Practical Membership Inference Attack Against Collaborative Inference in Industrial IoT

Hanxiao Chen, *Student Member, IEEE,* Hongwei Li (Corresponding author), *Senior Member, IEEE,* Guishan Dong, Meng Hao, *Student Member, IEEE,* Guowen Xu, *Student Member, IEEE,* Xiaoming Huang, and Zhe Liu, *Senior Member, IEEE*

***Abstract*—The effectiveness of state-of-the-art deep learning (DL) models has empowered the development of industrial Internet of Things (IIoT). Recently, considering resource-constrained and privacy-required IIoT devices, collaborative inference has been proposed, which splits DL models and deploys them in IIoT devices and an edge server separately. However, in this paper, we argue that there are still severe privacy vulnerabilities in collaborative inference systems. And we devise the first membership inference attack (MIA) against collaborative inference, to infer whether a particular data sample is used for training the model of IIoT systems. Existing MIAs either assume full access to the systems' APIs or availability of the target model's parameters, which is not applicable in realistic IIoT environments. In contrast to prior works, we propose *transfer-inherit* shadow learning and thus relax these key assumptions. We evaluate our attack on different datasets and various settings, and the results show it has high effectiveness.**

***Index Terms*—Industrial IoT, deep learning, collaborative inference, membership inference attack**

I. INTRODUCTION

Deep learning (DL) [1] has achieved state-of-the-art performance in a variety of real-world tasks for industrial Internet of Things (IIoT) [2], such as autonomous driving [3] and E-health [4]. However, deploying DL systems in IIoT scenarios still face two major challenges. One is that, DL based decision making requires extensive computation and storage resources, which is prohibitively expensive for resource-constrained IIoT devices [5]. The other is that, a naive alternative, e.g., outsourcing decision tasks to machine learning service providers, raises serious privacy issues [6]. To address these challenges, collaborative inference [7] [8] has recently been proposed and widely applied in IIoT scenarios. The main idea is to divide a DL model into two parts, and deploy the first part to local IIoT devices while offloading the resource-consuming latter part into an edge server. As a result, it not only saves the computation and storage resources of IIoT devices, but also provides better privacy protection of the training data, as IIoT devices only transmit intermediate values to the edge server, rather than the original data [9].

Despite tremendous success in a range of industrial IIoT applications, collaborative inference systems are inherently vulnerable to various privacy and security risks, such as model inversion attack [10] and property inference attack [11]. Different from these existing attacks, in this work, we concentrate on stealing the privacy of training data in collaborative inference systems, which is a more serious privacy threat. In other words, while collaborative inference was originally designed to mitigate the IIoT devices data privacy during the inference phase, we argue that *offloading the latter layers may leave some imprint of the training data and thus still face unintended privacy risks*. To the best of our knowledge, none of the recent works have explored the privacy threat of the training data itself in collaborative inference systems. Therefore, it is critical to quantify the current collaborative inference systems privacy threats and develop ideas for designing possible solutions.

In this paper, we devise the first practical membership inference attack (MIA) against collaborative inference to infer whether or not an individual input sample is in training dataset. In recent years, MIAs have been successfully carried out against DL models in a variety of attack scenarios [12] [13] [14] [15] [16]. For example, Shokri et al. [12] presented the first MIA against machine learning models in a black-box setting. Specifically, attackers train a binary classifier as the attack model and take as input a data sample's prediction value obtained by querying the target model to infer membership information. Subsequently, Nasr et al. [15] extended MIAs to white-box situation, where attackers launch more powerful attacks by utilizing more information including the activations and gradients of each layer. Existing MIAs either assume access to the systems' APIs or availability of the target model's parameters and training dataset. Unfortunately, in IIoT scenarios, it may be unpractical to query the target model embedded in the IIoT device, and be difficult to know the target model's parameters and training dataset. Therefore, a straightforward application of these existing attacks to IIoT scenarios is ineffective. Rather, we consider the realistic (and the worst) IIoT scenarios, where our attackers do not require to query the target model and make assumptions about the prior knowledge.

To this end, we design *transfer-inherit* shadow learning to achieve a practical MIA against collaborative inference in IIoT. Specifically, *transfer* means to transfer a general feature

Hanxiao Chen, Hongwei Li, Meng Hao and Guowen Xu are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China. (e-mail: hanxiao.chen@foxmail.com; hongweili@uestc.edu.cn; menghao0303@foxmail.com; guowen.xu@foxmail.com).

Guishan Dong is with the No.30 Institute of China Electronics Technology Group Corporation, Chengdu, 610041, China. (e-mail: mountain_dong@163.com).

Xiaoming Huang is with the Technology Marketing Department of CETC Cyberspace Security Research Institute Co., Ltd. (e-mail: apride@gmail.com).

Zhe Liu is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China. (e-mail: sduliuzhe@gmail.com).

extractor to replace the missing first part of model layers, and *inherit* means to fix the known latter layers that are specific to the target task. The objective of our *transfer-inherit* shadow learning is to imitate the behavior of the target model as much as possible, while relaxing key assumptions underlying existing attack schemes.

In summary, our contributions can be described as follows:

- We propose the *first* MIA against collaborative inference systems in IIoT environment, profoundly revealing that current collaborative inference systems still face serious privacy threats.
- In contrast to prior works that either assume access to the systems APIs or availability of the target models parameters and training data, we propose transfer-inherit shadow learning and thus relax these assumptions.
- We experimentally evaluate the performance of our attack using three datasets, and the results show that the proposed method significantly outperforms existing attacks in IIoT scenarios by a large margin. We further conduct ablation studies to illustrate the effectiveness of *transfer* and *inherit* methods.

The rest of this paper is organized as follows. In Section II, we introduce the related work including MIA, other attacks against machine learning and collaborative deep learning. The background including DL, collaborative inference and transfer learning are discussed in Section III. In Section IV, we discuss the threat model and our attack method in detail. Then Section V indicates the experimental setup. Section VI not only gives the experimental proof that our attack is effective and general but also proposes the mitigation strategies. Finally, summarizing the whole paper in Section VII.

II. RELATED WORK

A. MIA Against Machine Learning Models

MIA refers to that the attacker can determine whether or not a data item is used to train the *target model*. Shokri et al. [12] proposed the first MIA against the **black-box** target model. They pointed out that ML models often behave differently between the data it trained on and the data it does not trained on. The goal of the attacker is to conduct *attack models* to learn these differences and use them to distinguish between members and non-members of the training data. To train attack models, they introduced *shadow models* to imitate the behavior of the target model and provide training data for attack models.

Recently, Salem et al. [13] designed new membership inference attacks in the black-box setting, which relax the attack assumptions proposed by Shokri et al. from the perspectives of the shadow model and attack model. Specifically, they showed that only a single shadow model and a single attack model can still achieve high attack accuracy. They also pointed out that putting the three largest values of the confidence score vectors into the attack model instead of the entire vectors, would increase the success of MIA.

Besides the black-box setting, Nasr et al. [15] recently proposed comprehensive MIAs against the **white-box** target model. In this case, since the attacker has full access to the target model including architecture and all parameters, there is no longer shadow models. To train the attack model, they assumed that the attacker had a part of training data of the target model. And for a data record, they calculated its gradient and the output of hidden layers under the target model as its feature vector and put it into the attack model for inference.

Meanwhile, many recent works [17] [16] [14] have also carried out research on membership inference attacks from different perspectives.

