CMI: Client-Targeted Membership Inference in Federated Learning

Tianhang Zheng, and Baochun Li, *Fellow, IEEE,*

Abstract—Membership inference is a popular benchmark attack to evaluate the privacy risk of a machine learning model or a learning scheme. However, in federated learning, membership inference is still under-explored due to several issues. For instance, some assumptions in prior works may not be practical in federated learning. Most existing membership inference methods stand on those impractical assumptions or lack generalization ability, which may misestimate the privacy risk. To address these issues, we propose CMI, an attack framework armed by a targeted poisoning method, to conduct a critical evaluation of client-targeted membership inference in federated learning. Under CMI, we consider a strong adversary, refine the prior impractical assumptions, and apply simple but generalizable attack methods. The evaluation results on multiple datasets demonstrate the efficacy of CMI under identically independently distributed (i.i.d.) and non-i.i.d. settings. In terms of the defenses, although differetially private stochatic gradient descent (DP-SGD) is effective under the i.i.d. setting, it does not provide satisfactory protection under label-biased non-i.i.d. settings. Thus, we propose RR-Label, a modified random response algorithm, to defend against membership inference. Compared to DP-SGD and Random Response Top-k (RRTop-k), RR-Label enables a better trade-off between model utility and defensive performance under label-biased non-i.i.d. settings.

Index Terms—Federated Learning, Client-Targeted Membership Inference

1 Introduction

In the recent decade, deep learning has achieved monumental success in many applications with the assistance of big data. Meanwhile, the rapid growth of this data-demanding technique naturally increases data privacy concerns. To alleviate these concerns, the community has developed decentralized learning schemes, such as federated learning [1] and split learning [2], to keep the clients' private data on their local devices. In federated learning, the clients only need to share their locally trained models with the server to aggregate a high-performance global model. Each client's data is unseen to the server and the other clients. Therefore, federated learning is considered effective to prevent data leakage.

But prevention of direct data leakage does not indicate safety against advanced privacy attacks. Owing to the contributions of recent works [3], [4], [5], [6], federated learning is revealed to be vulnerable to membership inference, a benchmark attack for privacy risk evaluation. However, most prior works either stand on some impractical assumptions and requirements or apply a complicated but not generalizable inference method. Specifically, Nasr et al.'s attack [3] trains an auto-encoder on representations, gradients, model outputs, loss, and labels for membership inference, which significantly increases the complexity of the inference model and thus may suffer from overfitting and poor generalization on complicated datasets and networks. The attack in [4] assumes the server to be a passive adversary, which may underestimate the privacy risk in federated learning. Pichler et al.'s attack [5] needs to embed a special structure into the model architecture for membership inference, making the attack very easy to detect. Gu et al.'s attack [6] requires the server to have a large auxiliary dataset to train shadow models and the attack model. But in federated learning, it may not be easy for the server to obtain a large amount of private data. Moreover, Gu et al.'s attack [6] assumes the active adversary to select the targeted client in all the rounds of federated learning, which is almost impossible in practice, according to our analysis in Section 4.3.

To evaluate membership inference in federated learning under relatively more practical settings, we revise several misleading or impractical assumptions in the previous literature. Specifically, we assume an active adversary who can poison the model to improve the performance of membership inference. We assume that the server can select the targeted client for a reasonable number of rounds instead of all the rounds. We consider that the adversary only has a *small/medium-sized attack dataset* since most client data is private and thus may not be accessible to the server. Also, beyond i.i.d. settings, we consider non-i.i.d. settings where the attack dataset may not be distributed similarly to the targeted client's dataset. With these revised assumptions, we propose a new framework, called CMI, for client-targeted membership inference in federated learning. CMI considers a malicious server as the potential adversary and targets at attacking any client who participates in federated learning. Under CMI, the malicious server first selects the targeted client for a reasonable number of attack rounds (*i.e.*, $[Rp + 2\sqrt{Rp(1-p)}]$ in Section 4.3). In each attack round, the server poisons the model with our proposed targeted poisoning method, which misleads the model to predict the

- *T. Zheng is with Division of Computing, Analytics, and Mathematics, University of Missouri-Kansas City, Kansas City, Missouri, 64112, USA. E-mail: tzheng@umkc.edu.*
- *B. Li is with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ontario, M5S 1A1, Canada. Email: bli@ece.toronto.edu.*

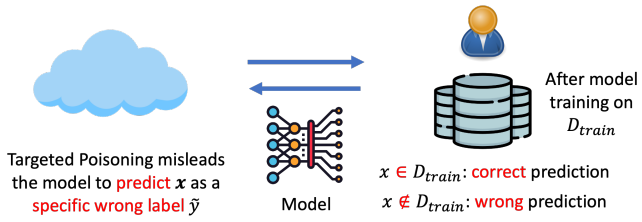


Fig. 1: The key idea of CMI: We propose targeted poisoned to mislead the model to predict the data that server attempts to infer as specific wrong labels. The model training process on the client side will correct the predictions of member data, which separates member and non-member data.

attack data samples as their most probable wrong labels. The server then sends the poisoned model to the targeted client. After the client optimizes the model on its training dataset, the model predictions of the member data in the attack dataset will be corrected as the true labels, while the model predictions of most non-member data still point to the wrong labels due to the poisoning effect. As a result, our targeted poisoning method increases the separation between the privacy metrics of member and non-member data (Fig. 5). Moreover, our targeted poisoning method can accumulate its poisoning effect since the most probable wrong labels remain the same for most attack data in the last several attack rounds.

Following the final attack round, CMI computes the privacy metrics of the attack data on the model from the targeted client. After that, CMI considers two methods for membership inference. The first method is to collect privacy metrics with membership labels by training shadow models on the attack dataset, and set sample-wise thresholds to distinguish between the privacy metrics of member and non-member data. Instead of using Song et al. [7]'s thresholding scheme, we employ support vector machine (SVM) to identify the sample-wise thresholds with larger margins so that the thresholds could be more tolerant to deviations. The first method is suitable for the i.i.d. setting where the distribution of the server's attack dataset is similar to the distribution of the client data so that the shadow models behave similarly to the targeted model. The second method is to cluster the privacy metrics of the attack data into member and non-member groups by spectral clustering with nearest neighbor affinity matrices. The clustering-based method is more effective under non-i.i.d. settings, where the attack dataset may not support the server to train shadow models with similar behavior as the targeted client's model.

With respect to the defenses, Naseri et al. [8] observed that local DP-SGD [9] is an effective defense against membership inference under the i.i.d. setting. This observation is also verified by our evaluation results. But under non-i.i.d. settings, especially when the targeted client only has data from a subset of classes, DP-SGD fails to provide satisfactory protection. This is because DP-SGD does not randomize labels; thus, DP-SGD still trains the local model on a label-biased distribution, leading to not low membership inference accuracy (See Section 6.1 for details). To address the label bias issue, we propose RR-Label, a modified random response method, to randomly flip the labels from the majority label group into minority labels. We show that RR-Label enables a much better trade-off between model utility and defensive performance against membership inference than DP-SGD [9] and RRTop-k [10] under the label-biased non-i.i.d. settings.

