IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

DOI:10.1587/transfun.2024CIP0004

Publicized:2024/09/04

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER Special Section on Cryptography and Information Security

Mitigation of Membership Inference Attack by Knowledge Distillation on Federated Learning

Rei UEDA[†a)], ***Student Member***, **Tsunato NAKAI**[††], ***Nonmember***, **Kota YOSHIDA**[†††], ***and*** **Takeshi FUJINO**[†††], ***Members***

SUMMARY Federated learning (FL) is a distributed deep learning technique involving multiple clients and a server. In FL, each client individually trains a model with its own training data and sends only the model to the server. The server then aggregates the received client models to build a server model. Because each client does not share its own training data with other clients or the server, FL is considered a distributed deep learning technique with privacy protection. However, several attacks that steal information about a specific client's training data from the aggregated model on the server have been reported for FL. These include membership inference attacks (MIAs), which identify whether or not specific data was used to train a target model. MIAs have been shown to work mainly because of overfitting of the model to the training data, and mitigation techniques based on knowledge distillation have thus been proposed. Because these techniques assume a lot of training data and computational power, they are difficult to introduce simply to clients in FL. In this paper, we propose a knowledge-distillation-based defense against MIAs that is designed for application in FL. The proposed method is effective against various MIAs without requiring additional training data, in contrast to the conventional defenses.

key words: *Federated Learning, Knowledge Distillation, Membership Inference Attack*

1. Introduction

Deep learning is now being used in various fields such as medicine and transportation. In particular, centralized learning, which accumulates training data and trains a deep neural network (DNN) model on a server, is commonly used. However, there is a risk of leaking private information in centralized learning, because data holders share their own data with the server. Some organizations cannot send training data to the server because of agreements on data privacy. This is a disadvantage because DNNs generally require a large amount of data for training.

Accordingly, federated learning (FL) was proposed to solve this problem of the sharing of private information. FL is a distributed deep learning technique that comprises multiple clients and a server and does not involve sharing of the clients' training data with other clients or the server [1]. In FL, clients train a model with their own training data and send only the trained model to the server. The server then aggregates the models received from the clients. FL is considered to have a lower risk of leaking private information than centralized deep learning because the clients do not share their own training data. However, there is a growing threat of privacy attacks that steal information about the clients' training data from the aggregated model on the server.

Among these attacks are membership inference attacks (MIAs), which reveal private information in the training data from the model [2, 3]. Specifically, MIAs can identify whether or not specific data was used to train a target model. The defender must design the target models to behave similarly on training and non-training data to mitigate MIAs. Existing defenses against MIAs can be divided into provable privacy defenses and empirical membership privacy defenses, as shown in Table 1 [4]. DP-SGD is a provable privacy defense using differential privacy, but DP-SGD is shown to greatly reduce the model utility (see Section 5.3). Because of this problem of the provable privacy defenses, empirical membership privacy defenses have been researched in recent years. In the empirical membership privacy defenses, the performance of their methods is evaluated based on tradeoffs among MIA privacy risk, model utility and computational cost. Here, adversarial regularization (AdvReg) and Memguard have a problem that requires a lot of computational cost to mitigate the MIA (see Section 6.2). DMP, KCD and SELENA are a defense with the use of knowledge distillation. Here, knowledge distillation is a technique in which knowledge acquired in one model (teacher) is transferred to another model (student) [5]. Knowledge distillation can mitigate overfitting of the model to the training data. Therefore, various MIA defenses using knowledge distillation have been proposed [4, 6, 7] in recent years. In particular, KCD and SELENA show a good tradeoff between MIA privacy risk and model utility. However, KCD and SELENA require a lot of training data and computational costs (see Section 2.2.2). This is not a problem for centralized learning, where the server has a lot of training data and computational resources. However, it can be a significant problem for clients in FL. In general, the amount of training data varies between clients, and not all clients have enough data and computing resources to implement defenses based on knowledge distillation. In addition, if the clients are implemented on edge devices, they will not have extensive computing resources. Considering the situation on edge devices, the conventional empirical membership privacy defenses proposed for the centralized learning scenario are not suitable for FL.

[†]The author is with the Graduate School of Science and Engineering, Ritsumeikan University, Kusatsu-shi 525-8577, Japan.

[††]The author is with Mitsubishi Electric Corporation, Kamakura-shi 247-8501, Japan.

[†††]The author is with the Department of Science and Engineering, Ritsumeikan University, Kusatsu-shi 525-8577, Japan.

a) E-mail: ri0098kv@ed.ritsumei.ac.jp

Table 1: Two categories of the MIA defenses: provable privacy defenses and empirical membership privacy defenses [4].

	Low model utility	High model utility
Provable privacy	DP-SGD [8]	No defences yet
Empirical membership privacy	Not considered	AdvReg [9] Memguard [10], DMP [6] KCD [7], SELENA [4] **FLKD (Our proposed defense)**

In this paper, we propose a new empirical membership privacy defense that introduces knowledge distillation into FL. Our proposed defense can mitigate MIAs without requiring a lot of training data and computational cost for FL clients.

Our contributions are as follows:

- We developed a defense against MIAs by using knowledge distillation in a way that is suitable for FL. Our proposed defense does not require a lot of training data or computational cost for FL clients.
- We evaluated our proposed defense on the CIFAR100, Purchase100, and Texas100 datasets with two MIA methods: single-query and label-only attacks. The proposed defense showed better privacy protection than the conventional defenses did.

2. Preliminaries

2.1 Federated learning

Federated learning (FL) is a distributed deep learning technique in which training parties are split into multiple clients and a server [1]. Figure 1 shows an overview of FL, and Algorithm 1 gives its operation procedure. In addition, Table 2 lists the settings used in FL. The model in FL is typically trained with the following steps [11].

1. A server sends a model ω^r to each client, where R is the number of rounds. Here, ω^1 is the initial model, which the server prepares.
2. Each client independently trains the model ω^r with its own training data.
3. Each client sends its trained model ω_k^r to the server, where ω_k^r denotes the distributed model ω^r as trained with the k-th client's training data.
4. The server aggregates the trained models ω_k^r and updates its own model to ω^{r+1}.

These four steps collectively comprise a round; the accuracy of the model on the server improves with repeated rounds.

In FL, the clients do not share their training data with the server or the other clients. Therefore, FL is considered a distributed deep learning technique with higher privacy protection. The model on the server contains private information on the clients' training data, however, and it is thus exposed to the threat of membership inference attacks (MIAs) [12, 13].

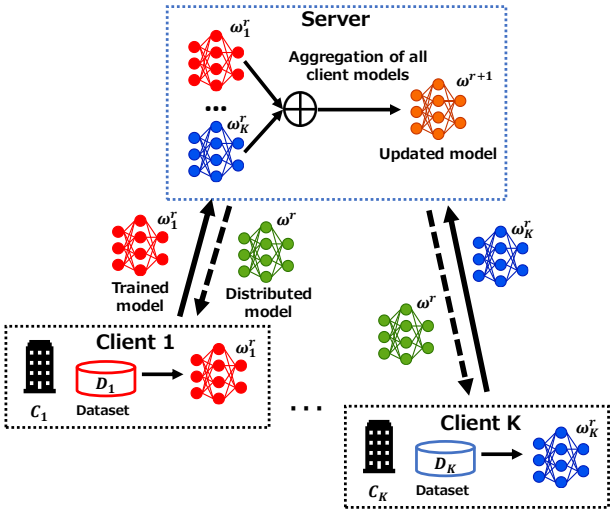


Fig. 1: Federated learning.

Table 2: Settings used in federated learning.