B. Attacks Against Machine Learning

Besides MIA, there is another type of attack against data privacy, i.e., *model inversion attack*, proposed by Fredrikson et al. [18]. They showed that attackers can infer corresponding prediction data based on the inference result. Later, this attack has been extended to a broader area [19]. And recently, He et al. [10] pointed out that the collaborative inference system is also vulnerable to this attack. In addition to the sensitive data itself, models also face serious privacy threats. Tramer et al.[20] indicated that via prediction APIs, attackers can duplicate the functionality of the target model, known as *model extraction attack*. Since then, many following works have been proposed, such as stealing the model's structures [21], parameters [22] and hyper-parameters [23]. Another major attack is *adversarial attack* [24] [25], where attackers add particular perturbation to the input to fool the ML model, leading to the expected misclassification.

C. Collaborative Deep Learning

A classic DL system contains dozens of layers and millions of parameters. For example, VGG16, a state-of-the-art convolutional network for classification and detection, contains 21 layers and more than 130 million parameters. Therefore, it's prohibitively expensive to deploy it entirely on edge devices with the limited computation and storage resources. A naive mitigation method is to outsource DL systems to the edge server, e.g., edge devices only simply transmit the collected private data to the server, which then perform specific tasks and return results. Unfortunately, it can incur significant communication costs and if the server is malicious, the privacy of raw data in edge devices will not be guaranteed [26].

To achieve a balance between efficiency and privacy, as well as reduce communication overhead, collaborative deep learning has been proposed. Typically there are two modes. One is collaborative training [27] [26], where multiple entities use their own private data to train a DL model locally and only exchange updates, and the global model is aggregated from each local update. The other is collaborative inference [5], which is widely used in IIoT environment. A well-trained DNN is divided into multiple parts, with each part distributed to a different entity. A data record sequentially passes through each part to obtain the final inference result.

Although the collaborative deep learning has achieved great success in IIoT scenarios, it still faces various privacy and security risks. He et al. [10] have shown that such method is susceptible to *model inversion attack*. The goal of this attack is to steal the privacy of inference data during the prediction phase. Song et al. [11] pointed out the phenomenon

of overlearning, which leads to *property inference attacks* against the attributes of data.

Different from these existing works, in this paper, we focus on revealing the privacy risks of training data in collaborative inference.

III. BACKGROUND

In this section, we describe the background of our attack method, including deep neural network, collaborative inference, membership inference attack and transfer learning.

A. Deep Neural Network

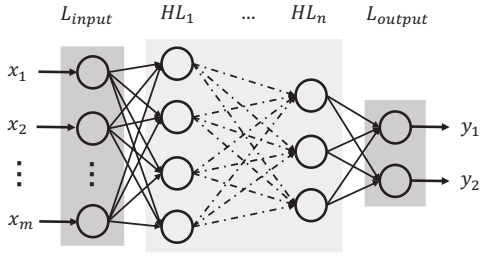


Fig. 1: A general DNN

A deep neural network (DNN) is a function f_ω: $\mathcal{X} \rightarrow \mathcal{Y}$ parameterized by parameters ω, which maps the input $\mathbf{x} \in \mathcal{X}$ to the output $\mathbf{y} \in \mathcal{Y}$. This function is learned from a training dataset with the goal of making inference on given data. Each element of the input $\mathbf{x}$ corresponds to one feature while each element of the output $\mathbf{y}$ represents a predefined class. A general DNN is shown in Fig. 1. It consists of an input layer L_{input}, one or more hidden layers $\{HL_1, \cdots, HL_n\}$ and an output layer L_{output}. Each layer consists of a large number of neurons that connected to other neurons in the adjacent layer.

The lifecycle of a DNN contains two stages: training phase and inference phase. The objective of training phase is to find the best parameters ω that accurately reflect the relationship between the feature vector $\mathbf{x}$ and the vectorized class $\bar{\mathbf{y}}$ of every data point in $\mathcal{X} \times \mathcal{Y}$. For this purpose, the loss function $\mathcal{L}(f_\omega(\mathbf{x}), \bar{\mathbf{y}})$ is proposed to quantify the gap between the true class $\bar{\mathbf{y}}$ and the model inference $\mathbf{y}$. Formally, the loss function over the training set D can be expressed as follows.

$$\mathcal{L}_D(f_\omega) = \frac{1}{|D|} \sum_{\langle \mathbf{x}_i, \bar{\mathbf{y}}_i \rangle \in D} \mathcal{L}(f_\omega(\mathbf{x}_i), \bar{\mathbf{y}}_i)$$

Therefore, the parameters ω can be found by minimizing $\mathcal{L}_D(f_\omega)$, where the stochastic gradient descent (SGD) is used to solve this optimization problem.

In the inference phase, the well-trained DNN f_ω is used to perform classification task. Specifically, for an input $\mathbf{x}$, f_ω returns the inference result $f_\omega(\mathbf{x})$.

B. Collaborative Inference

In a collaborative inference system, a full-trained DNN is divided into n parts: $f_\omega \rightarrow \{f_{\omega_1}, \cdots, f_{\omega_n}\}$ and allocated to n different entities: $\mathbb{E}_1, \cdots, \mathbb{E}_n$, respectively. Fig. 2 shows the structure of the collaborative inference system in our attack scenario, in which consists of two entities, the edge server and the IIoT device. In this scenario, the whole DNN f_ω is split into two parts: f_{ω_1} and f_{ω_2}, which are loaded on the IIoT device and the edge server respectively.

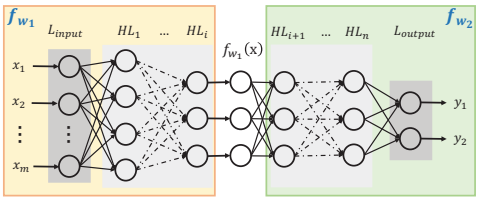


Fig. 2: The structure of collaborative inference

During inference phase, given a data record $\mathbf{x}$, the IIoT device generates intermediate value $\mathbf{v}_1 = f_{\omega_1}(\mathbf{x})$ and sends it to the edge server. Then the edge server calculates the final inference output $\mathbf{y} = f_{\omega_1}(\mathbf{v}_1)$.

C. Membership Inference Attack

Recall that the goal of MIA is to determine whether a data record is used to train the target model. The intuition behind it is that full-trained DNNs generally behave differently on their training data and the data they "see" for the first time, which can be exploited by attackers to distinguish members from non-members. This characteristic is associated with the degree of *overfitting*. Overfitting can be quantified as the gap between training accuracy and test accuracy. Large gap indicates serious overfitting. The deeper overfitting of the target model, the more vulnerable it is to attack.

Formally, MIA can be formulated as follows.

$$\mathcal{M}_{attack}\ (\mathcal{K}_{\mathcal{M}_{target}}\ ,\ (\mathbf{x}, y)) \rightarrow \{0, 1\}$$

Given a sample $(\mathbf{x}, y)$ and additional knowledge $\mathcal{K}_{\mathcal{M}_{target}}$ of the target model $\mathcal{M}_{target}$, attackers usually attempt to construct an attack model $\mathcal{M}_{attack}$ to execute attacks and finally output 0 or 1, where 0 means the sample is not a training data point, otherwise, attackers will consider it as training data.

D. Transfer Learning

The main idea of transfer learning is to transfer the knowledge including architecture and weights of a well-trained *Teacher* model to a new *Student* model [28], where both of two models have significant similarity tasks. Due to the knowledge student model has, it does not need to learn from scratch and thus the learning efficiency is improved.