We conduct extensive evaluations on commonly-used datasets for membership inference evaluation, including Purchase-100 [11], Texas-100 [12], CIFAR-10, and CIFAR-100 [13]. We demonstrate that our targeted poisoning method consistently improves the attack performance across varied datasets, networks, and experimental settings. In Table 2, we show that, if the clients do not apply defenses, CMI (Shadow + TP) could achieve over 90% attack accuracy. In Table 4, we show that our proposed RR-Label successfully defends against CMI under the label-biased non-i.i.d. setting, with superior performance than DP-SGD and RRTop-k.

Our contributions are summarized as follows:

1) We revise some assumptions in the previous literature to formulate a more practical threat model for evaluating membership inference in federated learning.
2) We propose a new framework, called CMI, with a targeted poisoning method to increase separation between member and non-member privacy metrics and two inference methods designed for i.i.d. and non-i.i.d. settings.
3) We observe the unsatisfactory performance of DP-SGD under label-biased non-i.i.d. settings, and we propose RR-Label to provide better protection.
4) We conduct extensive evaluations and highlight some takeaways summarized from the evaluation results.

We organize the remainder of the paper as follows: We first introduce the preliminary knowledge and related work in Section 2. In Section 4, we formulate the threat model with our revised assumptions. In Section 5, we present our targeted poisoning method and CMI's technical details. In Section 6, we present potential defenses against membership inference in federated learning including our proposed RR-Label. We conduct extensive evaluations in Section 7 and conclude the paper in Section 8.

2 Preliminaries

2.1 Definitions and Notations

In this paper, we mainly focus on classification problems in federated learning. We denote a data sample by $\boldsymbol{x}$ and its label by y. We denote the label set by $\mathcal{Y}_K = \{1, 2, ..., K\}$ with totally K labels. We denote a neural network by $\boldsymbol{f_\theta}(\cdot)$ with model parameters $\boldsymbol{\theta}$. $\boldsymbol{f_\theta}(\boldsymbol{x})$ refers to the softmax output of $\boldsymbol{x}$, and $\ell(\boldsymbol{f_\theta}(\boldsymbol{x}), y)$ refers to the cross-entropy between $\boldsymbol{f_\theta}(\boldsymbol{x})$ and y. i.i.d. is the abbreviation of independent and identically distributed. In terms of the hyperparameters of federated learning, we denote the total number of rounds by R and the total number of clients by M. We denote the number of selected clients in each round by m. ***We follow the prior works on membership inference in federated learning [3], [6] to compute membership inference accuracy***

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3346692

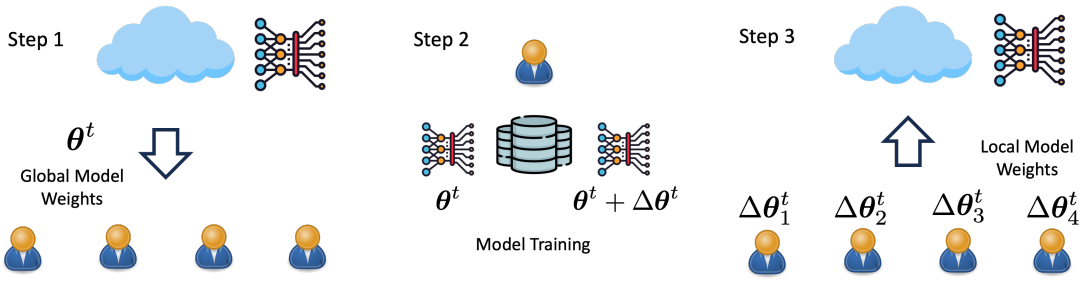


Fig. 2: The pipeline of each round of Federated Learning.

(attack accuracy) for evaluation and comparison. Future work can try to combine CMI and the likelihood-based method in [14] to compute False Positive Rate (FPR) for further evaluation.

2.2 Federated Learning

Federated Learning (FL) is a decentralized learning technique for data privacy protection [1], [15]. Typical FL involves a server and a number of clients to learn a high-performance model through multiple-round training and communication. As illustrated in Fig. 2, in each round of normal FL, the server first selects several clients and sends the current global model weights to the selected clients. After that, the selected clients optimize the received model weights on their local training datasets for several epochs, and send the updated local model weights back to the server. Finally, the server aggregates the local models to update the global model and starts a new round. A commonly-used method for model aggregation is FedAvg [16], which utilizes the average of model updates from the selected clients as the update for the global model. To faciliate the progress of federated learning, the community has implemented several open-source federated learning platforms, such as FATE [17], FedScale [18], Plato [19], and Flute [20]. Since FL keeps the clients' data on their local devices, it could prevent direct data leakage.

2.3 Membership Inference Attacks

Membership inference is a benchmark attack for privacy risk evaluation. In general, membership inference aims to infer whether certain data samples, referred to as attack data in this paper, belong to the training dataset of a model. A commonly-used threat model in most prior works assumes that the adversary only has access to the outputs of the targeted model. Under this assumption, some prior works propose to collect training model outputs with membership labels from shadow models, in order to train a neural network (NN) to label the targeted model outputs [21]. When the adversary has full access to the model, Nasr et al. [3] proposed to train an NN (auto-encoder) on the representations, gradients, model outputs, loss, and labels for membership inference. Although the idea of training NNs for inference is intuitive, the NN-based inference models are very complicated and thus may suffer sub-optimal performance or poor generalization due to inappropriate hyperparameter settings [7]. To avoid this issue, some recent works [7], [14], [22] propose non-NN based privacy metrics, which are computed on the targeted model outputs, for membership inference. We list those privacy metrics in Table 1. The intuition of using those privacy metrics for membership inference is: If the model is trained on certain data samples, then the cross-entropy (CE) or other metrics should be low for those samples. By empirical evaluations or theoretical analysis, previous works [7], [14], [22], [23] demonstrate that metric-based attacks can achieve comparable or better attack performance than NN-based attacks.

2.4 Membership Inference Defenses

Some recent works [7], [24], [25] also focus on defenses against membership inference. Here we roughly divide the existing defenses into two categories, *i.e.*, training-stage and inference-stage defenses. Adversarial regularization [24] is a typical training-stage defense, which employs an adversarial network [26] to infer the membership labels in the training stage, and updates the protected model to minimize the original loss and deceive the adversarial network. MemGuard [25] is a typical inference-stage defense, which applies adversarial perturbation to the model outputs to deceive the attack models. Nevertheless, most existing defenses are designed for the centralized scenario and may not be suitable for federated learning. For instance, adversarial regularization requires the defender to have additional reference data and computational resources. Moreover, competition between the adversarial network and the protected model could not converge with local training for a few epochs in each round. MemGuard is also not suitable for federated learning since the adversary can generate the unperturbed model outputs with the access to the model. The currently most appropriate defense for federated learning is local DP-SGD, also referred to as local differential privacy (LDP) in [8]. Therefore, we study local DP-SGD in this paper.

3 Related Work

The threat of membership inference was first explored by [21] with the assumption that the adversary only has access to the outputs of the targeted model. Following [21], Nasr et al. [3] studied membership inference under the white-box setting where the adversary has full access to the model [3]. Sablayrolles et al. [23] proved that representations and gradients used in [3] do not really provide useful information in addition to the model outputs for optimal membership inference. Pichler et al. [5] proposed to embed

a special structure into the model for membership inference in federated learning. Gu et al. [6] proposed to train a neural network on the prediction confidence series for membership inference in federated learning. [7], [14], [22] proposed some non-NN metrics computed on the model outputs, which are detailed in Section 5.2, for membership inference. In terms of defenses, Nasr et al. [3] proposed to improve model resistance against membership inference by competition with an adversarial network. Jia et al. [25] proposed to add adversarial noise on the model outputs to mislead the attack models. Naseri et al. [8] conducted extensive evaluations and demonstrated that local DP-SGD is effective in defending against membership inference in federated learning under the i.i.d. setting.

