and r' represent the clean data representation and perturbed data representation on the defended layer, respectively. We also define X and X' as the raw input and the reconstructed input via the perturbed data representation. To achieve Goal 1, we require that the distance between X and X', in terms of L_p norm, should be as large as possible; To reach Goal 2, we require that the distance between r and r', in terms of L_q norm, should be bounded. Formally, we have the following constrained objective function with respect to r':

$$\textbf{Achieving Goal 1:} \quad \max_{r'} ||X - X'||_p, \tag{3}$$

$$\textbf{Achieving Goal 2:} \quad \text{s.t.}, \ ||r - r'||_q \leq \epsilon, \tag{4}$$

where ϵ is a predetermined threshold. Note that X' depends on r'. Next, we design a solution to obtain r' and derive the certified robustness.

4.2. Defense Solution

Let $f : X \rightarrow r$ be the feature extractor before the defended layer. Prior to obtaining our solution, we make the following assumption and use the inverse function theorem.

Assumption 1. *The inverse of f, i.e., f^{-1}, exists on r and r', $\forall ||r - r'||_q \leq \epsilon$.*

Algorithm 1 Learning perturbed representation r' with $q = 0$ and $p = 2$.

Input: Training data $X \in \mathbb{R}^{M \times N}$; Feature extractor $f : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^L$ before the defended layer; Clean data representation $r \in \mathbb{R}^L$; Perturbation bound: ϵ;
Output: Perturbed data representation $r' \in \mathbb{R}^L$;
1: **function** PERTURB_REP(X, f, r, ϵ)
2: Compute $||r_i(\nabla_X f(r_i))^{-1}||_2$ for $i = 0, 1, ..., L-1$;
3: Find the set $\mathbb{S}$ which contains the indices of ϵ largest elements in $\{||r_i(\nabla_X f(r_i))^{-1}||_2\}_{i=1}^{L}$;
4: $r' \leftarrow r$;
5: Set $r'_i = 0$ for $i \in \mathbb{S}$;
6: **return** r';
7: **end function**

Lemma 1. *For $\forall f : X \rightarrow r$ and $f^{-1} : r \rightarrow X$, $\nabla_r f^{-1} = (\nabla_X f)^{-1}$.*

Then, our object function can be reduced as follows:

$$r' = \arg\max_{r'} ||X - X'||_p, \ s.t. ||r - r'||_q \leq \epsilon \tag{5}$$

$$= \arg\max_{r'} ||f^{-1}(r) - f^{-1}(r')||_p, \ s.t. ||r - r'||_q \leq \epsilon \tag{6}$$

$$\approx \arg\max_{r'} ||\nabla_r f^{-1} \cdot (r - r')||_p, \ s.t. ||r - r'||_q \leq \epsilon \tag{7}$$

$$= \arg\max_{r'} ||(\nabla_X f)^{-1} \cdot (r - r')||_p, \ s.t. ||r - r'||_q \leq \epsilon, \tag{8}$$

where we use Assumption 1 in Equ. (6), use the first-order Taylor expansion in Equ. (7), and use Lemma 1 in Equ. (8).

Note that, with different choices of $||.||_p$ and $||.||_q$, we have different defense solutions and thus have different defense effects. In this work, we set $p = 2$, i.e., we aim to maximize the MSE between the reconstructed input and the raw input. Meanwhile, we set $q = 0$ due to two reasons: our defense has an analytical solution and is communication efficient. Specifically, our solution is to find the ϵ largest elements in the set $\{||r_i(\nabla_X f(r_i))^{-1}||_2\}$. Moreover, the learnt perturbed representation is relatively sparse and thus improves the communication efficiency. Algorithm 1 details the solution to obtain the perturbed presentation r' with $q = 0$ and $p = 2$. Algorithm 2 details the local training process with our defense on a local device.

4.3. Certified Robustness Guarantee

We define our *certified robustness guarantee* as the certified minimal distance (in terms of L_p-norm) between the raw input and the reconstructed input. A larger defense bound indicates that our defense is more effective. Specifically, we have the following theorem on our defense bound:

Theorem 1. *Assuming Assumption 1 holds. Given a data input X, its representation r and any perturbed data representation r', we have:*

$$||X - X'||_p \geq \frac{||r - r'||_p}{||\nabla_X f||_p}. \tag{9}$$

Proof. See our proof in Appendix C. □

Algorithm 2 Local training process with our defense on a local device.

