MemDefense: Defending against Membership Inference Attacks in Federated Learning via Adversarial Perturbation

***Abstract*—Membership inference attacks can infer whether the target data records are the training data of the target model. Especially in the training phase of federated learning, participants and servers need to share model parameters in each round of interaction, which aggravates the membership inference attacks. The existing defenses against membership inference attacks use adversarial regularization and differential privacy to protect the membership privacy of target models. Unfortunately, these defenses aim to protect the membership privacy leakage from a well-trained model without considering the privacy leakage from the shared gradients and provide poor tradeoffs between the membership privacy and the accuracy loss of target models.**

In this paper, we design MemDefense – a defense mechanism to protect the membership privacy leakage from gradients in the model sharing phase of federated learning. MemDefense adds the adversarial noises to local models trained on participants at each round of federated learning. MemDefense contains two components (noise generator, noise optimizer) to find the adversarial noise that turn the model into an adversarial model so that the adversarial model can defend against the membership inference attacks, while maintaining the high-availability of target model. Specifically, noise generator is used to find the minimized adversarial noises to reduce the attack model's accuracy to around 50%. However, the noises added to the local models will be superimposed, and the global noise increases linearly with the number of participants. The noise optimizer bounds the global noise to a constant independent of the number of participants by scaling the adversarial noises.

We evaluate MemDefense on different deep learning models and using various benchmark datasets. The experimental results show that MemDefense incurs within 0.16% accuracy loss of target model for a 50±0.4 accuracy of attack model, while the state-of-the-art defense incurs 2.5% accuracy loss for a 55±5% attack accuracy.

***Index Terms*—federated learning, membership inference attacks, defenses, deep learning.**

I. Introduction

FEDERATED learning (FL) has been emerging as a distributed learning framework to protect the privacy of participants' data by sharing local models rather than their original data [1]. It has been widely used in many application scenarios, such as predictive models for patient survival [2], medical treatments [3], and predictive keyboards on participants' smartphones [1]. In the training phase of the FL setting, the target models are shared between the participants and the server. Several studies [4]–[6] have demonstrated that the sharing of target models makes FL vulnerable to *membership inference attacks*, where an adversary can infer whether a specific data record is a member or non-member of the target models' training dataset.

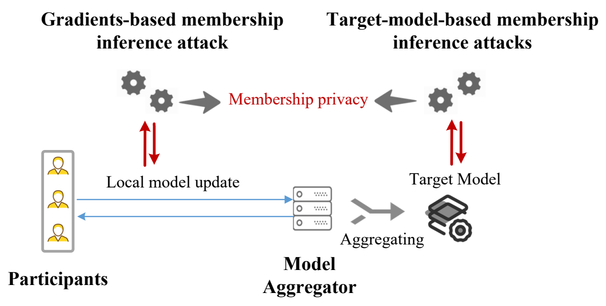

Gradients-based membership inference attack
Target-model-based membership inference attacks
Membership privacy
Local model update
Target Model
Aggregating
Participants
Model Aggregator

Fig. 1. The existing membership inference attacks in the federated setting.

Membership inference attacks expose the security risks of data privacy in both prediction and training phases of target models [4], [7]. According to the phase in which the attacks are launched, the existing membership inference attacks can be roughly classified into two categories: target model-based attacks and gradients-based attacks, as illustrated in Fig. 1.

In the *first* category, the attacks work in the prediction phase and attempt to extract membership privacy from well-trained target models. The reason for the success of these attacks is that certain features are still revealed to distinguish members from non-members in training dataset. Different prior knowledge can be employed by potential adversaries to conduct black-box attacks (e.g., using the outputs of the target models as features) and white-box attacks (e.g., using the prediction loss, the outputs of layers and the gradients as features). Many defenses have been proposed to defend against the target model-based attacks [8]–[12]. The basic idea of the existing defense mechanisms is to ensure similar outputs of the target models on member and non-member data.

In the *second* category, the adversaries execute white-box membership inference attacks using the shared gradients in the training process of target models. The intuition behind this type of attacks is that each sample in the training dataset will leave a distinguishable footprint on the gradients of the target model's loss function. The adversaries learn this feature to launch the white-box attacks. Multiple defenses [13]–[15] are proposed to protect the privacy leakage from gradients by encrypting the shared gradients. Other defenses leverage differential privacy [16]–[18] to protect the gradients.

In this paper, we focus on the membership privacy leakage from gradients in the training phase of federated learning. In summary, the existing defenses [8]–[12] resist the target model-based attacks by preventing the overfitting of target models and reducing the generalization errors, which cannot protect the shared gradients in the training phase of target models. The defenses for protecting gradients using the secure aggregation [13]–[15] and differential privacy [9], [11], [19] bring communication overhead and offer poor tradeoffs respectively. For instance, the state-of-the-art secure aggregation [14] achieves a secure aggregation overhead of $O(NlogN)$ with N users, which is a major bottleneck in large-scale federated learning. The differentially privacy model leads to the 24.5% accuracy loss of target model, while ensuring the accuracy of membership inference attack model is 51% [11]. Maintaining the time efficiency while protecting the privacy of the shared gradients in the federated settings are required. We observe that adding the perturbations to local models is an effective defense against the gradients-based attacks. However, providing the acceptable tradeoff between the membership privacy and the model utility is challenge.

We propose MemDefense to defend against the gradients-based attacks in the training process of FL, while maintaining the low accuracy overhead of target models. Specifically, MemDefense reduces the attack model's accuracy to around 50%, i.e., random guessing, while ensuring little effect on the training of target models, including the convergence rate and the accuracy of the final models. MemDefense adds well-designed adversarial noises to the local models trained on participants before sharing them to the server. In the large-scale federated learning, the adversarial noises added to the local models will be superimposed and the sum of noise increases linearly with the number of participants. Thus, we design two components: noise generator and noise optimizer to generate and optimize the adversarial noises.

The *noise generator* is used to find the minimized noise so that the accuracy of the attack model is about 50% at inferring the gradients added noise. Specifically, the defender first simulates the adversaries to train an attack model to infer the member and non-member. Then the defender generates and adjusts the generated adversarial noises using *noise generator* according to the feedback of the well-trained attack model.

The *noise optimizer* is used to solve the problem that the global noise is superposed linearly with the number of participants. Specifically, we design the scale factors to scale the adversarial noises of participants. We formulate finding a set of scale factors as solving a system of multivariate linear equations and design the algorithm for the solutions of this equation. In this way, the global noise maintains a constant level and does not increase linearly with the number of participants.

We evaluate MemDefense extensively on three real-world datasets and compare it with the state-of-the-art defenses [8], [12], [20]. Our experimental results show that MemDefense can effectively defend against membership inference attacks in the federated learning. For example, MemDefense reduces the attack model's accuracy to $50 \pm 0.4\%$, while the loss of training and testing accuracy is less than 0.1% and 0.16% respectively.

In summary, our contributions are as follows:

- We propose MemDefense to defend against the gradients-based membership inference attacks via the adversarial perturbation in the model sharing phase of federated learning.
- We design two components, i.e., noise generator and noise optimizer, to generate and optimize the generated noises. We formulate the noise optimization problem and design the optimization algorithm for weakening the accumulation of local noises.
- We evaluate MemDefense on three real-world datasets and the experimental results demonstrate that MemDefense can defend against the gradients-based attacks effectively, while maintaining low accuracy overhead of target models.

The remainder of this paper is organized as follows. We introduce the background and related work in Section II. Next, we describe the problem formulation and present the defense mechanism proposed in Section III and Section IV. We describe the experimental datasets and evaluation results in Section V and Section VI, respectively. Finally, we conclude this paper in Section VII.

II. Background and Related Work

In this section, we present the background of FL and review the existing membership inference attacks as well as the corresponding defenses.