Specifically, the student model is initialized by copying the first N layers, i.e. *feature extractor*, from the teacher model and followed by new K layers to match its classification task. Then the student model will be re-trained locally using its own dataset by freezing the feature extractor and only re-training the latter K layers. In other words, the weights of first N layers are fixed during training phase, while only the weights of the latter K layers are constantly updated. This is because the feature extractor can already correctly represent the features of training data. The student model can directly reuse the first

N layers and freeze them to speed up training and improve performance [29].

With the help of this idea, in this paper, we transfer the feature extractor from the existing well-trained model and stitch it with the inherited latter part of the target model to initialize our shadow model. Unlike transfer learning, we freeze the inherited latter layers that specific to classification tasks, and re-train the transferred feature extractor. The main reason is that the inherited latter part of the shadow model has been completely consistent with the latter part of the target model, and freezing the inherited latter layers can better achieve the goal of our shadow model, i.e., imitate the behavior of the target model as much as possible. Therefore, we only need to retrain the transferred feature extractor (rather than the entire shadow model) to make it have significant similarity with the first part of the target model. The ablation studies in Section VI-D prove the superiority of this method.

IV. Inference Attack

In this section, we first present our threat model and give the overview of our attack. Then we detail our inference attack against the collaborative inference system.

A. Threat Model

We conduct our attack against a collaborative inference system, where the well-trained target model is partitioned into two parts that are deployed in the IIoT device and the edge server separately. The IIoT device is trustworthy, and the first part of the target model configured on the IIoT device cannot be accessed and queried. This setting is reasonable in IIoT scenarios. We assume an attacker can compromise the edge server and thus can obtain the latter part of the target model. The attacker aims at inferring whether an individual data record is in the training dataset of the target model. The only prior knowledge of the attacker is the latter part of the target model, without access to hyper-parameter, exact training dataset, the architecture and weights of the first part of layers.

Therefore, the attacker needs to train a shadow model to imitate the behavior of the target model as much as possible, and then leverage the shadow model to obtain the membership information of training data. We also assume that the attacker has a dataset $\mathcal{D}_{shadow}$ with the same underlying distribution as the training data $\mathcal{D}_{target}$ of the target model. Note that the amount of data in $\mathcal{D}_{shadow}$ is much less than $\mathcal{D}_{target}$.

B. Overview of Inference Attack

As mentioned in section III-B, our attack method focuses on the collaborative inference system. Specifically, the target model $\mathcal{M}_{target}$ has been already full-trained on target training dataset $\mathcal{D}_{target}$ and divided into two parts, namely $\mathcal{M}_{target}^{first}$ and $\mathcal{M}_{target}^{latter}$. There are many methods to train the target model, such as collaborative training and secure multi-party computation, which are not considered in this paper. Because for a full-trained model, no matter how it is trained, it can be attacked by our method under collaborative inference scenarios, where attackers do not have any prior knowledge of target model's training process.

The first part $\mathcal{M}_{target}^{first}$ is kept at the IIoT device, and the latter part $\mathcal{M}_{target}^{latter}$ is offloaded to the edge server. The objective of the attacker (e.g., the edge server) is to train an attack model $\mathcal{M}_{attack}$ based on Shokri et al.'s method [12], to determine whether or not an individual input sample is in $\mathcal{D}_{target}$. It is emphasized that, compared with existing methods, our attack cannot query $\mathcal{M}_{target}$ at any phase and attackers' prior knowledge is very scarce. Thus, a straightforward application of existing methods is ineffective.

Therefore, we devise a new attack method, where the attacker reconstructs a shadow model by leveraging our *transfer-inherit* shadow learning method. Specifically, the attacker first transfers the feature extractor of existing well-trained models, and then fine-tunes the feature extractor using the dataset $\mathcal{D}_{shadow}$ drawn from the same distribution as $\mathcal{D}_{target}$.

The high-level overview of our attack is shown in Fig. 3.

C. Inference Attack

Based on the particularity of the collaborative inference system, this is, $\mathcal{M}_{target}^{first}$ is kept at the IIoT device and $\mathcal{M}_{target}^{latter}$ is offloaded to the edge server, we propose a new attack scheme. Algorithm 1 briefly describes our attack, which can be organized into three steps: shadow model construction (S_model), attack model construction (A_model) and membership inference (Mem_Infer).

Next we will introduce these three steps in detail.

Shadow model construction

To train an attack model, the edge server requires to train a shadow model $\mathcal{M}_{shadow}$ that behaves similarly to $\mathcal{M}_{target}$. We propose *transfer-inherit* shadow learning, and thus relax key assumptions such as knowledge of model weight and querying to model APIs. Specifically, *transfer* means to transfer a general feature extractor to replace the missing first layers and *inherit* means to fix the known latter layers that are specific to the target task. The objective of both is to make $\mathcal{M}_{shadow}$ imitate the behavior of $\mathcal{M}_{target}$ as much as possible. In detail, with the main idea of transfer learning, the attacker first transfers a feature extractor $\mathcal{M}_{shadow}^{first}$ of well-trained models such as AlexNet and then $\mathcal{M}_{shadow}$ is initialized by splicing $\mathcal{M}_{shadow}^{first}$ and $\mathcal{M}_{shadow}^{latter}$ that equals to $\mathcal{M}_{target}^{latter}$.

The attacker possesses the shadow dataset $\mathcal{D}_{shadow}$, which comes from the same distribution as $\mathcal{D}_{target}$, but does not intersect ($\mathcal{D}_{shadow} \cap \mathcal{D}_{target} = \emptyset$). The whole $\mathcal{D}_{shadow}$ is divided into two parts, including $\mathcal{D}_{shadow}^{train}$ that is used to retrain $\mathcal{M}_{shadow}$ and $\mathcal{D}_{shadow}^{test}$ that is the test dataset. Since only the parameters of $\mathcal{M}_{shadow}^{first}$ now maybe have a large discrepancy from $\mathcal{M}_{target}^{first}$, therefore, instead of training the entire shadow model, the parameters of $\mathcal{M}_{shadow}^{latter}$ are fixed and only retraining the feature extractor $\mathcal{M}_{shadow}^{first}$. It can be formulated by the following optimization process:

$$\mathcal{M}_{shadow}^{first} = argmin_{\mathcal{M}_{shadow}^{first}} \sum_{(x_i,y_i)\in\mathcal{D}_{shadow}^{train}} CE(\hat{y_i}, y_i)$$

where $\hat{y_i} = \mathcal{M}_{shadow}^{latter}(\mathcal{M}_{shadow}^{first}(x_i))$ and CE is cross-entropy. Note that leveraging *transfer-inherit* shadow learning, the training of our shadow model only requires a small number of data, because we just fine-tune the first part $\mathcal{M}_{shadow}^{first}$