Input: Training data $\boldsymbol{X} \in \mathbb{R}^{M\times N}$; Local objective function $F : \mathbb{R}^{M\times N} \to \mathbb{R}$; Feature extractor $f : \boldsymbol{W}_f \in \mathbb{R}^{M\times N} \to \mathbb{R}^L$ before the defended layer; The defended layer $g : \boldsymbol{W}_g \in \mathbb{R}^L \to \mathbb{R}^K$; Feature extractor after the defended layer $h : \boldsymbol{W}_h \in \mathbb{R}^K \to \mathbb{R}$; Local model parameters $\boldsymbol{W} = \{\boldsymbol{W}_f, \boldsymbol{W}_g, \boldsymbol{W}_h\}$; Learning rate η.
Output: Learnt model parameter $\boldsymbol{W}$ with our defense.

Initialize $\boldsymbol{W}$;
for $\mathbb{B}$ in local training batches **do**
  for $\boldsymbol{X} \in \mathbb{B}$ **do**
    $l \leftarrow F(\boldsymbol{X}; \boldsymbol{W})$;
    $\boldsymbol{r} \leftarrow f(\boldsymbol{X}; \boldsymbol{W}_f)$;
    $\boldsymbol{b} \leftarrow g(\boldsymbol{r}; \boldsymbol{W}_g)$; // e.g., $\boldsymbol{b} = \boldsymbol{W}_g\boldsymbol{r}$ for FC layers
    $l \leftarrow h(\boldsymbol{b}; \boldsymbol{W}_h)$;
    $\{\nabla\boldsymbol{W}_f, \nabla\boldsymbol{W}_g, \nabla\boldsymbol{W}_h\} \leftarrow \nabla_{\boldsymbol{W}} F(\boldsymbol{X}; \boldsymbol{W})$;
    $\boldsymbol{r}' \leftarrow Perturb\_rep(\boldsymbol{X}, f(;\boldsymbol{W}_f), \boldsymbol{r}, \epsilon)$;
    $\nabla\boldsymbol{W}'_g \leftarrow \tau(l, b, r', \boldsymbol{W}_g)$; // e.g., $\nabla\boldsymbol{W}'_g = \frac{\partial l}{\partial \boldsymbol{b}}\boldsymbol{r}'^T$ in FC
    $\nabla\boldsymbol{W} = \{\nabla\boldsymbol{W}_f, \nabla\boldsymbol{W}'_g, \nabla\boldsymbol{W}_h\}$;
    $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta\nabla\boldsymbol{W}$;
  end for
end for

5. Convergence Guarantee

In this section, we derive the convergence guarantee of FedAvg [16]—the most popular FL algorithm, with our proposed defense. We first describe the FedAvg algorithm with our defense and then present our theorem on the convergence guarantee.

5.1. FedAvg with Our Defense

In classical FedAvg, the objective function is defined as:

$$\boldsymbol{W} = \min_{\boldsymbol{W}} \{F(\boldsymbol{W}) \triangleq \sum_{k=1}^{N} p_k F_k(\boldsymbol{W})\}, \tag{10}$$

where p_k is the weight of the k-th device, $p_k \geq 0$ and $\sum_{k=1}^{N} p_k = 1$. F_k is the local objective in the k-th device.