A. *Federated Learning*

FL is a computing paradigm for distributed learning with multiple participants [1], where the participants upload local models trained over their private data, and then the server aggregates the local models and updates the global model. Table II shows the main notations used in this paper. Assuming the number of participants is N and each participant i has the private dataset D_i. The objective of each participant i is to minimize the loss function in Eq. (1).

$$L_i(\theta_i) = \frac{1}{|D_i|} \sum_{j \in D_i} l_j(\theta_i) \tag{1}$$

where θ_i is the local model of paricipant i and $l_j(\theta_i)$ is the loss function of participant i on data record (x_j, y_j). Define the global dataset as $D = \cup_{i=1}^{N} D_i$, the objective of FL is to train a global model θ to minimize the global loss function $L(\theta)$, as shown in Eq. (2).

$$L(\theta) = \frac{1}{|N|} \sum_{i \in N} \sum_{j \in D_i} \frac{L_i(\theta_i)}{|D_i|} \tag{2}$$

TABLE I
SUMMARY OF DEFENSES AGAINST MEMBER INFERENCE ATTACKS

Defense Goals	Solutions	Defense methods	Attack modes	Descriptions
Target model-based membership attacks	Shokri et al. [12]	L_2-Regularizer	White-box and black-box	Using conventional L_2 regularizer to train the target models, which leads to more than 70% attack models's accuracy with acceptable accuracy loss of target models.
	Nasr et al. [20]	Min-max game	Black-box	The adversarial regularization is used to train the target models, which incurs more than 10% accuracy loss of target models for a 50$\pm$5% attack accuracy.
	Salem et al. [8]	Dropout, Model stacking	Black-box	Using model stacking and dropout to train the target models, which provides similar tradeoff between the privacy and model utility with L_2-regularizer.
	Shejwalkar et al. [11]	Knowledge distillation	Black-box and white-box	Using knowledge distillation to train the target models, which can not protect the membership privacy leakage from shared gradients of FL.
	Jia et al. [9]	Adversarial examples	Black-box	Adding the well-designed noises to the predictions of target models, which is limited to the black-box attacks in the prediction phase.
Gradients-based membership attacks	Jinhyun et al. [14]	Secret sharing	White-box	The secure model aggregation is used to protect the local models, which brings communication overhead between the participants and the server.
	Shokri et al. [12]	Differential privacy	Black-box and white-box	Using DPSGD algorithm to train the target models, which provides poor tradeoffs between the privacy and the model utility.
	MemDefense (This paper)	Adversarial perturbations	White-box	Adding the adversarial noises to local models in the training phase of FL, which provides better tradeoff between the privacy and model utility.

B. Membership Inference Attacks

According to the different prior knowledge of adversaries, the membership inference attacks can be divided into the black-box membership inference attacks and white-box membership inference attacks [5], [21]. No matter in the scenario of black-box setting or white-box setting, the membership inference attacks require the construction of attack model, which is used to extract features from the target models, e.g., outputs, gradients, losses of layers and so on.

In the federated learning scenario, the participants and the server have access to the model's arthitectures, which means that they can perform the whitebox membership inference attacks. Consider a federated learning model θ and target data record (x, y). The goal of a membership inference attack is to infer whether (x, y) is the training data of participants.

The state-of-the-art white-box attack [4] is to train an attack model h using the target model's gradients, the loss of target model, and the outputs of model's hidden layers.

Let θ be the target model and h: $F(X, Y, \theta) \rightarrow [0, 1]$ be the attack model. In the white-box attack setting, given a data record (x, y), the attack model computes $F(X, Y, \theta)$ to infer whether the data record (x, y) is a member of training data. The input features of attack model h is a combination of different features of target model θ related to (x, y), e.g., θ's prediction on (x, y), θ's gradients on (x,y), etc, denoted by $F(X, Y, \theta)$. The output of h is the probability that (x, y) is a member of θ's training set based on the input vector $F(X, Y, \theta)$. Let $Pr_D(X, Y)$ and $Pr_{\bar{D}}(X, Y)$ be the conditional probabilities of the member and non-member, respectively. Given the above setting, the expected gain of attack model can be computed as:

$$\begin{aligned} G_\theta(h) = 0.5 \times E_{(x,y)\sim Pr_D(X,Y)}[log(h(F(x, y, \theta)))] \\ + 0.5 \times E_{(x,y)\sim Pr_{\bar{D}}(X,Y)}[log(1 - h(F(x, y, \theta)))] \end{aligned} \quad (3)$$

For the adversary, the objective of h is to maximize the privacy gain $G_\theta(h)$.

TABLE II
THE MAIN NOTATIONS USED IN THIS PAPER

Notations	Descriptions
SNR	Signal Noise Ratio of generated noise
θ	Target model of federated learning
h	The attack model
m	The number of selected participants at each iteration
T	The iterations of FL
N	The number of total participants in FL
Θ	The global model of FL
θ_i	The local model of i-th participant
θ_i^*	The noisy local model of i-th participant
$\boldsymbol{n}_i$	The noise added to local model of i-th participant
$\boldsymbol{n}_i^*$	The adversarial noise of i-th participant
γ	The scaling factor of MemDefense
γ_i	The scaling factor of i-th participant

C. Defenses against Membership Inference Attacks

Several defenses against membership inference attacks have been proposed [8]–[12], as show in Table I.

Defenses against the target model-based membership inference attacks. For the adversaries with blackbox access to the target models, overfitting is one major reason why blackbox membership inference attacks are effective [22]. Therefore, many methods are designed to reduce overfitting using regularization to protect membership privacy.

For instance, Shokri et al. [12] used conventional *L2 regularizer* in the training phase of federated learning. Nasr et al. [20] presented the machine learning with membership privacy using the min-max game with the *adversarial regularization*. Jia et al. [9] proposed a defense mechanism against the black-box membership inference attacks using *adversarial examples*. Instead of tampering the training process of the target models, the defense adds well-designed noise to each confidence score vector predicted by the target models. In addition, *dropout* [8], [23] and *model stacking* [8] have been used for preventing overfitting to defend against the blackbox membership inference attacks. Shejwalkar et al. [11] presented a new defense against whitebox and blackbox membership inference attacks using *knowledge distillation*, which trained the target models with membership privacy.

Unfortunately, the above defenses are not suitble for the membership privacy leakage in the sharing phase of FL. On the one hand, the design goal of these defenses is to protect the membership privacy of trained model by generalizing the model. However, the gradients in the model training process can still leak the membership privacy even for the well-generalization model. On the other hand, the existing defenses focus on the centralized training scenarios, whereas in the federated learning, the final model is aggregated by multiple local models iteratively. Whether using these defenses to local training will cause the unacceptable loss of global model accuracy remains to be studied.

Defenses against the gradients-based membership inference attacks. Several defenses [13]–[15] are proposed to encrypt the shared gradients using secure aggregation to protect the privacy leakage from gradients. However, these defenses bring high communication overhead (more than $O(NlogN)$ aggregation overhead) between the participants and the server.

Other defenses leverage differential privacy [16]–[18] to protect the privacy of target models. Differential privacy [24] has been widely used for privacy-preserving machine learning. In the training phase of machine learning, the defender adds differential noise to the objective function that is used to learn a model [25]–[27]. Shokri and Shmatikov [12] designed a differential privacy method for collaborative learning of deep neural networks. Abadi et al. [16] proposed the Differentially Private SGD Algorithm (DPSGD) to train the target models, which protects the data privacy in the training and prediction phase of target models. Geyer et al. [28] proposed the differentially private federated learning to ensure that a learned model does not reveal whether a client participated in the decentralized training. However, multiple studies [9], [11], [19] reveal that the differentially privacy models have a great impact on the target models' accuracy.

Novelty of proposed defense. The defense proposed in this paper aims to protect the membership privacy leakage from gradients by adding the well-designed perturbations to the local models. The proposed defense reduces the accuracy of membership inference attack models to around 50%, i.e., random guessing, while maintaining low accuracy overhead of target models. Specifically, the defense has little affect on the training of target models, including the convergence rate and the accuracy of the final model. Compared with the existing defenses, the proposed defense does not bring communication overhead, and provides better tradeoff between the membership privacy and the utility of the gradients in the federated learning scenarios.

III. Problem Formulation

In this section, we present a detailed description of threat model and design goals.

