DOI: 10.1109/TKDE.2023.3309858

FLGAN: GAN-Based Unbiased Federated Learning Under Non-IID Settings

Zhuoran Ma, Yang Liu, Yinbin Miao, *Member, IEEE*, Guowen Xu, *Member, IEEE*, Ximeng Liu, *Senior Member, IEEE*, Jianfeng Ma, *Member, IEEE*, and Robert H. Deng, *Fellow, IEEE*

***Abstract*—Federated Learning (FL) suffers from low convergence and significant accuracy loss due to local biases caused by non-Independent and Identically Distributed (non-IID) data. To enhance the non-IID FL performance, a straightforward idea is to leverage the Generative Adversarial Network (GAN) to mitigate local biases using synthesized samples. Unfortunately, existing GAN-based solutions have inherent limitations, which do not support non-IID data and even compromise user privacy. To tackle the above issues, we propose a GAN-based unbiased FL scheme, called FLGAN, to mitigate local biases using synthesized samples generated by GAN while preserving user-level privacy in the FL setting. Specifically, FLGAN first presents a federated GAN algorithm using the divide-and-conquer strategy that eliminates the problem of model collapse in non-IID settings. To guarantee user-level privacy, FLGAN then exploits Fully Homomorphic Encryption (FHE) to design the privacy-preserving GAN augmentation method for the unbiased FL. Extensive experiments show that FLGAN achieves unbiased FL with $10\% - 60\%$ accuracy improvement compared with two state-of-the-art FL baselines (i.e., FedAvg and FedSGD) trained under different non-IID settings. The FHE-based privacy guarantees only cost about 0.53% of the total overhead in FLGAN.**

***Index Terms*—Federated learning, fully homomorphic encryption, GAN, non-IID, user-level privacy.**

Manuscript received 12 January 2022; revised 30 June 2023; accepted 26 August 2023. Date of publication 29 August 2023; date of current version 8 March 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB3101100, in part by the National Natural Science Foundation of China under Grants 62302361, 62072361, and 62072352, in part by Shaanxi Province Key Industry Innovation Chain Project under Grant 2023-ZDLGY-52, in part by China Postdoctoral Science Foundation under Grant 2023M732735, in part by the Foundation for Innovative Research Groups of the National Natural Science Foundation of China under Grant 62121001, and in part by the Key R&D Program of Shaanxi Province under Grants 2019ZDLGY12-04 and 2020ZDLGY09-06. Recommended for acceptance by Y. Gao PhD. *(Corresponding author: Yang Liu.)*

Zhuoran Ma, Yang Liu, Yinbin Miao, and Jianfeng Ma are with the School of Cyber Engineering, Xidian University, Xi'an 710071, China, and also with the Shaanxi Key Laboratory of Network and System Security, Xidian University, Xi'an 710071, China (e-mail: emmazhr@163.com; bcds2018@foxmail.com; ybmiao@xidian.edu.cn; snbnix@gmail.com).

Guowen Xu is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: guowen.xu@ntu.edu.sg).

Ximeng Liu is with the Key Laboratory of Information Security of Network Systems, College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China, and also with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: snbnix@gmail.com).

Robert H. Deng is with the School of Information Systems, Singapore Management University, Singapore 188065 (e-mail: robertdeng@smu.edu.sg).

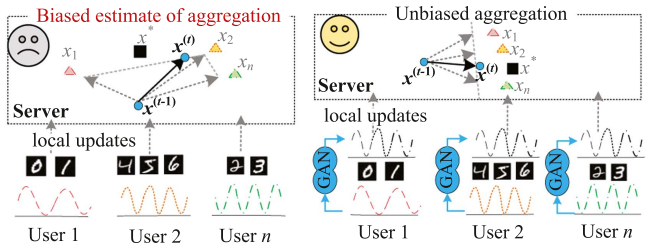
Digital Object Identifier 10.1109/TKDE.2023.3309858

I. INTRODUCTION

FEDERATED Learning (FL) [1], [2] is one of the most promising distributed learning paradigms, especially for privacy-sensitive applications. Instead of directly collecting user-held data, FL aims to protect user-level privacy [3] by implementing model training with sub-updates uploaded by users. It has been shown that the aggregated update of FL is an unbiased estimator of local updates under *Independent and Identically Distributed (IID)* settings [4]. Unfortunately, the uncontrollability and complexity of real applications always result in an uneven distribution of user-held data, which is hard to be ideally IID [5]. With ubiquitous non-IID data, FL generally brings various concerns, such as slow convergence and poor performance of the global model [6], [7]. As observed in Fig. 1(a), the most striking feature of non-IID FL is that the local data distribution of each user differs dramatically. If local updates are trained on user-held non-IID data, the aggregated parameter $x^{(t)}$ strays towards a local optimum $x_{i\in[1,n]}$, away from the true global optimum x^*. Therefore, *how to ensure FL performance with non-IID data poses a pervasive challenge for FL* [8].

To eliminate local biases in a non-IID setting, a direct approach is to add proper data samples that can smooth out the biased local data distribution caused by non-IID data. However, determining what is "deficient" in each user's data to fulfill the IID requirement would require learning the distribution of each user's data, which violates the privacy requirement in FL. To circumvent the privacy barrier and eliminate local biases in a non-IID setting, this paper designs a novel federated GAN model, called FLGAN, which enables Generative Adversarial Networks (GANs) to be privately trained to synthesize training samples. As depicted in Fig. 1(b), each user collaboratively trains a federated GAN model (containing a discriminator and a generator) to synthesize global IID data. The objective of the discriminator is to assess how likely the generated global samples are real or fake, which in turn trains the generator to improve its performance. The learned generator mimics IID samples to augment local non-IID data. In this way, FLGAN ensures the consistency of the objectives of both local and global convergence and eliminates locally biased updates, resulting in an unbiased FL aggregation.

Unfortunately, FLGAN must dive to the bottom to fix the following issues, which inhibit the development of federated GAN training. *The first issue* is that existing GAN solutions [9] face



(a) FL using non-IID data (b) FLGAN using non-IID data

Fig. 1. Non-IID FL approach, black squares denote the global convergence, and red, yellow and green triangles denote the local convergences. .

difficulties in constructing a global GAN in the non-IID setting. Mathematically, the objectives of local GAN and federated GAN remain biased in non-IID settings. As a result, it is easy to incur the model collapse, since the biased local model/update trained on non-IID data causes a biased estimator of the global GAN model. To address this issue, existing approaches [10], [11] swap the trained discriminator in a user-by-user manner to access massive data for constructing the global GAN. However, the computation overhead increases as users grow in number, which is too time-consuming and unsuited to a federated architecture. *The second issue* is that existing solutions ignore data privacy. The GAN training [12], [13] compromises data privacy as realistic-looking synthetic data containing sensitive information are transmitted between the server and users [3], [14]. Fully Homomorphic Encryption (FHE) is a promising solution for privacy guarantee. However, GAN training involves complicated non-linear activations such as ReLU and Sigmoid, which leads to the burdensome overhead by using FHE to implement secure computations. Therefore, it is challenging to design a federated GAN that is supportive of the non-IID setting and the federated architecture, while ensuring user-level privacy with a lightweight overhead.

To address the above challenges, we first design a federated GAN approach for diverse non-IID settings, which relies on local GAN training without data transmission in the FL architecture. We then propose a privacy-preserving mechanism to ensure user-level privacy with lightweight overhead in FLGAN. More importantly, FLGAN can be embedded in existing FL schemes to optimize non-IID FL training performance. Specifically, the main contributions of FLGAN are summarized as follows:

- We propose a novel federated GAN framework for the unbiased FL, which mitigates local biases by using the GAN augmentation approach, and counterbalances local non-IID data with generated samples. Besides, the federated GAN is constructed with a divide-and-conquer strategy, i.e., dividing the global data distribution estimation of the federated GAN into local training and global training.
- We propose a FHE-based GAN selection scheme to ensure user-level privacy for the federated GAN training without incurring high overheads, and also provides theoretical guarantees for user-level privacy.
- We deploy FLGAN into two state-of-the-art FL algorithms including FedSGD and FedAvg. We conduct extensive experiments using various datasets to show its performance by comparing with the FL model without FLGAN, and demonstrates the high performance of FLGAN in terms of test accuracy, fast convergence, compatibility and computation overheads.

The rest of the paper is organized as follows. We discuss the existing related work and some background knowledge in Section II. A set of machine learning and security primitives that are utilized in FLGAN are provided in Section III. We define the system model, threat model and design goals in Section IV. The FLGAN framework is explained in details in Section V. Sections VI and VII discuss the theoretical and performance analysis. We conclude the paper in Section IX.

II. Related Work

Our work is closely related to FL schemes with non-IID and federated GAN training.

A. Federated Learning With Non-IID

Existing FL schemes can be divided into two categories, i.e., FedSGD [15] and FedAvg [2]. FedSGD is based on averaging local gradients, while FedAvg is built on averaging local model weights. Specifically, FedAvg outperforms FedSGD in non-IID scenarios. However, compared with the IID scenario, the performance of FedAvg is also unstable and slides down significantly in the non-IID scenario, which brings an averaged model with low accuracy and incurs high communication burdens [16]. The non-IID scenario still has detrimental effects on the accuracy of FL.

To address the challenge, many solutions are proposed to optimize FL with non-IID data. Re-parameterized FL schemes [17], [18] are proposed to improve the performance of FedAvg in non-IID scenarios. Zhang et al. [17] proposed a federated primal-dual (FedPD) scheme with non-IID data, which addresses the non-convex objective functions to achieve optimization and reduce the communication complexity. FedProx [18] extended the FedAvg method to the non-IID setting by using a generalization and re-parametrization in local training. However, the above schemes rely on strong assumptions for solving linear FL system under non-IID settings [19]. Especially in FedProx [18], the value of a introduced parameter in the local training significantly affects the performance of FL, which should vary with different levels of non-IID scenarios. As we cannot learn the level of non-IID data, the value of the introduced parameter is difficult to adapt diverse non-IID scenarios. With the similar motivation, clustered FL schemes are proposed in order to alleviate the skewed distribution under non-IID scenarios. Ghosh et al. [20] proposed a iterative clustered FL scheme (IFCA), which estimates the cluster identities and minimizes the loss functions for the user clusters via gradient descent. Huang et al. [5] designed a federated attentive message passing scheme (FedAMP) by using similarity values among local personalized models, which aims to facilitate pairwise collaborations between clients with similar data. However, the clustered FL schemes are limited with a statistical estimation, which are not general for different non-IID scenarios, especially for an extreme non-IID scenario (i.e., 1 class, 1 device). FedMA [21] extended FedAvg to the non-IID scenario, which matches and averages hidden elements in local

TABLE I
FUNCTIONALITIES, SECURITIES AND TECHNIQUES IN VARIOUS SCHEMES: A COMPARATIVE SUMMARY

Functions	Fun_1	Fun_2	Fun_3	Fun_4	Fun_5
[9]	Weight average	✗	✗	✓	✗
[11]	Multi-discriminator	✓	✗	✗	✗
[10]	Multi-discriminator	✓	✓	✗	✗
[12]	Discriminator aggregation	✓	✗	✓	✗
FLGAN	Multi-Generator	✓	✓	✓	✓

Notes. Fun_1: GAN training solution; Fun_2: Whether supporting non-IID settings or not; Fun_3: Whether guaranteeing user-level privacy or not; Fun_4: Whether achieving a federated architecture or not; Fun_5: Whether supporting non-FL training or not.

models with similar feature extraction signatures. Unfortunately, FedMA still has the same issue of the above schemes, where the correlation degree of data distribution among users will directly affect the performance of the clustered FL schemes. Wang et al. [8] optimized FedAvg with non-IID data by using reinforcement learning, where the deep reinforcement learning agent selects k users in each training round for implementing the aggregation in FL. However, experimental results show that the above scheme cannot always outperform FedAvg, as the selection of k devices may introduce more bias to FL than devices selected randomly in FedAvg. Consequently, the skewed bias introduced by non-IID data remains a major barrier in FL, resulting in accuracy loss and slow convergence. Therefore, it is urgent to propose a solution to improve FL performance with non-IID data. By eliminating the bias caused by non-IID data, GAN is expected to be an effective solution for improving the performance of non-IID FL.