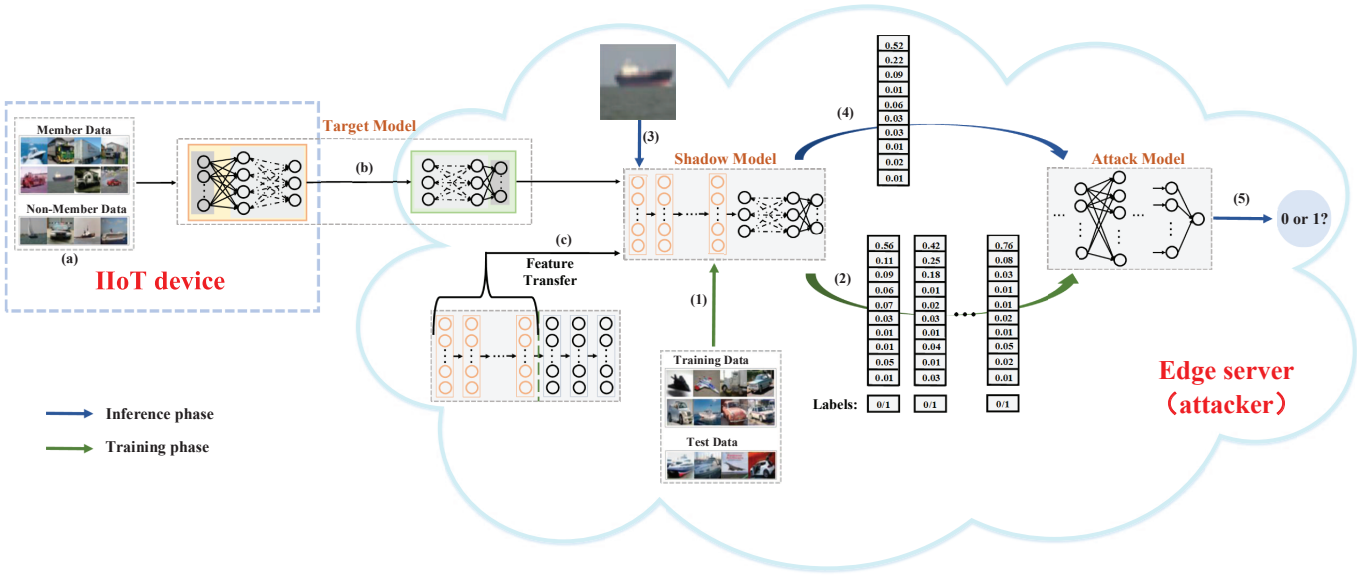


Fig. 3: The high-level overview of our attack. Assuming that the target model is trained with images shown in (a). The first layers of the well-trained model are deployed into a IIoT device and the latter layers are offloaded to the edge server (b). The edge server transfers a feature extractor of existing models to the latter part of the target model, to form the shadow model (c). *Training phase* includes (1) training a shadow model and (2) training a binary classifier as the attack model. During *inference phase*, given the target record, the attacker first (3) queries the shadow model to get the prediction vector (4) and then (5) obtains the inference result $(0$ or $1)$.

instead of training the entire model. A naive method is to train a shadow model from scratch on shadow dataset. However, the effectiveness of attack performance will be seriously reduced in this way, especially when the number of $\mathcal{D}_{shadow}$ is insufficient. We evaluate these different shadow model settings in Section VI-D.

Attack model construction

Now, the attacker has already got a shadow model that behaves similarly to the target model. The main idea behind our attack model is that the membership information of the shadow model is similar to that of the target model. In view of this, we initialize a binary classifier $\mathcal{M}_{attack}$ as our attack model, where the membership information of the shadow model is used as training data of $\mathcal{M}_{attack}$.

In detail, the attacker trains $\mathcal{M}_{attack}$ from scratch on the attack dataset $\mathcal{D}_{attack}$. In order to obtain $\mathcal{D}_{attack}$, the attacker queries $\mathcal{M}_{shadow}$ with its training data $\mathcal{D}^{train}_{shadow}$ and its test data $\mathcal{D}^{test}_{shadow}$, to get corresponding prediction vectors. If the data record belongs to $\mathcal{D}^{train}_{shadow}$, its prediction vector will be labeled as 1. Otherwise, the prediction vector is labeled as 0. Prediction vectors and $1/0$ labels together form $\mathcal{D}_{attack}$.

Specifically, for each $(\mathbf{x}, y) \in \mathcal{D}^{train}_{shadow}$, the attacker computes the prediction vector $\mathbf{y} = \mathcal{M}_{shadow}(\mathbf{x})$ and adds the sample $(\mathbf{y}, 1)$ into $\mathcal{D}_{attack}$. Similarly, for each $(\bar{\mathbf{x}}, y) \in \mathcal{D}^{test}_{shadow}$, the attacker computes the prediction vector $\bar{\mathbf{y}} = \mathcal{M}_{shadow}(\bar{\mathbf{x}})$ and adds the sample $(\bar{\mathbf{y}}, 0)$ into $\mathcal{D}_{attack}$. Finally, the attack dataset $\mathcal{D}_{attack}$ is obtained to train the binary classifier $\mathcal{M}_{attack}$.

Membership inference

Different from existing attacks that can query to target model with individual data records, the attacker in our setting has minimal prior knowledge, namely, the attacker is not allowed to access or query the target model at any phase. To overcome this challenge, the attacker queries the local shadow model instead of the target model to get prediction vector. The key reason why our attack can be successful is that the attacker's local shadow model contains the latter layers of the target model specific to the classification task.

In detail, for each individual data record $\mathbf{x}_{target}$, the attacker queries $\mathcal{M}_{shadow}$ to obtain the corresponding prediction vector $\mathbf{y}_{target} = \mathcal{M}_{shadow}(\mathbf{x}_{target})$, and then feeds it to $\mathcal{M}_{attack}$ to get the final inference result.

V. Experimental Setup

In this section, we introduce three state-of-the-art datasets our attack method used. And subsequently, the basic architecture of the target model, the shadow model and the attack model are illustrated respectively. Finally, we point out three metrics to evaluate our method.

We train all of the models on the server with an Intel(R) Xeon(R) E5-2620 v4(2.10GHz) 2 GPU, 16 CPU and 60 GB of RAM. And we use Pytorch[1] to implement our attack.

A. Datasets

We evaluate our attack over three datasets: MNIST, CIFAR10 and CIFAR100.

MINIST. MNIST[2] is a handwritten digital image dataset, containing $60,000$ training samples and $10,000$ test samples. Each picture in the dataset consists of 28×28 pixels and each

[1]https://pytorch.org/

[2]http://yann.lecun.com/exdb/mnist/

Algorithm 1 MIA Against Collaborative Inference

Input: $\mathcal{M}_{transfer}$: a well-trained model used to transfer, $\mathcal{M}^{latter}_{target}$, $\mathcal{D}_{shadow} = \mathcal{D}^{train}_{shadow} \cup \mathcal{D}^{test}_{shadow}$, $\mathbf{x}_{target}$

Output: 0/1: whether $\mathbf{x}_{target}$ is a member of target model's training dataset.

function S_MODEL($\mathcal{M}_{transfer}, \mathcal{M}^{latter}_{target}, \mathcal{D}_{shadow}$)
// Shadow model construction
  $\mathcal{M}^{first}_{shadow} \leftarrow Extract(\mathcal{M}_{transfer})$
  $\mathcal{M}_{shadow} \leftarrow Splic(\mathcal{M}^{first}_{shadow}, \mathcal{M}^{latter}_{target})$
  $\mathcal{M}_{shadow} \leftarrow train(\mathcal{M}_{shadow}, \mathcal{D}_{shadow})$
return $\mathcal{M}_{shadow}$
end function
function A_MODEL($\mathcal{D}_{shadow}, \mathcal{M}_{shadow}$)
// Attack model construction
  $precision_i \leftarrow \mathcal{M}_{shadow}(\mathbf{x}_i \in \mathcal{D}_{shadow})$
  if $\mathbf{x}_i \in \mathcal{D}^{train}_{shadow}$ **then**
    $label_i = 1$
  else if $\mathbf{x}_i \in \mathcal{D}^{test}_{shadow}$ **then**
    $label_i = 0$
  end if
  $\mathcal{M}_{attack} \leftarrow train(precision, label)$
return $\mathcal{M}_{attack}$
end function
function MEM_INFER($\mathbf{x}_{target}, \mathcal{M}_{shadow}, \mathcal{M}_{attack}$)
// Membership inference
  $pre_x \leftarrow \mathcal{M}_{shadow}(\mathbf{x}_{target})$
return $\mathcal{M}_{attack}(pre_x)$
end function

pixel is represented by a gray value. The label of each picture is encoded by an one-hot vector.