A. Threat Model

In this paper, we assume that an adversary aims to infer the membership privacy from the shared local gradients in the training phase of FL. Following the common assumptions in the literatures [4], [21], we consider the possible abilities an adversary obtains:

1) Having access to an auxiliary dataset with similar distribution to participants' private datasets, which is obviously owned by all participants of FL.
2) Obtaining the local models' gradients from participants during the training phase of FL. The server is required to aggragate the local models, which means that the server has access to the local models.
3) Knowing the structure of the target models, meaning that the adversary can launch white-box membership inference attacks.

According to different abilities the adversaries obtain, we consider three types of adversaries: *a semi-honest server* with abilities 2) and 3), *a semi-honest paricipant* with abilities 1) and 3), *a semi-honest server colludes with the semi-honest paricipants* with abilities 1), 2) and 3).

To defend against the membership inference attacks, the defender consider the attack scenarios with maximum membership privacy leakage, which means that the adversary has the above three abilities, i.e., a semi-honest server colludes with the semi-honest paricipants. If the defense mechanism can defend against the membership inference attacks by a "strong" adversary, it can effectively defend against the membership attacks by the "weaker" adversaries.

B. Design Goals

An ideal defense should defend against the membership inference attacks from the gradients, while preserving the quality of the main task models. Therefore, the design goals can be described as follows.

- *Effectiveness*. The attack accuracy represents the degree of membership privacy leakage. For the defender, the goal is to minimize the attack accuracy.
- *Time efficiency*. We evaluate the efficiency of MemDefense by comparing the time cost with and without the defense.

- *Low-accuracy overhead*. In each round of federated learning, the participants and the server are required to update the current model according to the sharing models to obtain a high-utility target model. Thus, the low-accuracy overhead of the target models are required.
- *Transferability*. In addition to membership interference attacks, MemDefense is effective defending against attacks based on gradient of federated learning, e.g., the reconstruction of data samples from the gradients in the training phase of FL.

IV. Design of MemDefense

In this section, we introduce the MemDefense that defending against the gradients-based membership inference attacks in the training of FL.

A. Overview

We present a high-level overview of MemDefense in Fig. 2. As mentioned above, the defender aims to add the "smallest noise" that can protect the membership privacy to the shared local model parameters, while maintaining the high-utility of aggregated global models of FL.

In order to find the appropriate adversarial noises, the defender first trains an attack model based on the observed local models' updates to explore the membership privacy form local gradients. Then, the defender generates the noises according to a certain distribution, i.e., Gaussian noise, Laplace noise, and adjusts the adversarial noises according to the feedback of the attack model. After that, the generated adversarial noises are added to the local models. We focus on the traditional FL with FedAvg [1] as the aggregation rule, which is widely adopted in recent studies [1], [4], [29], [30]. In this setting, the noise of aggregated global model is the sum of local noises generated by participants, which increases the accuracy overhead of global model. Therefore, we design a noise optimizer to optimize the noises added to the local models so that the aggregated global model has low accuracy overhead.

For such scenario, we design two components to defend against the gradients-based membership inference attacks: noise generator and noise optimizer.

Noise generator: generating adversarial noises that can protect membership privacy according to the attack model using the Algorithm 2 proposed. The adversarial noises are generated by a certain distribution, i.e., Gaussian noise, Laplace noise and random noise, and are adjusted by the value of Signal-Noise Ratio (SNR). The detailed noise generation algorithm is descripted below.

Noise optimizer: optimizing the generated noises to eliminate the cumulation of local noises. The objectives of noise optimizer are enabling the privacy of the local models and maintaining low accuracy overhead of aggregated global model. At each iteration of FL, each paricipant generates the adversarial noise and adds the noise to the local model to protect the membership privacy. In this way, the aggregated global noise is the sum of local noises. Therefore, we design the scaling factors for participants' local noises to achieve the above objectives. The noise optimizer is used to scale the local noises by the scaling factors before participants upload the noisy local models to the server. The detailed description about the noise components are given in the following subsections.

B. Noise Generator

In this section, we first generate the well-designed noises which are used to add to the local models trained on participants. Our goal is to find a noise vector $\boldsymbol{n}$ such that the utility loss of the target model is minimized and the accuracy of membership inference attack model h is around 50%, which means that the attack model can not determine whether a target data record is a member or not. Formally, we aim to generate the noises via solving the following optimization problem:

$$\min_{n} \ d(\theta, \theta + n) \tag{4}$$

$$s.t. \arg\min \{G(\theta + n), \theta)\} \tag{5}$$

$$|h(\theta + n) - 0.5| \leq \varepsilon \tag{6}$$

where θ is the model trained on participant's private data; the objective function Eq. (4) means that the noise added to the θ is minimized; the Eq. (5) means that the membership gain of adversary is minimized; the Eq. (6) means that defender's attack model outputs around 0.5, i.e., the attack model can not determine whether the target data record is a member or not. ε is the threshold that defenders can customize. We set it to 0.02 in this paper.

To acheive the target goals mentioned above, we design a noise generator to generate and adjust the noise vector. More details are descripted in Algorithm 1.

Algorithm 1 MemDefense Training

```
Input: Local model $\theta$, attack model $h$
Output: Noise added to local model $\boldsymbol{n}*$
1: for $t = 0$ to $T$ do
2:     Server selects $m$ paricipants and send the current global model to them
3:     for $k = 1$ to $m$ do
4:         $\theta_i \leftarrow$LocalUpdates($\theta$,$i$)
5:         /* Generating the adversarial noise */
6:         Initializing noise $\boldsymbol{n}$ by a certain distribution
7:         $\boldsymbol{n} \leftarrow Noise\ adjustment(\boldsymbol{n}, h, distribution)$
8:         /* End of generating noise */
9:         $\boldsymbol{n}_t^i = \gamma_i \boldsymbol{n}_t^i$ /* optimizing noise vector */
10:        Participants upload local update $\theta_i^t + \boldsymbol{n}_i^t$ to Server
11:    end for
12:    $\theta_t = \frac{1}{m} \sum_{i=1}^{m} \theta_i^t$
13: end for
```

Generating noise vector. (line 6-8 in Algorithm 1) In the training phase of federated learning, at each iteration t, the participant i trains the local model on their private data. Firstly, the defender generates a noise vector n according to a certain distribution, e.g., Gaussian distribution, Laplace distribution and so on. Secondly, the defender fits the local model θ_i^t to a trained attack model h and obtains the output of h. To solve

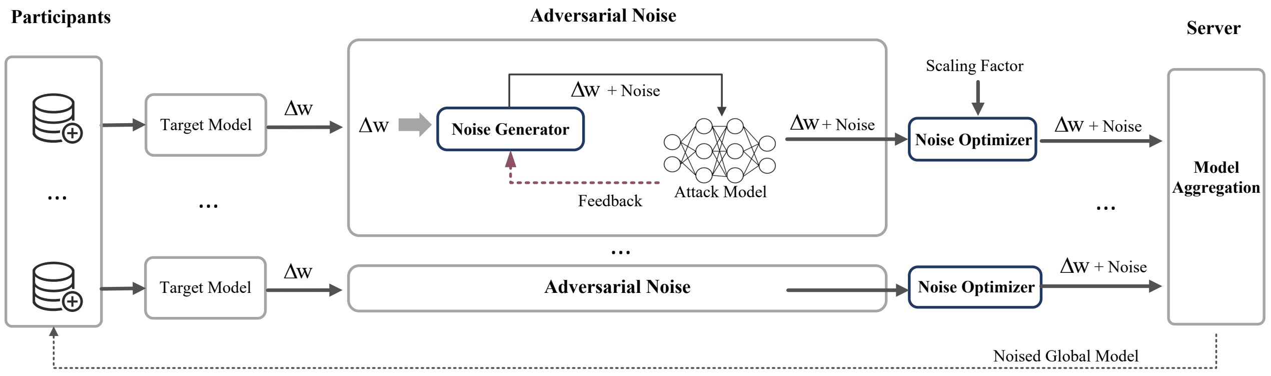


Fig. 2. The overview of the MemDefense.

the optimization problems of Eq. (5) and Eq. (6), the defender needs to adjust the generated noise vector $\boldsymbol{n}$, so that $|h(\theta + \boldsymbol{n}) - 0.5| \leq \varepsilon$. The noises are initialized according to a certain distribution and then are adjusted by a certain Signal-Noise Ratio (SNR). We set and adjust the upper and lower bounds of r iteratively to find the adversarial noises that satisfy Eq. (5) and Eq. (6). The larger the value of SNR, the smaller norm of the noises. When the output of attack model $h(\theta + \boldsymbol{n}) > 0.5 + \varepsilon$, the defender increases the noise vector as descripted in Algorithm 2. If $h(\theta + \boldsymbol{n}) < 0.5 - \varepsilon$, it means that the noise vector $\boldsymbol{n}$ is too larger, and the defender decreases the noise. The detailed noise adjustment algorithm is shown in Algorithm 2.