Equation 10 is solved via an iterative server-devices communication as follows: Suppose the server has learnt the global model $\boldsymbol{W}_t$ in a specific communication round, and randomly selects K devices $\mathbb{S}_t$ with replacement according to the sampling probabilities $p_1, ..., p_N$ for the next training round. Then FedAvg is performed as follows: First, the server sends the global model $\boldsymbol{W}_t$ to all devices. Then, all devices set their local model to be $\boldsymbol{W}_t$, i.e., $\boldsymbol{W}_t^k = \boldsymbol{W}_t, \forall k \in [1 : N]$, and each device performs I iterations of local updates. Specifically, for the i-th iteration, the local model in the k-th device applying our defense is updated as:

$$\nabla F'_k(\boldsymbol{W}^k_{t+i}, \xi^k_{t+i}) = \mathcal{T}(\nabla F_k(\boldsymbol{W}^k_{t+i}, \xi^k_{t+i})) \tag{11}$$

$$\boldsymbol{W}^k_{t+i+1} \leftarrow \boldsymbol{W}^k_{t+i} - \eta_{t+i}\nabla F'_k(\boldsymbol{W}^k_{t+i}, \xi^k_{t+i}), \tag{12}$$

where η_{t+i} is the learning rate and ξ^k_{t+i} is a data sample uniformly chosen from the k-th device. $\mathcal{T}(\cdot)$ is our defense scheme. Finally, the server averages the local models of the selected K devices and updates the global model as follows:

$$\boldsymbol{W}_{t+I} \leftarrow \frac{N}{K}\sum_{k\in\mathbb{S}_t} p_k \boldsymbol{W}^k_{t+I}. \tag{13}$$

5.2. Convergence Analysis

Our convergence analysis is inspired by [15]. Without loss of generality, we derive the convergence guarantee by applying our defense to a single layer. However, our results can be naturally generalized to multiple layers. We denote the input representation, parameters, and output of a single (e.g. s-th) layer in the k-th device and in the t-th round as r^k_t, $\boldsymbol{w}_{s^k_t}$ and b^k_t, respectively.

Before presenting our theoretical results, we first make the following Assumptions 2-5 same as [15] and an extra Assumption 6 on bounding the squared norm of stochastic gradients with respect to the single s-th layer.

Assumption 2. *$F_1, F_2, ..., F_N$ are L-smooth: $\forall \boldsymbol{V}, \boldsymbol{W}$, $F_k(\boldsymbol{V}) \leq F_k(\boldsymbol{W}) + (\boldsymbol{V} - \boldsymbol{W})^T\nabla F_k(\boldsymbol{W}) + \frac{L}{2}||\boldsymbol{V} - \boldsymbol{W}||_2^2$.*

Assumption 3. *$F_1, F_2, ..., F_N$ are μ-strongly convex: $\forall \boldsymbol{V}, \boldsymbol{W}$, $F_k(\boldsymbol{V}) \geq F_k(\boldsymbol{W}) + (\boldsymbol{V} - \boldsymbol{W})^T\nabla F_k(\boldsymbol{W}) + \frac{\mu}{2}||\boldsymbol{V} - \boldsymbol{W}||_2^2$.*

Assumption 4. *Let ξ^k_t be sampled from the k-th device's local data uniformly at random. The variance of stochastic gradients in each device is bounded: $\mathbb{E}||\nabla F_k(\boldsymbol{W}^k_t, \xi^k_t) - \nabla F_k(\boldsymbol{W}^k_t)||^2 \leq \sigma_k^2$ for $k = 1, ..., N$.*

Assumption 5. *The expected squared norm of stochastic gradients is uniformly bounded, i.e., $\mathbb{E}||\nabla F_k(\boldsymbol{W}^k_t, \xi^k_t)||^2 \leq G^2$ for all $k = 1, ..., N$ and $t = 0, ..., T-1$.*

Assumption 6. *For the single s-th layer, the squared norm of stochastic gradients on the output of each device is bounded: $||\nabla_{b^k_t} F_k(\boldsymbol{w}_{s^k_t}, \xi^k_t)||_2 \leq \Lambda_s$ for all $k = 1, ..., N$ and $t = 0, ..., T-1$.*

We define F^* and F^*_k as the minimum value of F and F_k and let $\Gamma = F^* - \sum_{k=1}^{N} p_k F^*_k$. We assume each device has I local updates and the total number of iterations is T. Then, we have the following convergence guarantee on FedAvg with our defense.