4 Threat Model

4.1 Adversary's Goal

In this paper, the potential adversary is a malicious server, similar to [3], [8]. The adversary's goal is to infer whether the attack data samples belong to the targeted client's training dataset (so-called client-targeted membership inference). A malicious client may also conduct membership inference attacks, referred to as local membership inference in the previous literature [3], [6]. ***Here we do not study the circumstance that the adversary is a malicious client, because a malicious client only receives the aggregated model and thus cannot conduct membership inference attacks on a targeted client.***

4.2 Adversary's Knowledge

We assume that the malicious server has access to the targeted client's local model weights. Even if the clients try to protect their model weights by secure multi-party computation, the server still can create some fake clients to obtain the targeted client's local model weights (See Section 6.3). The malicious server is likely to know the optimizer used by the clients, which means there is high probability that the malicious server knows the optimizer. This is because the clients should use the same optimizer for consistency; otherwise the global model may not converge. Since the server is the one who can communicate with all the clients, it is convenient for the server to set up the optimizer and share it with all the clients. Even if the optimizer is not set up by the server, the server may create a fake client to obtain the shared optimizer. In this paper, we consider the case that server knows the optimizer and the case that the server does not know the server. The server may also know whether the client uses a defense since it is usually proposed in the protocol beforehand. Even if the knowledge is not given in the protocol, the server still can try to infer whether a client uses defenses by the accuracy of the client's local model on the attack dataset. If the model accuracy does not increase or only increases a little bit in each round at the beginning of the learning process, then it is likely that the client uses certain defenses. In rare cases, if the server does not know the optimizer and whether the client applies a defense, it still can use spectral clustering for membership inference under CMI.

4.3 Adversary's Capabilities

One main capability of a server in federated learning is control over model aggregation, which means the malicious server can send a poisoned model to the clients. To be more specific, instead of aggregating the local models from the selected clients, the malicious server poisons the local model from the targeted client and then sends the poisoned model to the targeted client in each attack round*, as illustrated in Fig. 3. *This attack process is similar to the isolating attack process in [3], [8].* For the other clients or beyond the attack rounds, the server can execute normal model aggregation and send the real global model. As a result, in the whole attack process, the local model will only memorize the targeted client's data, and the poisoning method makes the model leak more information for membership inference. Moreover, we revise two assumptions in the prior works [3], [6], [8].

4.3.1 Reasonable Attack Rounds

Some prior works [3], [6], [8] assume that the malicious server could select the targeted client in all rounds of federated learning. This assumption is unrealistic since it is *almost impossible* with extremely low probability in practice. If we consider client selection as a Bernoulli sampling process: The probability of selecting each client in each round is $p = m/M$, where M is total number of the clients, and m is the number of selected clients per round. Given the total number of rounds R, a client will be selected for $n \sim \mathcal{B}(R, p)$ rounds, where $\mathcal{B}(R, p)$ is a Binomial distribution. *We then have the probability that the server selects the targeted client for all the rounds is* p^{-R}. *If we set* $p = 0.1$ *and* $R = 100$, *the probability of all-round selection is* 10^{-100}. If the malicious server insists on selecting the targeted client for all the rounds, the targeted client will be almost sure that the server is malicious.

A natural question to ask is—what is a reasonable number of attack rounds to select the targeted client? Given that the mean and standard deviation of n are $\mu(n) = Rp$ and $\sigma(n) = \sqrt{Rp(1-p)}$, our answer is $[Rp + 2\sqrt{Rp(1-p)}]$ (*i.e.,*$[\mu(n) + 2\sigma(n)]$). When Rp and $R(1-p)$ are greater than 5, we could approximate $\mathcal{B}(R, p)$ by Gaussian distribution $\mathcal{N}(Rp, \sqrt{Rp(1-p)})$. Thus, the probability that the server selects a client for greater than or equal to $[Rp + 2\sqrt{Rp(1-p)}]$ (*i.e.,* $\mu(n) + 2\sigma(n)$) rounds is approximately 2%. If there are 100 clients, the probability that *at least* one client is selected for greater than or equal to $[Rp + 2\sqrt{Rp(1-p)}]$ rounds is approximately 87%. Therefore, $[Rp + 2\sqrt{Rp(1-p)}]$ is a reasonable number.

4.3.2 Attack Dataset

Some prior works [6], [7], [21] assume that the server has a large auxiliary dataset with similar distribution as the target client's training data. *However, in federated learning, this assumption may not hold since the server may not have access to many* ***private*** *data records.* Moreover, under non-i.i.d. settings, it is very likely that the distribution of server's attack dataset is *not* similar to the distribution of the targeted client's dataset. Therefore, in this paper, we consider i.i.d. and non-i.i.d. settings. We also consider the

*. In this paper, attack round refers to the round that the malicious server selects the targeted client.

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3346692

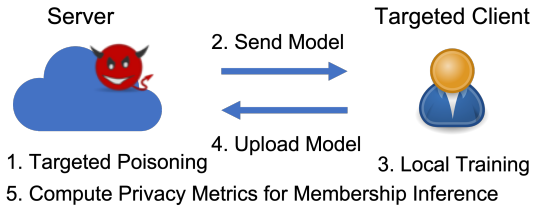


Fig. 3: Five steps for client-targeted membership inference in federated learning (CMI). The malicious client selects the targeted client for $[Rp+2\sqrt{Rp(1-p)}]$ rounds to repeat $1 \sim 4$ in each round and finally conduct membership inference.

circumstance that the server only has a small/medium-sized attack dataset. Under the i.i.d. setting, the adversary could train shadow models to set sample-wise thresholds for membership prediction. Under non-i.i.d. settings, the adversary could use spectral clustering to unsupervisedly cluster the privacy metrics for membership prediction.

5 Client-Targeted Membership Inference

In this section, we introduce our membership inference attack framework, called CMI. As illustrated in Fig. 3, in each attack round, the malicious server first poisons the model by our targeted poisoning method. The server then sends the model to the targeted client. The client updates the model on its local training dataset and sends the model back to the server. The server selects the targeted client for $[Rp + 2\sqrt{Rp(1-p)}]$ rounds and repeats the above procedure. Finally, the server computes the privacy metrics of the attack data on the model and infers the membership of the attack data by the shadow model-based method or unsupervised clustering. *The entire attack process is similar to the isolating attack process in [8] but with a different poisoning method and technical details of the inference methods.* We first introduce our targeted poisoning method in the following.