Methods of generating adversarial noises. We propose three methods to generate the adversarial noises. In this section, we give a detailed description of the generation of the noise vector.

Random noises. One naive method is to generate the noises randomly. It should be noted that the randomly noise vector $\boldsymbol{n}$ has the upper bound, i.e., n_{max}. In order to ensure that the generated noise is valid, we set $n_{max} = \theta_{max}$. The defender controls the noises by adjusting the upper and lower bounds iteratively. The noise vector $\boldsymbol{n}$ increases when the attack model's accuracy is more than 50%+ε, and vice versa.

Gaussian noises. Another method is to generate noises that follows Gaussian distribution. Formally, given the Gaussian noise vector $\boldsymbol{n} \sim N(\mu, \delta)$, we have:

$$f(\boldsymbol{n}) = \frac{1}{\sqrt{2\pi}\delta} e^{-\frac{n-\mu^2}{2}} \tag{7}$$

In this setting, we adjust the noise vector $\boldsymbol{n}$ by calculating the Signal-Noise Ratio (SNR). The formula of SNR is descripted as follows:

$$SNR = \frac{P_{original}}{P_{noise}} = 10\ log10\ \frac{s^2}{n^2} \tag{8}$$

where $P_{original}$ is the original signal and P_{noise} is the noise vector.

Algorithm 2 Noise adjustment

Input: The model θ, attack model h
Output: The adversarial noise $\boldsymbol{n}$

```
1:  Initialization: $min = 0, max = \theta_{max}$
2:  $\boldsymbol{n} \leftarrow generation(\theta, distribution, min, max)$
3:  for $k = 0$ to $R$ do
4:      if $h(\theta + \boldsymbol{n}) > 0.5 + \varepsilon$ then
5:          $min = \boldsymbol{n},\ max = max$
6:          $\boldsymbol{n} = randomly\ [min, max]$
7:      else
8:          $min = 0, max = min$
9:          $\boldsymbol{n} = randomly\ [min, max]$
10:     end if
11: end for
12: Return $\boldsymbol{n}$
```

In our defense mechanism, we slove the optimization function mentioned above by generating Gaussian noises with different SNR. The detailed algorithm is similar to Algorithm 2. The difference is that the min and max in Algorithm 2 represent SNR and the Gaussian noises are generated by a certain $SNR = \lfloor (min + max)/2 \rfloor$. The specific calculation of noise is shown in Eq. (9)

$$\begin{aligned} ps &= \frac{n^2}{|n|} \\ pn &= \frac{ps}{10^{\frac{SNR}{10}}} \\ noise &= N(0, \sqrt{pn}) \end{aligned} \tag{9}$$

In order to find the "minimized" noises, we dynamically adjust the upper and lower bounds of SNR to generate the noises that meet $\boldsymbol{n} \sim Gaussian(0, \sqrt{pn})$.

Laplace noises. Another method is to generate the Laplace noises $\boldsymbol{n} \sim Laplace(\lambda, \mu)$. Recently, Laplace noise has been applied in many technologies to protect data privacy. Differential privacy based on the Laplace noise mechanism has been widely used for the publication and training of machine learning models. Formally, given the noise vector $\boldsymbol{n}$ with Laplace distribution, we have:

$$f(\boldsymbol{n}) = \frac{1}{2\lambda} e^{-\frac{\lfloor n-\mu \rfloor}{\lambda}}. \tag{10}$$

Given the above definition, the two parameters of Laplace noise λ and μ can control the mean value and scale the noise vector respectively. The noise adjustment algorithm of Laplace noise is the same as the Gaussian noise.

C. Noise Optimizer

After obtaining the noise vector $\boldsymbol{n}$, the participant sends the noisy local model $\theta+\boldsymbol{n}$ to the server, and the server aggragates recieved local models to obtain the global model. It is noted that, the local noises are cumulative. If each participant adds noise to the local model, the noise of the aggregated global model is the sum of local noises, which means that the global noise is superposed linearly with the number of participants. It has an impact on the accuracy of main task models.

To slove this problem, each participant employs a scaling factor γ by restricting the generated noise vector added to the local model before uploading it to the server. It should be noted that the scaling factor γ is scalar and can be positive or negative. Then the noise optimization is formalized as finding the set of scale factors.

Optimizing noise vector. Recall the design goals of optimizing noise contains two parts.

1) *Enabling the privacy of the local models*: which means that the norm of the scaled noise by scaling factor γ_i : $\boldsymbol{n}_i^* = \gamma_i \boldsymbol{n}_i$ is larger than the original noise $\boldsymbol{n}_i$.

2) *Eliminating the accumulation of noises*: which means that the sum norm of scaled noises $\boldsymbol{n}^* = \sum_{i=1}^{m} \gamma_i \boldsymbol{n}_i$ from m local noises is approximately equal to a single noise $\boldsymbol{n}_i$.

To solve the problems described above, we design a scaling factor generation algorithm to generate a set of scaling factors $\gamma = \{\gamma_1, \gamma_2, ...\gamma_m\}$, where m is the number of participants.

Formally, $\theta_i^* = \theta_i + \boldsymbol{n}_i$ denotes the noisy local model and $\boldsymbol{n}_i$ denotes the generated noise for i-th participant. The original global model added noise Θ^* at each iteration of federated learning is calculated by the following:

$$\Theta^* = \frac{1}{m} \sum_{i=1}^{m} \theta_i^* = \frac{1}{m} \sum_{i=1}^{m} \theta_i + \boldsymbol{n}_i \tag{11}$$

We aim to generate a set of scaling factors γ that satisfy the following conditions:

$$\gamma \Theta^* = \frac{1}{m} \sum_{i=1}^{n} \theta_i + \gamma_i \boldsymbol{n}_i \tag{12}$$

$$\gamma_1 + \gamma_2 + ... + \gamma_m = 1 \tag{13}$$

$$|\gamma_i| \geq 1, \ \forall i \in [1, m] \tag{14}$$

Eq. (14), $|\gamma_i| \geq 1$ ensures that the norm of scaled local noise $|\gamma_i \boldsymbol{n}_i| \geq |\boldsymbol{n}_i|$. Meanwhile the Eq. (13) guarantees that the norm of aggregated global noise from m participants is close to the norm of noise generated by a single participant.

To find a set of scaling factor $\gamma : \{\gamma_1, \gamma_2, ..., \gamma_m\}$ that satisfy the Eq. (13) and Eq. (14), we design a symmetry algorithm to generate a set of scaling factors as descripted in the following.

Method of generating the scaling factors. As described above, the objective goal is the equations with m unknowns:

$$\begin{cases} \gamma_1 + \gamma_2 + ... + \gamma_m = 1 \\ |\gamma_i| \geq 1, \forall i \in [1, m] \end{cases} \tag{15}$$

Obviously, the system of m-ary linear equations have infinitely solutions. Because this is not the focus of this paper, we propose a naive symmetry algorithm to find a set of solutions of Eq. (15).