B. Federated Learning With GAN

GAN has already achieved success in the field of generating realistic "fake" data. Thus, GAN has been applied in FL in existing schemes. However, limited user-held data cannot be used to train a GAN model in the federated setting. To solve the issue, some federated GAN schemes are built in a FL architecture. Mohammad et al. [9] presented a federated GAN for federated settings (FedGAN), which can aggregate local discriminators and generators in a center server by using the weight average method. Unfortunately, FedGAN has an unacceptable performance in the non-IID scenario. Hardy et al. [11] proposed a multi-discriminator method to train a GAN in a federated fashion, but this scheme is required to swap discriminators among data devices in a peer-to-peer fashion. Xin et al. [10] proposed the private FL-GAN scheme, which uses differential privacy to protect the federated GAN scheme [11]. However, the global GAN is trained among local devices in order, which violates the FL framework. To adapt the FL architecture, Zhang et al. [12] designed a universal aggregation approach for training a federated GAN, which aggregates the mixture of all local discriminators for a centralized discriminator. Unfortunately, the trained federated GAN located on the server can generate realistic-looking synthetic data and infer the original data distribution, which undermines the purpose of FL to protect user-level privacy. As demonstrated in Table I, how to construct a federated GAN scheme with acceptable privacy budgets becomes a major challenge under the non-IID scenario.

TABLE II
NOTATION DESCRIPTIONS

Notations	Descriptions
$\lvert U\rvert = n$	Total number of data users
κ	Global label size of global data
$F(W), F_i(W)$	The global objective and local objective
W, W_i	The global model weights and local weights
μ_i, μ	The local gradient and aggregated gradient
$X = (X_1, ...X_n)$	The global training data
$W_{\mathbb{D}}, W_{\mathbb{G}}$	Model weights of discriminator $\mathbb{D}$ and generator $\mathbb{G}$
p_i	True local data distribution of the i-th user
p_g, p_z	Generator distribution and prior noise distribution
p_{data}	True global data distribution
N	Power-of-two integer (polynomial degree)
$\mathcal{R}, \mathcal{R}_q$	Ring of integers $\mathbb{Z}[x]/(x^N+1)$, $\mathbb{Z}_q[x]/(x^N+1)$
q	Modulus in the ciphertext space
Δ	Scaling factor
$\mathsf{Enc}_{pk}(x)$	Encrypt plaintext x into a ciphertext
$\mathsf{Dec}_{sk}(\llbracket c \rrbracket)$	Decrypt a ciphertext $\llbracket c \rrbracket$ into a plaintext
FHE.Add	Add two ciphertexts
FHE.Mul	Multiply two ciphertexts with the evaluation key evk

III. TECHNICAL BACKGROUND

This section first presents the FL framework and the Conditional GAN (CGAN) scheme, then introduces Fully Homomorphic Encryption (FHE). The notation descriptions are demonstrated in Table II.

A. Federated Learning

Based on a statistical analysis of the divide-and-conquer strategy, each data user $U_{i\in[1,n]}$ first learns a local model $\min\limits_{W_i} F_i(W_i)$, and only communicates these local updates. The server then aggregates local updates to return the aggregated model update to each user. The non-convex FL objective is defined as

$$\min_{W} F(W) = \frac{1}{n}\sum F_i(W_i), \tag{1}$$

where $F_i(W_i)$ is the local objective function, and $F(W)$ is the global objective function in FL.

In the t-th training iteration, each user first downloads the current FL model W, trains W on local data and communicates local updates to a center server for aggregation. Given n users, we consider two following FL algorithms:

- FedAvg [2]: Each user communicates local weight updates $W_i^{(t)}$ to the server. Then, the server implements the FedAvg aggregation to update the FL model such that

$$W^{(t+1)} \leftarrow \frac{1}{n}\sum W_i^{(t)}. \tag{2}$$

- FedSGD [15]: Each user communicates local gradients $\mu_i^{(t)}$, then the server implements FedSGD aggregation to update the FL model

$$W^{(t+1)} \leftarrow W^{(t)} - \eta\frac{1}{n}\sum \mu_i^{(t)}, \tag{3}$$

where η is the learning rate.

CKKS cryptosystem

- $\mathsf{KeyGen}(1^\rho) \rightarrow (pk, sk, evk)$: Given the security parameter ρ, the secret key $sk \leftarrow \mathcal{HWT}(h)$, public key $pk = (b, a) = (-a \cdot sk + e, a)$, and evaluation key $evk = (-a_0 \cdot sk + e_0 + sk^2, a_0)$ are generated, where $a, a_0 \leftarrow \mathcal{R}_q$, $e, e_0 \leftarrow \mathcal{E}$.
- $\mathsf{Enc}_{pk}(z, \Delta) \rightarrow [\![c]\!]$: Given the input vector $z \in \mathbb{C}^{N/2}$, the corresponding integral polynomial $x = \delta^{-1}(\lfloor \Delta \cdot \pi^{-1}(z) \rceil_{\delta(\mathcal{R})}) \in \mathcal{R}$ is encrypted as $[\![c]\!] = (c_0, c_1) = (x, 0) + pk = (b + x, a)$.
- $\mathsf{Dec}_{sk}([\![c]\!]) \rightarrow m$: Given the ciphertext $[\![c]\!]$ of level $l \leq N/2$, the plaintext $x \in \mathcal{R}$ is decrypted as $c_0 + c_1 \cdot sk = x + e$. Then, a vector z is returned as $z = \pi \circ \delta(\Delta^{-1} \cdot m)$. The $j \in \{0, 1, ..., N/2-1\}$-th entry in z is $z_j = \lfloor \Delta^{-1} \cdot m(\zeta_{2N}^{5^j}) \rceil$, and $\zeta_{2N} = exp(\frac{2\pi i}{2N})$ is the $2N$-th root of unity.
- $\mathsf{FHE.Add}([\![c_a]\!], [\![c_b]\!]) \rightarrow [\![c_{add}]\!]$: Given two ciphertexts $[\![c_a]\!]$ and $[\![c_b]\!]$, the addition ciphertext $[\![c_{add}]\!]$ is returned as $[\![c_{add}]\!] = (c_{a_0} + c_{b_0}, c_{a_1} + c_{b_1})$.
- $\mathsf{FHE.Mul}([\![c_a]\!], [\![c_b]\!]) \rightarrow [\![c_{mul}]\!]$: Given $[\![c_a]\!]$ and $[\![c_b]\!]$, the multiplication ciphertext $[\![c_{mul}]\!]$ is returned as $[\![c_{mul}]\!] = (d_0, d_1, d_2) = (c_{a_0} \cdot c_{b_0}, c_{a_0} \cdot c_{b_1} + c_{a_1} \cdot c_{b_0}, c_{a_1} \cdot c_{b_1})$. With the multiplication ciphertext $[\![c_{mul}]\!]$ and the evaluation key evk, the final result is relinearized as $[\![c_{mul}]\!] = (d_0, d_1) + \lfloor q^{-1} d_2 \cdot evk \rceil$.

Fig. 2. CKKS algorithms.

B. Conditional Generative Adversarial Network

Conditional Generative Adversarial Network (CGAN) [22], [23] is one of the most popular GAN schemes, which comprises of a discriminator $\mathbb{D}$ and a generator $\mathbb{G}$. $\mathbb{G}$ captures the data distribution p_{data}, and $\mathbb{D}$ estimates the probability of a data sample came from true data rather than synthetic data by $\mathbb{G}$. The model parameters W_d, W_g from $\mathbb{D}$ and $\mathbb{G}$ are trained simultaneously. The class labels y are auxiliary information provided to $\mathbb{D}$ and $\mathbb{G}$. Let $x \sim p_{data}(x)$ denote a random sample x drawn from p_{data}, and let $z \sim p_z(z)$ denote a random variable z drawn from p_z. The objective function of GAN is defined as a two-player min-max game with a value function $V(\mathbb{D}, \mathbb{G})$ such that

$$\min_{\mathbb{G}} \max_{\mathbb{D}} V(\mathbb{D}, \mathbb{G}) = \mathbb{E}_{x \sim p_{data}(x)}[\log \mathbb{D}(x|y)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathbb{D}(\mathbb{G}(z|y))]. \quad (4)$$

C. Fully Homomorphic Encryption

Given a plaintext message x, a ciphertext $[\![c]\!]$ is yielded with FHE. FHE schemes can support secure addition "$\oplus$" and multiplication "$\otimes$" operations over ciphertexts, whose results are the same as the plaintext operations such that

$$[\![c_a]\!] \oplus [\![c_b]\!] = [\![c_a + c_b]\!], \quad [\![c_a]\!] \otimes [\![c_b]\!] = [\![c_a \times c_b]\!]. \quad (5)$$

FLGAN exploits Cheon-Kim-Kim-Song (CKKS) cryptosystem [24], [25], which is a leveled FHE scheme based on the learning with errors problem [26]. Besides, CKKS scheme supports Single Instruction Multiple Data (SIMD) operations by packing plaintext messages into different slots in the ciphertext, with which the secure addition and multiplication can be operated in parallel. As demonstrated in Fig. 2, CKKS cryptosystem contains the following five algorithms, i.e., KeyGen, Enc, Dec, FHE.Add and FHE.Mul. For the pure presentation, Enc and Dec algorithms support encoding/decoding techniques for parallel computation. FHE.Mul supports the re-linearization to reduce the size of ciphertexts.