Theorem 2. *Let Assumptions 2-6 hold and $L, \mu, \sigma_k, G, \Lambda_s$ be defined therein. Choose $\kappa = \frac{L}{\mu}$, $\gamma = \max\{8\kappa, I\}$ and the learning rate $\eta_t = \frac{2}{\mu(\gamma+t)}$. Then FedAvg with our defense satisfies*

$$\mathbb{E}[F(\boldsymbol{W}_T)] - F^* \leq \frac{2\kappa}{\gamma + T}\left(\frac{Q + C}{\mu} + \frac{\mu\gamma}{2}\mathbb{E}||\boldsymbol{W}_0 - \boldsymbol{W}^*||^2\right),$$

where

$$Q = \sum_{k=1}^{N} p_k^2(\Lambda_s \cdot \epsilon + \sigma_k^2) + 6L\Gamma + 8(I-1)^2(\Lambda_s \cdot \epsilon + G^2)$$

$$C = \frac{4}{K}I^2(\Lambda_s \cdot \epsilon + G^2).$$

Proof. See our proof in Appendix D. □

6. Experiments

6.1. Experimental Setup

In our experiments, we evaluate our defense against two different model inversion attacks under non-IID settings. Experiments are conducted on a server with two Intel Xeon E5-2687W CPUs and four Nvidia TITAN RTX GPUs.

Attack methods. We evaluate our defense method against two model inversion attacks in FL. (1) **DLG attack [28]** assumes that a malicious server aims to reconstruct devices' data using their uploaded gradients. In *DLG* attack, the server optimizes reconstructed data to minimize the Euclidean distance between the raw gradients and the gradients that are generated by the reconstructed data in back propagation. (2) **Gradient Similarity (GS) attack [7]** shares the similar idea with *DLG*. Different from using Euclidean distance in DLG, GS attack utilizes cosine similarity between the raw gradients and the dummy gradients to optimize the reconstructed data during local updates.

Defense baselines. We compare our proposed defense with two existing defense methods: (1) **Gradient compression (GC)** [28] prunes gradients that are below a threshold magnitude, such that only a part of local updates will be communicated between devices and the server. (2) **Differential privacy (DP)** [17] protects privacy with a theoretical guarantee by injecting noise to the gradients uploaded to the server. In the experiments, we separately apply Gaussian and Laplacian noise to develop two DP baselines, i.e., DP-Gaussian and DP-Laplace.

Datasets. To evaluate our defense under more realistic FL settings, we use MNIST and CIFAR10 datasets and construct non-IID datasets by following the configurations in [16]. For each dataset, the data is distributed across 100 devices. Each device holds 2 random classes of data with 100 samples per class. By default, we perform training on CIFAR10 and MNIST non-IID dataset with 1000 and 200 communication rounds, respectively.

Hyperparameter configurations. In training, we set local epoch E as 1 and batch size B as 32. We apply SGD optimizer and set the learning rate η to 0.01. In each communication round, there are 10 devices which are randomly sampled to participate in the training. For model inversion attacks, the ideal case for the adversary is that there is only one sample in each batch, where the quality of reconstructed data will be very high [28]. We evaluate our defense in such an extreme case, but it should show much better performance in other general cases (i.e., more than one sample in each batch). With regard to *DLG* attack, we apply $L-BFGS$ optimizer and conduct 300 iterations of optimization to reconstruct the raw data. For *GS* attack, we utilize Adam optimizer with a learning rate of 0.1 and report the reconstructed results after 120 iterations. The base model architectures for two attacks are presented in Appendix B. For defense, the configurations of our method and the compared baselines are displayed in Tab. 2, where p_{model} in *GC* stands for the pruning rate of the local models' gradients, p_{fc} of our method represents the pruning rate of the the fully connected layer's gradients. Regarding DP-Gaussian and DP-Laplace, we set the mean and variance of the noise distribution as 0 and σ, respectively.

Table 2. Parameter configurations of different defense methods.

Configured Parameters	DLG	GS
GC: $p_{model}(\%)$	[1, 80]	[1, 90]
DP-Gaussian: $\sigma_{Gaussian}$	[$1e^{-4}$,$1e^{-1}$]	[$1e^{-4}$,$1e^{-1}$]
DP-Laplace: $\sigma_{Laplace}$	[$1e^{-4}$,$1e^{-1}$]	[$1e^{-4}$,$1e^{-1}$]
Ours: $p_{fc}(\%)$	[1, 40]	[1, 80]

Evaluation metrics. (1) **Privacy metric (MSE):** We use the mean-square-error (MSE) between the reconstructed image and raw image to quantify the effectiveness of defenses. A smaller MSE indicates a server privacy information leakage. (2) **Utility metric (Accuracy):** We use the accuracy of the global model on the testing set to measure the effectiveness of FL algorithms (i.e., FedAvg [16]). A smaller accuracy means a less practical utility.

