

Exploiting Unintended Property Leakage in Blockchain-Assisted Federated Learning for Intelligent Edge Computing

Meng Shen, *Member, IEEE*, Huan Wang, Bin Zhang, Liehuang Zhu, *Member, IEEE*,
Ke Xu, *Senior Member, IEEE*, Qi Li, *Senior Member, IEEE*, and Xiaojiang Du, *Fellow, IEEE*

Abstract—Federated learning (FL) serves as an enabling technology for intelligent edge computing, where high-quality machine learning (ML) models are collaboratively trained over large amounts of data generated by various IoT devices while preserving data privacy. To further provide data confidentiality, computation auditability and participant incentives, the blockchain framework has been incorporated into FL. However, it is an open question whether the model updates from participants in blockchain-assisted FL can disclose properties of the private data the participants are unintended to share.

In this paper, we propose a novel property inference attack that exploits the unintended property leakage in blockchain-assisted FL for intelligent edge computing. More specifically, we present an active attack to learn the property leakage from model updates of participants and to identify a set of participants with a certain property. We also design a dynamic participant selection strategy tailored to the setting of large-scale FL, which accelerates the selection process of target participants and improves attack accuracy. We evaluate the proposed attack through extensive experiments with publicly-available datasets. The experimental results demonstrate that the proposed attack is effective and efficient in inferring various properties of training data, while maintaining high quality of the main tasks in FL.

Index Terms—Edge computing, federated learning, property inference, blockchain, IoT

This work is partially supported by Beijing Nova Program with No. Z201100006820006, NSFC Projects with No. 61972039, 61932016 and 61872041, Beijing Natural Science Foundation with No. 4192050, Zhejiang Lab Open Fund with No. 2020AA3AB04, China National Funds for Distinguished Young Scientists with No. 61825204, Beijing Outstanding Young Scientist Program with No. BJJWZYJH01201910003011, BNRist with No. BNR2019RC01011, Science and Technology Planning Project of Guangdong Province with No. LZC0023 and LZC0024, PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019).

M. Shen and L. Zhu are with the School of Cyberspace Security, Beijing Institute of Technology, Beijing 100081, China. M. Shen is also with Peng Cheng Laboratory (PCL), Shenzhen 518066, China (e-mail: {shenmeng, liehuangz}@bit.edu.cn).

H. Wang is with the School of Computer Science, Beijing Institute of Technology, Beijing 100081, China (e-mail: wanghuan6693@163.com).

B. Zhang is with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: bin.zhang@pcl.ac.cn). B. Zhang is the corresponding author.

K. Xu is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China, also with the Beijing National Research Center for Information Science and Technology (BNRist), Beijing 100084, China, and also with Peng Cheng Laboratory (PCL), Shenzhen 518066, China (e-mail: xuke@tsinghua.edu.cn).

Q. Li is with the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China (e-mail: qi.li@sz.tsinghua.edu.cn).

X. Du is with the Department of Computer and Information Sciences, Temple University, Philadelphia PA19122, USA (e-mail: dxj@ieee.org).

Copyright (c) 20xx IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

I. INTRODUCTION

IN recent years, the rapid advancement of Internet of Things (IoT) results in a huge amount of data gathered from various IoT devices. The high-performance machine learning models require large amounts of data to perform data classification and prediction of future events [1]. To address the limitations of data privacy and network bandwidth, edge computing offloads computation resources and data to IoT devices. Federated Learning (FL) and blockchain are emerging paradigms of distributed learning [2–4] that stores and processes data locally, which has been extended to the edge computing and used in various domains [5, 6]. With the advanced features such as anonymity and traceability, blockchain has emerged as a promising technology to provide distributed secure solutions in FL which provides a guaranteed collaborative scheme among untrusted participants and server for efficient model training [4, 7–9].

Although the blockchain-assisted FL avoids the sharing of participants' data and guarantees the credibility and integrity of data, the shared model updates still reveal private information of participants' training datasets. An important question naturally arises: *what can be disclosed about the participants' private dataset from the model updates in blockchain-assisted FL?*

Existing studies have investigated various privacy violations in the federated setting, such as membership inference attacks and property inference attacks. In the membership inference attack, an adversary can determine if an exact data record was used to train the model [10–12]. In the property inference attack, an attacker can infer properties of the training data that is uncorrelated with the main task, e.g., inferring the hair color or the race (as a property) of the images used to train a gender classifier [10, 12, 13].

In this paper, we focus on the inference of *unintended properties* of the participants' training data, i.e., those properties that hold for the training data of certain subsets of the participants, but are not the *global* properties of the training data as a whole. Different from prior work [12] that assumes an adversarial participant, we explore the unintended property leakage in FL with an adversarial *server*, as the server potentially has a greater attack capability, e.g., having a global view of the model updates of all participants, or even determining the participants involved at each iteration of training. This inference enables the server to extract valuable properties that

participants are not intend to share and are independent of the features that characterize the main classifier.

In general, a practical property inference attack launched by a central server in FL has two goals. First, high inference accuracy is always desirable, which enables the adversary to correctly excavate as many private properties of participants' training data as possible. The second goal is to maintain high utility of participants' training data to ensure the quality of the classifier of the main task. An ideal property inference attack should have a negligible impact on the main task.

Recently, the cryptography is widely used to protect the privacy of various data, such as graph [14] and text data [15]. In order to reduce the risk of information leakage from model updates in FL, several secure aggregation schemes and multi-party secure computing algorithms [16, 17] have been proposed to protect participants' local updates. Thus, we assume that an active adversary only obtains the aggregated model without knowing each participant's plaintext model updates. We also assume that the adversary has white-box knowledge about the federated model. Instead of inferring the properties of individual participants, the goal in our attack is to infer a subset of participants with the target properties that are of interest to the adversary, while maintaining high quality of the classifier trained as the main task. It is a serious privacy risk for participants' sensitive data, especially in the field of health care and finance.

We design a participant selection strategy that can be actively launched by the server, which iteratively selects the participants whose training data is more likely with the target properties. Intuitively, the effectiveness and efficiency of the selection strategy are contradictory. Thus, we design a strategy with adjustable parameters, which enables the adversary to select appropriate parameters to meet their inference requirements.

We evaluate the effectiveness and efficiency of the proposed attack on several public datasets (i.e., CelebA [18], LFW [19], MNIST and CASIA-WebFace [20]) with up to 1,000 participants. The results show that the attack accuracy is mostly above 80%, even with 1,000 participants. For example, when the main task is the gender classifier trained on the CASIA dataset with 1,000 participants, an adversary can infer 100 participants with "hair-color:black" with a considerable accuracy of 82.3%. Moreover, it almost preserves the same utility of the gender classifier, where the prediction accuracy slightly drops from 82.4% to 82.0% (LFW) and from 80.7% to 80.2% (MNIST).

Our contributions are summarized as follows.

- We train a meta-classifier to exploit the property leakage in the training process of federated learning.
- We design a selection strategy to iteratively infer a subset of participants with target properties from all participants in the large-scale federated learning.
- We evaluate the proposed attacks on real-world datasets, which demonstrates that our attacks can efficiently infer the properties of participants' private data.

The remainder of this paper is organized as follows. We introduce the related work and problem statement in Section II

and Section III. Next, we present the property attacks in Section IV. We describe the experimental datasets and evaluation results in Section V and Section VI, respectively. Finally, we conclude this paper in Section VII.