R	Number of rounds
K	Number of clients
D_k	Training data on k-th client
n_k	Number of training samples on the k-th client
n	Number of training samples for all clients
$\mathcal{L}_k(\omega^r, D_k)$	Loss function of k-th client's model
ω_k^r	Local model of k-th client
ω^r	Server model in r-th round

Algorithm 1 Training procedure in federated learning

Input: Training data: $D_1, D_2, ..., D_K$; Initial server model: ω^1
Output: Trained model on server: ω^{R+1}

for $r = 1$ to R **do**
  for $k = 1$ to K **do**
    ※ Clients independently train model
    $\omega_k^r = \arg\min_{\omega^r} \mathcal{L}_k(\omega^r, D_k)$
  end for
  ※ Server aggregates trained client models
  $\omega^{r+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_k^r$
end for
return ω^{R+1}

2.2 Membership inference attacks and defenses

An MIA identifies whether specific data is used for training a target model [2, 3].

2.2.1 Existing membership inference attacks

MIAs are classified into white-box attacks and black-box attacks depending on the capabilities of the attacker [2, 7].

In white-box MIAs, the attacker has the target model and can use the target model parameters, the intermediate values for the target data, and the confidence scores. In black-box MIAs, the attacker does not have the target model

and only uses the confidence scores or prediction labels of the target data.

In this paper, we assume the black-box scenario; the server and all clients in FL are trustworthy. The trained model is securely stored on a server and only accepts users' (including adversary) access through APIs. The two types of black-box MIAs are single-query attacks and label-only attacks [4].

Single-query attack: This attack is based on the confidence score and is classified into two types, namely, neural-network- (NN-) based and metric-based attacks [7].

NN-based attacks [3, 9] assume that the attacker has prior knowledge about some of the target model's training (member) and non-training (non-member) data. The attacker thus trains a classification model with this prior knowledge and uses the model for attack.

Metric-based attacks also assume prior knowledge. Specifically, the attacker computes a metric $m = M(F(x), y)$ from the confidence score $F(x)$ and hard label y of the target data from the target model F. It then decides the membership of the target data via the computed metric m and a threshold determined from the prior knowledge. Such metric-based attacks can be classified into correctness-, confidence-, entropy-, and modified entropy-based attacks.

A correctness-based attack estimates that correctly predicted target data indicates member data [4, 7, 14]. The metric I_{corr} is defined as follows:

$$I_{corr}(F(x), y) = \mathbf{1}\{\underset{i}{\operatorname{argmax}}\, F(x)_i = y\}, \tag{1}$$

where $F(x)_i$ is a class-independent confidence score.

A confidence-based attack estimates that a high confidence score indicates member data [4, 7]. Specifically, it identifies the target data as member data when the confidence score is higher than either a class-dependent threshold $\tau_{(y)}$ or a class-independent threshold τ. The metric I_{conf} is defined as follows:

$$I_{conf}(F(x), y) = \mathbf{1}\{F(x)_y \geq \tau_{(y)}\}, \tag{2}$$

where $F(x)_y$ is a class-dependent confidence score.

An entropy-based attack estimates that low entropy indicates member data [4,7]. That is, it identifies the target data as member data when the entropy is lower than either a class-dependent threshold $\tau_{(y)}$ or a class-independent threshold τ. The metric I_{entr} is defined as follows:

$$I_{entr}(F(x), y) = \mathbf{1}\{-\sum_i F(x)_i \log(F(x)_i) \leq \tau_{(y)}\}. \tag{3}$$

Lastly, a modified entropy-based attack estimates that low entropy, together with the entropy metric and correct label, indicates member data [4, 7]. Specifically, this attack identifies the target data as member data when the combined entropy is lower than either a class-dependent threshold $\tau_{(y)}$ or a class-independent threshold τ. The metric I_{Mentr} is defined as follows:

$$I_{Mentr}(F(x), y) = \mathbf{1}\{-(1 - F(x)_y)\log(F(x)_y) - \sum_{i \neq y} F(x)_i \log(1 - F(x)_i) \leq \tau_{(y)}\}. \tag{4}$$

Label-only attack: This attack is based on a boundary estimation attack [4, 15, 16] using a prediction label. The boundary estimation attack estimates that data located far from the classification boundary is member data. The attacker computes the distance from the target data to the boundary by adding small noise to the target data that is not sufficient to cause misclassification. Then, the target data is determined as member data when the distance to the boundary is larger than either a class-dependent threshold $\tau_{(y)}$ or a class-independent threshold τ. Because this attack is based on the respective distances of member and non-member data to the classification boundary, obfuscation of the confidence score does not protect against it.

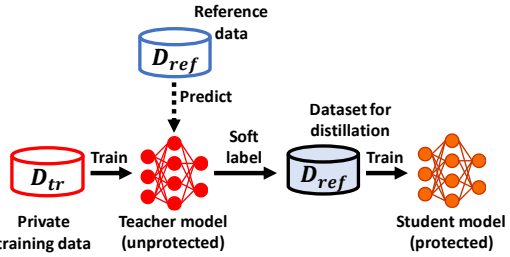


Fig. 2: DMP, a defense using knowledge distillation for MIAs.

2.2.2 Existing defenses using knowledge distillation

In this section, we introduce three existing MIA defenses that use knowledge distillation assuming centralized deep learning techniques.

First, DMP mitigates MIAs through knowledge distillation with reference data [6]. Here, knowledge distillation entails a technique to transfer knowledge acquired in one model (teacher) to another model (student). Figure 2 shows an overview of DMP. The defender trains a teacher model (unprotected) with private training data D_{tr} and builds a student model (protected) with knowledge distillation by using reference data D_{ref} and the teacher model. Here, the reference data D_{ref} is unlabeled and disjoint from the private training data D_{tr}.

Because it is difficult to prepare reference data for DMP, KCD [7] was proposed as an MIA defense to solve this problem. Specifically, it mitigates MIAs through knowledge distillation with the split training data [7]. Figure 3 shows an overview of KCD for the case of $s = 3$ divisions. The defender divides all the training data into multiple subsets and trains sub-models with each combination of the subsets. In

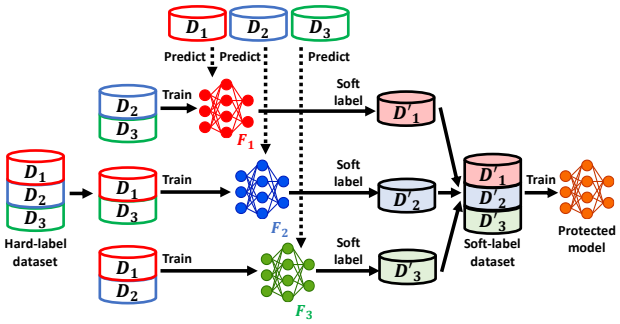


Fig. 3: KCD with $s = 3$ divisions.

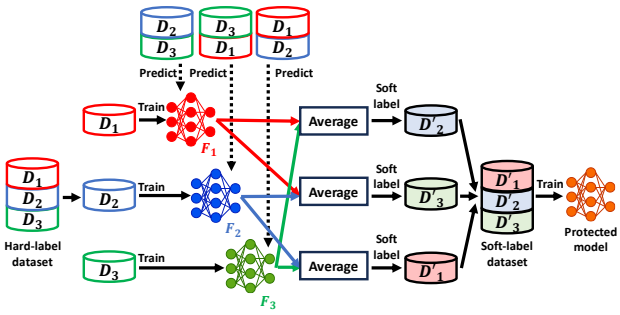


Fig. 4: SELENA with $s = 3$ divisions.

addition, it builds a protected model with knowledge distillation by using the sub-models. KCD requires a lot of training data and computational resources to achieve sufficient effects against MIAs.

Lastly, SELENA mitigates MIAs through knowledge distillation with the split training data and soft-label averaging [4]. Figure 4 shows an overview of SELENA for the case of $s = 3$ divisions. The defender divides the training data into multiple subsets and trains sub-models with each subset. In addition, it builds a protected model with knowledge distillation by using the sub-models and averaged soft labels. Like KCD, SELENA requires a lot of training data and computational resources to achieve sufficient effects against MIAs.

2.2.3 Existing defenses without knowledge distillation

Here, we introduce three existing defenses that do not use knowledge distillation assuming centralized learning.