CIFAR10. CIFAR10[3] consists of $60,000$ pictures, where each picture consists of 32×32 color pixels. The whole dataset is divided into 10 classes and each class contains $6,000$ pictures. Among them, $50,000$ pictures are used as the training data while $10,000$ pictures are used as the test data.

CIFAR100. This dataset[3] is similar to CIFAR10, except it has 100 classes, with 500 training images and 100 test images per class.

B. Target Model

We evaluate our target models on previously mentioned three datasets, namely MNIST, CIFAR10 and CIFAR100. Specifically, as shown in Table I, for the MNIST dataset, we train a convolutional neural network (CNN) as the target model with 2 convolutional layers and 2 fully connected layers, and we call it *Mn_CNN*. The *LeNet* including 2 convolutional layers and 3 fully connected layers is adopted on the CIFAR10 dataset and the *AlexNet*, including 5 convolutional layers followed by 3 fully connected layers, is adopted on the CIFAR100 dataset.

We train these target models from scratch using stochastic gradient descent (SGD) optimizer for 100 epochs with learning rate of 0.1 decayed by a factor of 0.1 every 30 epochs, where the momentum is 0.5 and the batch size is 64. The number of training data is 25000 and the test data is 5000. Finally we select one model for each dataset as the target model with the best test accuracy across all the epochs.

[3]http://www.cs.toronto.edu/ kriz/cifar.html

C. Shadow Model

To investigate the effectiveness of our *transfer-inherit* shadow learning (I-T), we compare with the following three conventional shadow model construction methods, namely I-NoT, NoI-T and NoI-NoT.

- *Inherit and Transfer (I-T)*: transfer a feature extractor from a well-trained model and connect it using a fully connected layer or a convolutional layer to the latter part of the DNN inherited from the target model. Specifically, we transfer 5 convolutional layers from a well-trained AlexNet as our feature extractor.
- *Inherit and No-Transfer (I-NoT)*: randomly initialize a feature extractor and connect it to the inherited latter layers. For comparability, we also transfer 5 convolutional layers with randomly initialized parameters.
- *No-Inherit and Transfer (NoI-T)*: transfer the feature extractor from the well-trained AlexNet model and followed by fully connected layers whose parameters are randomly initialized.
- *No-Inherit and No-Transfer (NoI-NoT)*: randomly initialize a CNN, where the architecture and the number of parameters for each layer are the same as our I-T.

We retrain the shadow model using the data that is the same distribution but disjoint with the training data of the target model, and only the transferred or randomly initialized layers are retrained. We use Adam optimizer for 100 epochs with the learning rate of 0.001 and pick a shadow model with the highest testing accuracy across all the epochs.

D. Attack Model

We use the LGBM (Light Gradient Boosting Machine) algorithm to generate our attack model. LGBM is an open source distributed high-performance gradient boosting framework of Microsoft, using a learning algorithm based on the decision tree. It grows trees leaf-wise (best-first) and it will choose the leaf with max delta loss to grow. Since Pytorch provides the API of this algorithm, we can only put in parameters and employ it directly, where we set the number of estimator to 10000 and a reg ambada to 10.

E. Evaluation Metrics

We use three metrics: accuracy, precision and recall to evaluate our attack scheme.

Assuming that b represents the real membership information of the target model. $b = 1$ means an instance $(\mathbf{x}, y)$ belongs to $\mathcal{D}_{target}$, otherwise, it means that this instance is not a training data.

Accuracy refers to the proportion of data records that the attack model correctly infers in the whole given dataset. It can be formulated as

$$Accuracy(attacker) = Pr(\mathcal{M}_{attack}(\mathbf{x}, y) = b \mid b)$$

Precision is the fraction of indeed members in the set inferred as members. It can be formulated as

$$Precision(attacker) = Pr(b = 1 \mid \mathcal{M}_{attack}(\mathbf{x}, y) = 1)$$

Recall is the proportion of data records correctly inferred as members in all indeed members. It can be expressed as

$$Recall(attacker) = Pr(\mathcal{M}_{attack}(\mathbf{x}, y) = 1 \mid b = 1)$$

Meanwhile, we set the baseline standard to 0.5, because the attacker that guesses at random has an accuracy of 0.5.

VI. PERFORMANCE EVALUATION

In this section, we conduct extensive experiments to evaluate the unintended membership privacy risks of the collaborative inference. We start by presenting the results of our attack in normal settings, followed by investigating the impact of different factors on attack performance and finally, comparing with existing works.

A. Performance of our attack

The goal of our attacker is to determine whether a given record is used to train the target model. We evaluate our attack method by inferring all data items in the training and test data of the target model. Table I indicates the training and test accuracy of different target models on the three datasets. They all show good performance on the training data, with accuracy of 99.98%, 98.59% and 99.79% respectively. And for the MNIST dataset, there is no significant gap between training accuracy and test accuracy, which indicates that the target model has strong generalization ability. However, for the other two datasets CIFAR10 and CIFAR100, the gaps are very large, which indicates that the two target models have been severely overfitted.

TABLE I: The training and test accuracy of different target models in three datasets, where the amount of training and test data of each model are 25000 and 5000 respectively.

	Architecture	Training Accuracy	Test Accuracy
MNIST	Mn_CNN (2 Conv+2 Fc)	99.98%	98.38%
CIFAR10	LeNet (2 Conv+3 Fc)	98.59%	57.06%
CIFAR100	AlexNet (5 Conv+3 Fc)	99.79%	37.54%

Fig. 4 depicts the performance of our attack on MNIST, CIFAR10 and CIFAR100, where we split each target model according to Table II. For example, we split the Mn_CNN trained by MNIST after the second convolution layer. The first two convolutional layers are deployed on the IIoT device, and the two fully connected layers are offloaded to the edge server. We can see that our attack shows significant performance for the CIFAR10 and CIFAR100 datasets. For instance, CIFAR10 has a 80.36% accuracy and CIFAR100 has a 81.55% accuracy. Interestingly, for the MNIST dataset, our attack performance has little difference from random guessing. The main reason is that the gap between training and test accuracy of MNIST is only 1.72. According to Section III-C, in this case, the attacker can hardly obtain additional knowledge about the target model's behavior, leading to the failure of our attack.

Since the results of these three metrics show the same trend, in the follow-up experiments, we choose the most representative metric, i.e., accuracy, to record.

TABLE II: The split point of target models.