II. RELATED WORK

FL provides a parallel scheme for participants to learn a collaborative model and achieves edge intelligent by learning from distributed data. The collaborative idea has been widely used in many fields [21, 22] and in the FL, it can reduce the privacy risk of directly sharing data [23]. Recently, blockchain is an emerging parallel that brings opportunities to the traditional information-centric networks [15, 24, 25] and has been widely used in FL for edge computing to provide data confidentially, computation auditability, device authentication for cross-domain industrial IoT and participant incentives [4, 7, 26, 27].

However, the model sharing does disclose the unintended leakage of the training data, which leads to various attacks against neural networks in the black-box setting or white-box setting. Here we briefly summarize the inference attacks from two aspects, membership inference attacks and property inference attacks.

Membership inference attacks. Shokri et al. [28] proposed a membership attack to infer whether the target data records are the training data of the target model in a black-box setting. Yeom et al. [29] analyzed the influence of overfitting on the membership attacks. Long et al. [30] showed that the membership attacks were still effective even in the well-generalized learning models, because of the complexity and memorability of deep neural networks. Hayes et al. [31] presented the membership attacks against generative models combining the generative and discriminative models of generative adversarial networks (GAN) [32].

In collaborative learning, shokir et al. [11] designed a membership attack model during the training phase in a white-box setting, including passive and active attacker based on the different adversary prior knowledge. Melis et al. [12] developed passive and active membership inference attacks to extract the unintended features from the model updates in the collaborative learning.

Property inference attacks. Model inversion attacks [33] were proposed to construct the inputs of a certain class. Ateniese et al. [34] built a meta-classifier to infer unexpected but useful statistical information of training dataset from ML classifiers. Ganju et al. [35] proposed the property inference attacks on fully connected neural networks to infer the global properties of training dataset and used the permutation invariant representations to simply the structures of networks.

In collaborative learning, the studies [10, 13] reconstructed the representation of a class based on the observing updates from participants using the generative adversarial networks. Wang et al. [13] proposed a novel GAN with a multi-task discriminator to recover participants' specified private data from the server side. Melis et al. [12] designed a mini-batch property attack model to extract the unintended features in the federated learning [2] and collaborative learning with syn-

TABLE I
THE MAIN NOTATIONS USED IN THIS PAPER

| Notations | Paraphrases |
|------------------|---|
| \mathbf{w} | Deep neural network model |
| $L(\mathbf{w})$ | Loss function of model \mathbf{w} |
| T | Required iterations of FT |
| \mathbf{w}_t | Global model at iteration t |
| \mathbf{w}_t^i | Local model of participant i at iteration t |
| N | The number of total participants in FL |
| \mathcal{P} | The target property that an adversary wants to infer |
| M | The number of participants with property \mathcal{P} |
| n_t | The number of selected participants at iteration t |
| K | The number of target participants with property \mathcal{P} |
| c | The size of participant sets |
| b | The number of screened participant sets |

chronized gradient updates [3] by active attacker and passive attacker.

By contrast, our property inference attacks could infer a set of participants with target properties by a server who can only obtains the aggregated global models, which achieves a more generic attack and has negligible impact on the FL tasks.

III. PROBLEM STATEMENT

In this section, we present the background, threat model and design goals of property inference attacks.

A. Background of Blockchain-Assisted Federated Learning

FL is an emerging paradigm of distributed learning with multiple participants. Participant i with dataset D_i , at iteration¹ t ($t \in [0, T]$), trains the local model \mathbf{w}_t^i that minimizes the local loss function $L_i(\mathbf{w}_t^i)$. The loss function is defined as

$$L_i(\mathbf{w}_t^i) = \frac{1}{|D_i|} \sum_{j \in D_i} l_j(\mathbf{w}_t^i), \quad (1)$$

where $l_j(\mathbf{w}_t^i)$ is the loss function of participant i on data sample (x_j, y_j) . Define the global dataset as $D = \bigcup_{i=1}^N D_i$, the objective of FL at iteration t is to train a global model \mathbf{w}_t to minimize the global loss function $L(\mathbf{w}_t)$ as shown in Equation (2).

$$L(\mathbf{w}_t) = \frac{1}{|N|} \sum_{i \in N} \sum_{j \in D_i} \frac{L_i(\mathbf{w}_t^i)}{|D_i|}, \quad (2)$$

where N is the number of participants and $|\cdot|$ denotes the size of sets.

The distributed technologies, e.g., federated learning and blockchain promote the development of edge computing. Edge computing provides the specific application scenarios for the combination of federated learning and blockchain which widely used for enhancing security in areas such as IoT [36] and vehicular networks [8]. The general process of blockchain-assisted FL architecture [8, 9] typically consists of three phases: nodes selection, local training, and global aggregation, as shown in Fig. 1.

¹The iteration is also called *round* in the literature [2, 12, 17].

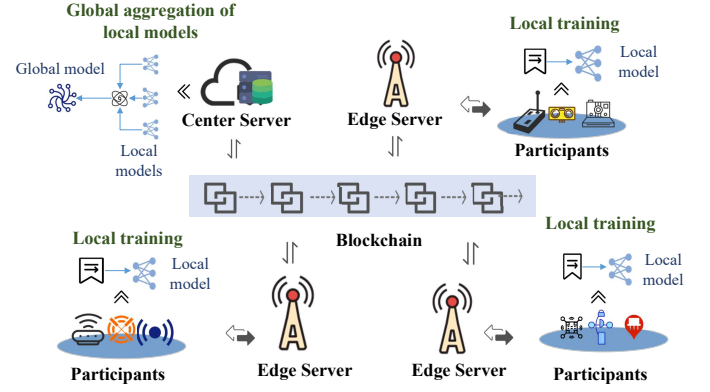


Fig. 1. A typical scheme of blockchain-assisted federated learning for intelligent edge computing.

1) *Nodes selection*: At the beginning, the central server initializes the global model. Then the central server selects the node to participate the model training.

2) *Local training*: The local training is implemented with gradient descent. Upon receiving \mathbf{w}_{t-1} in iteration t , each participant i updates the local model \mathbf{w}_t^i on the local data D_i as illustrated in Equation (1). Then participant i sends the updated model \mathbf{w}_t^i to the nearby edge server and uploads it to the blockchain for further verification and aggregation.

$$\mathbf{w}_t^i = \mathbf{w}_{t-1} - \eta \nabla L_i(\mathbf{w}_{t-1}) \quad (3)$$

3) *Global aggregation*: The aggregator (e.g., a centralized server) retrieves the updated models from the permissioned blockchain and aggregates local models \mathbf{w}_t^i from participating nodes to a global model \mathbf{w}_t .

$$\mathbf{w}_t = \frac{1}{n} \sum_{i=1}^n \mathbf{w}_t^i \quad (4)$$

where n is the number of nodes in iteration t .

B. Threat Model

In this paper, we assume the server is an adversary. In the blockchain-assisted federated learning settings, the adversary is the edge server. It is worth mentioning that in the federated learning commonly used, our attack is still valid and the adversary is the central server in this setting. We also assume that the adversary has a white-box access to the structure of federated model and federated algorithm. The assumption is also commonly used in the literature [11, 37].

In this setting, an adversary is unnecessary to strictly follow the procedures of the main task. Instead, he can take active actions (e.g., dynamically selecting the participants involved each iteration) to improve the accuracy or efficiency of the property inference attack. We assume that secure aggregation schemes [17] are employed to protect the model updates of individual participants. Thus, the adversary has only access to the global model aggregated from the selected participants at each iteration. In our attack, the goal turns to infer a certain number of participants that are most likely to have the desired properties in their training data.

In addition, to perform the property inference attacks, the adversary needs an auxiliary dataset that has the similar distribution with the participants' private data. That is because the datasets with similar distribution have similar gradients of deep neural networks [11, 28, 35]. The auxiliary data record is correctly labeled with the property an adversary wants to infer, as well as the label of main task.