First, DP-SGD mitigates MIAs by adding Gaussian noise $N(0, \sigma^2 C^2)$ to the gradient, where σ is a noise parameter and C is a clipping size [8]. Specifically, the defender clips the gradient according to the constant C and then adds the noise $N(0, \sigma^2 C^2)$ to the clipped gradient. A larger σ provides more privacy protection against MIAs but decreases the model accuracy.

Next, adversarial regularization (AdvReg) mitigates MIAs by training the model to reduce the discrepancy in confidence scores between member and non-member data [9]. First, the defender builds a binary classifier with the confidence scores of member and non-member data. Second, it defines a combination loss function using the binary classifier's outputs and the cross-entropy loss used in common classification models. Finally, the defender trains a protected model with the combination loss function. AdvReg determines the binary classifier's fraction on the defined loss function via a constant λ. A larger λ provides more privacy protection against MIAs but decreases the model accuracy. In addition, AdvReg entails additional computational cost for building the binary classifier.

Last, Memguard mitigates MIAs by adding noise to the target model's confidence score [10]. Because this defense obfuscates the confidence score without changing the prediction label, Memguard can mitigate MIAs without degrading the model accuracy. However, it cannot mitigate label-only MIAs, which use only the prediction label.

2.3 Application of conventional MIA defenses to FL

In this section, we discuss the application of conventional MIA defenses used in centralized learning, as described in Sections 2.2.2 and 2.2.3, to FL.

First, DMP requires reference datasets, but it is difficult for all clients to prepare such reference datasets in FL. Therefore, DMP is not a good defense against MIAs in FL. Moreover, KCD and SELENA require much training data and computational cost for training the sub-models, which is difficult for all clients to implement in FL. Therefore, KCD and SELENA also are not good MIA defenses in FL.

On the other hand, DP-SGD mitigates MIAs by simply adding noise to the gradient, which is not difficult to implement for all clients in FL.

Similarly, because AdvReg mitigates MIAs by training the model to reduce the discrepancy in confidence scores between member and non-member data, it also is not difficult for all clients to implement in FL. However, AdvReg requires building an additional binary classifier, so there are concerns in terms of its computational cost.

Lastly, Memguard mitigates MIAs by simply adding noise to the target model's confidence score. Therefore, it is easy to apply Memguard to the server model in FL, which is intended to be served to users who use the model as a service.

3. Our proposed defense

KCD and SELENA are superior defenses against MIAs in centralized learning. As explained above, however, they are difficult to apply for all clients in FL. In this paper, we propose a defense against MIAs, called federated learning knowledge distillation (FLKD), that uses applicable knowledge distillation in FL. Figure 5 shows an overview of FLKD's training procedure, which is specified in Algorithm 2. In addition, Table 3 lists the settings used in FLKD.

In FLKD, the server additionally sends distillation models $\omega_{d(\omega_k^r)}$ to the clients. This enables the clients to skip

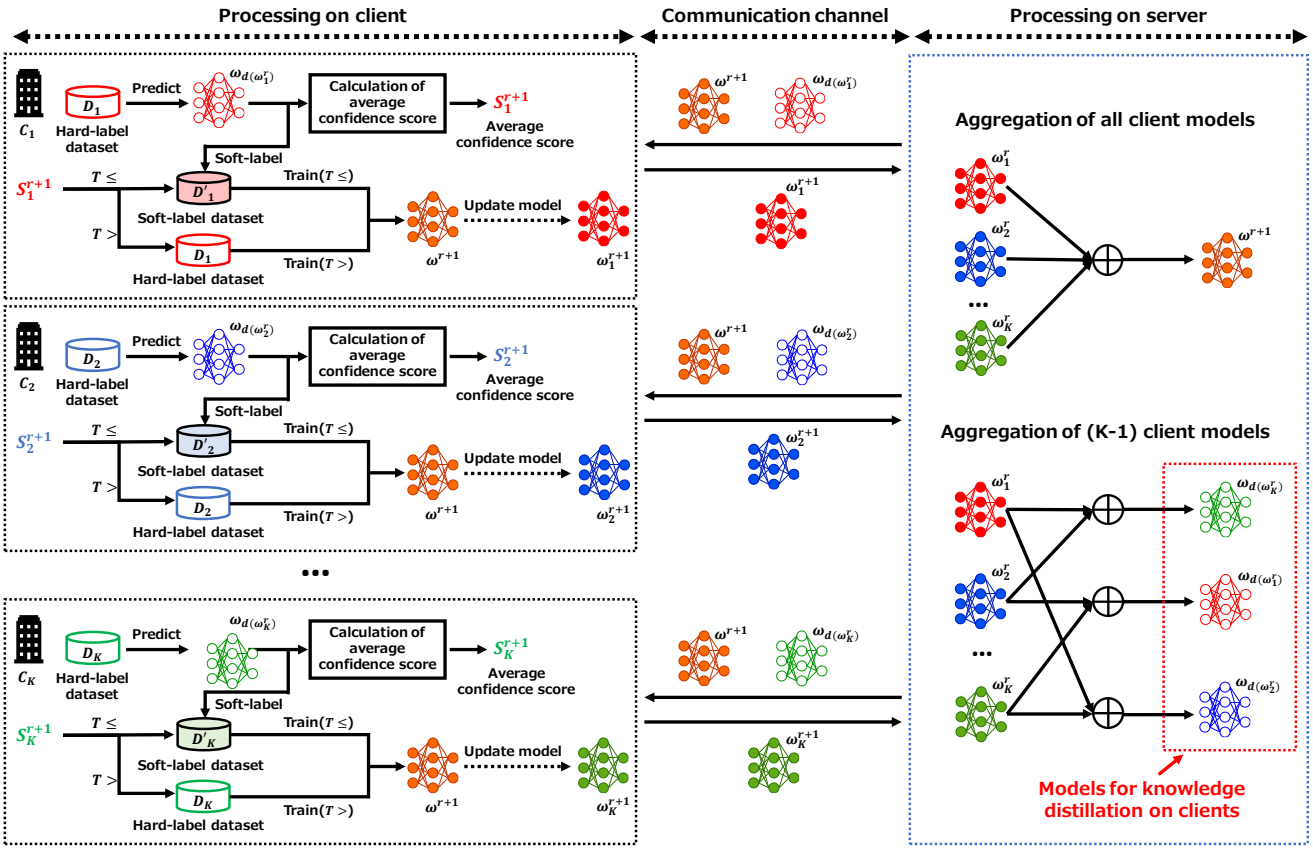


Fig. 5: Training phase of our proposed defense, FLKD.

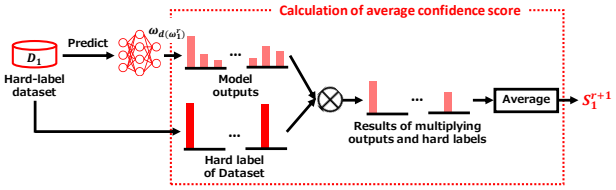


Fig. 6: Calculation of average confidence score.

Table 3: Settings used in FLKD.

ω^r	Server model in r-th round
$\omega_{d(\omega_k^r)}$	Distillation model of k-th client in $(r+1)$-th round
S_k^r	Average confidence score of k-th client in r-th round
T	Threshold for distillation
D_k	Hard-label training data of k-th client
$D_k^{'}$	Soft-label training data of k-th client

building a teacher model and focus only on student model training. FLKD comprises two phases: initialization and training. The details of each phase are given below.

Initialization phase

1. The server sends the initial server model ω^1 to the clients (Algorithm 2 L.3).
2. The clients train the received model ω^1 with their own training data and then send each trained model ω_k^1 back to the server (Algorithm 2 L.4).
3. The server calculates its next model ω^2 and a distillation model $\omega_{d(\omega_k^1)}$, which is calculated by aggregating models excluding the target client's trained model ω_k^1 (Algorithm 2 L.6).