	Split Point	The IIoT Device	The Edge Server
Mn_CNN	2_{nd} Conv	2 Conv	2 Fc
LeNet	1_{st} Conv	1 Conv	1 Conv + 3 Fc
AlexNet	1_{st} Conv	1 Conv	4 Conv + 3 Fc

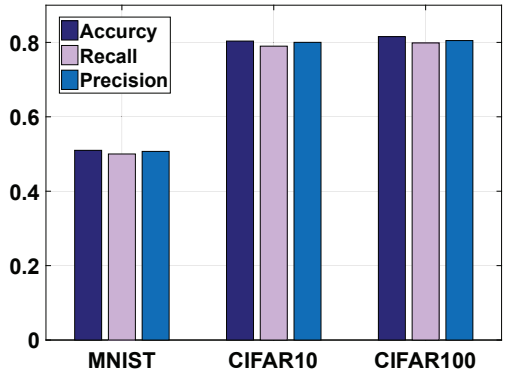


Fig. 4: Performance of our attack under three datasets, where the training and test data of each shadow model are 25000 and 5000 respectively.

B. Effect of the split point

Since our attack is similar to random guessing for the MNIST dataset, we just use two datasets, CIFAR10 and CIFAR100, to evaluate the relationship between the split point and the accuracy of membership inference. As shown in Table III, with different split points, the prior knowledge possessed by the attacker is different. We consider that the more model layers are offloaded to the edge server, the more knowledge the attacker will gain.

In Fig. 5, we show the attack accuracy against the LeNet trained by CIFAR10 at different split points, where the abscissa represents the split point. As we can see, when the attacker occupies a large number of model layers, especially when the split point is *1st Conv*, even with a small amount of training data, i.e., 5000, a high attack accuracy (64.33%) can be achieved. On the contrary, when the attacker has the least knowledge, i.e., only the last fully connected layer, our attack is failed (the attack accuracy is $43.62\% < 50\%$) under the same conditions, and in this case, we can only improve our attack accuracy to 68.94% by increasing the amount of training data of the shadow model to 25000.

The same phenomenon also appears in Fig. 6, which depicts the accuracy of our attack against the AlexNet trained by CIFAR100. Without loss of generality, we can observe that as the number of training data is consistent, the attack accuracy become much lower when the split point is in a deeper layer.

This is because as the split point moves backward, the edge server gains less and less knowledge of the target model, and then the shadow model is increasingly difficult to imitate the behavior of the target model.

TABLE III: The prior knowledge of the attacker under different split points.

	Split Point	The IIoT Device	The Edge Server	
			layers	knowledge
LeNet	1_{st} Conv	1 Conv	1 Conv+3 Fc	++++
	2_{nd} Conv	2 Conv	3 Fc	+++
	1_{st} Fc	2 Conv+1 Fc	2 Fc	++
	2_{st} Fc	2 Conv+2 Fc	1 Fc	+
AlexNet	1_{st} Conv	1 Conv	4 Conv+3 Fc	+++++
	2_{nd} Conv	2 Conv	3 Conv+3 Fc	+++++
	3_{rd} Conv	3 Conv	2 Conv+3 Fc	++++
	4_{th} Conv	4 Conv	1 Conv+3 Fc	+++
	5_{th} Conv	5 Conv	3 Fc	++
	1_{st} Fc	5 Conv+1 Fc	2 Fc	+

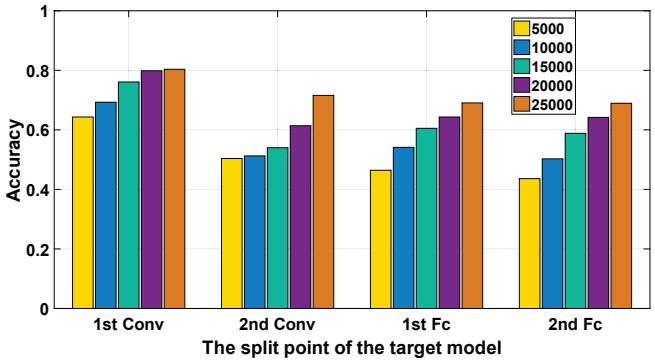


Fig. 5: The attack accuracy against CIFAR10 under different split points and different number of training data of the shadow model.

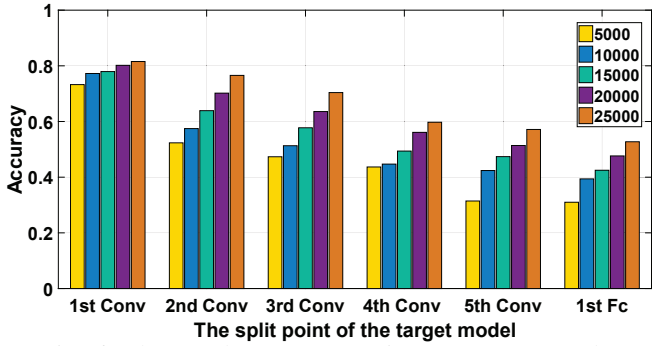


Fig. 6: The attack accuracy against CIFAR100 under different split points and different amount of training data of the shadow model.

C. Effect of the shadow training data

Fig. 5 and Fig. 6 also report the variation of our attack accuracy with the amount of training data of the shadow model. We can observe that for the CIFAR10 and CIFAR100 datasets, when the training data is 25000, the highest attack accuracy can reach 80.36% and 81.55%, and the lowest are 68.94% and 53.73%, respectively, which are still above the baseline standard. When the amount of training data is 5000, the highest attack accuracy can reach 64.32% and 72.26%, however, the lowest accuracy are only 43.62% and 31.03% respectively, where the attack fails.

Therefore, under the same split point, the attack accuracy drops as the amount of training data decreases. The main reason is that if the attacker does not have enough training data to train the shadow model, the shadow model cannot imitate the target model very well. Then in the membership inference stage, the prediction vector of the shadow model for a certain data item will be quite different from the prediction result obtained by querying the target model, resulting in the low performance of our attack.

D. Effect of shadow model initialization

In order to prove the effectiveness of our *transfer-inherit* shadow learning, we conduct ablation studies by comparing with other three initialization methods, namely I-NoT, NoI-T and NoI-NoT mentioned in Section V-C under CIFAR10 and CIFAR100 datasets, where the training data of the shadow model is 25000 and the split point of two target models is *2nd Conv*. We can observe from Fig. 7 that our *transfer-inherit* shadow learning is superior to the other three initialization methods. This means that both *transfer* and *inherit* are indispensable.

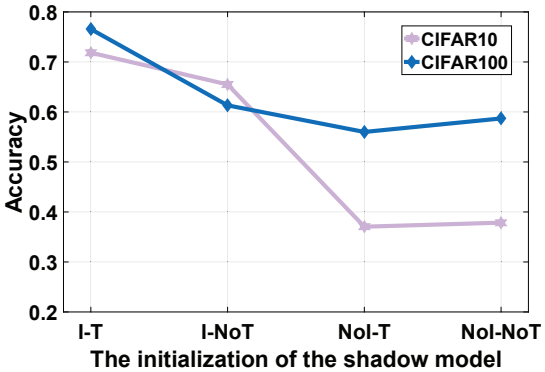


Fig. 7: The attack accuracy under different initialization of the shadow model, where the amount of training and test data are 25000 and 5000 respectively.

Specifically, for the CIFAR10 dataset, our attack accuracy can reach 71.85%, which is much better than the I-NoT method (65.51%), the NoI-T method (37.05%), and the NoI-NoT method (37.86%). For the CIFAR100 dataset, the attack accuracy can achieve 76.56% under our scheme and the gap with other three methods are 15.25%, 20.56% and 17.85% respectively. The main reasons are: (1) by *transfer*ring a feature extractor, the shadow model does not need to be trained from scratch, so a small amount of training data can achieve good performance, and (2) *inherit*ing the latter part of the target model allows the shadow model to better imitate the behavior of the target model.