If the adversary does not have access to similar training dataset (e.g., without publicly available datasets), he can compromise one of the participants to obtain the auxiliary dataset, as evidenced in FL with large-scale participants [38].

C. Design Goals

An ideal property inference attack should allow an adversary to extract the unintended property leakage of participants' private data from the model updates, while preserving the quality of the main task classifier. Therefore, the design goals can be described in the following aspects:

- *High accuracy.* For an active adversary, the goal is to infer a certain number of participants with the target property. The attack accuracy represents the ratio of correct inferences, i.e., the percentage of participants with the target property in the inferred participants.
- *High efficiency.* In the property inference attack, the attack efficiency is the required iterations of this attack during the federated learning process. We choose the iterations of federated training rather than running time of the attack for two reasons. Firstly, the iterations represent the communications between adversary and participants, the fewer communications, the more effective of attack. Another reason is that the time cost of each iteration depends on the performance of different machines.
- *Low overhead.* In the active attack scenario, the adversary actively selects certain participants in the training process of FL, which can have an impact on the accuracy of main tasks. Thus, we use overhead to evaluate the impact of the proposed attacks on the main tasks of FL.

IV. PROPERTY INFERENCE ATTACKS

In this section, we describe the attack model used to extract the property leakage of private datasets of participants from model updates.

A. Property Inference Attack Model

We present a high-level overview of the property inference attacks in the Fig. 2. As mentioned above, for the adversary, he takes steps 1, 2, 3 and 4 to train the attack model on the auxiliary dataset. Then, he feeds the global updates \mathbf{w}_t to the attack model and uses the selection strategy (step 6) to select participants with the target property based on the outputs of the attack model.

Define \mathcal{P} as the target property the adversary aims to infer, and \mathcal{H} as a meta-classifier to determine whether the target participant's private data has the property \mathcal{P} or not.

Following researches [11, 35], to train \mathcal{H} , the adversary has an auxiliary dataset denoted by D_a , which has similar

distribution with the training data for the federated model. $D_a = \{D_a^{\mathcal{P}}, D_a^{\bar{\mathcal{P}}}\}$, where $D_a^{\mathcal{P}}$ and $D_a^{\bar{\mathcal{P}}}$ are the auxiliary datasets with and without the property \mathcal{P} , respectively. The property inference attack mainly consists of three phases: training data generation, attack model training, and property inference, as illustrated in Algorithm 1.

Generating training dataset of attack models. To train the attack model \mathcal{H} , an adversary first generates the training dataset of \mathcal{H} . \mathcal{H} is used to learn the differences of gradients of model updated on the private datasets with and without \mathcal{P} . Therefore, the server mimics the behavior of the target model, and divides the auxiliary dataset into N pieces. Then, at each iteration t , the adversary trains the federated model \mathbf{w}_t on N auxiliary datasets and obtains gradients on auxiliary datasets $D_a^{\mathcal{P}_i}$ ($i \in [1, x]$) and $D_a^{\bar{\mathcal{P}}_j}$ ($j \in [1, y]$) after obtaining the global model \mathbf{w}_{t-1} , just like the participants in the federated learning. x is the number of divided pieces with the property \mathcal{P} and y is the number of divided pieces without \mathcal{P} , where $D_a^{\mathcal{P}} = \sum_{i=1}^x D_a^{\mathcal{P}_i}$, $D_a^{\bar{\mathcal{P}}} = \sum_{j=1}^y D_a^{\bar{\mathcal{P}}_j}$. The update rule is defined in Equation (5),

$$\begin{aligned} g_a^{\mathcal{P}} &= \nabla L(D_a^{\mathcal{P}_i}, \mathbf{w}_{t-1}) \\ g_a^{\bar{\mathcal{P}}} &= \nabla L(D_a^{\bar{\mathcal{P}}_j}, \mathbf{w}_{t-1}) \end{aligned} \quad (5)$$

where $L(D, \mathbf{w})$ is described in Equation (1).

To simplify the input features and attack model, we extract only the parameters of the last fully-connected layer. The reason is that the last layer of the neural network leaks more information about the training dataset [11]. Hereafter, \mathbf{g} represents the gradients of the last layer.

Training phase of the attack model. The adversary builds the training data of the attack model \mathcal{H} by labeling the gradients $g_a^{\mathcal{P}}$ and $g_a^{\bar{\mathcal{P}}}$ as \mathcal{P} and $\bar{\mathcal{P}}$, respectively.

Prediction phase of the attack model. The adversary aggregates the global updates and uses it as the input feature of \mathcal{H} . Then, the attack model \mathcal{H} outputs a score in $[0, 1]$: the closer the score is to 1, the more likely the updates have property \mathcal{P} , and vice versa. After that, the adversary leverages the participant selection strategy to iteratively select participants with the property \mathcal{P} based on the predictions of \mathcal{H} .

B. Naive Selection Strategy

To select a set of participants with the target property, we first consider a naive selection strategy (NSS). At each iteration of FL, the server randomly selects n participants involved in model training. If we aim to infer K participants with the target property, a naive selection strategy is to select K ($K = n$) participants at each iteration, and infer the set of participants with the largest probability of having the target property.

The naive selection strategy can be described as follows:

1) *Partitioning participants.* The adversary divides N participants into multiple sets, each of which contains K participants. Thus, the number of selected participants n at each iteration equals K .

2) *Screening out participants.* At each iteration, the server selects a set of K participants and predicts the probability that

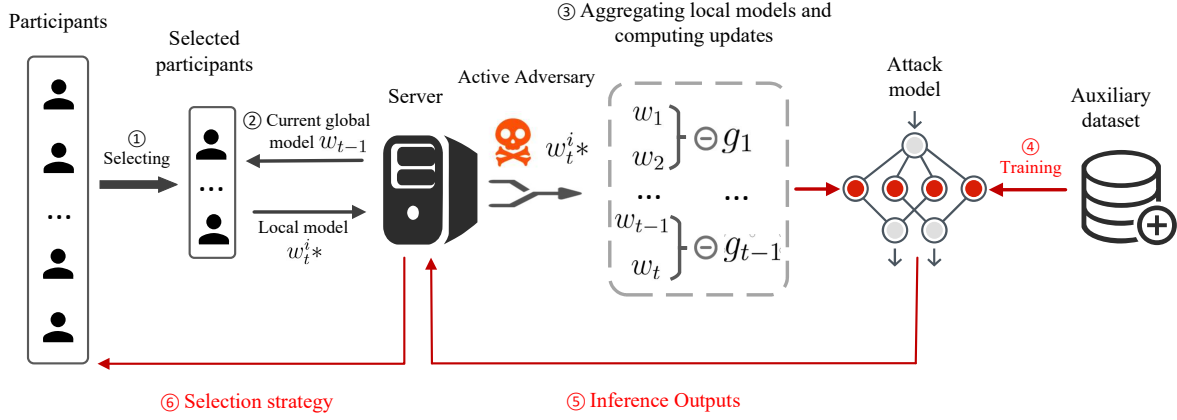


Fig. 2. The overview of the property inference attack. A honest server performs Steps 1, 2 and 3. The active adversary only obtains global model w_t at t -th iteration and conducts Steps 4-6 to perform the attack.