CKKS Primitive: Let $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ be sets of integers, rational numbers, real numbers, and complex numbers, respectively. For a power-of-two integer N, the plaintext space $\mathcal{R} = \mathbb{Z}[x]/(x^N + 1)$ and the ciphertext space $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^N + 1)$ are denoted as a ring of integers of the $(2^{\tilde{}}N)$-th cyclotomic field, q is the modulus in the ciphertext space. $\mathcal{E}$ means the error distribution which is a truncated discrete Gaussian distribution. $\mathcal{HWT}(h)$ is a subset of vector in $\{0, 1\}^N$ with Hamming weight h. The canonical embedding σ embeds $a \in \mathbb{Q} = \mathbb{Q}[x]/(x^N + 1)$ into an element of $\mathbb{C}^N$ whose elements are values of a evaluated at the distinct roots of $x^N + 1$. CKKS can encrypt a complex vector with $(l \leq N/2)$ entries as a ciphertext. Let Δ be the scaling factor, σ be the standard deviation of $\mathcal{E}$, $\mathbb{H} = \{(z_j)_{j \in \mathbb{Z}_{2^{\tilde{}}N}^*} : z_j = \overline{z_{-j}}\}$, and π be a nature projection from $\mathbb{H}$ to $\mathbb{C}^{N/2}$.

IV. Problem Formulation

In this section, we discuss the system model, threat model and design goals in FLGAN.

A. System Model

As depicted in Fig. 3, the system model consists of a central server S and multiple users $U = \{U_1, \ldots, U_n\}$ $(n \geq 2)$. Each role of the entity is defined as follows:

- *User:* $U_{i \in [1,n]}$ takes charge of generating and distributing system keys (Step ①). Besides, $U_{i \in [1,n]}$ holds a localized dataset X_i, and trains a local GAN model. The local data distribution and local generator parameters are sent to the server (Step ②).
- *Central server:* To achieve the goal of learning the global data distribution p_{data} with privacy guarantees, S performs the privacy-preserving GAN selection process. During the process, S selects the appropriate local generators according to encrypted local data distribution to build the federated GAN on $U_{i \in [1,n]}$ (Step ③).

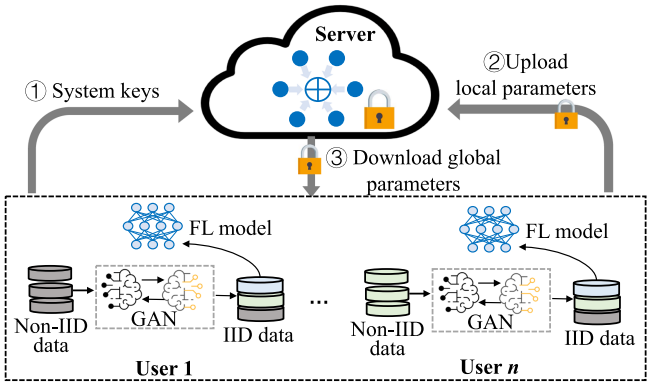


Fig. 3. System model.

B. Threat Model

In general, users in U are fully trusted as the goal of FLGAN is to protect user-level privacy. Consider the prevalence of privacy leakage in servers [27], S is viewed as *honest-but-curious* that rigidly executes the predefined protocol but aims to reconstruct local private distribution for prying user-level privacy. Without loss of generality, U and S are assumed to be non-colluding. The collusion between users in U and S is highly unlikely due to U's unwillingness. The reason is that the collusion would harm U's own interests by revealing user-level privacy [28].

We consider the external adversaries. The potential threats caused by these adversaries are shown as follows:

- The adversary strictly follows the *honest-but-curious* model [29] and tries to learn user-level privacy, which is restricted from probabilistic polynomial-time computation capacity.
- The adversary can observe encrypted local updates and encrypted aggregated updates, with which the adversary tries to reconstruct user-level data [3], [14].

C. Design Goals

FLGAN is designed to achieve two following design goals. Before that, we give a definition of *User-level privacy* (Definition 1). Without loss of generality, all local data are treated as non-IID. Specifically, we give a formal definition of IID and non-IID data in Definition 2 [12], [16]. In a non-IID scenario, the global distribution is composed of distinct local data distributions $p_{i\in[1,n]}$. The biased data distribution p_i of U_i is considered as user-level privacy [3].

*Definition 1 (*User-Level Privacy*):* Let the local data distribution p_i, local training data X_i, local FL model, and locally-trained GAN model (i.e., $\mathbb{D}_i$ and $\mathbb{G}_i$) be private information of $U_{i\in[1,n]}$, with which local data can be observed and inferred by the adversary.

*Definition 2 (*IID versus Non-IID*):* Give the global data distribution p_{data} and local distributions $p_{\forall i\in[1,n]}$, the federated setting is IID if $p_i = p_{data}$, $\forall i \in [1, n]$. The federated setting is non-IID if $\exists x, p_i(x) \neq p_j(x)$, $\exists i, j \in [1, n]$, where p_i represents the empirical distribution of the local data X_i, and $p_{data} = \sum_{i\in[1,n]} p_i$.

Formally, these two goals are outlined as follows.

*Goal 1 (*User-Level Privacy Guarantee*):* FLGAN should strictly follow the federated architecture, which ensures the construction of the federated GAN without accessing user-held training data. Besides, the adversary should fall to observe and infer any user-level privacy from locally-submitted parameters.

*Goal 2 (*Unbiased FL With non-IID Data*):* FLGAN should construct the federated GAN under non-IID data without affecting the FL process. The federated GAN is adopted to implement the fine-grained data augmentation according to diverse local distributions of user-held training data, which can use the global generator to generate "true" data with the labels lacking in local non-IID data. These generated data are added to user-held data to convert the local non-IID data into IID data, implementing the unbiased non-IID FL.

V. Proposed Scheme

In this section, we first show the technique overview about how to design FLGAN, then demonstrate the concrete construction of FLGAN.

A. Technique Overview

The overall structure of FLGAN is illustrated in Fig. 4. The goal of FLGAN is to construct a federated GAN to generate IID data for the unbiased non-IID FL, which strictly follows the federated architecture and ensures user-level privacy. To achieve this, we first design the *divide-and-conquer* strategy, which divides the federated GAN training problem into local training (Phase 1) and conquers the problem through global training. Instead of transmitting local training data, each user shares local generator and local data distribution in FLGAN. Then, S selects a subset of local generators trained from dissimilar data distributions for each user. To protect user-level privacy, FHE is exploited to provide the *privacy-preserving generator selection* method (Phase 2), with which local updates and intermediate parameters are indistinguishable in S. Eventually, the federated GAN engages to implement the *fine-grained GAN augmentation* (Phase 3) to convert local non-IID data into IID data, thereby implementing the unbiased non-IID FL.

In the *divide-and-conquer* strategy, we define the local loss and global loss for training a federated GAN in FLGAN. Under the non-IID setting, true data distributions of U are presented as $p_{data}(x) = \{p_1(x), \ldots, p_n(x)\}$. U_i locally trains his/her local GAN models $(\mathbb{G}_i, \mathbb{D}_i)$.

Local loss. Given true training samples $x \in X_i$ and true local data distribution $p_i(x)$, U_i optimizes $(\mathbb{G}_i, \mathbb{D}_i)$ by minimizing the local loss $\mathcal{L}_{U_i}$

$$\mathcal{L}_{U_i} = \min_{\mathbb{G}_i} \max_{\mathbb{D}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i) = \mathbb{E}_{x\sim p_{data}}[\log \mathbb{D}_i(x|y)] + \mathbb{E}_{z\sim p_z}[\log(1 - \mathbb{D}_i(\mathbb{G}_i(z|y)))]. \quad (6)$$

To reach the local optimum, we divide $V_{U_i}(\mathbb{D}_i, \mathbb{G}_i)$ into two sub-objective functions based on (4):

- $\mathbb{D}_i$ is trained to discriminate generated samples from true data by $\max V_{U_i}(\mathbb{D}_i, \mathbb{G}_i)$.

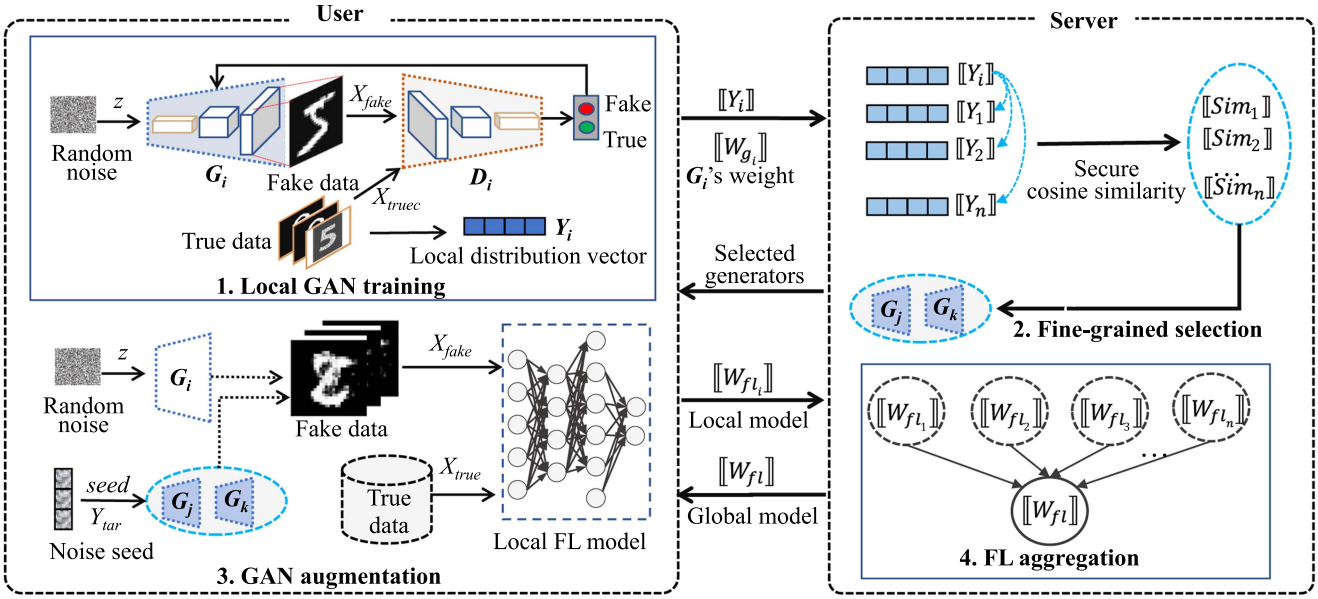


Fig. 4. Overview of FLGAN.

- The feedbacks of a local optimum $\mathbb{D}_i$ guide $\mathbb{G}_i$ to flow the data regions that are more likely to be classified as "true" data. $\mathbb{G}_i$ is updated to $\min V_{U_i}(\mathbb{D}_i, \mathbb{G}_i)$.