5.1 Targeted Poisoning

We propose a targeted poisoning method to improve the performance of membership inference. The basic idea of targeted poisoning is deceiving the model to predict the attack data as certain wrong labels. With targeted poisoning, we attempt to achieve the effect shown in Fig. 4—For the member data in the attack dataset, the model predictions will be corrected as true labels after the targeted client optimizes the model on its training dataset (including the member data) with true labels. For the non-member data in the attack dataset, most of the model predictions will still be wrong since the targeted client does *not* optimize the model on the non-member samples to correct their predictions. *As a result, targeted poisoning can increase the separation between the privacy metrics of member and non-member data, as indicated by Fig. 5.*

The main objective of our targeted poisoning method is

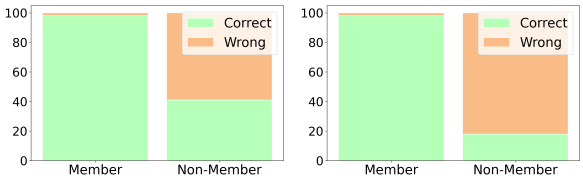
$$\min_{\boldsymbol{\theta}} \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}), \tilde{y}), \tag{1}$$

where $\tilde{\boldsymbol{y}}$ refers to the wrong labels. *Instead of using random labels, we propose to employ the most probable wrong labels for the current model as the wrong labels to magnify the poisoning effect, i.e.,*

$$\tilde{y} = \underset{k \neq y}{\operatorname{argmax}}\, \boldsymbol{f}_{\boldsymbol{\theta},k}(\boldsymbol{x}). \tag{2}$$

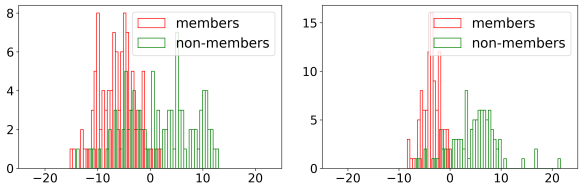
To realize targeted poisoning in practice, the malicious server can optimize the objective in (1) with the same optimizer† used by the clients for $\lfloor E/2 \rfloor$ epochs in each round. E refers to the number of local epochs. The server can start targeted poisoning from the $[R_{attack}/3]$-th round as the local model training process becomes more stable than the beginning, where R_{attack} is the number of attack rounds.

Compared to gradient ascent proposed by [3], one advantage of targeted poisoning is that targeted poisoning can accumulate its poisoning effect: For instance, in the last five attack rounds on Purchase-100, $\tilde{y}$ remains the same for over 95% of the attack data $\boldsymbol{x}$. Therefore, our targeted poisoning method deceives the model to predict $\boldsymbol{x}$ as the same $\tilde{y}$ for most attack data across different rounds. *Also, gradient ascent maximizes the loss and thus may suffer from* ***loss explosion*** *in some extreme cases, while targeted poisoning does not have this issue.* As shown in Table 2, target poisoning (TP) consistently improves membership inference and outperforms gradient ascent (GA) in most cases.



(a) No Poisoning (b) Targeted Poisoning

Fig. 4: Results on CIFAR10: Targeted poisoning deceives the model to predict the attack data as wrong labels. The predictions of member data in the attack dataset will be corrected as the true labels by local training, but most predictions of the non-member data will not be corrected since the targeted client does not trains the model on the non-member data.



(a) No Poisoning (b) Targeted Poisoning

Fig. 5: The histograms of SCL on CIFAR-10.

The readers may have a concern: Targeted poisoning decreases the model accuracy, so the attack may be easy to detect. But we note that, even if there is no attack, it is still common that the accuracy of the client's received global model is lower than the accuracy of the client's local model

†. In most circumstances, all the clients should use the same optimizer for consistency. Otherwise, the global model may fail to converge. Thus, even if the server does not know the optimizer, it still can create a fake client to obtain the shared optimizer.

Metric	Expression
PE	$-\log \boldsymbol{f}_{\text{argmax}\, \boldsymbol{f}(\boldsymbol{x})}(\boldsymbol{x})$
CE	$-\log \boldsymbol{f}_y(\boldsymbol{x})$
Mentr	$-(1-\boldsymbol{f}_y(\boldsymbol{x}))\log \boldsymbol{f}_y(\boldsymbol{x}) - \sum_{i\neq y} \boldsymbol{f}_i(\boldsymbol{x}) \log(1-\boldsymbol{f}_i(\boldsymbol{x}))$
SCL	$\log((1-\boldsymbol{f}_y(\boldsymbol{x}))/\boldsymbol{f}_y(\boldsymbol{x}))$

TABLE 1: Expressions of privacy metrics. *For SCL, we use the opposite of the definition in [14].*

(only) on this client's own data. Furthermore, in practical federated learning, model accuracy may decrease due to other factors. The decrease caused by targeted poisoning depends on the ratio between the attack dataset and the targeted client's training dataset. If victim dataset is larger than the attack dataset, then the decrease in accuracy is relatively mild. Otherwise, the victim may observe significant decrease in the accuracy. Therefore, if the server attempts to infer the membership of a large number of data samples in one federated learning process, we do not recommend the server to use targeted poisoning, which may increase the chance for the victim to detect the attack. In addition, we note that gradient ascent in [3] also decreases the model accuracy but is still considered a valid attack method that can be leveraged by an active adversary (malicious server).

5.2 Privacy Metrics

We provide the expressions of some privacy metrics studied in the recent works [7], [14] in Table 1, including prediction entropy (PE), cross-entropy (CE), Mentr [7], and scaled logit (SCL) [14]. ***For SCL, we use the opposite of the definition in [14] for consistency with the other privacy metrics***‡**.** We do not consider to use prediction entropy under CMI, because both correct and wrong predictions with high confidence lead to small prediction entropy [7]. As shown in Table 1, Mentr seems more informative since both CE and SCL are the functions of $\boldsymbol{f}_y(\boldsymbol{x})$, while Mentr contains the information of $\boldsymbol{f}(\boldsymbol{x})$. But according to [23], if the targeted client uses the CE loss for local training, the additional information in Mentr is not helpful to optimal membership inference. Our evaluation also indicates that, when we set thresholds for membership inference, CE, Mentr, and SCL have comparable performance, *i.e.,* No one is always better than the others.

In terms of unsupervised clustering for membership inference, SCL has the overall best performance in most cases. We conjecture that this is because (1) SCL has the largest value range, *i.e.,* $(-\infty, +\infty)$, so that the member and non-member clusters are more separable according to Euclidean distance. (2) The distributions of SCL are more close to Gaussian distributions than the other metrics [14], and [27] proves the optimality of spectral clustering on the mixture of Gaussian distributions. By default, we use SCL in the experiments.

5.3 Threshold Setting or Clustering

After computing the privacy metrics, the adversary needs to predict membership labels based on those privacy metrics. There are two methods for membership prediction based on privacy metrics: (1) Shadow model based threshold setting

‡. We expect the privacy metrics of member data to be smaller.

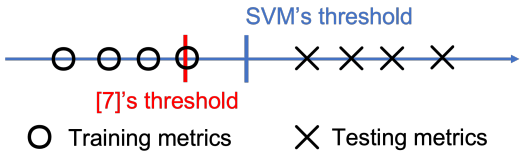


Fig. 6: An illustration of the difference between SVM-based threshold setting method and the threshold setting method in [7]. SVM's threshold is more tolerant to deviations.