Algorithm 1 Inference Attack Model

Input: auxiliary dataset $D_a = \{D_a^P, D_a^{\bar{P}}\}$, model w_t
Output: Attack model \mathcal{H}

```

1: Server executes: Initialize global model  $w_0$ ,  $T$ ;
2: for  $t = 0$  to  $T$  do
3:   /* Generating training datasets of  $\mathcal{H}$  */
4:    $D_c^P = \phi, D_c^{\bar{P}} = \phi$ 
5:   for  $k = 1$  to  $K$  do
6:     for each  $D_a^{P_i}$  ( $i \in [1, x]$ ),  $D_a^{\bar{P}_j}$  ( $j \in [1, y]$ )
7:        $g_a^P = \nabla L(D_a^{P_i}, w_{t-1}), g_a^{\bar{P}} = \nabla L(D_a^{\bar{P}_j}, w_{t-1})$  do
8:          $D_c^P \leftarrow D_c^P \cup (g_a^P, P), D_c^{\bar{P}} \leftarrow D_c^{\bar{P}} \cup (g_a^{\bar{P}}, \bar{P})$ 
9:       end for
10:    end for
11:    Train attack model  $\mathcal{H}$ , given  $D_c^P, D_c^{\bar{P}}$ 
12:     $S_t = \text{Select}(n)$  /* Participant selection strategy */
13:    for each  $i \in S_t$  do
14:       $w_t^i = \text{ClientUpdate}(w_{t-1})$ 
15:    end for
16:     $w_t = \sum_{k=1}^n \frac{1}{n} w_t^k$ 
17:    /* Prediction phase of attack model */
18:  end for

```

the global update of the selected participants has a certain property. After selecting all participant sets, the adversary obtains the corresponding predictions of \mathcal{H} . Then, he screens out the last one sets of participants ($b \geq 1$) based on the rank of predictions, as the prediction represents the probability that the set of participants has the property. Thus, the adversary selects the participants may have the property \mathcal{P} . Repeat the above steps until only one set of participants is left.

There are two main problems in NSS. First, the accuracy of \mathcal{H} will drop as the number of target participants K increases. This is because local updates are averaged, making it more difficult to extract information from the updates. Second, given the number of participants N , the required iterations of this attack are only affected by the number of target participants K . For instance, if $N = 100$ and $K = 10$, the number of participant sets is $\lfloor N/K \rfloor = 10$, and the adversary needs 10 iterations to traverse all participant sets. This limits the flexibility in participant selection and thus reduces the attack efficiency.

C. Dynamic Selection Strategy

To select a given number of participants more efficiently, while achieving high accuracy of the attack model, we propose a dynamic selection strategy (DSS) to adaptively select participants at each iteration.

The basic idea of DSS is illustrated in Fig. 3. We use two parameters c and b to dynamically change the number of selected participants and the number of screened participant sets. At each iteration, the number of selected participants is set as $n = \lfloor N/c \rfloor$, where c represents the proportion of the selected participants among all participants. Also, the number of screened participants equal $b \times n$, where n is the size of the selected participants set and b is the number of screened participant sets.

Algorithm 2 Dynamic Selection Strategy (DSS)

Input: Attack model \mathcal{H} and global model w_t
Output: The set of participants \mathcal{S}_P , $|\mathcal{S}_P| = K$

```

1: Initialize  $N_0 = N$ ,  $\mathcal{S}_P = \emptyset$ 
2: for  $r = 0$  to  $R$  do
3:   Divide  $N_r$  participants into  $c$  sets,  $S_r = \{S_r^1, S_r^2, \dots, S_r^c\}$ 
4:   for each  $S_r^i \in S_r$  do
5:      $w_t = \sum_{k=1}^{|S_r^i|} \frac{1}{|S_r^i|} w_t^k, g(S_r^i) = w_t - w_{t-1}$ 
6:      $\text{Pre}(S_r^i) \leftarrow (\mathcal{H}, g(S_r^i))$ 
7:   end for
8:   Filter out the  $b$  sets  $S_r^x$  with smallest prediction,  $x \in [1, c]$ 
9:    $N_r = N_r - b \times \lfloor N_r/c \rfloor, S_r = S_r - \{S_r^{x_1}, S_r^{x_2}, \dots, S_r^{x_b}\}$ 
10: end for
11:  $\mathcal{S}_P = S_r$ 

```

Algorithm 2 exhibits the process of DSS. We first divide the participants into c sets randomly, and iteratively screen out b sets of participants with the minimum probability of owning the property \mathcal{P} , until the required number of participants are left. However, it is important to note that these c sets do not overlap. R is the round of selection strategy, and we initialize the $N_1 = N$ and $M_1 = M$, where M denotes the number of participants with property \mathcal{P} . At each round r ($r \in [1, R]$), we perform the selection strategy as shown in Fig. 3.

1) *Partitioning participants.* We divide the N_r candidate participants into c sets, each participant set contains $\lfloor N_r/c \rfloor$

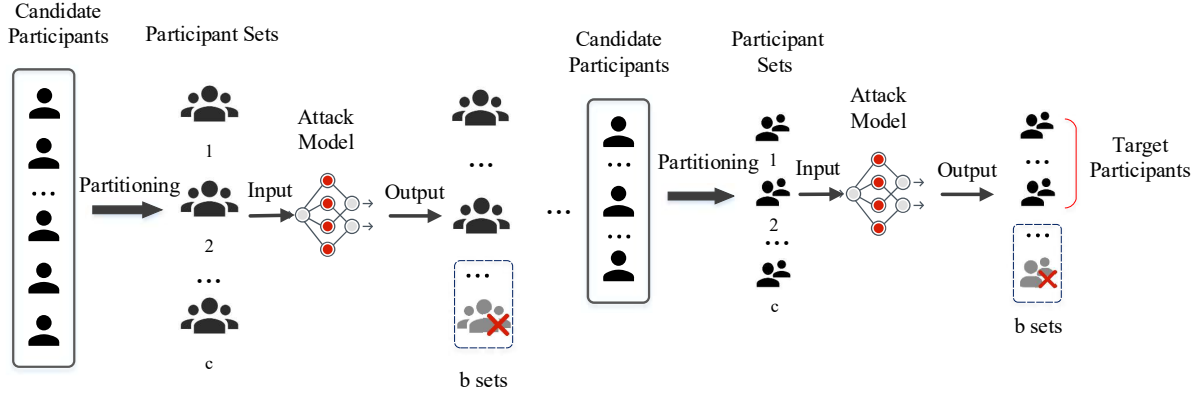


Fig. 3. The dynamic selection strategy used in the active attack scenario.

participants and the number of selected participants $n = \lfloor N_r/c \rfloor$.

2) *Screening out participants.* At each iteration, the server selects a set of participants S_r^i ($i \in [1, c]$) and predicts the probability that the global update from S_r^i has the property \mathcal{P} using the meta-classifier \mathcal{H} . After c iteration, we obtain the corresponding predictions of \mathcal{H} . Then we screen out the last b sets of participants S_r^j ($j \leq c$) based on the rank of predictions, since the predictions represent the probability that the set of participants has the property \mathcal{P} . Thus, we screen out the participants most likely without the property.

After steps 1) and 2), the number of candidate participants is $N_{r+1} = N_r - b \times \lfloor N_r/c \rfloor$, and remaining participants with the property \mathcal{P} satisfies Definition 1. Repeat above steps until the number of candidate participants satisfies $N_R \approx K$, where K is the number of participants with the property \mathcal{P} we aim to infer. It should be noted that some participants who are not selected in this iteration (less than $\lfloor N_r/c \rfloor$) will not be screened out.