To solve this equations, we first decompose the equations into two sub-equations:

$$\begin{cases} \gamma_1 + \gamma_2 + ... + \gamma_{2x} = 0 \quad \forall x \in [1, \lfloor m/2 \rfloor] \\ \gamma_{2x+1} + \gamma_m = 1 \quad 2x+1 \neq m \\ \gamma_m = 1 \quad 2x+1 = m \end{cases} \tag{16}$$

Let x denotes the segmentation point $x = \lfloor m/2 \rfloor$. As shown in Eq. (16), we set $\gamma_m + \gamma_{2x+1} = 1$, if $2x+1 \neq m$, i.e., $\{\gamma_{2x+1}, \gamma_m\} \in \{0, 1\}$. The problem is reduced to finding a set of integers whose sum is 0 as described in Eq. (16).

A naive strategy is to generate a set of random numbers with size of x. Let $V = \{v_1, v_2, ...v_x\}$ denotes the random numbers, we set scaling factors $\gamma_i \in \{v_1, v_2, ...v_x\}$, $\forall i \in [1, x]$. Then, we set scaling factors $\gamma_i \in \{-v_1, -v_2, ... - v_x\}$,$\forall i \in [x+1, 2x]$ to ensure the sum of the first equation of Eq. (16) is zero, as descripted in Eq. (17).

$$\begin{cases} \gamma_i \in \{v_1, v_2, ...v_x\} \quad \forall i \in [1, x] \\ \gamma_i \in \{-v_1, -v_2, ... - v_x\} \quad \forall i \in [x+1, 2x] \end{cases} \tag{17}$$

In this way, we find a set of solutions of objective Eq. (16). The server generates the scaling factors using the symmetry-and-scaling algorithm descripted above and sends the scaling factors randomly to participants at each iteration. In the next subsection, we will give a detailed theoretical analyzes of MemDefense.

D. Security Analysis of MemDefense

In this section, we theoretically analyze the security of the generated adversarial noises. An ideal defense mechanism is to defend against the membership inference attacks from model updates of FL. MemDefense is defined as the definition Eq. (1).

Definition 1. *Formally, the generated adversarial noise is expressed as n_i for i-th participant, the number of participants is m. The optimization of adversarial noises is descripted above, which γ_i denotes the scaling factor of i-th participant. The MemDefense is defined as following:*

$$\mathcal{M}(\Theta, \gamma, \boldsymbol{n}) = \sum_{i=1}^{m} (\theta_i + \gamma_i \boldsymbol{n}_i) \tag{18}$$

where $\boldsymbol{n}_i$ denotes the noise vector with the same size as local model θ_i, and $\boldsymbol{n}_i = \mathcal{F}(SNR, distribution)$, i.e., the Gaussian noise and the Laplace noise.

We analyze the information that an adversary can get from participant's view and server's view, so as to valid the security of MemDefense. A formally security theorem is presented in Theorem 1.

Theorem 1. *Assuming $\mathcal{A}$ is the adversary who can obtain the noisy local models received from participants, θ and θ^* denote the trained model and the noisy model using MemDefense, respectively. We require the existence of an efficient simulator* $\mathsf{Sim_A}$ *such that*

$$\mathsf{View_A} \equiv_{\mathsf{c}} \mathsf{Sim_A}(\mathsf{S}, \mathsf{S}^*) \tag{19}$$

where $\equiv_c$ denotes computational indistinguishability against Probabilistic Polynomial Time (PPT) adversaries. $\mathsf{View_A}$ *denotes the adversary's view in the execution of MemDefense.* S *denotes the input to MemDefense, which includes the local models, adversarial noises, and the scaling factors.* S^* *denotes the outputs the adversary obtains from MemDefense.*

Case 1: Semi-honest participants. Assuming i-th participant is semi-honest, we use the notation $\mathcal{A}$ indicates the semi-honest participant. $\mathcal{A}$ can obtains the noisy global model aggregated from the server Θ_t^* at each iteration of FL, where Θ_t^* is computed by the definition 1. We construct an ideal simulator to interact with the adversary $\mathcal{A}$ to prove that the real world distribution obtained by the adversary is computational indistinguishable to the simulated distribution.

The simulator Sim proceeds as follows: Server sends the scaling factor γ_i and global model Θ to $\mathcal{A}$. $\mathcal{A}$ trains the local model θ and adds the adversarial noise $\boldsymbol{n}$ to θ.

The simulator Sim can obtain nothing, except the noisy local model $\theta + \gamma_i \boldsymbol{n}$ and previous noisy global model Θ. $\mathcal{A}$ obtains nothing except the noisy global model Θ, which means that the curious participant learns nothing in the real world, because that the noisy global model can effectively defend against the membership inference attacks.

Case 2: Semi-honest server and semi-honest server colludes with the paricipants. Assuming the model aggragator (server) is semi-honest, we use the notation $\mathcal{A}$ indicates the semi-honest server. Similar to the curious participant, only the noisy local models $\theta_i + \gamma_i \boldsymbol{n}_i$ $\mathcal{A}$ can obtains and the noisy local models can protect the membership privacy. This means that the sever learns "nothing more" than their own information. Thus, in this case, even the semi-honest server that colludes with the semi-honest participants can learn nothing about the participants' membership privacy from the shared models.

In the section of experiments, we evaluate the effectiveness of MemDefense as descripted above. The experimental results further prove that MemDefense can effectively defend against the gradients-based membership attacks in the training phase of federated learning.

V. Datasets and Model Arthitectures

In this section, we describe the datasets used to evaluate MemDefense and introduce the architectures of the target models and the membership inference attack models.

A. Datasets

CIFAR-100. CIFAR-100 is a image dataset which is widely used for evaluating image classification [31]. It contains 60,000 color (RGB) images, including 50,000 images for training and 10,000 images for testing. The image size is 32×32 pixels. In this dataset, the main task is to train a deep learning model which can cluster the images into 100 classes. Each class has 500 training and 100 test images.

CIFAR-10. CIFAR-10 is a image dataset and contains 60,000 color (RGB) images. The number of training images is 50,000 and the number of testing images is 10,000. The image size is 32×32 pixels. In this dataset, the main task is to train a deep model for image recognition. Each class has 5,000 training and 1,000 test images.

Purchase-100. Purchase-100 is a popular benchmark dataset used to classify the user's purchases. The Purchase-100 dataset contains the shopping recoreds of serveral thousand online customers. Each record in this dataset is the shopping history of a single customer. The dataset contains 600 different products. The label of each user is a binary record which represents whether the user bought each of products. The dataset contains 197,324 data records. The shopping records are clustered into 100 classes based on the similarity of the purchases. The main task on this dataset is to identify the class of each user's purchases.

B. Model Arthitectures

Target models. For the Purchase-100 classification task, we use the fully connected network with four hidden layers of sizes {1024, 512, 256, 128}. For the dataset CIFAR-100, we use AlexNet and DenseNet12 models. DenseNet12 corresponds to DenseNet-BC(L=100, k=12). For CIFAR-10, we use AlexNet that is widely used for image classification model. For CIFAR-100 dataset, we use SGD optimizer to train the models, and we set the learning rate for 0.001, 0.0001,0.00001 for epochs 0-100, 100-200, 200-300 accordingly. We use the ReLU as the activation function and SGD learning algorithm for all models. The batch sizes are 128 on these datasets.

Following defenses, we meansure the training accuracy of main tasks A_{train} by computing the percentage of the correct labels predicted by the target model on the training data. A_{test} represents the percentage of the correct labels predicted by the trained model. We use the generalization error E_{gen} to measure the difference of the training and testing accuracy.

Membership inference attack model. We use the state-of-the-art attack models proposed by Nasr et al. [4]. For the given data samples, we use gradients of the loss with respect to last two layers and outputs of the last two layers of the target model as the input features. The label is 1 if input data is a member of target model, and label is 0 if input data not belongs to the training data of target model. We flatten the input features into a one-dimensional vector and use a CNN with 100 kernels of size (1,100) to extract the input features. The max pooling size is (1,2). Two fully-connected layers are of size 128 and 64. We use the ReLU as activation function and Adam optimizer for all attack models. The learning rate is 0.001 and the output of attack model is a softmax layer.

TABLE III
SIZES OF DATASETS USED IN MAIN TASKS AND INFERENCE ATTACK TASKS.