Training phase

1. The server sends its model ω^{r+1} $(r \geq 1)$ and a distillation model $\omega_{d(\omega_k^r)}$, which is aggregated from the models of all clients but the k-th one (Algorithm 2 L.9).
2. A client calculates the average confidence score S_k^{r+1} for training data D_k with the distillation model $\omega_{d(\omega_k^r)}$. Figure 6 illustrates the calculation of S_k^{r+1} (Algorithm 2 L.11).
3. If the average confidence score S_k^{r+1} is higher than a predefined threshold T, then the client trains the received model ω^{r+1} with the soft-label training data $D_k^{'}$ (Algorithm 2 L.12-13). Otherwise, if S_k^{r+1} is lower than T, then the client trains ω^{r+1} with the hard-label training data D_k (Algorithm 2 L.14-15).
4. The client sends the trained model ω_k^{r+1} to the server, which calculates the updated server model ω^{r+2} and the updated distillation model $\omega_{d(\omega_k^{r+1})}$ (Algorithm 2 L.18).

In FLKD, as described above, the client controls the

Algorithm 2 Training phase of FLKD

Input: Hard label training data: $D_1, D_2, ..., D_K$; Initial server model: ω^1
Output: Trained model on server: ω^{R+1}
※ **Initialization phase**
for $k = 1$ to K **do**
  Server sends initial server model ω^1 to each client.
  Clients independently train model ω^1 with their own training data and send trained model ω_k^1 to server.
end for
Server calculates server model ω^2 and distillation model $\omega_{d(\omega_k^1)}$.
※ **Training phase**
for $r = 1$ to $R - 1$ **do**
  Server sends server model ω^{r+1} and distillation model $\omega_{d(\omega_k^r)}$ aggregated from all models except k-th client's model to clients.
  for $k = 1$ to K **do**
    Client calculates average confidence score S_k^{r+1} for training data D_k with distillation model $\omega_{d(\omega_k^r)}$.
    if $S_k^{r+1} \geq T$ **then**
      Client trains server model ω^{r+1} with soft-label training data D_k' and sends trained model ω_k^{r+1} to server.
    else
      Client trains server model ω^{r+1} with hard-label training data D_k and sends trained model ω_k^{r+1} to server.
    end if
  end for
  Server aggregates client's model ω_k^{r+1} and calculates server model ω^{r+2} and distillation model $\omega_{d(\omega_k^{r+1})}$.
end for
return ω^{R+1}

knowledge distillation implementation with a threshold T (Algorithm 2 L.11-15). In this way, FLKD achieves both model accuracy and privacy protection against MIAs.

The accuracy of the client model ω_k^r is expected to increase with each round. According, the accuracy of ω_k^r is low in early rounds, as is the accuracy of the distillation model $\omega_{d(\omega_k^r)}$ built from the client models ω_k^r. Soft labels calculated by the distillation model are considered to have a negative effect on model training, because they are close to random in the early round. FLKD addresses this problem by switching the knowledge distillation via the threshold T. That is, a client trains the server's model ω^{r+1} with the hard-label training data D_k in the early rounds, and if the average confidence score S_k^{r+1} representing the model accuracy exceeds T, then the client's training is switched to knowledge distillation. Here, the average confidence score S_k^{r+1} is calculated from the hard- and soft-label outputs of the distillation model $\omega_{d(\omega_k^r)}$ (see Figure 6). Because S_k^{r+1} is calculated from a one-hot vectorized confidence score, it is a nonnegative number in $[0, 1]$.

If the threshold T is set higher, then clients only conduct knowledge distillation near the last round, and the model is more likely to overfit. On the other hand, if it is set lower, then clients conduct knowledge distillation from early rounds, and overfitting is less likely to occur.

An advantage of FLKD is that the distillation model for each client is quickly built by changing the aggregated combination (model parameter averaging). Table 4 summarizes the client processing required for knowledge distillation with different defenses. As seen in the table, when KCD or SELENA is applied in FL, a client must train $s + 1$ models with a limited number of training samples, such as $n_k(s-1)/s$ or n_k/s. In contrast, FLKD only requires one additional inference for knowledge distillation and can mitigate MIAs without additional model training. Accordingly, clients can mitigate MIAs via FLKD without a lot of training data or computational resources.

4. Experimental setup

In this section, we introduce our experimental setup.

4.1 Datasets

We used the 100-class classification datasets of CIFAR100, Purchase100, and Texas100, as in existing studies on MIAs [4].

CIFAR100: This dataset contains 60,000 images, each of which is a color image with 32×32 pixels [17].

Purchase100: This dataset is based on Kaggle's "Acquire Valued Shoppers Challenge" [1]. Specifically, we used a dataset that was preprocessed by Shokri et al. [3] and contains 197,324 records. Each record has 600 binary features about the items purchased by a customer.

Texas100: This dataset is based on the "Texas Department of State Health Services" data, and we use a version that was again preprocessed by Shokri et al. [3]. This dataset contains 67,330 records, where each record has 6,170 binary features about a patient.

Table 5 lists the training and test data sizes used to train and evaluate the target model. In addition, as listed in the table, an MIA attacker has some of the target model's member and non-member data as prior knowledge.

4.2 Membership inference attacks

For this paper, we selected the trained server model as the target model and applied both single-query and label-only attacks against it. As in existing studies on MIAs, we used the "Accuracy" as the metric [4, 7, 9]:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FP+TN+FN}}. \quad (5)$$

Here, TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively, in terms of whether the prediction and ground-truth labels are members.

A random guess has $Accuracy = 0.5$ because MIA is a binary classification task. The Accuracy thus diverges from 0.5 as the risk of leaking private information increases.

[1] https://www.kaggle.com/c/acquire-valued-shoppers-challenge/data

Table 4: Comparison of client processing required for knowledge distillation in FL. Here, n_k is the number of training samples for the k-th client, and s is the number of divisions for KCD and SELENA.

Defense	Number of training models	Number of training samples for each model	Number of inferences for knowledge distillation
None	1	n_k	0
FLKD	**1**	$\boldsymbol{n_k}$	**1**
KCD	$s+1$	$n_k(s-1)/s$	s
SELENA	$s+1$	n_k/s	s

Table 5: Datasets.

Dataset	Target model		Prior knowledge		MIA evaluation	
	Training	Test	Training	Test	Training	Test
CIFAR100	50,000	5,000	5,000	5,000	5,000	5,000
Purchase100	20,000	5,000	5,000	5,000	5,000	5,000
Texas100	50,000	5,000	5,000	5,000	5,000	5,000

We also defined a "mitigation ratio" (MR) to evaluate the tradeoff between the MIA accuracy and model accuracy:

$$\text{Mitigation Ratio (MR)} = \frac{\text{MIA Accuracy}}{\text{Model Accuracy}}, \tag{6}$$

where a smaller MR indicates a better defense against MIAs.

4.3 Target model setups

We compared FLKD with DP-SGD, AdvReg, and Memguard, the existing defenses described in Section 2.3. DP-SGD and AdvReg were applied in FL by introducing them to all clients individually, while Memguard was introduced as a defense in the server model.

4.3.1 Model architectures

We used ResNet18 as the model architecture for CIFAR100 [4]. For Purchase100 and Texas100 [4], we used a 4-layer fully connected NN with various number of nodes in each layer [1024, 512, 256, 128].

4.3.2 Hyperparameters of target model

We evaluated the defenses against MIAs in FL by varying the number of clients K among 5, 10, and 20. Each client had the same number of training samples, where all the training data without duplicates was divided equally among the clients. The hyperparameters of FLKD, DP-SGD, and AdvReg were set as follows.