(2) Unsupervised clustering. The first method is widely used in the centralized training scenario in the previous literature [7], [21], which predicts a data sample as a member if its privacy metric is smaller than a threshold. [7] proposes to set class-wise thresholds. *However, in practice, the server may not have sufficient data from all the classes with similar distribution as the client's dataset.* For instance, if the server has 200 data samples from with 100 classes, and the server split the data into 100 samples for training the shadow models and 100 samples for testing, then the probability that each class has one training sample is only $\frac{100!}{100^{100}} \approx \frac{\sqrt{200\pi}}{e^{100}}$ (according to Stirling's approximation). *Therefore, it is not practical to set class-wise thresholds when the server has limited data.*

Other choices include setting a general threshold or sample-wise thresholds. Since different data samples may have different properties, some data samples may be easy to fit, while others may not. In this sense, different data samples may have different distributions of privacy metrics [14]. Therefore, by default, we set sample-wise thresholds instead of a general threshold here. *We use support vector machine (SVM) to identify the threshold with large margins rather than use [7]'s method to set one of the privacy metrics as the threshold.* As shown in Fig. 6, the threshold set by SVM is more tolerant to deviations than the threshold set by the method in [7].

Although the first method is intuitive and effective, it has limitations in practical federated learning. One limitation is that the server needs to know the learning scheme used by the targeted client, including the optimizer, number of local epochs, and the defensive mechanism. If the server uses a different learning scheme to train the shadow models, then the attack performance of the first method could be unsatisfactory. Moreover, under non-i.i.d. settings, it is very likely that the distribution of the server's dataset is different from the distribution of the targeted client's data, so the threshold set by the attack dataset does not generalize to the victim dataset. Consequently, setting thresholds by training shadow models on the server's dataset may lead to suboptimal attack performance.

Under the above circumstances, we may need to use the second method, *i.e.,* unsupervised clustering. This is because clustering method is an unsupervised approach. Since the privacy metrics of member and non-member data are significantly separated by targeted poisoning, we could use the clustering method for membership inference. Here the idea of unsupervised clustering is to cluster the privacy metrics into two clusters, and predict the cluster with the smallest privacy metric as the member cluster. According to

our analysis in Section 5.2, we use SCL here. We follow [3] to use spectral clustering, which performs better than K-means here. We use the nearest neighbor method to construct the affinity matrix for spectral clustering, which has better empirical performance than the radial basis function (RBF) kernel method here.

6 Potential Defenses

As introduced in Section 2.4, the community has developed several defenses against membership inference, but not all of them are suitable for federated learning. For instance, MemGuard is not applicable to federated learning since the malicious server has the access to the local model. Adversarial regularization requires the clients to have additional data and computational resource for the adversarial learning process, which is not affordable for some clients in federated learning. Local DP-SGD seems to be a suitable defense, according to [8][§]. Although DP-SGD has good defensive performance under the i.i.d. setting, which is verified by [8] and our experimental results, it may not provide satisfactory protection under non-i.i.d. settings, as discussed below.

6.1 DP-SGD's Limitation under label-biased Distribution

In some cases, a client may only have data from a subset of classes. In that case, even if the client applies DP-SGD locally, the local model is still only trained on label-biased data distribution. As a result, DP-SGD may not provide satisfactory protection. Here we provide a simple example for the readers to understand the reason: Suppose that the total number of classes is K, and the targeted client only has training data from K_1 classes. After local training, the local model will tend to recognize a data sample as one of those K_1 classes. Consequently, the local model will tend to have relatively smaller privacy metrics on the member data from those K_1 classes and larger privacy metrics on the non-member data from the other $K - K_1$ classes, leading to not low membership inference accuracy. The privacy metrics of non-member data from those K_1 classes may also affect the attack accuracy, but anyway, the attack accuracy is usually not very low according to the results in Table 4.

6.2 RR-Label against Membership Inference

To defend against membership inference under the non-i.i.d. settings with label-biased data distribution, we propose to randomize the labels by a modified random response method, namely RR-Label. Ghazi et al. [10] recently proposed a random response based algorithm, called RRTop-k, to protect label privacy in deep learning. Our random response method is different from the method in [10] since our objective is to reduce membership inference accuracy. We specify RR-Label in Alg. 1 and introduce the notations in Alg. 1 in the following. Suppose that a client's data is mainly from K_1 classes ($K_1 < K$). We name the K_1 classes as the majority label group, denoted by $\mathcal{Y}_{K_1}$. We name the other classes as the minority label group, denoted by $\mathcal{Y}_K/\mathcal{Y}_{K_1}$.

§. Local differential privacy (LDP) in [8] means that the clients apply DP-SGD to local training for privacy protection.

Algorithm 1 RR-Label (for Unbalanced Non-i.i.d. Settings)

RR-Label(y):
Sample b from Bernoulli distribution $Bernoulli(q)$.
if $b < 0.5$ and $y \in \mathcal{Y}_{K_1}$ **then**
  Output a random label $y' \in \mathcal{Y}_K/\mathcal{Y}_{K_1}$
else
  Output $y' = y$
end if

The main difference between RR-Label and RRTop-k [10] is that RRTop-k ($k = K_1$[¶]) always outputs a label from $\mathcal{Y}_{K_1}$. Although RRTop-k can guarantee ϵ-DP privacy on a single label [10], it does not address the label bias issue. Since RRTop-k only trains the local model on labels from $\mathcal{Y}_{K_1}$, the RRTop-k trained model tends to recognize a data sample as the label from $\mathcal{Y}_{K_1}$. Thus, RRTop-k has the same limitation as DP-SGD here. In contrast, RR-Label always optimizes the model on labels from both $\mathcal{Y}_{K_1}$ and $\mathcal{Y}_K/\mathcal{Y}_{K_1}$. So the predictions of RR-Label trained model point to the labels from both $\mathcal{Y}_{K_1}$ and $\mathcal{Y}_K/\mathcal{Y}_{K_1}$. As indicated by results in Table 4, RRTop-k has similar defensive performance as DP-SGD since they both fail to address the label bias issue, while RR-Label reduces the membership inference to nearly the accuracy of random guess and maintains acceptable model utility.

6.3 Multi-Party Secure Computation

Secure multi-party computation (SMC) is another potential defense against client-targeted privacy attacks in federated learning. To apply SMC to federated learning. we could first set up a random vector with the same dimension as the model weights between Client i and j, denoted by $\boldsymbol{r}_{ij}$. In each round, before the n selected clients send their model updates to the server, they randomize their model weights as follows:

$$\tilde{\boldsymbol{\theta}}_i = (\boldsymbol{\theta}_i + \sum_{j=i+1}^{n} \boldsymbol{r}_{ij} - \sum_{k=1}^{i-1} \boldsymbol{r}_{ki}) \mod \Theta,$$

where Θ is a certain bound on the model weights [28]. By the shared randomness between the n selected clients, SMC seems to prevent the leakage of the clients' local model weights. *However, if the server creates a few fake clients or collaborates with some clients, and intentionally selects those clients and the targeted client in the same round, then the server will know all the $\boldsymbol{r}_{ij}$ in this round. With this trick, the server still can obtain the targeted client's local model weights.* Therefore, in federated learning, we recommend the clients to use DP-SGD or RR-Label to defend against membership inference.

7 Experiments

7.1 Experimental Setup

7.1.1 Datasets

We follow [3], [7], [8] to use Purchase-100, Texas-100, CIFAR-10, and CIFAR-100 for evaluation. For Purchase-100 and Texas-100, we randomly select 50000 samples for training

¶. The optimal k is K_1 if the client mainly has data from K_1 classes with a uniform prior, according to [10].

and 10000 samples for testing as CIFAR-10 and CIFAR-100. We randomly divide the training samples into 100 training datasets and allocate them to 100 clients. We measure the testing accuracy (of the aggregated model) on all the testing samples.