Fig. 7 also demonstrates that *inherit* is the most important factor of our membership inference attack. For the CIFAR10 dataset, the attack accuracy under I-NoT is 65.51%, which is still top the baseline, however, with the NoI-T method, the attack accuracy is only 37.05%. For the CIFAR100, the attack accuracy also drops the most without our *inherit* method. Therefore, *inherit* is the strongest guarantee of our attack in the query-free case, because the latter layers are specific to the classification task.

E. Effect of querying the target model

During the membership inference phase, traditional black-box MIAs need to query the target model to get the prediction value and then put this value into the attack model to get the inference result. In this subsection, we visualize the impact of query or not on our attack accuracy with CIFAR10 and CIFAR100 datasets under different split points, where the results are shown in Fig. 8 and Fig. 9.

For the LeNet trained by CIFAR10, when the split point is *1st Conv*, regardless of the amount of training data, whether to query the target model has almost no impact on the attack performance. The attack accuracy of both are very high. The same phenomenon also occurs on the AlexNet trained by CIFAR100. Recall that the attacker's prior knowledge is the most with *1st Conv* split point. Because except for the first layer, other layers of the shadow model (including the architecture and parameters) are exactly the same as the target model. The attacker only needs to retrain the first layer and finally, the shadow model must be extremely similar to the target model. Therefore, for a data point, the prediction vector of the target model is approximately similar to the predicted output of the shadow model, resulting in the same membership inference of the attack model.

As the split point moves backward, the attacker gains less and less knowledge of the target model, leading to a large gap between the local shadow model and the target model. Therefore, the prediction value obtained by query-free has little in common with the prediction value obtained by querying the target model (the real prediction), resulting in a poor attack performance. And the same trend is shown when the amount of training data changes. That's a limitation we can't ignore, and we'll leave it for the future work.

F. Comparison with Shokri et al.'s attack

We compare our attack scheme with the attack proposed by Shokri et al. [12]. For the Shokri et al.'s attack, we train multiple shadow models and attack models following the original configuration. And for our attack, the split point is *1st Conv*. Fig. 10 quantifies this comparison under different number of training data. We can observe that in the case of query-free, Shokri et al.'s attack is completely useless, while our attack can achieve good performance. Specifically, For CIFAR10 and CIFAR100 datasets under Shokri et al.'s attack, the accuracy only up to 37.86% and 58.7% respectively, and the attack precision is all around 50%, which are much lower than our attack.

We also firmly believe that in the real IIoT environment, where the first layers of the target model is completely black-box and query-free, existing works can only achieve similar performance to Shokri et al.'s and far below our attack. Therefore, it is ineffective for a straightforward extension of existing attack methods.

G. Mitigation strategies

We tentatively propose two possible mitigation methods to reduce the privacy risk during collaborative inference.

Use regularizer to reduce overfitting. Recall that overfitting of the target model is the main culprit that triggers our attack. And it's also a serious problem in AI because it limits the predictive and generalization ability of DNNs. Therefore, the regularization techniques can be used to defend our attack. Shokri et al. [12] pointed out that standard L_2-norm regularizer is an effective mitigation. This regularizer is added as a penalty term to the loss function and it can be expressed as $\lambda \sum_i \theta_i^2$, where θ_is are the weights of the target model.

Split the target model into three parts. Another key reason why our attack can be successful is that the latter layers of the target model still remember information about training data. A fundamental mitigation is to reduce the latter offloaded layers that specific to target task. Further, maybe we can split the target model into three parts [30] and only offload the computationally large middle part to the edge server.

VII. Conclusion

In this paper, we design the first membership inference attack against collaborative inference to profoundly reveal the serious privacy threat of training data. Specifically, we consider the more realistic situation of collaborative learning in the IIoT environment, where attackers only know the latter part of the target model, and we propose *transfer-inherit* shadow learning to relax the key assumptions of existing works. We evaluate our attack on different datasets and various settings, and the experimental results show that it has high effectiveness. Through this work, we hope to re-examine the privacy issues of training data in collaborative inference. To remedy our attack, we also point out two mitigation methods. In future work, we will focus on designing attacks targeting more generalized models and further improving the attack performance.

Acknowledgement

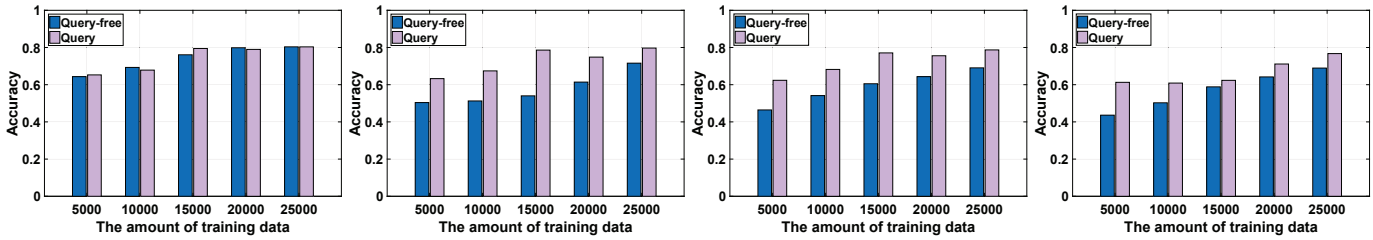
This work is supported by the National Key R&D Program of China.

References

[1] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

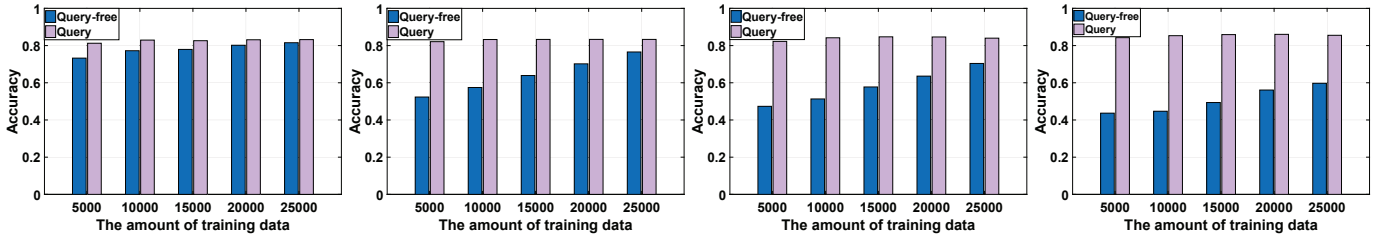
[2] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial internet of things (iiot): An analysis framework," *Computers in industry*, vol. 101, pp. 1–12, 2018.

[3] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue *et al.*, "An empirical evaluation of deep learning on highway driving," *arXiv preprint arXiv:1504.01716*, 2015.