- **FLKD:** The threshold T was varied from 0.50∼0.95 in increments of 0.05 for all three datasets.
- **DP-SGD:** For CIFAR100, the clipping size C was 40, and the noise parameter was varied among 0.001, 0.01, and 0.03. For Purchase100, the clipping size C was 0.001, and the noise parameter was varied among 0.03, 0.05, and 0.07. Lastly, for Texas100, the clipping size C was 0.0001, and the noise parameter was varied among 0.001, 0.01, and 0.05.
- **AdvReg:** The privacy parameter λ was varied among 1, 10, 30, and 50 for all three datasets.

5. Experimental results

Table 6 and Figure 7 give the MIA accuracy and model accuracy of the target model. In the figure, results plotted at the bottom right in each graph indicate that both the accuracy and privacy are better. According to these results, FLKD showed the best tradeoff between the MIA accuracy and model accuracy as compared to the existing defenses of DP-SGD, AdvReg, and Memguard. The results in the table indicate that FLKD reduced single-query attacks by 14% and label-only attacks by 15% with a 1% reduction in the model accuracy, as compared to an undefended model ("None"), on CIFAR100 with five clients. In addition, on Purchase100 with five clients, FLKD reduced single-query attacks by 7% and label-only attacks by 7% with a 3% reduction in the model accuracy, as compared to the undefended model. Finally, on Texas100 with five clients, FLKD reduced single-query attacks by 21% and label-only attacks by 14% without degrading the model accuracy.

5.1 Results of changing number K of clients in FL

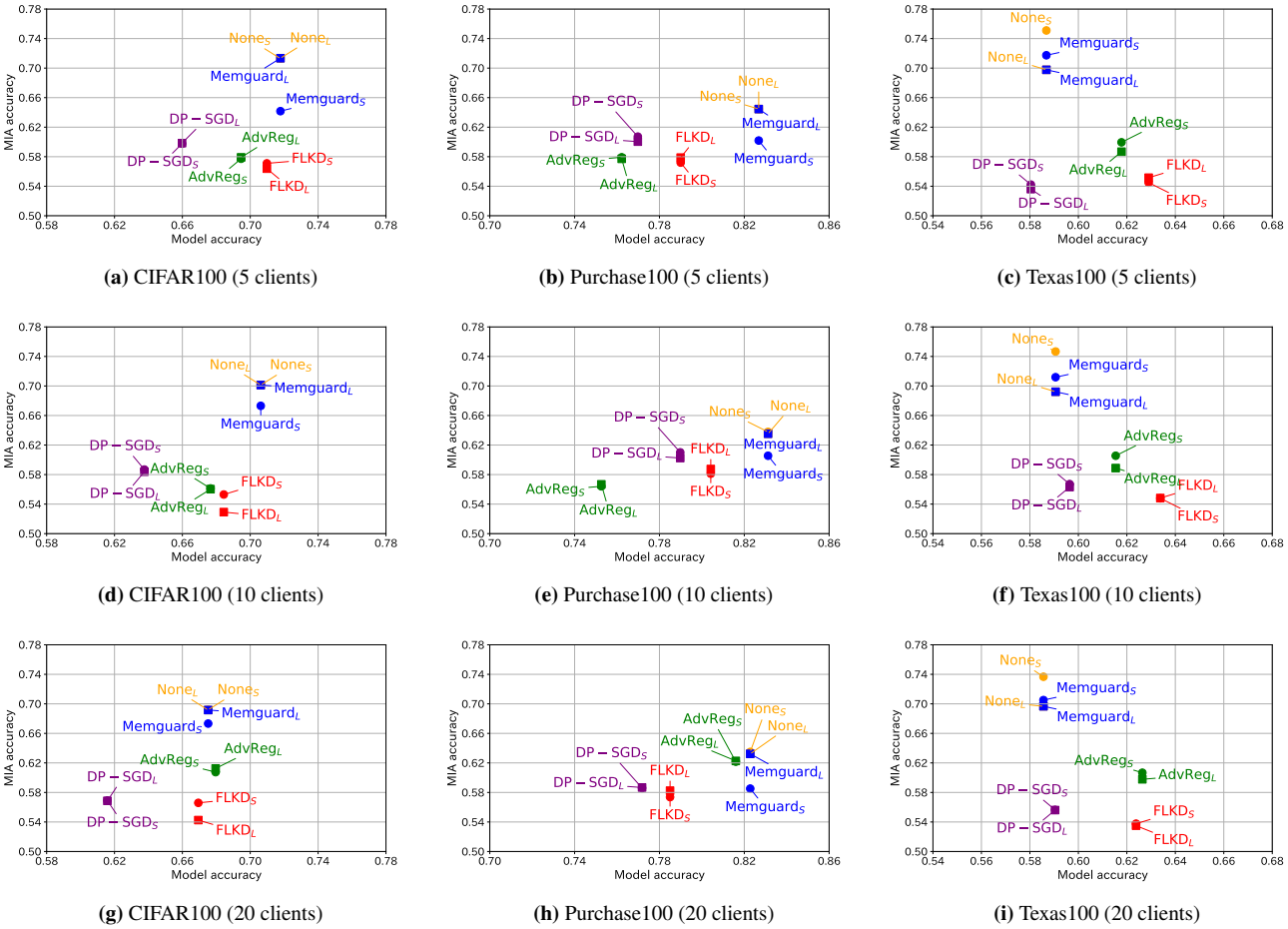
As introduced above, Figure 7 shows the MIA and model accuracies in FL when the number of clients, K, was 5, 10 or 20. In this scenario, the number of training samples for each client decreased as the number of clients increased (see Section 4.3.2). For example, on CIFAR100, each client had 10,000 training samples when K was 5, but only 2,500 training samples when K was 20. According to Figure 7, FLKD showed the best tradeoff between the MIA and model accuracies when the number of each client's training samples was limited to 2,500. FLKD can mitigate MIAs without requiring a lot of training data (see Section 3); therefore, it can sufficiently mitigate MIAs without depending on the number of training samples for clients in FL.

5.2 Results of changing FLKD threshold T

Next, Figure 8 shows the MIA and model accuracies when the threshold T was varied in FLKD. On all three datasets, a larger threshold T yielded a higher MIA accuracy. This was because a client switched its knowledge distillation according to T: as described in Section 3, a client conducted knowledge distillation only when the average confidence score S_k^{r+1} is greater than or equal to T. Accordingly, when a client sets

Table 6: Comparison of the MIA and model accuracies on three datasets for FLKD, an undefended model ("None"), and the existing defenses DP-SGD, AdvReg, and Memguard.

Dataset	Client	Defense		Model accuracy		MIA accuracy		MR	
		Name	Parameter	Train	Test	Best single	Label	Best single	Label
CIFAR100	5	None	-	0.9993	0.7178	0.7129	0.7134	0.9932	0.9939
		FLKD	**T=0.70**	**0.8501**	**0.7098**	**0.5710**	**0.5638**	**0.8045**	**0.7943**
		DP-SGD	$\sigma = 0.01$	0.8544	0.6598	0.5977	0.5984	0.9059	0.9069
		AdvReg	$\lambda = 30$	0.8513	0.6946	0.5772	0.5791	0.8310	0.8337
		Memguard	-	0.9993	0.7178	0.6417	0.7134	0.8940	0.9939
Purchase100	5	None	-	0.9991	0.8268	0.6446	0.6445	0.7796	0.7795
		FLKD	**T=0.75**	&textbf0.9155	**0.7900**	**0.5720**	**0.5788**	**0.7241**	**0.7327**
		DP-SGD	$\sigma = 0.05$	0.9172	0.7698	0.6072	0.6004	0.7888	0.7799
		AdvReg	$\lambda = 30$	0.9109	0.7622	0.5792	0.5771	0.7599	0.7572
		Memguard	-	0.9991	0.8268	0.6019	0.6445	0.7280	0.7795
Texas100	5	None	-	0.9969	0.5868	0.7510	0.6977	1.2798	1.1890
		FLKD	**T=0.60**	**0.7023**	**0.6290**	**0.5449**	**0.5516**	**0.8663**	**0.8769**
		DP-SGD	$\sigma = 0.05$	0.6578	0.5804	0.5422	0.5361	0.9342	0.9237
		AdvReg	$\lambda = 50$	0.7790	0.6178	0.5995	0.5868	0.9704	0.9498
		Memguard	-	0.9969	0.5868	0.7174	0.6977	1.2226	1.1890