7.1.2 Networks

On Purchase-100 and Texas-100, we follow [3], [7] to use a multi-layer perception network with four hidden layers. The numbers of neurons in the hidden layers are $[1024, 512, 256, 128]$, and the activation function is Tanh function [7], [21]. On CIFAR-100, we use ResNet-18 [29].

7.1.3 Federated Learning

We set the number of clients as 100 rather than only 4 in [8]. By default, we use an SGD optimizer with momentum 0.9 and learning rate 0.01 for the clients to optimize their local models. We set the number of local epochs as 5 and the batch size as 32. The total number of rounds is set at 100. The server selects 10 clients to aggregate their local models in each round. We evaluate the non-i.i.d. setting where each client only has training data from half of the total classes, *i.e.*, $K_1 = K/2$.

7.1.4 Attack Settings

According to the settings of federated learning, the malicious server selects the targeted client for 16 rounds ($Rp = 10$ and $\sqrt{Rp(1-p)} = 3$). By default, we set the size of the attack dataset at 200, in which 100 samples are randomly sampled from the targeted client's training dataset, and 100 samples are randomly sampled from the testing dataset. Therefore, under the i.i.d. setting where the client's training dataset and the testing dataset are from the same distribution, the attack dataset and the victim dataset share a similar distribution. But under the non-i.i.d. setting where the client's training dataset and the testing dataset are not from the same distribution, the attack dataset and the victim dataset do not share a similar distribution. For targeted poisoning, we also use an SGD optimizer with momentum 0.9 and learning rate 0.01, and we optimize the objective in (1) for $\lfloor E/2 \rfloor = 2$ epochs. By default, we use SCL. For the shadow model based method, we train 256 shadow models to set thresholds. *For the attack baselines [3] and [6], the server also **only** selects the targeted client for 16 rounds, **not** all 100 rounds. We do not use [4] and [5] as baselines here, because [4]'s attack is designed for a completely different threat model, and [5] needs to embed a special structure into the neural networks.*

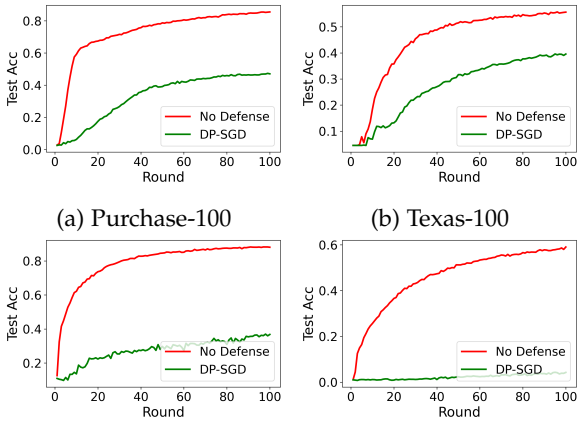
7.1.5 Defense Settings

We use Opacus [30] to implement DP-SGD in federated learning. For DP-SGD, set $\epsilon = 5$ or 10 and $\delta = 10^{-5}$ for each round of training. The maximum gradient norm is set at 1. For RR-Label, we set $q = 0.15 \sim 0.3$ for $Bernoulli(q)$ in Alg. 1. *We do not use SMC as a baseline, because SMC will be broken if the server uses a simple trick, as explained in Section 6.3.*

7.2 Empirical Results under i.i.d. Setting

7.2.1 Attack Results

We report the attack results in Table 2, where GA refers to gradient ascent, TP refers to targeted poisoning. Shadow refers to training shadow models to set sample-wise thresholds (by SVM) for membership inference. Cluster refers to spectral clustering with nearest neighbor affinity matrices for membership inference. As shown in Table 2, our proposed targeted poisoning consistently improves the attack performance and outperforms gradient ascent [3] in most cases. Notably, CMI (Shadow + TP) achieves over 90% accuracy on all the datasets and significantly outperforms [3] and [6] under the i.i.d. setting.



(a) Purchase-100 (b) Texas-100

(c) CIFAR-10 (d) CIFAR-100

Fig. 7: The test accuracy of FedAvg and FedAvg + DP-SGD *under the i.i.d. setting*. DP-SGD causes significant degradation in model accuracy in federated learning, especially on more complicated datasets such as CIFAR-100.

7.2.2 Defense Performance

We evaluate the defensive performance of local DP-SGD and provide the evaluation results in Table 3. We do not evaluate RR-Label under the i.i.d. setting because RR-Label is proposed to address the issue of DP-SGD under label-biased non-i.i.d. setting, and RR-Label is not applicable to i.i.d. settings where there is no majority label group. We confirm [8]'s observation that local DP-SGD is an effective defense against membership inference under the i.i.d. setting. *Nevertheless, DP-SGD also significantly reduces the model accuracy, which is **not** reported in [8].* We also show the evolution of testing accuracy in Fig. 7, which indicates federated learning with local DP-SGD converges slower than standard federated learning. A surprising result is that on CIFAR-100, if the clients apply DP-SGD, the global model accuracy will be very low. This is probably because CIFAR-100 is more complicated than Purchase-100 and Texas-100 and has $10\times$ more classes than CIFAR-10.

A natural question is—why [8] does not observe the severe degradation in model performance caused by DP-SGD? One reason is that [8] only considers four participants in the membership inference experiments. If we use similar settings, the testing accuracy of DP-SGD can increase by about 10% on CIFAR-100. *Another reason is that the results reported by [8] are questionable*: According to [31] (a Github project with 1.5K stars), even with centralized training and no defense, AlexNet only achieves approximately 44% testing accuracy on CIFAR-100. But [8] reports 82% testing accuracy with AlexNet on CIFAR-100. In fact, even in the centralized

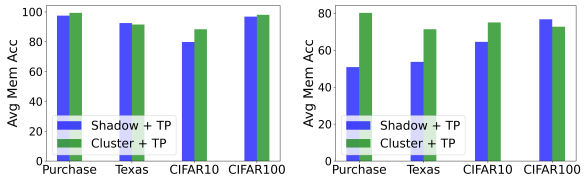
This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3346692

Attack Method		Nasr's Attack [3]	Gu's Attack [6]	Shadow	Shadow + GA	Shadow + TP	Cluster	Cluster + GA	Cluster + TP
Purchase-100	Best	96.0%	89.5%	93.0%	96.0%	99.0%	97.0%	98.5%	100%
	Avg	91.5%	88.2%	91.8%	92.8%	97.3%	96.3%	98.0%	99.8%
Texas-100	Best	87.5%	80.0%	84.5%	88.0%	91.5%	84.0%	88.5%	87.5%
	Avg	85.5%	78.0%	82.8%	87.7%	90.3%	83.0%	86.7%	86.3%
CIFAR-10	Best	65.0%	71.0%	75.0%	79.5%	93.5%	78.0%	74.0%	89.0%
	Avg	58.7%	67.3%	73.8%	78.0%	90.8%	76.0%	70.3%	86.0%
CIFAR-100	Best	82.0%	94.0%	97.5%	99.0%	99.0%	98.0%	98.5%	100%
	Avg	65.5%	89.5%	96.5%	97.3%	98.3%	85.8%	95.5%	99.5%

TABLE 2: Comparison between different attack methods under the *i.i.d.* setting: *We report the best and the averaged membership inference accuracy over three runs.* The targeted client updates its local model on 500 training samples, while the server only has 100 training samples to train the shadow models. Nevertheless, the shadow model based inference method is more suitable for the i.i.d. setting according to the results. GA: Gradient Ascent; TP: Targeted Poisoning.