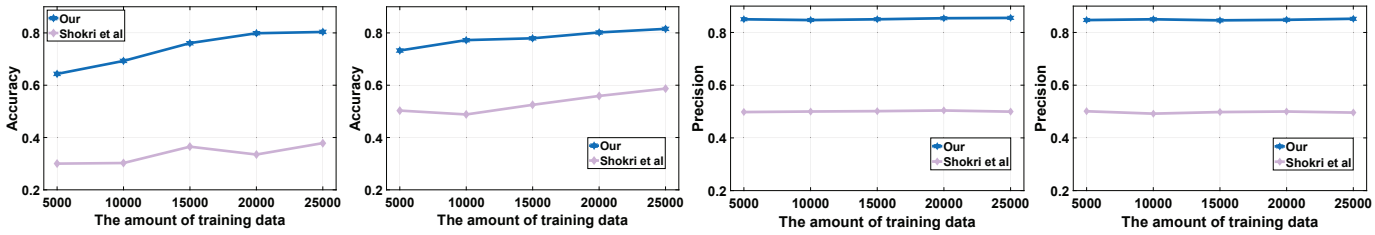
(a) (b) (c) (d)

Fig. 8: The impact of query or not on our attack against the LeNet trained by CIFAR10 under different number of training data from 5000 to 25000 and different split points. The split point at Fig. 8a is *1st Conv*, at Fig. 8b is *2nd Conv*, at Fig. 8c is *1st Fc* and Fig. 8d is *2nd Fc*.



(a) (b) (c) (d)

Fig. 9: The impact of query or not on our attack against the AlexNet trained by CIFAR100 under different number of training data from 5000 to 25000 and split points. The split point at Fig. 9a is *1st Conv*, at Fig. 9b is *2nd Conv*, at Fig. 9c is *3rd Conv* and Fig. 9d is *4th Conv*.



(a) (b) (c) (d)

Fig. 10: Comparison of our attack with Shokri et al.'s under different amount of training data based on two datasets, CIFAR10 ((a), (c)) and CIFAR100 ((b), (d)).

[4] S. Tuli, N. Basumatary, S. S. Gill, M. Kahani, R. C. Arya, G. S. Wander, and R. Buyya, "Healthfog: An ensemble deep learning based smart healthcare system for automatic diagnosis of heart diseases in integrated iot and fog computing environments," *Future Generation Computer Systems*, vol. 104, pp. 187–200, 2020.

[5] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 615–629, 2017.

[6] G. Xu, H. Li, H. Ren, K. Yang, and R. H. Deng, "Data security issues in deep learning: Attacks, countermeasures and opportunities," *IEEE Communications Magazine*, vol. 57, no. 11, pp. 116–122, 2019.

[7] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *Journal of Network and Computer Applications*, vol. 116, pp. 1–8, 2018.

[8] J. Chi, E. Owusu, X. Yin, T. Yu, W. Chan, P. Tague, and Y. Tian, "Privacy partitioning: Protecting user data during the deep learning inference phase," *arXiv preprint arXiv:1812.02863*, 2018.

[9] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *Proceedings of IEEE ICDCS*, 2017, pp. 328–339.

[10] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *Proceedings of ACSAC*, 2019, pp. 148–162.

[11] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," *arXiv preprint arXiv:1905.11742*, 2019.

[12] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proceedings of IEEE S&P*, 2017, pp. 3–18.

[13] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proceedings of NDSS*, 2019.

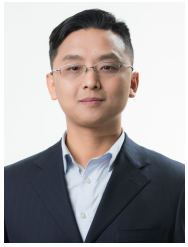
[14] K. Leino and M. Fredrikson, "Stolen memories: Leveraging model memorization for calibrated white-box membership inference," *arXiv preprint arXiv:1906.11798*, 2019.

[15] 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 *Proceedings of IEEE S&P*, 2019, pp. 739–753.

[16] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, "Logan: evaluating privacy leakage of generative models using generative adversarial networks," *arXiv preprint arXiv:1705.07663*, 2017.

[17] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," *arXiv preprint arXiv:2003.10595*, 2020.
[18] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *Proceedings of USENIX Security*, 2014, pp. 17–32.
[19] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of ACM CCS*, 2015, pp. 1322–1333.
[20] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction apis," in *Proceedings of USENIX Security*, 2016, pp. 601–618.
[21] W. Hua, Z. Zhang, and G. E. Suh, "Reverse engineering convolutional neural networks through side-channel information leaks. in 2018 55th acm/esda," in *Proceedings of IEEE Design Automation Conference*, vol. 10, no. 3195970.3196105, 2018.
[22] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *Proceedings of IEEE CVPR*, 2019, pp. 4954–4963.
[23] S. J. Oh, B. Schiele, and M. Fritz, "Towards reverse-engineering black-box neural networks," in *Proceedings of ICLR*, 2018, pp. 121–144.
[24] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of ACM AsiaCCS*, 2017, pp. 506–519.
[25] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, "Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks," in *Proceedings of USENIX Security*, 2019, pp. 321–338.
[26] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2019.
[27] T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project adam: Building an efficient and scalable deep learning training system," in *Proceedings of USENIX OSDI*, 2014, pp. 571–582.
[28] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.
[29] B. Wang, Y. Yao, B. Viswanath, H. Zheng, and B. Y. Zhao, "With great training comes great vulnerability: Practical attacks against transfer learning," in *Proceedings of USENIX Security*, 2018, pp. 1281–1297.
[30] 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.

Hanxiao Chen (S'20) is currently a Ph.D. student at the School of Computer Science and Engineering (School of Cyber Security), University of Electronic Science and Technology of China, China. Her research interests are the intersection of AI & Machine Learning with Security & Privacy, such as membership inference attack/defenses and secure multi-party computation.



Hongwei Li (M'12-SM'18) is currently the Head and a Professor at Department of Information Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China. He received the Ph.D. degree from University of Electronic Science and Technology of China in June 2008. He worked as a Postdoctoral Fellow at the University of Waterloo from October 2011 to October 2012. His research interests include network security and applied cryptography. He is the Senior Member of IEEE, the Distinguished Lecturer of IEEE Vehicular Technology Society.

Guishan Dong is currently a professor in the No.30 Institute of China Electronics Technology Group Corporation. He received the Ph.D. degree from University of Electronic Science and Technology of China in December 2009. His research interests include network security and applied cryptography. He is the standing director of Association for Cryptologic Research in China, and a member of the Cryptography Standardization Committee.

Meng Hao (S'19) received his B.S. degree in Information Security from University of Electronic Science and Technology of China (UESTC) in 2018. Currently, he is working toward the Ph.D. degree at School of Computer Science and Engineering (School of Cyber Security), UESTC. His research interests include Secure Multi-Party Computation and Machine Learning Security.

Guowen Xu (S'15) is currently a Ph.D. student at the School of Computer Science and Engineering, University of Electronic Science and Technology of China, China. He has published more than 30 papers in international conferences and journals, such as ACM CCS, ACM ACSAC, ACM ASIACCS, IEEE TDSC, IEEE TIFS, etc. His research interests include Secure Outsourcing Computing and Security&Privacy issues in Deep Learning.

Xiaoming Huang is the senior engineer and the general manager of Technology Marketing Department of CETC Cyberspace Security Research Institute Co., Ltd. His research interests include cognitive domain security, cryptography and information security theory, trusted computing and trusted network technology, computer and communication security issues. He is a member of Sichuan electronic information expert group.

Zhe Liu (SM'18) is a professor in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. He received the B.S. and M.S. degrees from Shandong University, China, in 2008 and 2011, respectively, and the Ph.D. degree from the Laboratory of Algorithmics, Cryptology and Security, University of Luxembourg, Luxembourg, in 2015. His research interests include security, privacy and cryptography solutions for the Internet of Things. He has co-authored over 100 research peer-reviewed journal and conference papers. He was a recipient of the prestigious FNR Awards-Outstanding Ph.D. Thesis Award in 2016, ACM CHINA SIGSAC Rising Star Award in 2017 as well as DAMO Academy Young Fellow in 2019. He has served as program committee member in more than 50 international conferences.