6.2. Defense Results: Utility-Privacy Tradeoff

We compare our defense with the baselines against the two attack methods in terms of model accuracy and MSE. Ideally, we want to maintain high model accuracy while achieving high MSE. The results are shown in figure 5.

We have the following two key observations. First, when achieving the MSE such that the reconstructed image is not recognizable by humans, our method shows no drop in accuracy while the other baselines sacrifice as high as 6% and 9% accuracy under the DLG and GS attacks, respectively.

Second, without sacrificing accuracy, our defense can achieve 160x MSE than the baseline defenses. The accuracy can be maintained by our defense until MSE being 0.8, while the baselines show significant accuracy drop with a much smaller MSE. The reason is two folded: 1) our defense does not perturb parameters in the feature extractor (i.e., convolutional layers), which preserves the descriptive power of the model; and 2) the representations embedded in the gradients that are pruned by our defense are mostly inference-irrelevant, and hence pruning these parameters would be less harmful to the global model performance.

To perceptually demonstrate the effectiveness of our defense, we also visualize the reconstructed images. We compare our defense with GC, which is the defense baseline that also utilizes pruning. To save the space, we only show the results using the *GS* attack but we have a similar observation in the *DLG* attack. Figure 6 shows the reconstructed image of a random sample in CIFAR10: the reconstructed image generated by our defense becomes unrecognizable

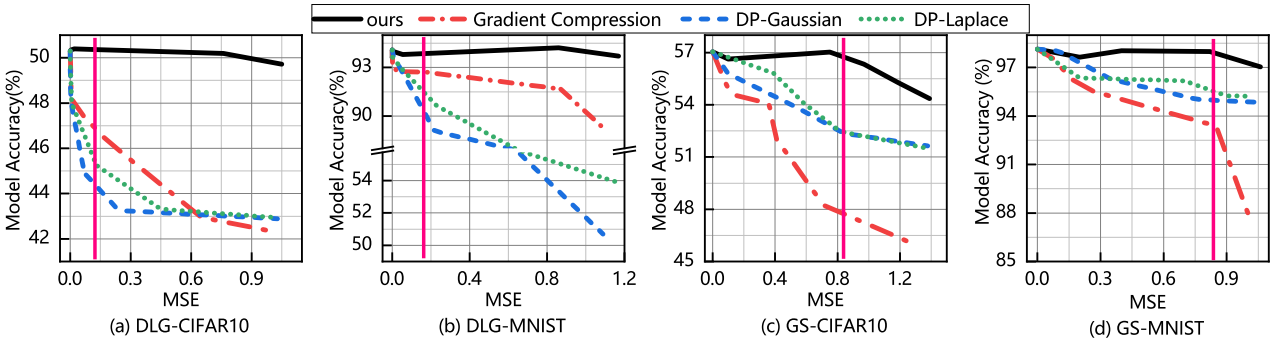

Figure 5. Compared defenses on model accuracy and MSE between reconstructed image and raw image for different attack baselines and datasets. The pink vertical line is the boundary that the reconstructed image is unrecognizable by human eyes if MSE is higher.

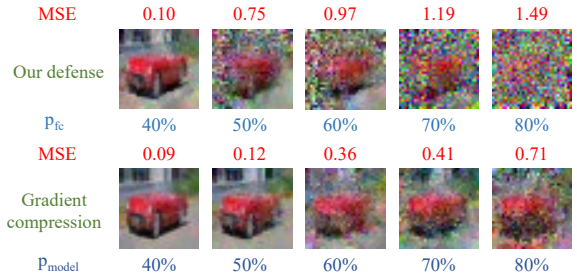

Figure 6. Comparing our defense with GC under the same pruning rate on model accuracy and MSE on a random image in CIFAR10.