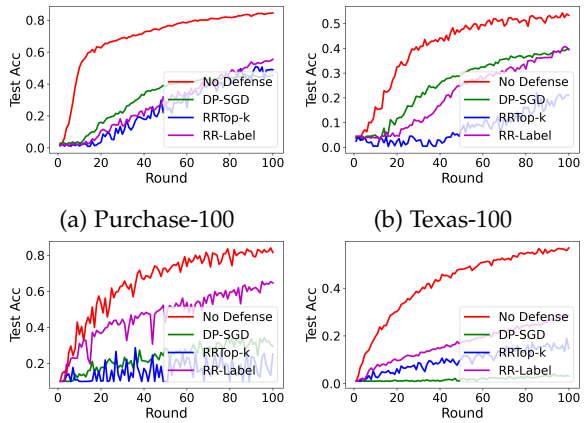
Defense Method		No Defense		DP-SGD	
		Test Acc	Mem Acc	Test Acc	Mem Acc
Purchase-100	Best	85.6%	100%	47.0%	70.5%
	Avg	85.2%	99.8%	46.8%	68.8%
Texas-100	Best	55.8%	87.5%	40.1%	64.0%
	Avg	55.7%	86.3%	39.7%	63.8%
CIFAR-10	Best	88.5%	89.0%	36.8%	57.0%
	Avg	88.2%	86.0%	36.3%	54.5%
CIFAR-100	Best	59.4%	100%	4.4%	59.0%
	Avg	59.1%	99.5%	4.3%	56.7%

TABLE 3: The defensive performance of DP-SGD under the *i.i.d.* setting. We use Cluster + TP which does not need a similarly-distributed dataset to train shadow models.



(a) No Defense (b) DP-SGD

Fig. 8: Compare the *averaged* membership inference accuracy of Shadow + TP with Cluster + TP over three runs under the non-i.i.d. setting. In most cases, the clustering method achieves better attack performance under the non-i.i.d. setting.



(a) Purchase-100 (b) Texas-100

(c) CIFAR-10 (d) CIFAR-100

Fig. 9: The test accuracy of FedAvg, FedAvg + DP-SGD, FedAvg + RRTop-k, and RR-Label *under the label-biased non-i.i.d. setting*.

scenario, we may need to use more complicated neural networks such as WRN-28-10 and ResNeXt-29 to achieve over 80% accuracy on CIFAR-100, according to [31]. Apparently, our experimental results are more convincing than [8]'s results.

All in all, we confirm that local DP-SGD is an effective defense against membership inference in federated learning. But we reveal that, in practical federated learning, DP-SGD significantly degrades model accuracy, especially on more complicated datasets such as CIFAR-10 and CIFAR-100.

7.3 Empirical Results under non-i.i.d. Setting

We report the main results in Table 4. We show that, even sacrificing a lot of model accuracy, DP-SGD [8], [9] still can not achieve satisfactory defensive performance under the non-i.i.d. setting. To be specific, under the non-i.i.d. setting, CMI achieves over 70% membership inference accuracy against DP-SGD. This is because DP-SGD does not address the label bias issue since in the training process, DP-SGD still only optimizes data pairs with labels from $\mathcal{Y}_{K_1}$ ($K_1 = K/2$).

Although RRTop-k [10] guarantees ϵ-DP on a single label, it still only optimizes the model on the data with labels from $\mathcal{Y}_{K_1}$. Therefore, RRTop-k also does not address the label bias issue and thus can not provide satisfactory protection. In contrast to DP-SGD and RRTop-k, RR-Label optimizes the model on the data with random labels from $\mathcal{Y}_K/\mathcal{Y}_{K_1}$ to address the label bias issue. As shown in Table 4, RR-Label reduces the membership inference accuracy of CMI by $35\% \sim 45\%$ with a sacrifice of $15\% \sim 30\%$ model accuracy. All in all, RR-Label achieves a much better trade-off between model accuracy and defensive performance against membership inference, compared to DP-SGD and RRTop-k.

Besides, we compare Shadow + TP and Cluster + TP under the non-i.i.d. setting in Fig. 8, which shows that the clustering method is more suitable for membership prediction here. We also plot the evolution of testing accuracy in Fig. 9, which shows that, under non-i.i.d. settings, the global model accuracy experiences more fluctuations than under the i.i.d. setting.

7.4 Ablation Studies

7.4.1 Attack Rounds

One important revised assumption is that the server only selects the targeted client for a reasonable number of rounds.

Defense Method		No Defense		DP-SGD		RRTop-k		RR-Label	
		Test Acc	Mem Acc	Test Acc	Mem Acc	Test Acc	Mem Acc	Test Acc	Mem Acc
Purchase-100	Best	84.9%	99.5%	45.7%	80.5%	50.7%	79.0%	56.1%	58.5%
	Avg	84.2%	99.3%	45.4%	80.2%	49.8%	77.3%	55.1%	56.2%
Texas-100	Best	53.3%	92.5%	40.4%	71.5%	21.6%	73.0%	40.2%	57.0%
	Avg	52.9%	91.5%	39.7%	71.3%	19.4%	72.3%	39.5%	55.3%
CIFAR-10	Best	82.5%	91.0%	35.2%	75.0%	26.2%	75.0%	65.0%	55.0%
	Avg	81.0%	88.3%	31.4%	75.0%	24.9%	75.0%	63.9%	51.8%
CIFAR-100	Best	57.0%	99.0%	3.6%	74.5%	17.4%	74.5%	29.5%	60.5%
	Avg	56.7%	98.0%	3.0%	72.7%	15.6%	73.7%	28.2%	54.8%

TABLE 4: The model performance and defensive performance of different methods under the *non-i.i.d.* setting. We use Cluster + TP here. *Test Acc refers to model testing accuracy, and Mem Acc refers to membership inference accuracy.*

Attack Rounds	$\mu + 2\sigma$	$\mu + 3\sigma$	$\mu + 4\sigma$	$\mu + 5\sigma$
Texas-100	86.3%	91.8%	95.6%	97.3%
CIFAR-10	86.0%	91.5%	92.3%	93.2%

TABLE 5: The *averaged* attack accuracy of Cluster + TP over three runs with different attack rounds.

Small Attack Dataset	P100	T100	C10	C100
Shadow + TP	100%	90%	83%	98%
Cluster + TP	100%	87%	85%	100%

TABLE 6: The *averaged* attack accuracy over three runs on a small attack dataset with 20 samples. *To save space, we use the initial plus the number of classes to denote a dataset here.*

We set the number as $[\mu(n) + 2\sigma(n)]$, according to the analysis in Section 4.3. But readers may wonder what will happen if we increase the number of attack rounds. To answer this question, we use Cluster + TP as an example and show the results in Table 5 ($\mu = 10$ and $\sigma = 3$). We only show the results on Texas-100 and CIFAR-10 because on Purchase-100 and CIFAR-100, CMI already achieves nearly 100% attack accuracy with $\mu + 2\sigma$ attack rounds. According to Table 5, as the number of attack rounds increases, it is not surprising that the membership inference accuracy also increases. Therefore, the clients should reject uploading the model if they are selected for an unreasonable number of rounds in federated learning.