Definition 1. (Candidate Participants) Given the number of candidate participants N , the number of participants M with the property \mathcal{P} , the number of selected participants n at each iteration of federated learning, the number of sets of participants c , a screen parameter b , the remaining candidate participants N_r and the remaining participants with the property \mathcal{P} , M_r following the rules:

$$N_r = N(1 - \frac{b}{c})^r \quad (6)$$

$$M_r = M - \sum_{j=1}^r \sum_{i=1}^b x_j^i \quad (7)$$

where $x_j = \sum x_j^i$ ($i \in [1, b]$) is the number of screened participants with property \mathcal{P} at j -th round of DSS.

D. Effectiveness Analysis of DSS

In this section, we formalize the effectiveness and efficiency of the above selection strategy and analyze the impact of the number of participants sets c and screened participant sets b .

We use the \mathcal{S} to represent the above selection strategy, \mathcal{A} and \mathcal{B} represent the accuracy and efficiency of this attack. Attack accuracy \mathcal{A} is the proportion of participants with the property \mathcal{P} among the inferred participants.

We use the required iterations of this attack to formalize its efficiency, because the time cost of each iteration in the federated learning depends on the performance of machines.

The selected participants with the property \mathcal{P} at each round subject to the hypergeometric distribution defined as follows.

Definition 2. (Hypergeometric Distribution) Given the above variable N, M, n . The probability of the number of participants X with property \mathcal{P} in n selected participants satisfies the hypergeometric distribution $X \sim H(N, M, n)$. It follows the rule, and the mathematical expectation of X is $E(x) = \frac{nM}{N}$.

$$P(X = x) = \frac{C_M^x C_{N-M}^{n-x}}{C_N^n} \quad (8)$$

Given the above selection strategy \mathcal{S} , the impacts of c and b on attack accuracy are defined as follows:

Definition 3. (Selection Strategy) Given a hypergeometric distribution $X \sim H(N, M, n)$, the parameters of b and c , target number of participants K with the property \mathcal{P} . We define the selection \mathcal{S} to maximize the accuracy of our objective by solving the following problems:

$$\arg \min_{c, b} \sum_{r=1}^R \sum_{i=1}^b P(X = x_r^i), X \sim H(N_r, M_r, n_r) \quad (9)$$

The efficiency of our attack is iterations in Algorithm 1, and is calculated as follows:

Definition 4. (Iterations Required) Given the selection strategy \mathcal{S} , the parameters of c, b , the round required R of selection strategy, the iterations required is $r \times c$, thus, the efficiency objective aim to solving the following problem:

$$\arg \min_{c, b} (c \times \log(1 - \frac{b}{c})^{\frac{K}{N}}) \quad (10)$$

From the Equations (9) and (10), we can see that the effectiveness and efficiency of DSS are related to the parameters b and c . The larger the number of participant sets c , the higher

the accuracy of attack and the more iterations needed. The smaller the screen value b , the higher the accuracy of attack and the more iterations needed.

We demonstrate the efficiency and effectiveness of DSS in Section VI.

V. DATASETS AND MODEL ARCHITECTURES

In this section, we describe the datasets and the architecture of the target models and the inference attack models.

A. Datasets

Labeled Faces In the Wild (LFW). It is a standard dataset designed for face recognition, and contains 13,233 face images of 5,749 individuals. Each image has multiple labels, such as race, age, gender, hair color and eyewear. For LFW₁, the main task is the gender classifier and the target property is “race:black”; for LFW₂, the main task is the race classifier and the target property is male. Each participant in FL has 32 images (i.e., the batch size).

CelebFaces Attributes (CelebA). It contains 202,599 face images for 10,177 celebrities. Each image labels 40 binary attribute annotations, such as race, smile, age and gender. The main tasks of FL on CelebA are gender and smile classifiers, and the target properties are race, smile and black hair, which are denoted by CelebA₁, CelebA₂, CelebA₃ and CelebA₄, respectively. Each participant in FL has 64 images, composing a total of 128,000 images.

CASIA-WebFace. It contains more than 400,000 face images of 10,575 individuals. For CASIA₁, the main task is gender classifier and the target property is black race; and for CASIA₂, the main task is race classifier and the target property is male. Each participant in FL has 64 images.

MNIST. It contains 70,000 handwritten digits. We select a subset of the original dataset, named MNIST₁. The main task is to recognize the digit shown in image, and we want to infer whether the target model was trained using noisy images [35]. We create the noisy images by adding a random brightness jitters to each images as suggested in [35]. Each participant in FL has 32 images.

We divide the datasets randomly into two parts for adversary and participants, as shown in Table III. It should be noted that there is no overlapping between the auxiliary dataset of adversary and the training datasets of participants. The dataset sizes for all participants are same on the same dataset. Half of the participants have the training datasets with property \mathcal{P} , and another half of the participants have the training datasets without property \mathcal{P} . Note that each participant’s private data only has pure properties, e.g., either black hair or non-black hair.

B. Model Architecture Description

Target models. For the face classification task, we use a three-layer CNN classifier with 32, 64, and 128 filters at each layer, a kernel size of (3,3), followed by two fully-connected layers of size 256 and 2. We use the FaceNet [39] to align

TABLE II
DATASETS AND TASKS IN OUR EXPERIMENTS. WE USE THE PEARSON’S COEFFICIENT TO SHOW THE RELEVANCE OF THE TARGET PROPERTIES TO THE MAIN TASK LABELS.

| Datasets | #Records | Main task | Target property | Corr |
|---------------------|----------|-------------|-----------------|--------|
| LFW ₁ | 12.8K | Gender | Race (black) | 0.084 |
| LFW ₂ | 16.6K | Race | Gender (male) | 0.084 |
| CelebA ₁ | 128K | Gender | Smile | -0.139 |
| CelebA ₂ | 128K | Smile | Gender (male) | -0.139 |
| CelebA ₃ | 128K | Gender | Black Hair | 0.114 |
| CelebA ₄ | 128K | Smile | Attractive | 0.054 |
| CASIA ₁ | 102.4K | Gender | Race (black) | - |
| CASIA ₂ | 102.4K | Race | Gender (male) | - |
| MNIST ₁ | 57.6K | Recognition | Noisy images | - |

all face images to 160*160 pixels. On MNIST, we use a two-layer CNN classifier with 32, 64 filters, followed by two fully-connected layers of size 128 and 10. We use the ReLU as the activation function and SGD learning algorithm for all models. The learning rate is 0.00001. The batch sizes are 32 on LFW and MNIST and 64 on the rest datasets.

Attack model. We flatten the input features into a one-dimensional vector and use a CNN with 100 kernels of size (1,100) to extract the input features. The max pooling size is (1,2). Two fully-connected layers are of size 128 and 64. We use the ReLU as activation function and Adam optimizer for all inference attack models. The learning rate is 0.001 and the output of attack models is a softmax layer.

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

| Datasets | Target model | | Attack model | |
|---------------------------|------------------|---------------------|------------------|---------------------|
| | Dataset with p | Dataset without p | Dataset with p | Dataset without p |
| LFW ₁ | 3,200 | 3,200 | 3,200 | 3,200 |
| LFW ₂ | 3,200 | 3,200 | 9,600 | 640 |
| CASIA ₁ | 32,000 | 32,000 | 6,400 | 32,000 |
| CASIA ₂ | 25,600 | 12,800 | 32,000 | 32,000 |
| CelebA _{1,2,3,4} | 32,000 | 32,000 | 32,000 | 32,000 |
| MNIST ₁ | 16,000 | 16,000 | 12,800 | 12,800 |

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed attacks.