Datasets	Target model		Attack model	
	Dataset member	Dataset non-member	Dataset member	Dataset non-member
Purchase-100	10,000	10,000	5,000	5,000
CIFAR-100	50,000	10,000	5,000	5,000
CIFAR-10	50,000	10,000	5,000	5,000

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the MemDefense. The experimental settings of detailed datasets and models are descripted in III.

The member and non-member are the training set and test set used in target model, respectively. In the attack model, the member and non-member data are used for training the attack model. Specifically, the adversary labels the data samples used for training target model as member and the data samples used for testing as non-member. After that, the adversary trains the attack model on the member and non-member data samples.

We have carried out several experiments on three datasets with the aims of: (i) evaluating the effectiveness of MemDefense that defending against the gradients-based membership inference attacks during the training phase of federated learning; (ii) comparing the performance of MemDefense with existing defenses, i.e., L2 regularization, adversarial regularization, differential privacy and dropout; (iii) assessing the performance of the main tasks' models with and without MemDefense; (iv) characterizing the transferability of MemDefense across the data reconstruction attacks.

A. Effectiveness of MemDefense

In this section, we show the evaluations of MemDefense. Since there is no standard metric for evaluating the quality of such a tradeoff, we use the attack accuracy as the metric to evaluate the effectiveness of MemDefense, the accuracy loss of main task's model as the metric to assess the "negligibility". Following researches [11], [20], we use the generalization error E_{gen} to measure the difference of training and test accuracy. A_{train} and A_{test} represent the accuracy of target models. A_{att} represents the membership inference risks as the accuracy of the corresponding attack models.

The detailed results are shown in Table IV. As Table IV shows, MemDefense can protect the membership privacy in the training phase of federated learning and maintain the high accuracy of main tasks' models. For instance, attack accuracy A_{att} decreases from 82% to 49% on CIFAR-100.

B. Impact on Main Task

In this subsection, we evaluate the impact of MemDefense on the main tasks of federated learning.

Accuracy. As descripted in section III, low accuracy loss of the target model is required. We evaluated the impact of generated noises added to the local models on the main tasks' accuracy and the experimental results are shown in Table IV. The experimental results show that MemDefense is "negligibility", and the training accuracy A_{train} and test accuracy A_{test} only drops within 2%.

Effectiveness. Fig. 3 shows the convergence of the losses of the main tasks with the increase of epochs in the federated setting. As expected, the performance of target models with and without Memdefense is almost the same. The training losses and training accuracy of target models varying with epochs are shown on the left and right of Fig. 3 respectively. As shown in Fig. 3, the yellow line indicates MemDefense in this paper, and the blue line indicates the model without defense, yellow line and the blue line are almost coincident.

We evaluated the time cost of generating noise added to model compared to the participant's training time cost. Under the same equipment, we set up five epochs for participants' training at each round of FL and set the batch-size to 128. In order to speed up the process of adjusting the adversarial noises, we calculate the initial adversarial noises offline. Because the membership inference attacks can be transfered [8], we first implement MemDefense on a trained model to find the noises that can defend against the membership inference attacks as the initialized noises. Table V shows the experimental results on different datasets. The experimental results show that the training time of each participant is about half of the time cost of generating adversarial noises, which demonstrates that the time cost of MemDefense is acceptable for the participants.

C. Comparison with Other Defenses

In this section, we compare MemDefense with serveral state-of-the-art regularization methods. In our defense, we focus on the white-box membership inference attacks during the learning phase of federated learning. Nasr et al. [20] proposed an adversarial regularization (Ad-reg) against the membership inference attacks. Salem et al. [8] revealed the methods of preventing model's overfitting can defend against the membership inference attacks, i.e., L2 regularization and dropout.

Comparison with defenses based on differential privacy. The existing defenses based on the differential privacy [12], [16], [28] can not protect the record level membership privacy of model updates in the training phase of federated learning. In order to compare our defense with the defense mechanism based on differential privacy, we evaluate the membership inference attack in centralized training scenario and compare the model trained using DPSGD algorithm [16] with the model trained using MemDefense. Our evaluation shows that the performance of MemDefense is better than defenses based on differential privacy. Table VI shows the results for Alexnet trained on CIFAR-10 dataset. The experiments demonstrate that the DPSGD incurs %10 loss in test accuracy of trained model, while the attack model can still achieve 63% attack

TABLE IV

THE EFFECTIVENESS OF MEMDEFENSE. A_{att} REPRESENTS THE ATTACK ACCURACY. IN ALL EXPERIMENTS, THE NOISE TYPE IS GAUSSIAN AND THE SIGNAL TO NOISE RATIO IS 15, THE ROUND OF FEDERATED LEARNING IS 100 AND THE NUMBER OF PARICIPANTS IS 100. IN THE DATASET CIFAR-100, THE DEFAULT NUMBER OF PARICIPANTS IS 20, BECAUSE OF THE LIMINATION OF IMAGES.

Experiments	Without defense				With MemDefense			
	A_{att}	A_{train}	A_{test}	E_{gen}	A_{att}	A_{train}	A_{test}	E_{gen}
P-FC	0.73	0.98	0.93	0.05	0.51	0.96	0.91	0.05
C10-A	0.79	0.95	0.82	0.13	0.53	0.95	0.81	0.14
C100-A	0.82	0.99	0.48	0.51	0.49	0.99	0.48	0.51
C100-D	0.77	0.91	0.63	0.28	0.50	0.91	0.47	0.44

TABLE V

THE EXPERIMENTAL RESULTS OF MEMDEFENSE BY COMPARING THE TIME COST WITH AND WITHOUT MEMDEFENSE. THE NUMBER OF PARICIPANTS IS 100 AND THE BATCH-SIZE IS 128.

Datasets	CIFAR-10	CIFAR-100	Purchase-100
Training time (s)	14±0.6	15±0.2	0.9 ±0.08
Defending time (s)	7±0.3	8±0.5	0.6±0.03

TABLE VI

THE EXPERIMENTAL RESULTS TRAINED ON CIFAR-10 USING MEMDEFENSE AND DPSGD. THE PARAMETERS δ, ϵ OF DPSGD ARE SET TO $1e-5$ AND 2 RESPECTIVELY. THE BATCH-SIZE AND LOT-SIZE ARE SETTED TO 16.

Defense mechanisms	A_{train}	A_{test}	A_{att}
No defense	0.95	0.82	0.79
MemDefense	0.95	0.81	0.53
DPSGD	0.94	0.72	0.63

accuracy. Compared with DPSGD algorithm, the model trained using MemDefense incurs less test accuracy (1%) than the DPSGD-trained model (10%) while the attack accuracy decreases from 63% to 53%.

Comparison with adversarial regularization. We compare the models trained using adversarial regularization and MemDefense proposed. We set the parameter of Ad-reg $\lambda = 2$, and the experimental results are shown in Fig. 3. Meanwhile, the corresponding attack models' accuracy are 50% and 59% respectively. As shown in Fig. 3, MemDefense has insignificant impact on the main tasks, and decreases the attack model's accuracy to around 50%.

Comparison with other regularization. We compare MemDefense with other regularization methods, e.g., L2 regularization and dropout to evaluate the performance of MemDefense. Fig. 3 shows the experimental results on various datasets. In all experiments, the L2 generalization factor and dropout rate are set to 1e-3 and 0.5, respectively. The accuracy of attack models is still above 60% trained using L2 regularization and dropout. The results show that MemDefense can effectively defend against the gradients-based membership inference attacks.

D. The Impact of the Noise Parameters

In this section, we evaluate the impact of noise parameters, icnluding the noise type and the SNR that are used to genreate the adversarial noises.

TABLE VII

THE IMPACT OF DIFFERENT METHODS GENERATING NOISES. IN ALL EXPERIMENTS, THE SIGNAL TO NOISE RATIO IS 15, THE ROUND OF FEDERATED LEARNING IS 100 AND THE NUMBER OF PARICIPANTS IS 100. IN THE DATASET CIFAR-100, THE DEFAULT NUMBER OF PARICIPANTS IS 20, BECAUSE OF THE LIMINATION OF IMAGES.