7.4.2 Attack Dataset

In the previous experiments, we use a medium-sized attack dataset with 200 data samples for evaluation. Here we evaluate CMI on 20 attack samples to verify the applicability of CMI to small-sized attack datasets. As indicated by the results in Table 6, CMI can also achieve good attack results on small attack datasets. Therefore, the effectiveness of CMI does not highly depend on the data size.

7.4.3 Defense Budgets

A natural question regarding defense is—Could we improve defensive performance by tighter budgets? Here we use DP-SGD as an example. We conduct evaluations with $\epsilon = 1$ and show the results in Table 7. As indicated by Table 7, with a tighter budget, DP-SGD could reduce the attack accuracy to about 50% (approximately the accuracy of random guess). But the model accuracy also drops to an unacceptable level on all datasets. Thus, it is crucial to select an appropriate defense budget.

Tighter Budget		P100	T100	C10	C100
DP-SGD (i.i.d.)	Test Acc	2.1%	8.5%	16.6%	1.4%
	Mem Acc	48.3%	51.2%	51.0%	52.7%

TABLE 7: The *averaged* results of DP-SGD with a tighter budget ($\epsilon = 1$) over three runs.

7.5 Takeaways

We highlight some takeaways summarized from the evaluation results: (1) Even if the malicious server only has a small/medium-sized dataset and only selects the victim for limited rounds, federated learning is still vulnerable to client-targeted membership inference. (2) Although DP-SGD is an effective defense against membership inference under the i.i.d. setting, it causes significant degradation in model accuracy. (3) DP-SGD could not provide satisfactory protection when a client only has data from a subset of classes. (4) Under label-biased non-i.i.d. settings, appropriately randomizing the labels in the local training process could provide protection against membership inference.

8 Conclusions

In this paper, we provide a more practical threat model for membership inference in federated learning by revising several impractical assumptions in prior works. Under the revised threat model, we propose CMI, an attack framework with a targeted poisoning method. The targeted poisoning method could increase the separation between the privacy metrics of member and non-member data and accumulate its poisoning effect to improve the attack performance. Under CMI, we further refine some technical details such as setting the thresholds by SVM to improve the tolerance of the thresholds to deviations. Due to the limited effectiveness of DP-SGD under label-biased non-i.i.d. settings, we propose a modified random response algorithm, called RR-Label. Since RR-Label addresses the label bias issue by randomizing the majority labels into minority labels, it could provide protection against membership inference under label-biased non-i.i.d. settings. Extensive evaluations on multiple datasets verify the efficacy of our proposed attack and defense methods.

References

[1] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[2] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," *arXiv preprint arXiv:1812.00564*, 2018.

[3] 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)*. IEEE, 2019, pp. 739–753.

[4] H. Hu, Z. Salcic, L. Sun, G. Dobbie, and X. Zhang, "Source inference attacks in federated learning," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 1102–1107.

[5] G. Pichler, M. Romanelli, L. R. Vega, and P. Piantanida, "Perfectly accurate membership inference by a dishonest central server in federated learning," *arXiv preprint arXiv:2203.16463*, 2022.

[6] Y. Gu, Y. Bai, and S. Xu, "Cs-mia: Membership inference attack based on prediction confidence series in federated learning," *Journal of Information Security and Applications*, vol. 67, p. 103201, 2022.

[7] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 2615–2632.

[8] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," in *29th Annual Network and Distributed System Security Symposium, NDSS 2022*, 2022.

[9] M. Abadi, A. Chu, I. 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*, 2016, pp. 308–318.

[10] B. Ghazi, N. Golowich, R. Kumar, P. Manurangsi, and C. Zhang, "Deep learning with label differential privacy," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 131–27 145, 2021.

[11] "Acquire Valued Shoppers Challenge," https://www.kaggle.com/c/acquire-valued-shoppers-challenge.

[12] "Texas Department of State Health Services," https://www.dshs.texas.gov/THCIC/Hospitals/Download.h.

[13] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[14] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2022, pp. 1519–1519.

[15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[17] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "Fate: An industrial grade platform for collaborative learning with data protection," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 10 320–10 325, 2021.

[18] F. Lai, Y. Dai, S. S. Singapuram, J. Liu, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "FedScale: Benchmarking model and system performance of federated learning at scale," in *International Conference on Machine Learning (ICML)*, 2022.

[19] B. Li, N. Su, C. Ying, and F. Wang, "Plato: An open-source research framework for production federated learning," in *Proceedings of the ACM Turing Award Celebration Conference-China 2023*, 2023, pp. 1–2.

[20] M. Hipolito Garcia, A. Manoel, D. Madrigal Diaz, F. Mireshghallah, R. Sim, and D. Dimitriadis, "Flute: A scalable, extensible framework for high-performance federated learning simulations," *arXiv e-prints*, pp. arXiv–2203, 2022.

[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] L. Song, R. Shokri, and P. Mittal, "Privacy risks of securing machine learning models against adversarial examples," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 241–257.

[23] A. Sablayrolles, M. Douze, C. Schmid, Y. Ollivier, and H. Jégou, "White-box vs black-box: Bayes optimal strategies for membership inference," in *International Conference on Machine Learning*. PMLR, 2019, pp. 5558–5567.

[24] 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*, 2018, pp. 634–646.

[25] 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*, 2019, pp. 259–274.

[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[27] M. Löffler, A. Y. Zhang, and H. H. Zhou, "Optimality of spectral clustering in the gaussian mixture model," *The Annals of Statistics*, vol. 49, no. 5, pp. 2506–2530, 2021.

[28] V. Mugunthan, A. Polychroniadou, D. Byrd, and T. H. Balch, "Smpai: Secure multi-party computation for federated learning," in *Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services*, 2019.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[30] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao *et al.*, "Opacus: User-friendly differential privacy library in pytorch," in *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021.

[31] "Pytorch-Classification," https://github.com/bearpaw/pytorch-classification.

Tianhang Zheng received his B.S. degree from Peking University, China, in 2016, M.S. degree from University at Buffalo, NY, USA, in 2019, and Ph. D. degree from University of Toronto, ON, Canada. He is currently an assistant professor at University of Missouri-Kansas City. His research is focused on adversarial learning, data poisoning, privacy and fairness.

Baochun Li received his B.Engr. degree from the Department of Computer Science and Technology, Tsinghua University, China, in 1995 and his M.S. and Ph.D. degrees from the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, in 1997 and 2000. Since 2000, he has been with the Department of Electrical and Computer Engineering at the University of Toronto, where he is currently a Professor. He holds the Bell Canada Endowed Chair in Computer Engineering since August 2005. His research interests include cloud computing, distributed systems, datacenter networking, and wireless systems. He was the recipient of the IEEE Communications Society Leonard G. Abraham Award in the Field of Communications Systems in 2000. In 2009, he was a recipient of the Multimedia Communications Best Paper Award from the IEEE Communications Society, and a recipient of the University of Toronto McLean Award. He is a member of ACM and a Fellow of IEEE.