A. Evaluation of Property Inference Attack

Recall that the goal of the adversary is to infer a certain number of participants with the target properties. We set the number of target participants $K = \frac{N}{10}$, unless otherwise noted. Table II shows an overview of our experiments. We use the Pearson’s coefficient to show the relevance of the target properties to the main task labels. The Pearson’s coefficients are missing in CASIA and MNIST datasets, because there are no data labels in CASIA and MNIST, only the images.

Attack accuracy. To evaluate the accuracy of the active attack, in each experiment in Table II, we randomly initialize

TABLE IV
ACCURACY AND ITERATION REQUIRED FOR ACTIVE PROPERTY INFERENCE ATTACKS ($K = \frac{N}{10}$)

| Experiments | N=100 | | N=200 | | N=400 | | N=600 | | N=800 | | N=1,000 | |
|---------------------|-------|------------|-------|------------|-------|------------|-------|------------|-------|------------|---------|------------|
| | Acc. | Iterations | Acc. | Iterations | Acc. | Iterations | Acc. | Iterations | Acc. | Iterations | Acc. | Iterations |
| LFW ₁ | 90.7% | 60 | 87.6% | 96 | - | - | - | - | - | - | - | - |
| LFW ₂ | 89.2% | 60 | 84.5% | 96 | - | - | - | - | - | - | - | - |
| CelebA ₁ | 77.3% | 60 | 77.6% | 96 | 72.2% | 148 | 71.0% | 187 | 70.2% | 260 | 71.3% | 402 |
| CelebA ₂ | 72.1% | 34 | 76.0% | 96 | 72.6% | 209 | 70.8% | 236 | 71.2% | 385 | 70.2% | 462 |
| CelebA ₃ | 76.2% | 60 | 73.1% | 57 | 74.8% | 148 | 69.5% | 297 | 71.2% | 228 | 70.1% | 298 |
| CelebA ₄ | 70.0% | 40 | 70.1% | 96 | 71.5% | 185 | 69.5% | 297 | 68.8% | 385 | 66.5% | 402 |
| CASIA ₁ | 91.3% | 34 | 96.0% | 96 | 90.3% | 154 | 86.2% | 150 | 80.7% | 139 | 82.3% | 221 |
| CASIA ₂ | 93.6% | 60 | 91.4% | 96 | 83.7% | 105 | 83.0% | 150 | 83.6% | 228 | 82.6% | 298 |
| MNIST ₁ | 89.0% | 60 | 87.5% | 96 | 81.3% | 105 | 81.2% | 150 | 81.6% | 228 | 81.5% | 307 |

TABLE V
COMPARISON OF ATTACK ACCURACY AND EFFICIENCY BETWEEN DSS AND NSS ($N = 100, b = 2$).

| Experiments | DSS | | NSS | |
|---------------------|----------|------------|----------|------------|
| | Accuracy | Iterations | Accuracy | Iterations |
| LFW ₁ | 83.9% | 34 | 78.4% | 54 |
| LFW ₂ | 84.8% | 34 | 72.5% | 54 |
| CelebA ₁ | 73.8% | 41 | 60.3% | 54 |
| CelebA ₂ | 73.4% | 40 | 60.6% | 54 |
| CelebA ₃ | 75.0% | 34 | 64.8% | 54 |
| CASIA ₁ | 90.9% | 34 | 75.4% | 54 |
| CASIA ₂ | 87.2% | 34 | 72.0% | 54 |
| MNIST ₁ | 81.9% | 39 | 68.2% | 54 |

TABLE VI
THE ACCURACY OF THE MAIN TASKS WITH OR WITHOUT ATTACKS FOR 240 ITERATIONS ($N = 100, b = 2$).

| Experiments | Attack Accuracy | Accuracy of Main Tasks | |
|---------------------|-----------------|------------------------|-------------|
| | | Attack | Attack-free |
| LFW ₁ | 83.9% | 81.1% | 81.3% |
| LFW ₂ | 84.8% | 88.9% | 89% |
| CelebA ₁ | 73.8% | 95.3% | 95.4% |
| CelebA ₂ | 73.4% | 90.6% | 90.7% |
| CelebA ₃ | 75.0% | 85.2% | 85.5% |
| CASIA ₁ | 90.9% | 85.2% | 86% |
| CASIA ₂ | 87.2% | 86.4% | 86.4% |
| MNIST ₁ | 81.9% | 80.0% | 80.2% |

model parameters to repeat the inference attack 100 times and obtain the averaged attack accuracy. We also vary the participant number N from 100 to 1,000 to evaluate its performance with different participant populations.

Table IV shows the accuracy with different datasets. The attack accuracy decreases as the number of participants N increases, but remains above 70% even with 1,000 participants.

To verify the effectiveness of the active selection strategy, we compare DSS with NSS. Table V shows the accuracy of the active attacks using NSS or DSS with $c = 7$ and $b = 2$. The number of participants N equals 100 in all experiments. The results show that DSS helps increase the attack accuracy while significantly reducing the iterations required.

Attack efficiency. We use the required iterations during the inference attack to evaluate the effectiveness of active attacks, as illustrated in Equation (10). Table IV shows the required iterations in different datasets with varying participant numbers.

Attack overhead. The only difference of the active attack from the passive attack lies in that the adversary can actively select participants involved in the training of main tasks. Table VI shows the impact of the active attack on the accuracy of main tasks with different datasets. We can see that the accuracy of the main tasks only slightly drops when the active attacks are launched.

The impact of target participants on inference attack. Table VII shows attack accuracy for different number of target participants. As expected, an increasing number of target participants decreases the accuracy of the property inference

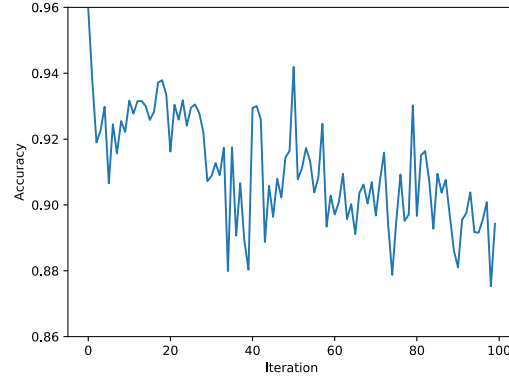


Fig. 4. The accuracy of inference attack in various iterations on LFW₁ ($N = 100, K = 10$).

attack. It should be noted that in all experiments, a half of the participants have the target property \mathcal{P} , which means that $\frac{N}{2}$ is the upper bound of K .

The impact of training iteration on inference attack. Fig. 4 shows the attack accuracy performed in different iterations of main tasks. As expected, the training iteration has a marginal impact on the attack accuracy, and increasing the training iterations weakens the accuracy of the property inference attack. When the models of main tasks have been fitted, the update gradients of the main task models will be less obvious. From the above, the more training iterations, the higher the accuracy of the main task models, and the less significant the update gradients.

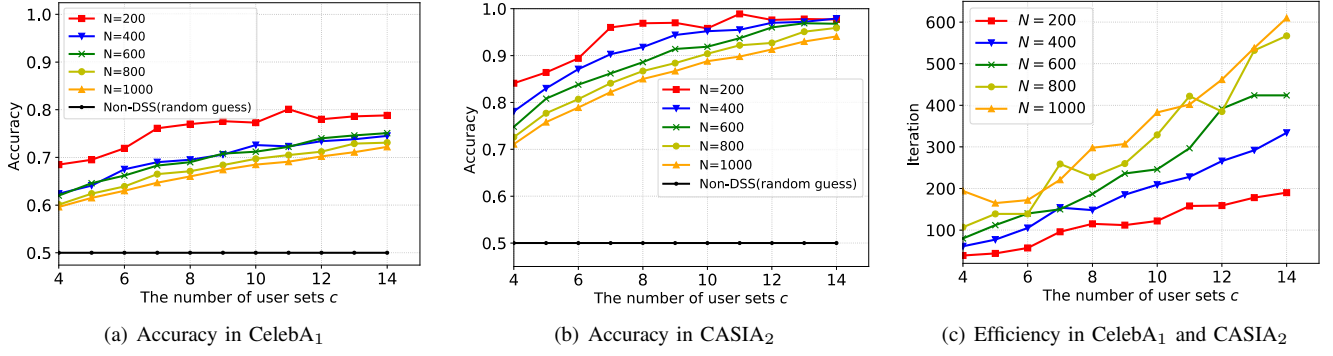


Fig. 5. Attack accuracy and efficiency with varying N and c ($K = \frac{N}{10}$, $M = \frac{N}{2}$, $b = 2$).