Global loss. Since $(\mathbb{D}_i, \mathbb{G}_i)$ is trained on the local distribution p_i, we redefine a global loss $\mathcal{L}_\Omega$ to train the federated GAN $(\mathbb{D}_\Omega, \mathbb{G}_\Omega)$ as

$$\mathcal{L}_\Omega = \min_{\mathbb{G}_i} \max_{\mathbb{D}_i} V_\Omega(V_{U_1}(\mathbb{D}_1, \mathbb{G}_1), \ldots, V_{U_n}(\mathbb{D}_n, \mathbb{G}_n)). \quad (7)$$

In practice, the locally-trained $\mathbb{G}_i^*$ cannot generate any IID samples with the standards of local optimum $\mathbb{D}_i^*$, as a local $\mathbb{D}_i^*$ is inferior to impede the $\mathbb{G}_\Omega$ training. Our goal is to construct the global optimum $p_g(x) = p_{data}(x)$. To guide $\mathbb{G}_i$ toward amassing $p_g(x)$ in approximately correct region of the global data X, it is necessary to introduce all local distributions by using generators $\mathbb{G}_j^*(x)$ [30]. Thus, we reformulate the global loss $\mathcal{L}_\Omega$ as

$$\mathcal{L}_\Omega = \min_{\mathbb{G}_i} \max_{\mathbb{D}_i} V_\Omega(V_{U_1}(\mathbb{D}_\Omega, \mathbb{G}_\Omega), \ldots, V_{U_n}(\mathbb{D}_\Omega, \mathbb{G}_\Omega)). \quad (8)$$

B. Construction of FLGAN

The concrete construction of FLGAN consists of four algorithms, namely initialization, local GAN training, privacy-preserving generator selection, and GAN augmentation, which are defined as follows.

1) Initialization: As described in Algorithm 1, U implements KeyGen with the security parameter ψ to generate pk for all entities (i.e., $U_{i\in[1,n]}$ and S), sk for $U_{i\in[1,n]}$, and evk for S. The public key pk is used to encrypt local parameters for $U_{i\in[1,n]}$, and the secret key sk is used to decrypt encrypted parameters. The evaluation key evk is used to relinearize the multiplication result in FHE.Mul for S. Then, $U_{i\in[1,n]}$ downloads and initializes

Algorithm 1: Initialization.

Input: Security parameter ψ.
Output: System keys pk, sk, evk.

```
/*Key distribution*/
U generates (pk, sk, evk) ← KeyGen(1^ψ);
U broadcasts pk and distributes evk to S, sk is
  shared among U_{i∈[1,n]};
/*Model distribution*/
for U_{i∈[1,n]} ∈ U do
    U_i downloads (W_d^(0), W_g^(0)) and W^(0) from S;
    U_i initializes (W_d^(0), W_g^(0)) and W^(0).
```

a common GAN model weights $(W_d^{(0)}, W_g^{(0)})$ of $(\mathbb{D}^{(0)}, \mathbb{G}^{(0)})$ and a common FL model $W^{(0)}$ from S.

2) Local GAN Training: At the t-th training round, $U_{i\in[1,n]}$ trains the local GAN $(\mathbb{D}_i^{(t)}, \mathbb{G}_i^{(t)})$ over individual non-IID data X_i. The details are demonstrated in Algorithm 2. In advance, U_i agrees on the label size κ of global training data. Then, U_i samples a mini-batch of noise from the prior noise distribution $p_z(z)$ as a seed vector $seed = \{z_1, \ldots, z_m\} \sim p_z(z)$. After that, $\mathbb{D}_i^{(t)}$ obtains the generated "fake" data $X_{fake}^{(t)} = \{\widetilde{x}_1, \ldots, \widetilde{x}_m\} \sim \mathbb{G}_i^{(t)}(seed)$ from the generated samples $\mathbb{G}_i^{(t)}(seed)$. Subsequently, U_i samples a mini-batch of true data samples $X_{true}^{(t)} = \{x_1, \ldots, x_m\} \sim p_i(x)$ from the local distribution of training data $p_i(x)$. The generated data $X_{fake}^{(t)}$ are labeled as "fake", the original training data $X_{true}^{(t)}$ are labeled as "true". To minimize (6), these data $\overline{X}_i^{(t)} = X_{true}^{(t)} \cup X_{fake}^{(t)}$ are fed into the local GAN $(\mathbb{D}_i^{(t)}, \mathbb{G}_i^{(t)})$ with the following phases.

Algorithm 2: Local GAN Training.

Input: "Fake" samples $X_{fake}^{(t)}$ and true samples $X_{true}^{(t)}$.
Output: Encrypted local parameters $[\![W_{g_i}^{(t)}]\!]$, $[\![Y_i]\!]$.

for $U_{i\in[1,n]} \in U$ **do**
  U_i samples $seed = \{z_1, ..., z_m\} \sim p_z(z)$;
  $\mathbb{D}_i^{(t)}$ receives $X_{fake}^{(t)} = \{\widetilde{x}_1, ..., \widetilde{x}_m\} \sim \mathbb{G}_i^{(t)}(seed)$;
  U_i samples $X_{true}^{(t)} = \{x_1, ..., x_m\} \sim p_i(x)$;
  $\overline{X}_i^{(t)} = X_{true}^{(t)} + X_{fake}^{(t)}$ are fed into $\mathbb{D}_i^*$;
  for $x_i \in X_{true}^{(t)}$, $\widetilde{x}_i \in X_{fake}^{(t)}$ **do**
    /*Local discriminator training*/
    U_i maximizes $V_{U_i} = \frac{1}{m}\sum_{i=1}^{m} \log \mathbb{D}_i^{(t-1)}(x_i) + \frac{1}{m}\sum_{i=1}^{m} \log(1 - \mathbb{D}_i^{(t-1)}(\widetilde{x}_i))$ by Eq. 6;
    U_i updates $W_{d_i}^{(t)} \leftarrow W_{d_i}^{(t-1)} + \eta \nabla V_{U_i}(W_{d_i}^{(t-1)})$;
    /*Local generator training*/
    U_i updates $W_{g_i}^{(t)} \leftarrow W_{g_i}^{(t-1)} - \eta \nabla V_{U_i}(W_{g_i}^{(t-1)})$ to minimize V_{U_i};
  /*Local feedbacks*/
  U_i constructs $Y_i = \{\rho_1 ..., \rho_\kappa\}$ and normalizes Y_i as $\|Y_i\| = 1$;
  U_i uploads $[\![W_{g_i}^{(t)}]\!] \leftarrow \mathsf{Enc}_{pk}(W_{g_i}^{(t)}, \Delta)$ and $[\![Y_i]\!] \leftarrow \mathsf{Enc}_{pk}(Y_i, \Delta)$ to S.
return $[\![W_{g_i}^{(t)}]\!]$ and $[\![Y_i]\!]$.

Local discriminator training. U_i inputs the generated "fake" data $X_{fake}^{(t)}$ and true data $X_{true}^{(t)}$ into $\mathbb{D}_i$ to maximize the objective function $\max_{\mathbb{D}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i)$. Then, the local discriminator weight $W_{d_i}^{(t)}$ is updated to maximize V_{U_i} such that

$$\max_{\mathbb{D}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i) = \frac{1}{m}\sum_{i=1}^{m} \log \mathbb{D}_i^{(t-1)}(x_i) + \frac{1}{m}\sum_{i=1}^{m} \log(1 - \mathbb{D}_i^{(t-1)}(\widetilde{x}_i)),$$

Local generator training. U_i updates the local generator $W_{g_i}^{(t)}$ to minimize V_{U_i}. Note that gradient-based updates can use any standard gradient-based learning methods, e.g., Stochastic Gradient Descent (SGD) and momentum.

Local feedbacks. After reaching the local optimization, U_i constructs a κ-dimensional vector $Y_i \in \mathbb{C}^\kappa$ that contains local distribution information, $\kappa \ll N/2$. The element $\rho_k \in Y_i$ is set as $\rho_k = \frac{|\varpi_k|}{|X_i|}$, where $|\varpi_k|$ is the size of samples of the $k \in [1, \kappa]$-th label y_k. The local distribution p_i is denoted as

$$p_i = \rho_1 \cdot P_{label_1} + \rho_2 \cdot P_{label_2} + \cdots + \rho_\kappa \cdot P_{label_\kappa} = \sum\nolimits_{k\in[1,\kappa]} \rho_k \cdot P_{label_k},$$

where P_{label_k} is the global distribution of data samples belonging to the k-th label. To implement secure outsourcing, we use the SIMD technique, with which the number of slots is $N/2$. Note that a slot can encode a real number. $U_{i\in[1,n]}$ outsources weights $[\![W_{g_i}^{(t)}]\!]$ of $\mathbb{G}_i^{(t)}$ by using the Enc algorithm such that

$$[\![W_{g_i}^{(t)}]\!] = (c_0, c_1) = (x, 0) + pk = (b + x, a) \in \mathcal{R}_q^2,$$
$$x = \delta^{-1}(\lfloor \Delta \cdot \pi^{-1}(W_{g_i}^{(t)}) \rceil_{\delta(\mathcal{R})}) \in \mathcal{R}, \quad (9)$$

where $W_{g_i}^{(t)} \in \mathbb{C}^{N/2}$ is represented as a vector, and the size is $|W_{g_i}^{(t)}| < N/2$. Finally, U_i uploads $[\![W_{g_i}^{(t)}]\!]$ and $[\![Y_i]\!]$ to S, where Y_i is normalized as $\|Y_i\| = 1$.

3) Privacy-Preserving Generator Selection: FLGAN should guarantee user-level privacy. $U_{i\in[1,n]}$ needs to submit sensitive parameters (i.e., local generators and local labels) to S, in which local data are easily reconstructed with local parameters. To keep these user-level parameters secret from S, FLGAN designs the privacy-preserving generator selection method based on FHE, which is depicted in Algorithm 3.

Algorithm 3: Privacy-Preserving Generator Selection.

Input: Encrypted local parameters $[\![W_{g_i}^{(t)}]\!]$, $[\![Y_i]\!]$.
Output: The selected local generators $[\![W_{g_i}^{(t)}]\!]$ for U_i.

/*Secure cosine similarity*/
for $i, j \in [1, n], i \neq j$ **do**
  S calls $\mathsf{FHE.Mul}$ and $\mathsf{FHE.Add}$ to obtain $[\![\cos_{i,j}]\!] \leftarrow [\![Y_i]\!] \cdot [\![Y_j]\!]^T$ by Eq. 10;
/*Local selection*/
for $U_{i\in[1,n]} \in U$ **do**
  S returns $[\![\mathsf{CS}_i]\!] = \{[\![\cos_{i,1}]\!], ..., [\![\cos_{i,n}]\!]\}$ to U_i;
  for $j \in [1, n], j \neq i$ **do**
    $\cos_{i,j} \leftarrow \mathsf{Dec}_{sk}([\![\cos_{i,j}]\!])$;
    if $\cos_{i,j} < tv_i$ **then**
      $[\![W_{g_j}^{(t)}]\!]$ is selected by U_i;
return The selected generators $[\![W_{g_j}^{(t)}]\!]$ for U_i.