(a) CIFAR100 (5 clients) **(b)** Purchase100 (5 clients) **(c)** Texas100 (5 clients)

(d) CIFAR100 (10 clients) **(e)** Purchase100 (10 clients) **(f)** Texas100 (10 clients)

(g) CIFAR100 (20 clients) **(h)** Purchase100 (20 clients) **(i)** Texas100 (20 clients)

Fig. 7: Comparison of the MIA and model accuracies on three datasets for FLKD, an undefended model ("None"), and the existing defenses DP-SGD, AdvReg, and Memguard. The vertical axis indicates the MIA accuracy, while the horizontal axis indicates the model accuracy for testing. The subscripts S and L indicate the best single-query and label-only attack results, respectively, in each graph. Points toward the bottom right in each graph represent better defenses.

a high threshold (e.g., $T = 0.95$), it conducts knowledge distillation only near the last round and not mitigate MIAs. On

the other hand, when it sets a low threshold (e.g., $T = 0.50$), it conducts knowledge distillation from early rounds, and the model accuracy may possibly decrease. Therefore, a client needs to set an appropriate threshold T depending on the model application in FLKD.

5.3 Results of changing DP-SGD noise parameter σ

Figure 9 shows the MIA and model accuracies when the noise parameter σ of DP-SGD was varied. Larger values of σ yielded lower MIA and model accuracies. In addition, to achieve the same privacy protection as FLKD, DP-SGD induced a large reduction in the model accuracy. Specifically, for an MIA accuracy of about 56% on CIFAR100, the model accuracy was about 12% lower for DP-SGD than for FLKD. In addition, for an MIA accuracy of about 58% on Purchase100, the model accuracy was about 6% lower for DP-SGD than for FLKD. Finally, for an MIA accuracy of about 55% on Texas100, the model accuracy was about 5% lower for DP-SGD than for FLKD.

5.4 Results of changing AdvReg parameter λ

Figure 10 shows the MIA and model accuracies when the parameter λ of AdvReg was varied. On CIFAR100 and Purchase100, larger values of λ yielded lower MIA and model accuracies. Specifically, for an MIA accuracy of about 57% on CIFAR100, the model accuracy was about 3% lower for AdvReg than for FLKD. In addition, for an MIA accuracy of about 58% on Purchase100, the model accuracy was about 3% lower for AdvReg than for FLKD.

On Texas100, on the other hand, as the AdvReg parameter λ was increased, the MIA accuracy decreased, but the model accuracy increased. For $\lambda = 30, 50$, however, there was no difference in the MIA or model accuracy. Therefore, AdvReg had a limited MIA accuracy reduction and could not achieve better privacy protection than FLKD could on Texas100.

6. Discussion

In this section, we discuss the application of KCD in FL, the computational cost of FLKD and the communication overhead of FLKD.

6.1 Application of KCD to client in FL

As explained above (see Section 2.3), it is difficult to apply KCD to clients in FL. In this section, we experimentally demonstrate this problem. Figure 11 shows the MIA and model accuracies when KCD was applied to clients in FL. As the number of clients, K, increased, the model accuracy decreased, because the amount of training data for each client decreased. In our setup, each client had the same number of training samples (see Section 4.3.2). Therefore, as the number of clients increased, each client was limited in the amount of training data for KCD, and the model accuracy decreased.

From this result, KCD is not a good defense against MIAs in FL.

6.2 Computational cost of FLKD

We also compared the training and inference times of FLKD with those of the existing defenses DP-SGD, AdvReg, Memguard, and KCD. We calculated these times by using an NVIDIA GeForce RTX 3090 with 24 GB of VRAM, 64 GB of DRAM, and an Intel Corei7-12700 CPU. The calculation conditions were as follows. On each dataset, the number of training epochs was 10, and the batch sizes for training and evaluation were 64.

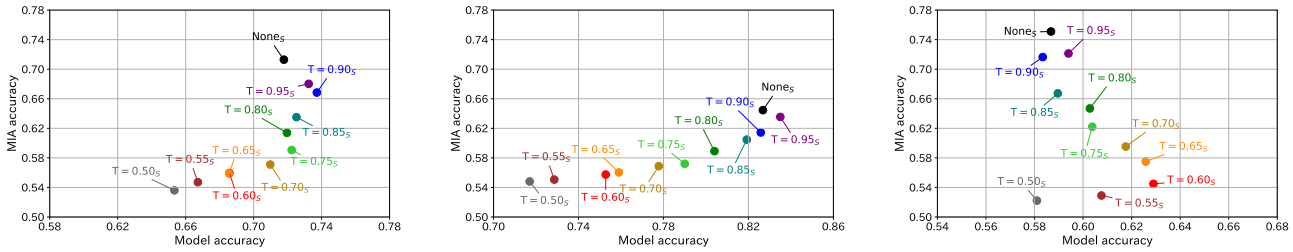
- **CIFAR100:** We calculated the time for the client to train the model with 10,000 training samples and the inference time for the client to evaluate 5,000 test samples with the model.
- **Purchase100:** We calculated the time for the client to train the model with 4,000 training samples and the inference time for the client to evaluate 5,000 test samples with the model.
- **Texas100:** We calculated the time for the client to train the model with 10,000 training samples and the inference time for the client to evaluate 5,000 test samples with the model.

Table 7 lists the computational costs for an undefended model ("None"), FLKD, DP-SGD, AdvReg, Memgurad, and KCD. Here, the undefended model ("None") is faster than other defense techniques in the training and inference phases. As seen in the table, FLKD could mitigate MIAs while having approximately the same training and inference times as those of an undefended model ("None").

As explained above, DP-SGD requires clipping and adding the gradient for each batch training (see Section 2.2.3). Therefore, it had a longer training time than None, but it had the same inference time. AdvReg requires building an additional binary classifier (see Section 2.2.3). Therefore, it had a significantly longer training time than None, but with the same inference time. Memguard mitigates MIAs by adding noise to the confidence score in the inference phase (see Section 2.2.3). Therefore, its training time was the same as that of None, but the inference time was significantly increased. Finally, in KCD, the client trains multiple sub-models with split training data (see Section 2.2.2). Therefore, its training time was significantly increased over that of None, but with the same inferecen time.

6.3 Reduction method of communication overhead on FLKD

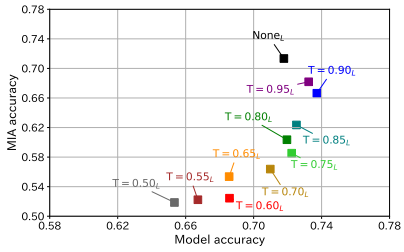
In Figure 5, the server additionally sends distillation models $\omega_{d(\omega_k^r)}$ to the k-th client in FLKD. Therefore, FLKD has the disadvantage that the communication cost from the server to the clients is twice as high as for standard FL. However, there is a technique to perform our FLKD with the



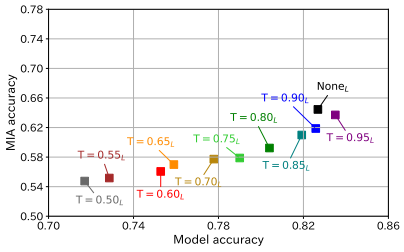
(a) CIFAR100 (single-query attacks)

(b) Purchase100 (single-query attacks)

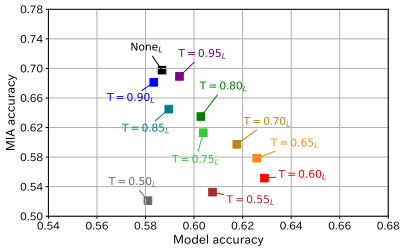
(c) Texas100 (single-query attacks)