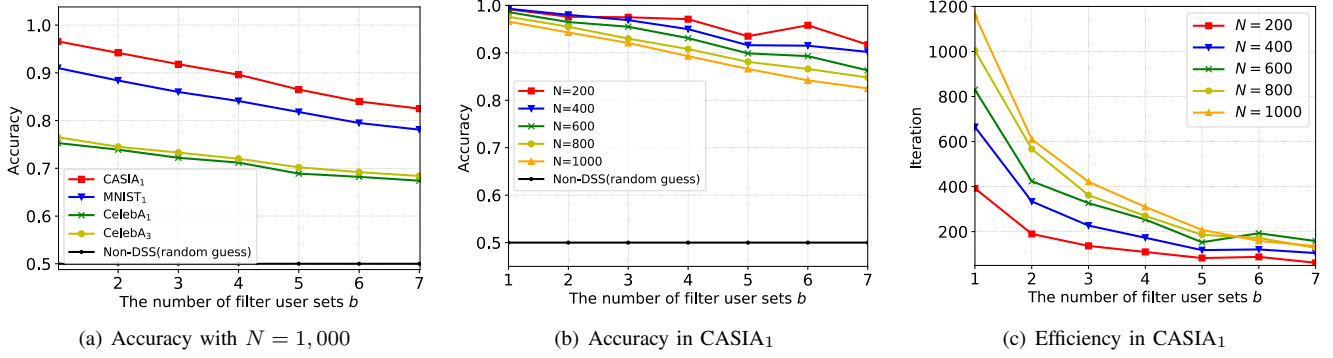


Fig. 6. Attack accuracy and efficiency with varying b ($c = 14$).

TABLE VII
ATTACK ACCURACY AT INFERRING VARIOUS SIZE OF TARGET PARTICIPANTS IN CASIA₁ ($c = 9$, $b = 1$)

| Participants (N) | Attack accuracy with different (K) | | | |
|----------------------|--|--------|--------|--------|
| | $1/10N$ | $1/5N$ | $1/3N$ | $1/2N$ |
| 100 | 89.0% | 89.9% | 81.3% | 70.7% |
| 200 | 91.2% | 87.9% | 79.5% | 71.1% |
| 400 | 89.7% | 84.6% | 76.3% | 68.6% |
| 800 | 85.4% | 79.7% | 72.1% | 65.2% |
| 1000 | 84.1% | 78.1% | 70.5% | 64.3% |

TABLE VIII
THE ATTACK ACCURACY FOR VARIOUS DISTRIBUTIONS OF PARTICIPANT'S PRIVATE DATA. THE SIZE OF PARTICIPANT'S DATA IS 64. THE RATIO OF THE DATA WITH \mathcal{P} TO THE DATA WITH $\bar{\mathcal{P}}$ ARE $7/1$, $4/1$, $3/1$, $2/1$ ($N = 100$, $c = 7$, $b = 3$).

| Ratios of Pro. Sizes and Non-Pro. Sizes | Pro. Sizes | Non-Pro. Sizes | Attack Acc. |
|---|------------|----------------|-------------|
| 1/0 | 64 | 0 | 86.3% |
| 7/1 | 56 | 8 | 84.8% |
| 4/1 | 51 | 13 | 83.6% |
| 3/1 | 48 | 16 | 71.9% |
| 2/1 | 43 | 21 | 71.3% |

The impact of distributions of participants' datasets on inference attack. We evaluate the attack accuracy for various distributions of participant's private data. If the size of participant's data with property \mathcal{P} is larger than those with $\bar{\mathcal{P}}$, we label the model updates as \mathcal{P} and vice versa. As expected, the closer the size of data with \mathcal{P} and $\bar{\mathcal{P}}$, the worse our attack performance.

B. Impact of Selection Strategy on Inference Attack

To select the suitable parameters of dynamic selection strategy, we analyze the experimental results with varying the size of selected participants c , the number of screened participant sets b and the number of participants N .

Impact of the number of participants N . Figs. 5(a) and 5(b) demonstrate the impact of the number of participants N on CASIA-WebFace and CelebA datasets. From Fig. 5, the attack accuracy of $N = 200$ is obviously higher than the

accuracy of $N = 1,000$ with the same parameters c and b . As expected, the larger N is, more iterations are required, the lower efficiency and accuracy the attack will get.

Impact of the size of selected participants c . Fig. 5 shows the accuracy and efficiency of this attack on CelebA and CASIA datasets with different c . We set the beginning value of c as 4 rather than 1. The reason behind this is twofold. The first one is that c represents the proportion of selected participants among all participants which means it has a lower bound. The second reason is that when c is smaller, the number of selected participants $n = \lfloor N/c \rfloor$ would be larger, and the accuracy of the attack model \mathcal{H} would be lower. When c increases, the number of iterations increases, and the efficiency of this attack would decrease as well.

Impact of screened participant sets b . As shown in Figs. 6(a) and 6(b), the attack accuracy is negatively correlated with

b . The attack efficiency is positively correlated with b . When the value of b increases, the required iterations decrease and the efficiency of this attack would increase (cf. Fig. 6(c)).

From these experiments in several real-world datasets, we verify the impact of parameters N , c and b on the effectiveness and efficiency of the attacks and the selection strategy we proposed.

C. Limitations of Attacks

Attack model. The attack model we proposed is used to predict whether a model update has a certain property. The adversary generates meta-training data to train the attack model based on the global model and auxiliary dataset. When the number of required iterations is large, the time cost of training the attack model is high.

The number of selected participants. The output of attack model is the probability that the model update has target property. When the number of selected participants increase, the accuracy of the attack model will decrease. In order to ensure the accuracy of attack model and iteratively inferring the participants with target property, the number of selected participants is limited.

VII. CONCLUSION

In this paper, we proposed a novel attack to exploit the unintended properties leakage from model updates in blockchain-assisted FL for intelligent edge computing. This attack is practicable and enables a server to infer a set of participants with target properties, which is a risk to the sensitive data of IoT devices. We evaluated the proposed attacks on real-world datasets and demonstrated that the proposed attacks are effective and efficient in inferring various properties of training data while having negligible impact on the main tasks of FL. Our attacks suggest that there are a number of privacy risks even if the local updates of participants are encrypted and servers observe only the aggregated global updates. In the future work, we will improve the attack efficiency and explore better defenses to protect sensitive information of participants in the training process of blockchain-assisted FL.