Dataset	Laplace Noise		Gaussian Noise		Randomly Noise	
	A_{att}	E_{gen}	A_{att}	E_{gen}	A_{att}	E_{gen}
P-FC	0.51	0.05	0.50	0.05	0.50	0.06
C10-A	0.52	0.13	0.49	0.14	0.50	0.13
C100-A	0.50	0.51	0.49	0.52	0.51	0.50
C100-D12	0.49	0.38	0.51	0.37	0.50	0.38

TABLE VIII

THE IMPACT OF DIFFERENT METHODS GENERATING NOISES OF MEMDEFENSE

Dataset	$SNR = 15$		$SNR = 20$		$SNR = 25$	
	A_{att}	E_{gen}	A_{att}	E_{gen}	A_{att}	E_{gen}
P-FC	0.49	0.04	0.50	0.05	0.52	0.05
C10-A	0.52	0.14	0.50	0.13	0.51	0.14
C100-A	0.49	0.51	0.51	0.50	0.52	0.51
C100-D12	0.54	0.38	0.51	0.39	0.50	0.41

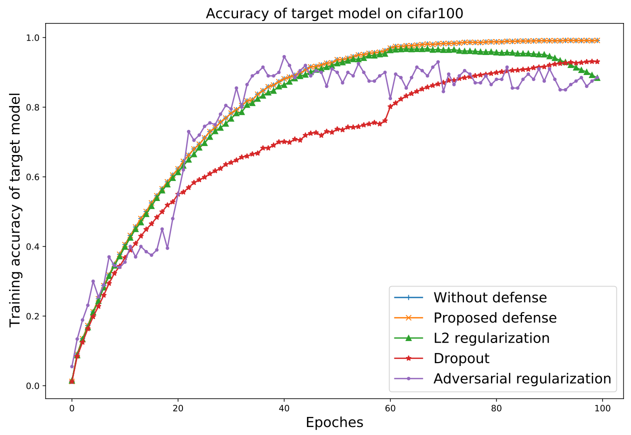
The impact of noise type. In MemDefense, we design three methods to generate the noises added to the sharing models, i.e., randomly noises, Laplace noises and Gaussian noises as descripted in Section V. We evaluate the impact of different methods generating the adversarial noises. The experimental results show that the performance of MemDefense is almost independent of noise type, as shown in Table VII.

The impact of Signal to Noise Ratio. In this subsection, we explore the effectiveness of MemDefense against the membership inference attacks with varying SNR. As expected, the smaller the SNR, the greater the noise, and the better the performance of MemDefense.

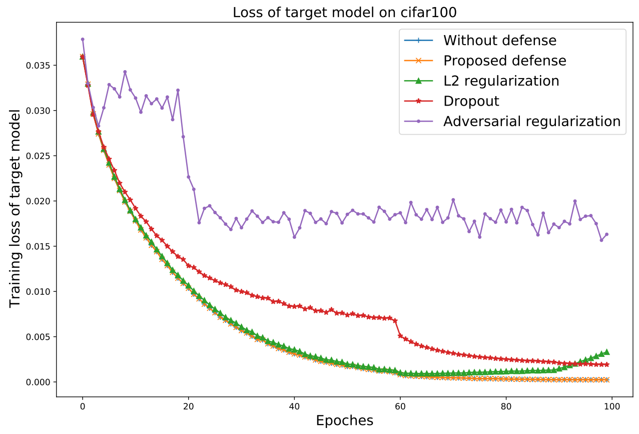
E. The Impact of the Number of Participants

In this section, we evaluate the impact of the number of participants on MemDefense.

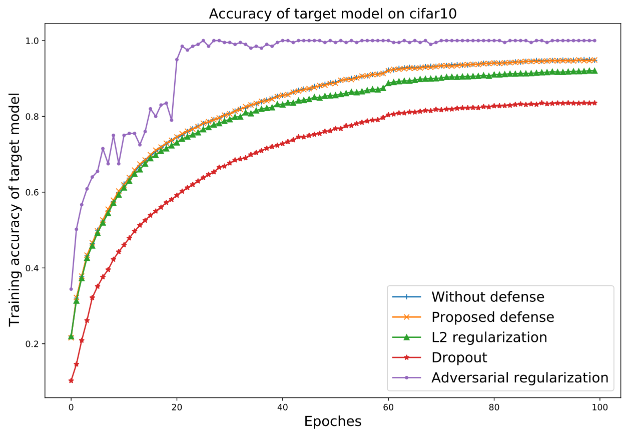
Accuracy. In this section, we show the evaluation of MemDefense with varying the number of participants. We measure the generalization error E_{gen}, the accuracy of target models A_{train}, A_{test} and the accuracy of the corresponding attack models A_{att}.



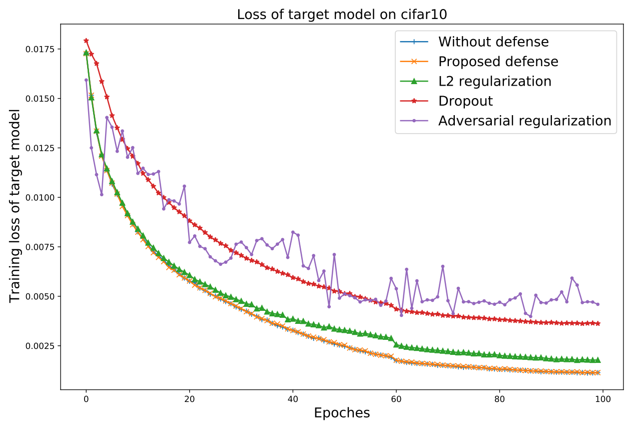
(a) The accuracy of target model on CIFAR-100



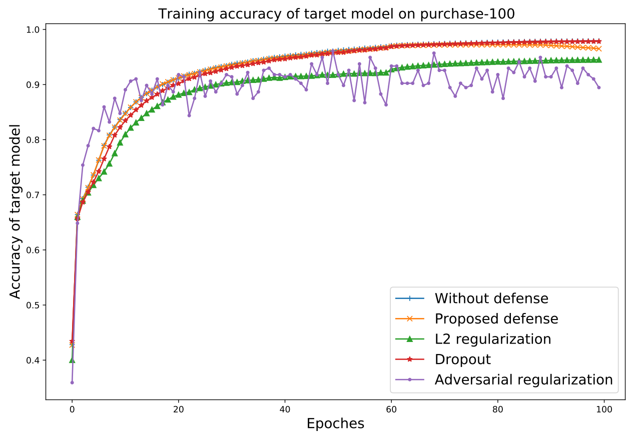
(b) The training loss of target model on CIFAR-100



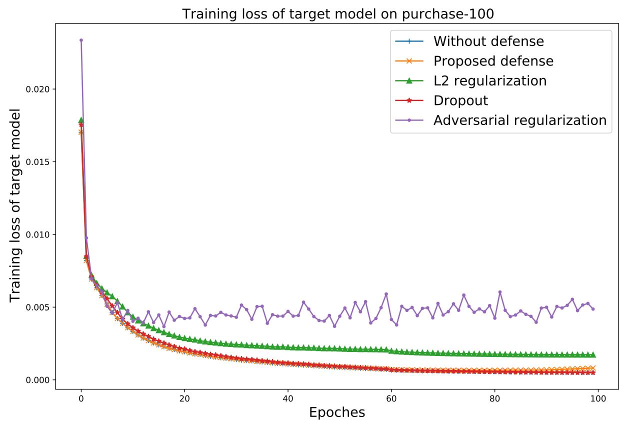
(c) The accuracy of target model on CIFAR-10



(d) The training loss of target model on CIFAR-10



(e) The accuracy of target model on Purchase-100



(f) The training loss of target model on Purchase-100

Fig. 3. The comparison with other defenses. The round of federated learning is 100 and the number of paricipants is 100.

TABLE IX
THE IMPACT OF THE NUMBER OF PARTICIPANTS.