(d) CIFAR100 (label-only attacks)

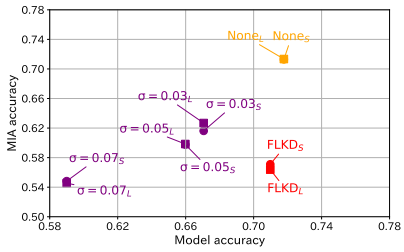


(e) Purchase100 (label-only attacks)

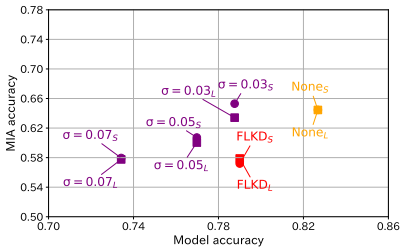


(f) Texas100 (label-only attacks)

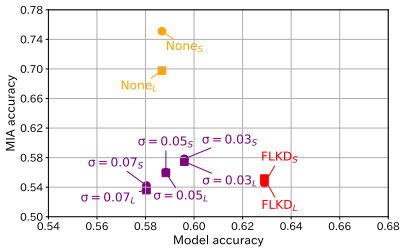
Fig. 8: Comparison of the MIA and model accuracies on three datasets for various threshold values T in FLKD, with five clients in FL. The vertical axis indicates the MIA accuracy, while the horizontal axis indicates the model accuracy for testing. Points toward the bottom right in each graph represent better defenses.



(a) CIFAR100

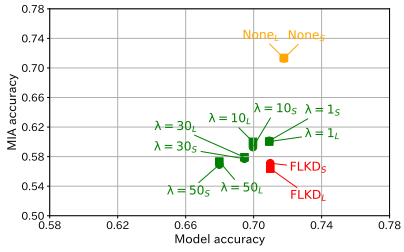


(b) Purchase100

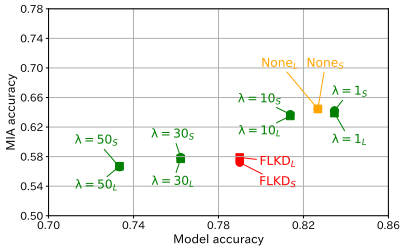


(c) Texas100

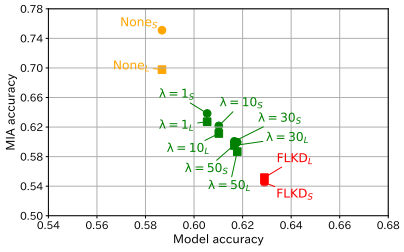
Fig. 9: Comparison of the MIA and model accuracies on three datasets for various noise parameters σ in DP-SGD, with five clients in FL. The vertical axis indicates the MIA accuracy, while the horizontal axis indicates the model accuracy for testing. The subscripts S and L indicate the best single-query and label-only attack results, respectively, in each graph. Points toward the bottom right in each graph represent better defenses.



(a) CIFAR100

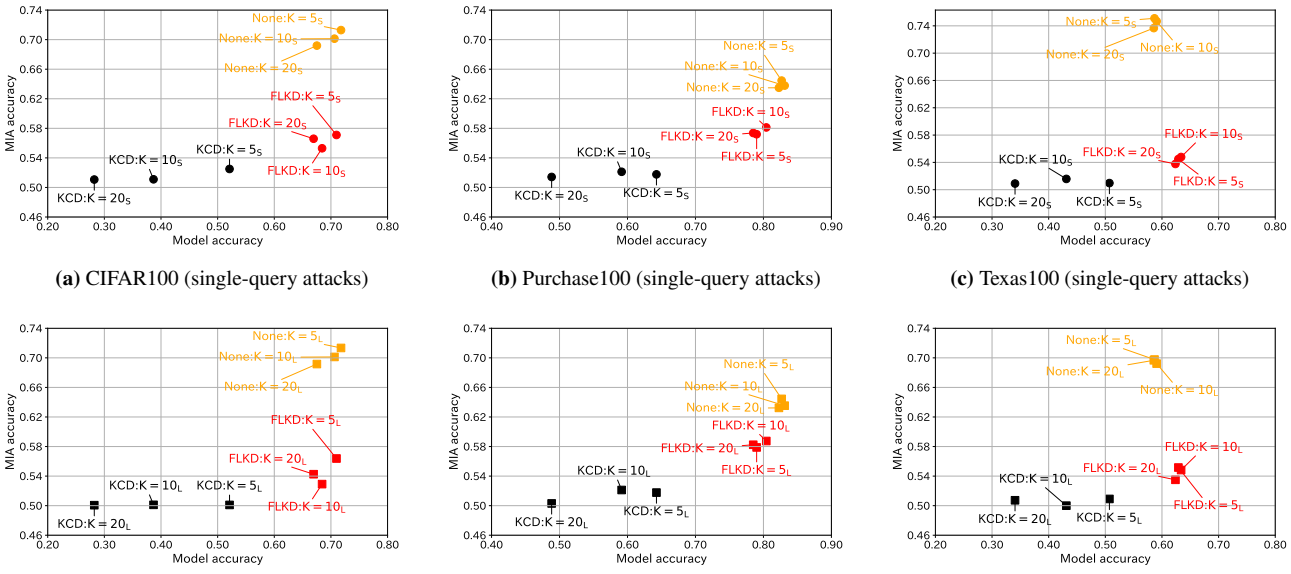


(b) Purchase100



(c) Texas100

Fig. 10: Comparison of the MIA and model accuracies on three datasets when the parameter λ of AdvReg was varied, with five clients. The vertical axis indicates the MIA accuracy, while the horizontal axis indicates the model accuracy for testing. The subscripts S and L indicate the best single-query and label-only attack results, respectively. Points toward the bottom right in each graph represent better defenses.



(a) CIFAR100 (single-query attacks) **(b)** Purchase100 (single-query attacks) **(c)** Texas100 (single-query attacks)

(d) CIFAR100 (label-only attacks) **(e)** Purchase100 (label-only attacks) **(f)** Texas100 (label-only attacks)

Fig. 11: Comparison of the MIA and model accuracies on three datasets when the number of clients was varied in KCD, with the number of divisions, s, set to 10. The vertical axis indicates the MIA accuracy, while the horizontal axis indicates model accuracy for testing. Points toward the bottom right represent better defenses.

Table 7: Comparison of the computational costs on three datasets for an undefended model ("None"), FLKD, and the existing defenses DP-SGD, AdvReg, Memguard, and KCD.

		None	FLKD	DP-SGD	AdvReg	Memguard	KCD
CIFAR100	Training time	36.00s	**37.87s**	95.43s	123.36s	36.00s	2361.84s
	Inference time	1.87s	**1.87s**	1.87s	1.87s	1256.61s	1.87s
Purchase100	Training time	0.62s	**0.64s**	4.85s	3.46s	0.62s	21.71s
	Inference time	0.05s	**0.05s**	0.05s	0.05s	1156.80s	0.05s
Texas100	Training time	2.44s	**2.48s**	28.59s	6.25s	2.44s	74.65s
	Inference time	0.06s	**0.06s**	0.06s	0.06s	1314.39s	0.06s