Secure cosine similarity. S receives encrypted label vectors $[\![Y_{i\in[1,n]}]\!]$. Then, S executes a secure cosine similarity among $[\![Y_{i\in[1,n]}]\!]$ to calculate the cosine similarity $\cos_{i,j}$ between $[\![Y_i]\!]$ and $[\![Y_j]\!]$ as

$$\cos_{i,j} = \frac{Y_i \cdot Y_j^T}{\|Y_i\| \cdot \|Y_j\|} = Y_i \cdot Y_j^T, \quad s.t.\ \|Y_i\| = \|Y_j\| = 1. \quad (10)$$

We approximate the cosine similarity to the secure inner product [31] between two vectors, which calls $\mathsf{FHE.Mul}$ and $\mathsf{FHE.Add}$ algorithms. Finally, S returns the $(n-1)$-dimensional encrypted set $[\![\mathsf{CS}_i]\!]$ to U_i for a fine-grained selection over encrypted local generators $[\![W_{g_i}^{(t)}]\!]$.

Local Selection. Once receiving $[\![\mathsf{CS}_i]\!]$, U_i obtains CS_i by calling Dec. To train FLGAN, it is crucial to learn the global data distribution, which means capturing the underlying patterns and structures present across the global dataset in FL. The GAN training starts harshly for the standard $\max_{\mathbb{D}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i)$ with limited training knowledge. To obtain constructive feedbacks to $\mathbb{G}_i$, local generators can capture different aspects of local

training data, and the combination of these generators can provide a complete representation of the global data distribution. It is necessary for U_i to select local generators from distinct data distributions. U_i sets a threshold value tv_i to select local generators in S, where tv_i is the similarity threshold of label distribution ($tv_i > 0.5$). If $\cos_{i,j} < tv_i$, then $[\![W_{g_j}^{(t)}]\!]$ is selected for U_i. The lower $\cos_{i,j}$ is, the lower similarity of data distribution between local data X_i and X_j is. The bigger possibility of $\mathbb{G}_j^{(t)}$ can be selected by U_i. Thus, it is more important to select U_j's generator $\mathbb{G}_j^{(t)}$ to construct $\mathbb{D}_\Omega$ for U_i. Therefore, S is more likely to select the local generator $[\![W_{g_j}^{(t)}]\!]$ to U_i.

Remark. As GAN training improves, tv_i needs to be more critical of the local training of $\mathbb{G}_i$ with the growth of tv_i. To construct FLGAN, it is highly recommended to prioritize the inclusion of diverse generators with distinct data distributions in order to maximize the model's potential for capturing the global data distribution and producing high-quality outputs. Thus, U_i adaptively sets the threshold value of tv_i closer to 0, which grows with the training rounds. tv_i is obtained as

$$tv_i = \frac{1}{1 + e^{(t-\phi)}}, \tag{11}$$

where $0 < tv_i < 1$, t is training rounds and ϕ is the threshold rounds. Without loss of generality, we set $\phi = 10$ in FLGAN.

4) GAN Augmentation: To train the federated GAN $(\mathbb{D}_\Omega, \mathbb{G}_\Omega)$, U_i executes the GAN augmentation process with selected generators to minimize (7), which is shown in Algorithm 4.

Fine-grained augmentation. Upon receiving selected generators $[\![W_{g_j}^{(t)}]\!]$ and according local label vectors $[\![Y_j]\!]$, U_i obtains $\mathbb{G}_j^{(t)}$ and $Y_j = \{\rho_0, \rho_1 \ldots, \rho_\kappa\}$ by calling the Dec algorithm. Since Y_j includes the labels of local training data X_j, U_i identifies the target label vector $Y_{tar} = \{y_1, \ldots, y_k\}$ lacked in individual true training data X_i and fed in selected generators $\mathbb{G}_j^{(t)}$. Given the selected generators $[\![W_{g_j}^{(t)}]\!]$, U_i uploads the data seed z of the label vector Y_{tar}. U_i replenishes the local data X_i with the generated "true" data $\overline{X}_{true}$ of target labels being lacking labels in X_i. Finally, U_i balances the local non-IID training data X_i to IID training data $\overline{X}_i$.

Local GAN Tuning. With generated "true" data $\overline{X}_{true}$ and original local data X_i, U_i trains the local GAN by sampling a mini-batch of true data samples from $\overline{X}_i$. These true data samples from X_{true} are labeled as "true". U_i labels the generated "fake" data X_{fake} as "fake". These data $\overline{X}_i$ are all fed into the local GAN to train $\mathbb{D}_i$ and $\mathbb{G}_i$.

FL Training. FLGAN supports different FL strategies, e.g., FedAvg and FedSGD. $U_{\forall i \in [1,n]}$ locally trains local updates (i.e., local FL weights $W_i^{(t)}$ or local gradients $\mu_i^{(t)}$) on true data $\overline{X}_i$ ($\overline{X}_i = \overline{X}_{true} \cup X_i$), which is demonstrated in Section III-A. Upon receiving encrypted inputs, S implements the aggregation process on local updates by calling FedAvg or FedSGD algorithms.

Algorithm 4: GAN Augmentation.

Input: Encrypted selected generators $[\![W_{g_i}^{(t)}]\!]$.
Output: The FL model $W^{(t+1)}$.

for $U_{i\in[1,n]} \in U$ **do**
  */*Fine-grained augmentation*/*
  for $j \in [1,n]$, $j \neq i$ **do**
    U_i decrypts $W_{g_j}^{(t)} \leftarrow \mathsf{Dec}_{sk}([\![W_{g_j}^{(t)}]\!])$ and $Y_j \leftarrow \mathsf{Dec}_{sk}([\![Y_j]\!])$;
    U_i sets $Y_{tar} = \{y_1, ..., y_k\}$ being lacking in X_i;
    U_i generates $\overline{X}_{true} = \{\widetilde{x}_1, ..., \widetilde{x}_m\} \sim \mathbb{G}_j^{(t)}(z)$ of Y_{tar} using $W_{g_j}^{(t)}$;
  $\overline{X}_i = \overline{X}_{true} \cup X_i$;
  */*Local GAN tuning*/*
  U_i samples $X_{fake} = \{\widetilde{x}_1 ... \widetilde{x}_m\} \sim \mathbb{G}_i^{(t)}(z)$ and $\overline{X}_i$;
  $W_{d_i}^{(t)} \leftarrow W_{d_i}^{(t-1)} + \eta \nabla V_{U_i}(W_{d_i}^{(t-1)})$;
  $W_{g_i}^{(t)} \leftarrow W_{g_i}^{(t-1)} - \eta \nabla V_{U_i}(W_{g_i}^{(t-1)})$;
  */*FL training*/*
  U_i locally trains $W_i^{(t)}$ on $\overline{X}_i = \overline{X}_{true} \cup X_i$;
  if *FL strategy is FedAvg* **then**
    U_i sends the locally trained weight $W_i^{(t)}$ to S;
  if *FL strategy is FedSGD* **then**
    U_i sends the locally trained gradient $\mu_i^{(t)}$ to S;
*/*FL aggregation*/*
if *FL strategy is FedAvg* **then**
  S aggregates $W^{(t+1)} \leftarrow \frac{1}{n}\sum W_i^{(t)}$ to $U_{i\in[1,n]}$;
if *FL strategy is FedSGD* **then**
  S aggregates $W^{(t+1)} \leftarrow W^{(t)} - \eta\frac{1}{n}\sum \mu_i^{(t)}$;
for $U_i \in U$ **do**
  U_i updates $W_{fl}^{(t+1)} \leftarrow W_{fl}^{(t)} - \eta\frac{1}{n}\sum \nabla F_i(W_{fl_i}^{(t)})$;
return The FL model $W^{(t+1)}$.

VI. Theoretical Analysis

In this section, we theoretically prove the security and unbiasedness of FLGAN.

A. Privacy Analysis

The security goal of FLGAN is to ensure that user-level privacy cannot be leaked to S. As the only process implemented in S is the *privacy-preserving generator selection* phase, we give a following theorem of the security goal of FLGAN. Our privacy analysis is based on the proofs from the CKKS cryptosystem [24], [25].

Theorem 1 (User-Level Security of FLGAN): FLGAN can guarantee user-level security, if S cannot distinguish an executed functionality Σ in the real-world $\text{REAL}_{\text{ADV}}^{\Sigma}(\lambda)$ from an execution in the ideal-world $\text{IDEAL}_{\text{ADV,SIM}}^{\Sigma}(\lambda)$ for all polynomial-time-bounded adversaries ADV on a polynomial parameter λ, there

exists a straight-line simulator SIM such that

$$|\mathsf{Pr}[\text{REAL}^{\Sigma}_{\text{ADV}} = 1] - \mathsf{Pr}[\text{IDEAL}^{\Sigma}_{\text{ADV,SIM}} = 1]| \leq \mathsf{negl}(\lambda).$$

Proof: The $\text{REAL}^{\Sigma}_{\text{ADV}}(\lambda)$ and $\text{IDEAL}^{\Sigma}_{\text{ADV,SIM}}(\lambda)$ are defined as follows.

- $\text{REAL}^{\Sigma}_{\text{ADV}}(\lambda)$: ADV performs the privacy-preserving generator selection on local label vectors $[\![Y_{i\in[1,n]}]\!]$ by calling FHE.Add and FHE.Mul algorithms. ADV then observes encrypted inputs $[\![Y_i]\!]$, and finally outputs n encrypted sets $[\![\mathsf{CS}_i]\!] = \{[\![\text{cos}_{i,1}]\!], \dots [\![\text{cos}_{i,n}]\!]\}$.
- $\text{IDEAL}^{\Sigma}_{\text{ADV,SIM}}(\lambda)$: ADV performs the privacy-preserving generator selection on simulated local label vectors $\text{SIM}([\![\widetilde{Y}_{i\in[1,n]}]\!])$ by calling FHE.Add and FHE.Mul algorithms. Finally, ADV observes all simulated parameters $[\![\widetilde{Y}_{i\in[1,n]}]\!]$, and outputs n encrypted sets $[\![\widetilde{\mathsf{CS}}_i]\!]$.

Let $[\![W^{(t)}_{g_i}]\!]$ and $[\![Y_i]\!]$ be the uploaded local parameters that include the user-level privacy of local data distribution p_i. We define the real-world $\text{REAL}^{\Sigma}_{\text{ADV}}(W^{(t)}_{g_i}, Y_i)$ and the simulated world $\text{IDEAL}^{\Sigma}_{\text{ADV,SIM}}(W^{(t)}_{g_i}, Y_i)$, aiming to prove that the user-level sensitive information is computationally indistinguishable on S. The security proof in FLGAN mainly focuses on the *privacy-preserving generator selection* phase.