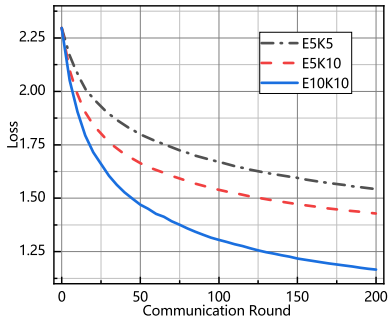

Figure 7. Convergence of LR+FedAvg with our defense.

when pruning only $50\% - 60\%$ parameters in the FC layer. However, when applying the *GC* defense, the reconstructed image is still recognizable even when 80% parameters of the whole model are pruned. Note that being unrecognizable to humans is not the ultimate goal of defense, as the private information might still reside in the image though the image is not perceptually recognizable [12]. Nonetheless, a MSE higher than the threshold that makes the image recognizable still serves as a meaningful indicator of privacy defense.

6.3. Convergence Results

Following the experimental setup in [15], we use a logistic regression (LR) to examine our convergence results on FedAvg using our defense. We distribute the MNIST dataset among $N = 100$ devices in a non-IID setting where each device contains samples of 2 digits. Here, ϵ in Equ. (4) is set to be 50, local batch size $B = 32$, local epoch $E = \{5, 10\}$, number of sampled devices K in each communication round is selected from $\{5, 10\}$.

Figure 7 shows the results of loss vs. communication rounds. We observe that LR+FedAvg with our defense converges well, which validates our theoretical analysis.

7. Conclusions and Future Work

In this work, we present our key observation that the data representation leakage from gradients is the essential cause of privacy leakage in FL. We also provide an analysis of this observation to explain how the data presentation is leaked. Based on this observation, we propose a defense against model inversion attack in FL. This is done by perturbing data representation such that the quality of the reconstructed data is severely degraded, while FL performance is maintained. In addition, we derive certified robustness guarantee to FL and convergence guarantee to FedAvg—the most popular FL algorithm, when applying our defense. We conduct extensive experiments to evaluate the effectiveness of our defense, and the results demonstrate that our proposed defense can offer a much stronger privacy guarantee without sacrificing accuracy compared with baseline defenses.

Our further research include: 1) Investigating the impact of various p-norm and q-norm on both defense and accuracy, as well as designing norms that consider structural information in the data; 2) Extending our analysis of data representation leakage to other types of layers, e.g., convolutional layer, to have a more comprehensive understanding of privacy leakage in FL.

Acknowledgements

This work was supported in part by NSF-2040588, AFRL FA8750-18-2-0057, NSF-1822085, and NSF IUCRC for ASIC memberships from Ergomotion, etc. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of AFRL, NSF and their contractors.

References

[1] Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning: Revisited and enhanced. In *International Conference on Applications and Techniques in Information Security*, 2017. 2, 3

[2] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 2015. 2

[3] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017. 1, 2

[4] Gábor Danner and Márk Jelasity. Fully distributed privacy preserving mini-batch gradient descent learning. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, 2015. 1, 2

[5] 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*, 2015. 1, 2

[6] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, 2014. 2

[7] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients–how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, 2020. 1, 2, 7

[8] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv*, 2017. 1, 2

[9] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. Learning privately from multiparty data. In *International Conference on Machine Learning*, 2016. 1, 2

[10] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv*, 2018. 1

[11] 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*, 2017. 2

[12] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, 2019. 8

[13] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4

[14] Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets. *arXiv*, 2020. 4

[15] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019. 6, 8, 11, 12

[16] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 2017. 1, 6, 7

[17] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018. 1, 2, 7

[18] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019. 1, 2, 3

[19] Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017. 1, 2

[20] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, 2019. 2

[21] Manas Pathak, Shantanu Rane, and Bhiksha Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems*, 2010. 1, 2

[22] Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Françoise Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv*, 2019. 1

[23] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, 2019. 2

[24] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015. 1, 2

[25] 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)*, 2017. 2

[26] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019. 1, 2, 3

[27] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv*, 2020. 2, 3

[28] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, 2019. 1, 2, 3, 4, 7