Dataset	$N = 50$		$N = 100$		$N = 150$	
	A_{att}	E_{gen}	A_{att}	E_{gen}	A_{att}	E_{gen}
P-FC	0.51	0.06	0.52	0.07	0.51	0.05
C10-A	0.49	0.11	0.49	0.11	0.52	0.13

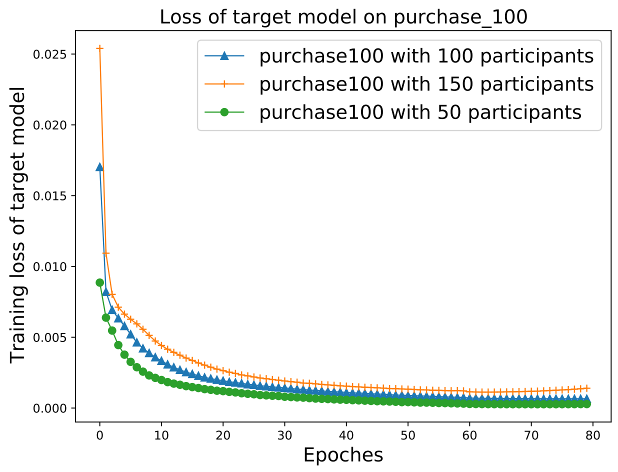


Fig. 4. The loss of target models with varying the number of paricipants. The noise type is Gaussian noise and SNR is 15.

Table IX shows the accuracy of main tasks on various datasets with varying the number of participants. The number of participants varies from 50 to 150. The accuracy of attack models reach about 50% with different numbers of participants, which is equivalent to the probability of random guess.

Effectiveness. We explore the training efficiency of the main tasks with the increase of the number of participants. In order to show intuitively the efficiency of the main tasks' models with and without MemDefense, we draw a line chart of the losses of the target models.

As the number of participants increases, the convergence rate of the global model decreases slightly. The reason behind this is that when the number of participants increases, each participant has less data samples, and the number of aggregation rounds required by the main tasks' models increases. The experimental results are descripted in Fig. 4.

F. The Transferability of MemDefense

In this section, we evaluate the transferability of MemDefense across the data reconstruction attack from model's gradients. Following research [32], we add the adversarial noises generated by MemDefense to the gradients. We characterized the effectiveness of MemDefense with different SNR, as shown in Fig. 5. In all experiments of Fig. 5, the noise type is Gaussian noise. As expected, the data reconstruction attack fails when the signal noise ratio $SNR = 15$ and the reconstructed data is the same as the initial data generated

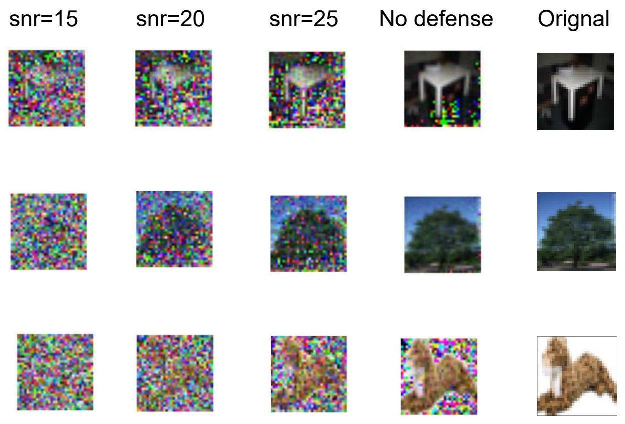


Fig. 5. The effectiveness of MemDefense on the data reconstruction attacks (the noise type is Gaussian noise in this setting).

randomly. The experimental results show that MemDefense is transferable and the generated adversarial noises can defend against the data reconstruction attacks from gradients.

VII. CONCLUSION

In this paper, we proposed MemDefense to defend against the gradients-based membership inference attacks in the sharing model phase of federated learning. Specifically, the MemDefense contains two components, noise generator and noise optimizer, which are used to generate and optimize the adversarial noises respectively to protect the membership privacy with negligible accuracy loss of target models. The experimental results demonstrated that MemDefense can effectively defend against the gradients-based membership inference attacks, while maintaining the high utility of main tasks' models.

There is still an open problem to be solved. For instance, how to accelerate the noise generation process. A future research direction is the optimization of the adversarial noise to reduce the time cost of the defense mechanism.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, ser. Proceedings of Machine Learning Research, vol. 54. PMLR, 2017, pp. 1273–1282.
[2] A. Jochems, T. M. Deist, I. El Naqa, M. Kessler, C. Mayo, J. Reeves, S. Jolly, M. Matuszak, R. Ten Haken, J. van Soest *et al.*, "Developing and validating a survival prediction model for nsclc patients through distributed learning across 3 countries," *International Journal of Radiation Oncology* Biology* Physics*, vol. 99, no. 2, pp. 344–352, 2017.
[3] A. Jochems, T. M. Deist, J. Van Soest, M. Eble, P. Bulens, P. Coucke, W. Dries, P. Lambin, and A. Dekker, "Distributed learning: developing a predictive model based on data from multiple hospitals without data leaving the hospital–a real life proof of concept," *Radiotherapy and Oncology*, vol. 121, no. 3, pp. 459–467, 2016.
[4] M. Nasr, R. Shokri, and A. 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, San Francisco, CA, USA, May 19-23, 2019*, pp. 739–753.

[5] Y. Long, V. Bindschaedler, L. Wang, D. Bu, X. Wang, H. Tang, C. A. Gunter, and K. Chen, "Understanding membership inferences on well-generalized learning models," *arXiv preprint arXiv:1802.04889*, 2018.

[6] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pp. 691–706.

[7] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 587–601.

[8] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019.

[9] J. Jia, A. Salem, M. Backes, Y. Zhang, and N. Z. Gong, "Memguard: Defending against black-box membership inference attacks via adversarial examples," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*. ACM, 2019, pp. 259–274.

[10] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, D. Lie, M. Mannan, M. Backes, and X. Wang, Eds. ACM, 2018, pp. 634–646. [Online]. Available: https://doi.org/10.1145/3243734.3243855

[11] V. Shejwalkar and A. Houmansadr, "Reconciling utility and membership privacy via knowledge distillation," *CoRR*, vol. abs/1906.06589, 2019. [Online]. Available: http://arxiv.org/abs/1906.06589

[12] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[13] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 911–926, 2020.

[14] J. So, B. Guler, and A. S. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 167, 2020.

[15] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. ACM, 2017, pp. 1175–1191.

[16] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, Eds. ACM, 2016, pp. 308–318.

[17] M. Pathak, S. Rane, and B. Raj, "Multiparty differential privacy via aggregation of locally trained classifiers," in *Advances in Neural Information Processing Systems*, 2010, pp. 1876–1884.

[18] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013, Austin, TX, USA, December 3-5, 2013*. IEEE, 2013, pp. 245–248.

[19] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*, N. Heninger and P. Traynor, Eds. USENIX Association, 2019, pp. 1895–1912.

[20] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 634–646.

[21] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 3–18.

[22] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 2018, pp. 268–282.

[23] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2670313

[24] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[25] D. Kifer, A. D. Smith, and A. Thakurta, "Private convex optimization for empirical risk minimization with applications to high-dimensional regression," in *COLT 2012 - The 25th Annual Conference on Learning Theory, June 25-27, 2012, Edinburgh, Scotland*, ser. JMLR Proceedings, S. Mannor, N. Srebro, and R. C. Williamson, Eds., vol. 23. JMLR.org, 2012, pp. 25.1–25.40. [Online]. Available: http://proceedings.mlr.press/v23/kifer12/kifer12.pdf

[26] R. Iyengar, J. P. Near, D. Song, O. Thakkar, A. Thakurta, and L. Wang, "Towards practical differentially private convex optimization," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 2019, pp. 299–316. [Online]. Available: https://doi.org/10.1109/SP.2019.00001

[27] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?id=2021036

[28] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017.

[29] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1175–1191.

[30] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[31] A. Krizhevsky and G. Hinton, "Learning multiple layers of features fro tiny images," 2009.

[32] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 14 747–14 756. [Online]. Available: http://papers.nips.cc/paper/9617-deep-leakage-from-gradients