S knows nothing about local parameters $[\![W^{(t)}_{g_i}]\!]$, $[\![Y_i]\!]$ and the secure cosine similarity $\text{cos}_{i,j}$. As illustrated in (9), these parameters are encrypted with pk. The original values are unlikely to be leaked to S, unless S acquires sk. During the privacy-preserving generator selection process, the adversary ADV can only hold evk for FHE.Mul and FHE.Add. Following the security definition of CKKS, the securities of FHE.Mul and FHE.Add are also captured using the real-world versus ideal-world game. We refer readers to [24], [25] for formal analysis. Based on computationally indistinguishable algorithms (i.e., FHE.Mul and FHE.Add), ADV cannot observe any sensitive information without sk. Hence, the adversary ADV cannot distinguish between the experiments $\text{REAL}^{\Sigma}_{\text{ADV}}(W^{(t)}_{g_i}, Y_i)$ and $\text{IDEAL}^{\Sigma}_{\text{ADV,SIM}}(W^{(t)}_{g_i}, Y_i)$. Because we exclude the possibility of privacy leakage among $\mathcal{U}$, FLGAN is secure if the user-level privacy cannot be leaked to S. By combining contributions from all games, FLGAN can guarantee user-level privacy against polynomial-time-bounded adversaries, without leaking user-level privacy to S. □

B. Convergence Analysis

The goal of FLGAN is to guarantee the unbiasedness of FLGAN under the non-IID setting, wherein the federated GAN learns the global data distribution. We prove the convergence of FLGAN with the following theorem and lemma.

Theorem 2 (Convergence of FLGAN): Let p_{g_i} be the local optimum of data generated by $\mathbb{G}_{i\in[1,n]}$, and p^*_g be the optimal generated distribution of the federated GAN. FLGAN has p^*_g equals to the true global data distribution p_{data}. Formally, the global minimum of $\mathcal{L}_{\mathbb{D}_i}$ given the federated discriminator $\mathbb{D}_i$ is achieved if and only if $p^*_g = p_{data}$.

Proof: Assume all local generators $\mathbb{G}_{i\in[1,n]}$ are locally optimal. i.e., they learn to generate realistic-looking data that are distinguished as "true" by local discriminators $\mathbb{D}_{i\in[1,n]}$. A local generator $\mathbb{G}_{i\in[1,n]}$ is optimal if $p_{g_i} = p_i$.

Lemma 1: Each local discriminator $\mathbb{D}_i$ is trained optimally from true data distribution $p_i(x)$ and the generated distribution p_{g_i} of local generator $\mathbb{G}_i$. The global minimum of $\mathcal{L}_{\mathbb{D}_i}$ given $\mathbb{D}_i$ is achieved if and only if $\mathbb{D}_i(x) = \frac{p_i(x)}{p_i(x)+p_{g_i}(x)}$.

To update local generators $\mathbb{G}_i$ under different non-IID levels, we augment $\mathcal{L}_{\mathbb{G}_i}$ with selected generators $\mathbb{G}_j$. Thus, the selected generators $\mathbb{G}_{j\in[1,n]/i}$ synthesize the optimally generated distributions p_{g_j}, which can augment the local data distribution p_i as

$$p^*_i = p_i + \sum_{j\in[1,n]/i} p_{g_j} = p_i + \sum_{j\in[1,n]/i} p_j \approx p_{data}. \quad (12)$$

These generated data from selected generators $\mathbb{G}_{j\in[1,n]/i}$ and local training data are fed into the local discriminator $\mathbb{D}_i$ for a local optimum $\mathbb{D}^*_i(x)$. U_i inputs generated "fake" data and true data into $\mathbb{D}_i$ to maximize the objective function V_{U_i} as

$$\begin{aligned}
&\max_{\mathbb{D}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i) \\
&= \mathbb{E}_{x\sim p_i(x)}[\log \mathbb{D}_i(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-\mathbb{D}_i(\mathbb{G}_i(z))] \\
&= \int_x p_i(x)\log \mathbb{D}_i(x)dx + \int_x p_{g_i}(x)\log(1-\mathbb{D}_i(x))dx \\
&= \int_x \left[p_i(x)\log \mathbb{D}_i(x) + p_{g_i}(x)\log(1-\mathbb{D}_i(x))\right]dx.
\end{aligned}$$

We define a function $F(\mathbb{D}_i(x))$ as

$$F(\mathbb{D}_i(x)) = p_i(x)\log \mathbb{D}_i(x) + p_{g_i}(x)\log(1-\mathbb{D}_i(x)).$$

Thus, the optimally local $\mathbb{D}^*_i(x)$ is obtained as

$$\begin{aligned}
\frac{\partial F}{\partial \mathbb{D}_i(x)} &= p_i(x)\times\frac{1}{\mathbb{D}_i(x)} - p_{g_i}(x)\times\frac{1}{1-\mathbb{D}_i(x)} = 0, \\
&\Rightarrow \mathbb{D}^*_i(x) = \frac{p_i(x)}{p_i(x)+p_{g_i}(x)}.
\end{aligned}$$

Hence, $\mathbb{D}^*_i(x)$ is obtained as

$$\mathbb{D}^*_i(x) = \frac{p^*_i(x)}{p^*_i(x)+p_{g_i}(x)} \approx \frac{p_{data}(x)}{p_{data}(x)+p_{g_i}(x)}.$$

Given a local optimum $\mathbb{D}^*_i(x)$, $\mathbb{G}^*_i(x)$ is optimized by minimizing V_{U_i} as

$$\begin{aligned}
&\min_{\mathbb{G}_i} V_{U_i}(\mathbb{D}_i, \mathbb{G}_i) \\
&= \mathbb{E}_{x\sim p_i(x)}[\log \mathbb{D}^*_i(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-\mathbb{D}^*_i(\mathbb{G}_i(z))] \\
&= \mathbb{E}_{x\sim p_i}[\log\frac{p_i(x)}{p_i(x)+p_{g_i}(x)}] + \mathbb{E}_{z\sim p_z}[\log\frac{p_{g_i}(x)}{p_i(x)+p_{g_i}(x)}] \\
&= -2\log 2 + KL(p_i(x)||\frac{p_i(x)+p_{g_i}(x)}{2}) \\
&= -2\log 2 + 2JSD(p_i(x)||p_{g_i}(x)).
\end{aligned}$$

KL means the Kullback-Leibler divergence [32], JSD means the Jensen-Shannon divergence [33], which are used to

TABLE III
DATASET DESCRIPTIONS

Datasets	Data size	Features	Classes	U	Data size/ a user
MNIST	70,000 (10,000 test samples)	$28 * 28$	10	10	6,000
Fashion MNIST	70,000 (10,000 test samples)	$28 * 28$	10	10	6,000
CIFAR-10	60,000 (10,000 test samples)	$32 * 32$	10	10	5,000
USPS	9,298 (1,000 test samples)	$16 * 16$	10	10	800
CIFAR-100	60,000 (10,000 test samples)	$32 * 32$	100	10	5,000

Notes. Convert the USPS samples to MNIST format.

measure the similarity between two probability distributions. More detailed proofs are shown in [22]. Based on (12), the federated $\mathbb{G}_{i\in[1,n]}$ is optimal if $p_{g_i} = p_i^* \approx p_{data}$. □

VII. PERFORMANCE EVALUATION

In this section, comprehensive experiments are conducted to evaluate that FLGAN can achieve the unbiased non-IID FL.

A. Experimental Setup

We evaluated our experiments with PyTorch running on a six-core Ubuntu 18.04 machine with Intel i7-9750H processor 4.50 GHz and 64 GB RAM, NVIDIA GTX 3080 TI GPU. We simulated the FL setting in a single machine with multiple threads, each of which implements real-world PyTorch users. The execution time reports an average time of 10 trials. We adopt the Simple Encrypted Arithmetic Library (SEAL) homomorphic encryption library [34] to implement the CKKS cryptosystem. We set $N = 2^{13}$ to achieve the 128-bit security level with $\log q = 50$-bit and $\Delta = 2^{30}$.

Non-IID datasets. We use the following four standard image datasets to evaluate FLGAN, namely MNIST [35], Fashion MNIST [36], CIFAR-10, CIFAR-100 [37], and USPS [38]. The specified description of these dataset are shown in Table III. Besides, we build the non-IID setting with $|U| = 10$ and divide these datasets into 10 subsets, U_i holds a subset X_i. We set a challenging non-IID setting, wherein X_i only contains data of the same class, the data class held by each subset is non-overlapping. For MNIST, Fashion MNIST, CIFAR-10 data, the subsets $X_1, X_2, \ldots, X_{10}$ contain ground-truth data samples of $\{1\}, \{2\}, \ldots, \{0\}$-th classes, respectively. For CIFAR-100 data, the subsets $X_1, X_2, \ldots, X_{10}$ contain ground-truth data samples of $\{0-9\}, \{10-19\}, \ldots, \{90-99\}$-th classes, respectively.

Performance Metrics. The following two metrics are employed for a comprehensive evaluation of the GAN performance in FLGAN.

- *Fr é chet Inception Distance* (FID) [39] computes the distance between feature vectors of true and generated data samples.
- *Inception Score* (IS) [40] judges the quality of generated data samples, especially for synthetic data samples outputed by GAN models.

Training Parameters. Each user holds two components of FLGAN including GAN and FL model. For the GAN, we utilize the Conditional Deep Convolutional GAN (CDCGAN) model to construct GAN in FLGAN. The generator is constructed with deconvolution layers ($Deconv$), the discriminator and FL model consists of convolutional neural networks. The specified GAN architecture is shown in Table IV.

For the FL model, the LeNet-5 [41] model is adopted as the FL network. The specified parameters are shown in Table V. We compare FLGAN with two FL baselines (i.e., FedAvg and FedSGD) to evaluate the performance of the unbiased non-IID FL. The mini-batch size is 100. For the local GAN held by each user, the SGD optimizer is adopted with the learning rate of 0.0002 and the momentum term of 0.5. For the FL model, the learning rate and the momentum term is 0.01 and 0.9, respectively. We evaluate the test accuracy of FL models using testing data from each dataset. We set the threadshold value $tv_i = 0.8$ for the local GAN selection ($\forall i \in [1, n]$).

B. Experimental Results

1) GAN Augmentation: After multiple training rounds, FLGAN conducts the federated GAN with the privacy-preserving generator selection scheme. To evaluate the quality of the federated GAN, we compare ground-truth data with generated data in FLGAN. The results are depicted in Fig. 5(a)–(c). For these datasets (i.e., MNIST, Fashion MNIST, and USPS), the last column is true samples from ground-truth data. The other columns are generated samples from different training rounds in FLGAN. With the growth of training rounds, we observe that synthesized data are more and more clear. Eventually, synthesized data are indistinguishable from ground-truth data after 50 rounds. Especially, the CIFAR-10 dataset contains diverse data images, making the training convergence more challenging than that of MNIST, Fashion MNIST, and USPS. Fig. 5(d) demonstrates the synthesized data of the "bird" class with 1-100 training rounds. For CIFAR-10, synthesized samples can still be reused as training samples in FL with a slight loss of accuracy, which is demonstrated in Section 7.2.2.