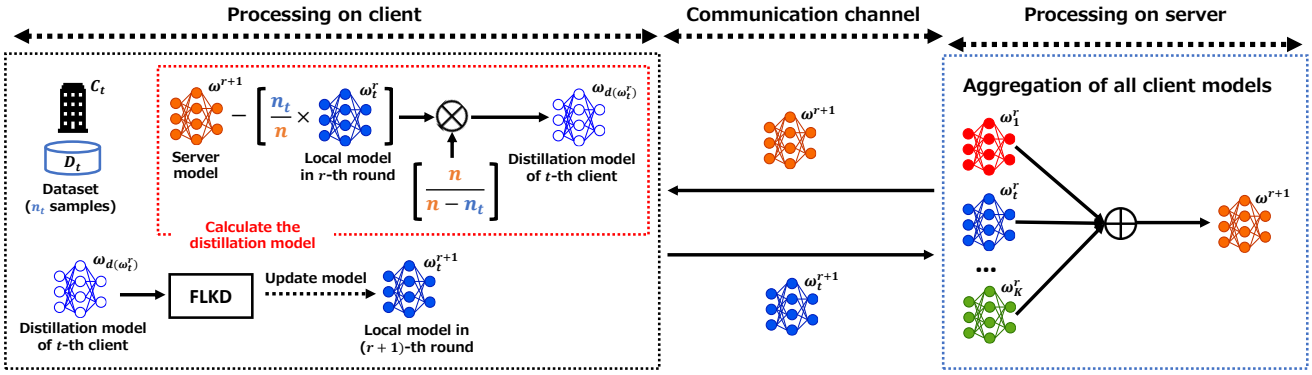


Fig. 12: Reduction method of communication overhead on FLKD.

comparable communication costs as standard FL. Figure 12 shows the reduction method of communication overhead on FLKD. In this technique, clients calculate their distillation model instead of receiving them from the server. To compute the distillation model, the client needs the number of training samples for all clients n, which is not confidential information and can be received from the server. Each client can calculate its distillation model with the following steps:

$$\omega^{r+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_k^r, \tag{7}$$

$$\begin{aligned} \omega_{d(\omega_t^r)} &= \sum_{k=1, k\neq t}^{K} \frac{n_k}{n - n_t} \omega_k^r \\ &= \left\{ \sum_{k=1}^{K} \frac{n_k}{n} \omega_k^r - \frac{n_t}{n} \omega_t^r \right\} \cdot \frac{n}{n - n_t} \\ &= \left\{ \omega^{r+1} - \frac{n_t}{n} \omega_t^r \right\} \cdot \frac{n}{n - n_t}, \end{aligned} \tag{8}$$

where ω^{r+1} is the server model in $(r+1)$-th round, ω_t^r is the local model of the t-th client and n_t is the number of training samples on the t-th client. Note that the t-th client has the server model ω^{r+1}, the number of training samples on the t-th client n_t, and the local model of the t-th client ω_t^r in the standard FL procedure; the only additional knowledge required to perform our FLKD is n. Since n is an integer scalar value, sending this is small enough to be negligible compared to the model parameter exchange performed in the standard FL. Therefore, FLKD can mitigate MIAs at the equivalent communication costs as standard FL. In addition, the computational cost that the client additionally calculates the distillation model is negligible, since the distillation model can be calculated by only a few multiplications and subtraction.

7. Conclusion

In this paper, we proposed a defense against MIAs that uses knowledge distillation in a way that is suitable for FL. Our proposed defense, called federated learning knowledge distillation (FLKD), can mitigate MIAs without a lot of training data and computational cost, in contrast to existing defenses such as KCD and SELENA. We compared FLKD with the existing defenses DP-SGD, AdvReg, Memguard, and KCD in terms of the tradeoff between the task and MIA accuracies on the CIFAR100, Purchase100, and Texas100 datasets.

In our experimental results, FLKD showed the best accuracy tradeoff on these tasks, as compared to the existing defenses. Specifically, on CIFAR100, FLKD reduced single-query attacks by 14% and label-only attacks by 15% with a 1% reduction in model accuracy as compared to an undefended model. On Purchase100, as compared to the undefended model, it reduced single-query attacks by 7% and label-only attacks by 7% with a 3% reduction in model accuracy. Finally, on Texas100, FLKD reduced single-query attacks by 21% and label-only attacks by 14% without degrading the model accuracy.

In addition, FLKD achieved the best computational cost in terms of the training and inference times. In FLKD, each client mitigates MIAs by using a distillation model built on the server, rather than building the distillation model itself. Therefore, FLKD requires only as much as training and inference time as an undefended model. By the clients calculating their distillation model instead of the server, the communication costs can be reduced to the same level as standard FL.

In our future work, we evaluate the MIA privacy risk and the model utility of the FLKD in white-box MIAs [2].

References

[1] 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.

[2] 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," 2019 IEEE symposium on security and privacy (SP), pp.739–753, IEEE, 2019.

[3] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," 2017 IEEE symposium on security and privacy (SP), pp.3–18, IEEE, 2017.

[4] X. Tang, S. Mahloujifar, L. Song, V. Shejwalkar, M. Nasr, A. Houmansadr, and P. Mittal, "Mitigating membership inference attacks by self-distillation through a novel ensemble architecture," 31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022, ed. K.R.B. Butler and K. Thomas, pp.1433–1450, USENIX Association, 2022.

[5] G.E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," CoRR, vol.abs/1503.02531, 2015.

[6] V. Shejwalkar and A. Houmansadr, "Membership privacy for machine learning models through knowledge transfer," Proceedings of the AAAI conference on artificial intelligence, pp.9549–9557, 2021.

[7] R. Chourasia, B. Enkhtaivan, K. Ito, J. Mori, I. Teranishi, and H. Tsuchida, "Knowledge cross-distillation for membership privacy," Proceedings on Privacy Enhancing Technologies, vol.2, pp.362–377, 2022.

[8] M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp.308–318, 2016.

[9] M. Nasr, R. Shokri, and A. Houmansadr, "Machine learning with membership privacy using adversarial regularization," Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018, ed. D. Lie, M. Mannan, M. Backes, and X. Wang, pp.634–646, ACM, 2018.

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

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

[12] Y. Liu, J. Peng, J. Kang, A.M. Iliyasu, D. Niyato, and A.A.A. El-Latif, "A secure federated learning framework for 5g networks," IEEE Wireless Communications, vol.27, no.4, pp.24–31, 2020.

[13] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," arXiv preprint arXiv:2009.03561, 2020.

[14] S. Yeom, I. Giacomelli, A. Menaged, M. Fredrikson, and S. Jha, "Overfitting, robustness, and malicious algorithms: A study of potential causes of privacy risk in machine learning," J. Comput. Secur., vol.28, no.1, pp.35–70, 2020.

[15] C.A. Choquette-Choo, F. Tramèr, N. Carlini, and N. Papernot, "Label-only membership inference attacks," Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July

2021, Virtual Event, ed. M. Meila and T. Zhang, Proceedings of Machine Learning Research, vol.139, pp.1964–1974, PMLR, 2021.
[16] Z. Li and Y. Zhang, "Membership leakage in label-only exposures," CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021, ed. Y. Kim, J. Kim, G. Vigna, and E. Shi, pp.880–895, ACM, 2021.
[17] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

Rei Ueda received his B.E. in electronic engineering fron Ritsumeikan University in 2023. He is currently a master's student at the Graduate School of Science and Engineering, Ritsumeikan University. His research interests include machine learning. He is a member of IEICE.

Tsunato Nakai received the B.E., M.E., and Ph.D. in engineering from Ritsumeikan University, Japan, in 2013, 2015, and 2022. His research interests include machine learning, hardware and industrial control system security. He is a member of IPSJ.

Kota Yoshida received his B.E., M.E., and Ph.D. in engineering from Ritsumeikan University in 2017, 2019, and 2022. He is currently an assistant professor in Department of Electronic and Computer Engineering at Ritsumeikan University. His research interests include machine learning and hardware security. He is a member of IEICE, IEEE, JSAI.

Takeshi Fujino was born in Osaka, Japan, on March 17, 1962. He received his B.E. and M.E., and Ph.D. in electronic engineering from Kyoto University, Kyoto, Japan, in 1984, 1986, and 1994. He joined the LSI Research and Development center, Mitsubishi Electric Corp. in 1986. Since then, he had been engaged in the development of micro-fabrication processes, such as electron beam lithography, and embedded DRAM circuit design. He has been a professor at Ritsumeikan University since 2003. His research interests include hardware security such as side-channel attacks and physically unclonable functions. He is a member of IEICE, IPSJ, IEEE.