REFERENCES

- [1] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, 2019.
- [2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, vol. 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282, PMLR, 2017.
- [3] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321, ACM, 2015.
- [4] J. Weng, J. Weng, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IACR Cryptology ePrint Archive*, vol. 2018, p. 679, 2018.
- [5] A. Jochems, T. M. Deist, J. Van Soest, M. Eble, P. Bulens, P. Coucke, W. Dries, P. Lambin, and A. Dekker, "Distributed learning: developing a predictive model based on data from multiple hospitals without data leaving the hospital—a real life proof of concept," *Radiotherapy and Oncology*, vol. 121, no. 3, pp. 459–467, 2016.
- [6] M. Shen, Y. Deng, L. Zhu, X. Du, and N. Guizani, "Privacy-preserving image retrieval for medical IoT systems: A blockchain-based approach," *IEEE Network*, vol. 33, no. 5, pp. 27–33, 2019.
- [7] X. Zhu, H. Li, and Y. Yu, "Blockchain-based privacy preserving deep learning," in *Information Security and Cryptology - 14th International Conference, Inscrypt 2018*, vol. 11449, pp. 370–383, Springer, 2018.
- [8] Y. Zhang, Y. Lu, X. Huang, K. Zhang, and S. Maharjan, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, 2020.
- [9] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Transactions on Industrial Informatics*, 2019.
- [10] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 603–618, ACM, 2017.
- [11] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pp. 739–753.
- [12] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pp. 691–706.
- [13] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 2512–2520.
- [14] M. Shen, B. Ma, L. Zhu, R. Mijumbi, X. Du, and J. Hu, "Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 4, pp. 940–953, 2018.
- [15] M. Shen, J. Zhang, L. Zhu, K. Xu, and X. Tang, "Secure SVM training over vertically-partitioned datasets using consortium blockchain for vehicular social networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5773–5783, 2020.
- [16] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2019*, pp. 1–11.
- [17] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for federated learning on user-held data," *arXiv preprint arXiv:1611.04482*, 2016.
- [18] Z. Liu, P. Luo, X. Wang, and X. Tang, "Large-scale celebfaces attributes (celeba) dataset," *Retrieved August*, vol. 15, 2018.
- [19] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," 2008.
- [20] S. Li, D. Yi, Z. Lei, and S. Liao, "The casia nir-vis 2.0 face database," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 348–353, 2013.
- [21] X. Wang, X. Li, S. Pack, Z. Han, and V. C. M. Leung, "STCS: spatial-temporal collaborative sampling in flow-aware software defined networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 999–1013, 2020.
- [22] D. Zhang, X. Chen, D. Wang, and J. Shi, "A survey on collaborative deep learning and privacy-preserving," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 652–658, IEEE, 2018.
- [23] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.
- [24] M. Shen, J. Duan, L. Zhu, J. Zhang, X. Du, and M. Guizani, "Blockchain-based incentives for secure and collaborative data sharing in multiple clouds," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1229–1241, 2020.
- [25] R. Li, H. Asaeda, and J. Li, "A distributed publisher-driven secure data sharing scheme for information-centric IoT," *IEEE Internet of Things Journal*, vol. 4, no. 3, pp. 791–803, 2017.
- [26] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.
- [27] M. Shen, H. Liu, L. Zhu, K. Xu, H. Yu, X. Du, and M. Guizani, "Blockchain-assisted secure device authentication for cross-domain in-

- dustrial iot,” *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 942–954, 2020.
- [28] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, IEEE, 2017.
- [29] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, “Privacy risk in machine learning: Analyzing the connection to overfitting,” in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282, IEEE, 2018.
- [30] Y. Long, V. Bindshaedler, L. Wang, D. Bu, X. Wang, H. Tang, C. A. Gunter, and K. Chen, “Understanding membership inferences on well-generalized learning models,” *arXiv preprint arXiv:1802.04889*, 2018.
- [31] J. Hayes, L. Melis, G. Danezis, and E. D. Cristofaro, “LOGAN: membership inference attacks against generative models,” *PoPETs*, vol. 2019, no. 1, pp. 133–152, 2019.
- [32] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.
- [33] M. Fredrikson, S. Jha, and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 1322–1333, ACM, 2015.
- [34] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, “Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers,” *IJSN*, vol. 10, no. 3, pp. 137–150.
- [35] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, “Property inference attacks on fully connected neural networks using permutation invariant representations,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 619–633.
- [36] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, “Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities,” *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7702–7712, 2019.
- [37] C. Song, T. Ristenpart, and V. Shmatikov, “Machine learning models that remember too much,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 587–601.
- [38] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” *arXiv preprint arXiv:1807.00459*, 2018.
- [39] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Meng Shen (M’14) received the B.Eng degree from Shandong University, Jinan, China in 2009, and the Ph.D degree from Tsinghua University, Beijing, China in 2014, both in computer science. Currently he serves in Beijing Institute of Technology, Beijing, China, as an associate professor. His research interests include privacy protection for cloud and IoT, blockchain applications, and encrypted traffic classification. He received the Best Paper Runner-Up Award at IEEE IPCCC 2014. He is a member of the IEEE.

Huan Wang received the B.Eng degree in computer science from Ocean University of China, Qingdao, China in 2018. Currently she is a master student in the Department of Computer Science, Beijing Institute of Technology. Her research interests include machine learning and data privacy.

Bin Zhang received his Ph.D. degree in Department of Computer Science and Technology, Tsinghua University, China in 2012. He worked as a post doctor in Nanjing Telecommunication Technology Institute from 2014 to 2017. He is now a researcher in the Cyberspace Security Research Center of Peng Cheng Laboratory. He publish more than 40 papers in refereed international conferences and journals. His current research interests focus on network anomaly detection, Internet architecture and its protocols, network traffic measurement, information privacy security, etc.

Liehuang Zhu (M’16) is a professor at the Department of Computer Science at Beijing Institute of Technology. He is selected into the Program for New Century Excellent Talents in University from the Ministry of Education, P.R. China. His research interests include Internet of Things, Cloud Computing Security, Internet and Mobile Security.

Ke Xu received his Ph.D. from the Department of Computer Science & Technology of Tsinghua University, Beijing, China, where he serves as a full professor. He has published more than 200 technical papers and holds 11 US patents in the research areas of next-generation Internet, Blockchain systems, Internet of Things (IoT), and network security. He is a member of ACM and senior member of IEEE. He has guest-edited several special issues in IEEE and Springer Journals. He is an editor of IEEE IoT Journal. He is Steering Committee Chair of IEEE/ACM IWQoS.

Qi Li (Senior Member, IEEE) received the Ph.D. degree from Tsinghua University. He is currently an Associate Professor with the Institute for Network Sciences and Cyberspace, Tsinghua University. He has worked with ETH Zurich and The University of Texas at San Antonio. His research interests include network and system security, particularly in Internet and cloud security, mobile security, and big data security. He is currently an Editorial Board Member of the IEEE TDSC and ACM DTRAP.

Xiaojiang Du (S’99-M’03-SM’09-F’20) is a tenured professor in the Department of Computer and Information Sciences at Temple University, Philadelphia, USA. Dr. Du received his B.S. and M.S. degree in electrical engineering from Tsinghua University, Beijing, China in 1996 and 1998, respectively. He received his M.S. and Ph.D. degree in electrical engineering from the University of Maryland College Park in 2002 and 2003, respectively. His research interests are wireless communications, wireless networks, security, and systems. He has authored over 400 journal and conference papers in these areas, as well as a book published by Springer. Dr. Du has been awarded more than \$5 million US dollars research grants from the US National Science Foundation (NSF), Army Research Office, Air Force, NASA, the State of Pennsylvania, and Amazon. He won the best paper award at IEEE GLOBECOM 2014 and the best poster runner-up award at the ACM MobiHoc 2014. He serves on the editorial boards of three international journals. Dr. Du is a Fellow of IEEE and a Life Member of ACM.