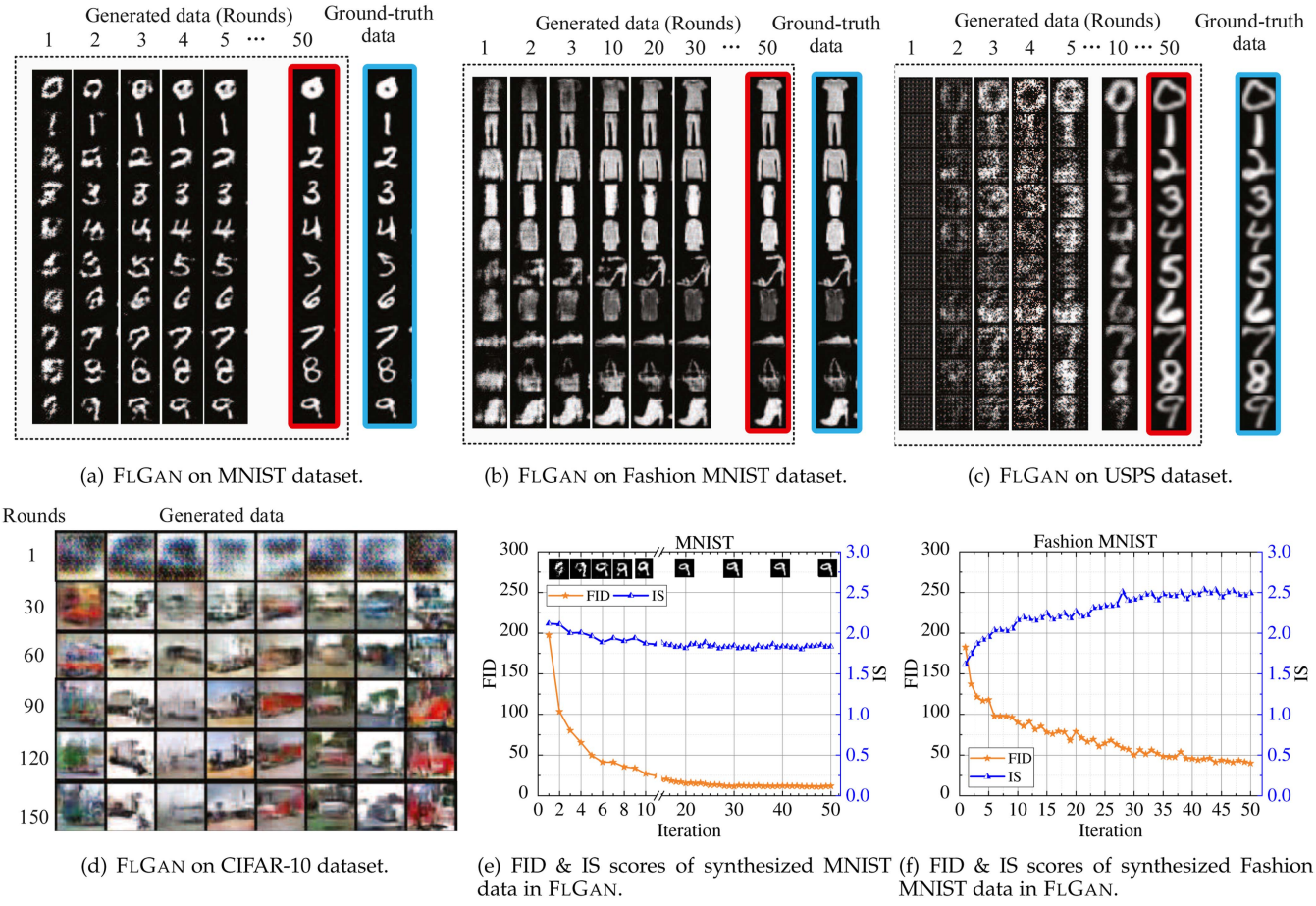
As demonstrated in Fig. 5(e) and (f), we adopt FID and IS scores to evaluate the quality of synthesized data in MNIST and Fashion MNIST. Lower FID scores correlate well with higher quality of synthesized images. Conversely, higher IS scores represent well with higher quality of synthesized samples. We randomly select 6,000 data samples from synthesized data to calculate FID and IS scores. Fig. 5(e) plots that IS scores increase and FID scores decrease with the growth of training rounds. The synthesized image sample (i.e., class="9") gets clearer and clearer. After 20 training rounds, the quality of a federated GAN is gradually stabilized over MNIST data in FLGAN. Fig. 5(f) depicts that ever-greater IS scores and ever-diminishing FID scores are yielded with the increase of training iterations. i.e., The quality of a federated GAN trained on Fashion MNIST data is gradually stabilized in FLGAN. Table VI demonstrates FID & IS scores for each dataset.

Computation overhead. We test the computation overhead of FLGAN, where the privacy-preserving generator selection adopts

TABLE IV
GAN ARCHITECTURE

Dataset	Generator	Discriminator
MNIST, USPS, Fashion MNIST	$100 * 1 * 1 \xrightarrow{\text{Embedding}} 100 \xrightarrow{\text{FC}_1} 3^2 * 384 \xrightarrow{\text{Deconv}_1} 7^2 * 192 \xrightarrow{\text{Deconv}_2} 14^2 * 96 \xrightarrow[\tanh]{\text{Deconv}_3} 28^2 * 1$	$28^2 * 1 \xrightarrow[stride=2]{\text{Conv}_1} 14^2 * 96 \xrightarrow[stride=2]{\text{Conv}_2} 7^2 * 192 \xrightarrow[stride=2]{\text{Conv}_3} 3^2 * 384 \xrightarrow[stride=2]{\text{Conv}_4} 1 * 1 * 1$
CIFAR-10, CIFAR-100	$100 * 1 * 1 \xrightarrow{\text{Embedding}} 100 \xrightarrow{\text{FC}_1} 4^2 * 256 \xrightarrow{\text{Deconv}_1} 8^2 * 128 \xrightarrow{\text{Deconv}_2} 16^2 * 64 \xrightarrow[\tanh]{\text{Deconv}_3} 32^2 * 3$	$32^2 * 3 \xrightarrow[stride=2]{\text{Conv}_1} 16^2 * 64 \xrightarrow[stride=2]{\text{Conv}_2} 8^2 * 128 \xrightarrow[stride=2]{\text{Conv}_3} 4^2 * 256 \xrightarrow[stride=2]{\text{Conv}_4} 1 * 1 * 1$

Notes. FC denotes the fully connection layer, Conv denotes the convolution layer, and Deconv denotes the deconvolution layer.



(a) FLGAN on MNIST dataset.

(b) FLGAN on Fashion MNIST dataset.

(c) FLGAN on USPS dataset.

(d) FLGAN on CIFAR-10 dataset.

(e) FID & IS scores of synthesized MNIST data in FLGAN.

(f) FID & IS scores of synthesized Fashion MNIST data in FLGAN.

Fig. 5. Synthesized samples in FLGAN, "Red" square means synthesized data, and "Blue" square means ground-truth data.

TABLE V
FL ARCHITECTURES

Dataset	Parameters
MNIST, USPS, Fashion MNIST	28^2*1 $\xrightarrow[stride=2]{\text{Conv}_1}$ 14^2*32 $\xrightarrow[stride=1]{\text{Conv}_2}$ 14^2*64 $\xrightarrow[stride=2]{\text{Conv}_3}$ 7^2*128 $\xrightarrow[stride=1]{\text{Conv}_4}$ 7^2*256 $\xrightarrow{\text{FC}_1}$ 12,544 $\xrightarrow[\text{Softmax}]{\text{FC}_2}$ 10
CIFAR-10, CIFAR-100	$32^2 * 3 \xrightarrow[stride=1]{\text{Conv}_1} 28^2$*6 $\xrightarrow[stride=2]{\text{Pool}_1} 14^2$*6 $\xrightarrow[stride=1]{\text{Conv}_2} 10^2$*16 $\xrightarrow[stride=2]{\text{Pool}_2} 5^2$*16 $\xrightarrow{\text{FC}_1}$ 120 $\xrightarrow[\text{ReLU}]{\text{FC}_2}$ 84 $\xrightarrow[\text{ReLU}]{\text{FC}_3}$ 1*10*1

TABLE VI
FID & IS SCORES

Datasets	FID	IS	Training Rounds
MNIST	11.8	2.1075 ± 0.0399	50
Fashion MNIST	40.0	2.4910 ± 0.0792	50
CIFAR-10	102.8	2.9925 ± 0.1125	150
USPS	245.8	1.5805 ± 0.0816	50
CIFAR-100	422.7	11.3269 ± 0.0914	500

Notes. 6,000 samples selected from the synthesized data.

the CKKS cryptosystem to achieve user-level privacy. The local GAN training phase costs 23 min for a training epoch. The privacy-preserving selection process costs 7.3 s (i.e., 0.53% of the total time) over 10 encrypted 10-dimensional label vectors $[\![Y_{i\in[1,n]}]\!]$ $(n = 10)$, which guarantees user-level privacy with

TABLE VII
QUALITATIVE COMPARISON WITH EXISTING GAN SCHEMES

Schemes	MNIST	Fashion MNIST	CIFAR-10	USPS	CIFAR-100
MD-GAN [11]	38.9	57.1	179.4	354.4	452.1
UA-GAN [12]	39.1	51.2	168.8	329.2	465.3
F2A-GAN [13]	19.9	46.4	134.7	**219.4**	445.3
FLGAN	**11.8**	**40.0**	**102.8**	245.8	**372.7**

Notes. 6,000 samples are selected from synthesized unbiased data.

TABLE VIII
ACCURACY COMPARISON BETWEEN FLGAN AND BASELINES

Baseline	Schemes	MNIST	Fashion MNIST	USPS	CIFAR-10
FedSGD	Baseline	90.49%	81.74%	72.78%	25.14%
	FLGAN	**98.78%**	**90.29%**	**84.13%**	**86.40%**
FedAvg	Baseline	90.31%	80.46%	73.15%	19.77%
	[18]	81.32%	47.54%	36.93%	16.39%
	[8]	**98.96%**	74.33%	69.73%	50.85%
	FLGAN	94.13%	**84.34%**	**84.67%**	**83.38%**

Notes. FL Training until accuracy convergence.

lightweight overhead. The average time of the GAN augmentation phase spends 5 min to generate 6,000 samples to balance local non-IID data setting.

The computation overhead of the privacy-preserving selection process relies on the size n of $\mathcal{U}$ and the dimension κ of label vectors $[\![Y_{i\in[1,n]}]\!]$. Let $\mathsf{T}_{\mathsf{FHE.Mul}}$ and $\mathsf{T}_{\mathsf{FHE.Add}}$ be the computational complexities of FHE.Mul and FHE.Add, respectively. The computational complexity of the privacy-preserving selection process is demonstrated as $\mathcal{O}(n^2 * \kappa * (\mathsf{T}_{\mathsf{FHE.Mul}} + \mathsf{T}_{\mathsf{FHE.Add}}))$, where $\mathsf{T}_{\mathsf{FHE.Mul}} \gg \mathsf{T}_{\mathsf{FHE.Add}}$. As the number n of users increases, $\mathcal{S}$ can perform secure computations in parallel, improving the efficiency of the privacy-preserving selection process.

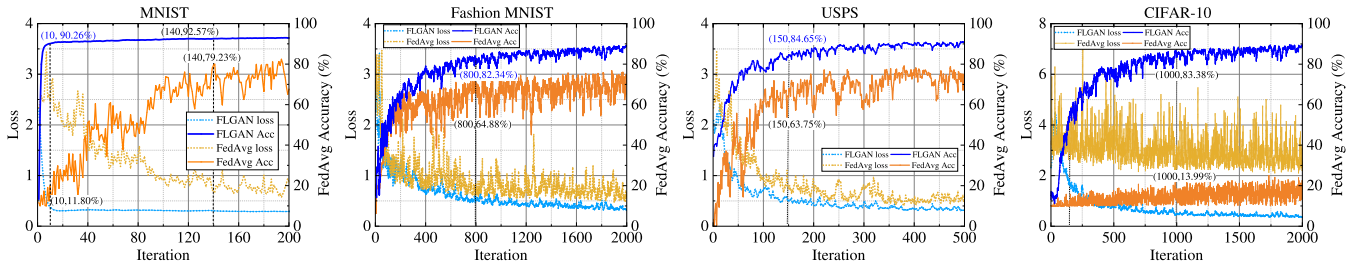
Comparative experiments. To quantitatively compare FLGAN with the proposed federated GAN schemes, we demonstrate the FID scores in Table VII under the challenging non-IID setting, i.e., each user holds samples belonging to a non-overlapping class. Since the training data is non-IID, these schemes trained with IID data [11], [12] are difficult to train a global GAN model that can accurately capture the underlying data distribution across distinct local data. Comparing the results from these schemes [11], [12], F2A-GAN [13] shows outstanding permance, because F2A-GAN can effectively train a global GAN model, even when the data distributions are non-IID. However, F2A-GAN [13] performs poorly on complex image datasets, such as CIFAR-10 and CIFAR-100, due to complex image data making it difficult for the generator and discriminator to learn the global data distribution and easily prone to overfitting. For USPS data, F2A-GAN [13] outperforms with a lower FID score. This is because the larger size of the USPS data is larger than those in MNIST, Fashion MNIST, CIFAR-10, and CIFAR-100. The USPS dataset has dimensions of 16×16 pixels compared to 28×28 pixels for MNIST and Fashion MNIST, and 32×32 pixels for CIFAR-10. This larger size may make it easier for F2A-GAN to capture fine-grained details with excellent performance. For MNIST, Fashion MNIST, CIFAR-10, and CIFAR-100, FLGAN shows the lowest FID score compared with the results from these schemes [11], [12], [13], demonstrating that FLGAN clearly outperforms existing schemes. For these schemes [11], [12], [13], the challenging non-IID setting impedes the high-quality GAN training. Conversely, FLGAN still can generate data with high quality for data augmentation.

2) Unbiased FL in FLGAN: With the GAN augmentation, the federated GAN generates IID data samples to convert local non-IID data into IID data, aiming to construct the unbiased FL in FLGAN. We use two state-of-the-art FL baselines (i.e., FedAvg and FedSGD) to testify the performance of the unbiased FL in FLGAN, which compare FLGAN with FL baselines without data augmentation in terms of test accuracy and loss. To evaluate the efficiency of FLGAN, we investigate different non-IID data on all datasets. Figs. 6 and 7 plot the test accuracy and loss of FL with 10 users and the mini-batch of 100. The training starts at the initialization of a FL task and ends with the FL convergence.

FedAvg Algorithm. Fig. 6 compares the training process of FLGAN using FedAvg with the original FedAvg algorithm. For *non-IID MNIST*, the unbiased FL in FLGAN using FedAvg achieves the test accuracy of 90.26%, while FedAvg baseline reaches the accuracy of 11.80% after 10 iterations. Besides, FedAvg baseline converges to a test accuracy of 79.23% after 140 iterations. FLGAN has significantly improved the convergence rate via reducing training iterations by up to 92% on the MNIST. As depicted in Fig. 6(a), FLGAN has greatly increased the test accuracy of 13.3% after 140 iterations. For *non-IID Fashion MNIST*, the unbiased FL in FLGAN using FedAvg reaches the test accuracy of 82.34%, while the FedAvg baseline reaches the accuracy of 64.88% after 800 iterations. As demonstrated in Fig. 6(b), FLGAN has significantly increased the test accuracy by up to 17.5% compared to the FedAvg baseline. For *non-IID USPS*, the unbiased FL in FLGAN using FedAvg implements the test accuracy of 84.65%, compared with the test accuracy 63.75% of the FedAvg baseline after 150 iterations. The unbiased FL in FLGAN has improved the accuracy by up to 20.9% compared to the FedAvg baseline, which is shown in Fig. 6(c). For *non-IID CIFAR-10*, diverse data samples increase the training difficulty of FL. Hence, FL requires more training iterations for model convergence. FLGAN using FedAvg reaches the test accuracy of 83.38% at 1,000 iterations, while the FedAvg baseline cannot achieve the model convergence with the test accuracy of 13.99%. As depicted in Fig. 6(d), FLGAN accelerates the convergence by constructing unbiased FL, which can also greatly increase the accuracy by up to 69.3% compared to the FedAvg baseline.

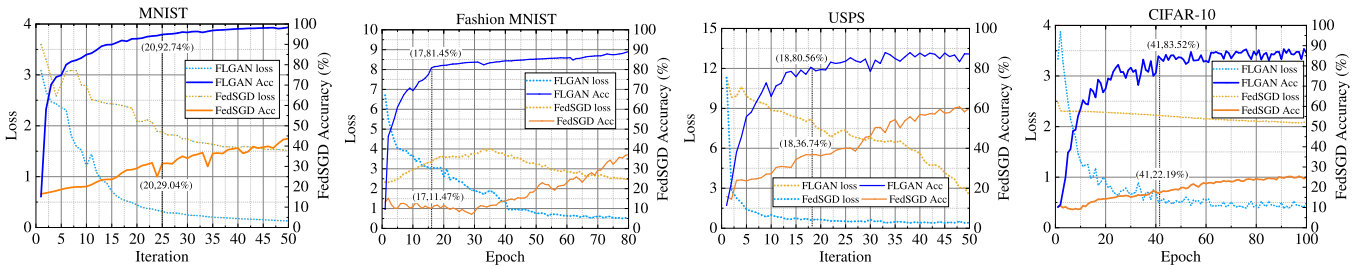
FedSGD Algorithm. Under different non-IID settings, Fig. 7 and Table VIII compare the unbiased FL in FLGAN using FedSGD with the original FedSGD baseline. Compared with the FedAvg baseline, FedSGD baseline has a poor convergence rate, tenfold times increasing the training iterations[1]. Therefore, Fig. 7 uses a training epoch instead of a training iteration. An epoch indicates 60 iterations for Fashion MNIST, and 80 iterations for USPS, 50 iterations for CIFAR-10. As MNIST involves

[1]The reason is that FedSGD is based on gradient descent, which assumes that the training data is IID. For non-IID data, FedSGD faces the problem of sample bias, which leads to the decline of model accuracy. FedAvg aggregates the model parameters of local models that can adapt to distinct data distributions and thus perform better on non-IID data.



(a) Non-IID MNIST, FedAvg (b) Non-IID Fashion MNIST, FedAvg (c) Non-IID USPS, FedAvg (d) Non-IID CIFAR-10, FedAvg

Fig. 6. Test accuracy *Acc* and loss with FEDAVG.



(a) MNIST (b) Fashion MNIST (c) USPS (d) CIFAR-10

Fig. 7. Test accuracy *Acc* and loss with FedSGD.

fewer iterations, we still use the training iterations. Fig. 7 shows a comparison of the convergence performance between the unbiased FL in FLGAN using FedSGD and the original FedSGD baseline. It can be observed that FedSGD baseline significantly decreases the convergence speed, while the FL convergence speed enhanced by FLGANaccelerates significantly. Therefore, while FedSGD baseline is still in slow model convergence, FLGAN is demonstrating high model accuracy.

- For *non-IID MNIST data*, the unbiased FL in FLGAN using FedSGD achieves the test accuracy of 92.74%, while FedSGD baseline reaches the accuracy of 29.04% after 20 training iterations. As depicted in Fig. 7(a), FLGAN has significantly improved the convergence rate and reduced the training iterations by up to 92% on the MNIST. Besides, FLGAN has increased the test accuracy by up to 63.7% after 20 iterations, compared with the FedSGD baseline.
- For *non-IID Fasion MNIST data*, Fig. 7(b) plots that the unbiased FL in FLGAN achieves a faster convergence after 17 epochs compared with the FedSGD baseline, which has significantly improved the accuracy by up to 69.9%.
- For *non-IID USPS data*, Fig. 7(c) demonstrates that FLGAN can reach the test accuracy of 80.56% with 18 training epochs. Compared with the FedSGD baseline, FLGAN has accelerated the convergence rate by greatly reducing training epochs.
- For *non-IID CIFAR-10 data*, Fig. 7(d) shows that FLGAN can achieve the test accuracy of 83.52% around 41 training epochs, which testifies the outstanding convergence rate and stable accuracy.

Table VIII compares the final accuracy among FLGAN, two FL baselines and existing non-IID FL solutions [8], [18]. With the above observations, we demonstrate that FLGAN outperforms these two FL baselines and existing non-IID FL solutions [8], [18], in the aspect of convergence rate and test accuracy. Specifically, the re-parameterized solution [18] can accelerate the training convergence, but cannot achieve the accuracy improvement with biased local updates. The existing clustered FL solution [8] has a significant accuracy improvement with MNIST data, but has poor performance with other datasets. The reason is that more skewed bias is introduced by clustering randomly selected users. With training on IID data constructed by the global augmentation, FLGAN provides the unbiased FL training under different non-IID settings, which can maintain the stable training accuracy and fast FL convergence with different FL algorithms.

VIII. DISCUSSION OF FLGAN

A. Efficiency Optimization

We emphasize that our experiments are evaluated on an Ubuntu 18.04 machine equipped with an Intel i7-9750H processor boasting six cores running at 4.50 GHz, 64 GB RAM, and NVIDIA GTX 3080 TI GPU. However, we anticipate a significant improvement in the efficiency of FLGAN by utilizing more powerful computers and GPUs for both users and the server in real-world scenarios. In FLGAN, the privacy-preserving generator selection process only costs about 0.53% of the total

overhead, while the computational overheads are mainly spent on local GNN training.

As demonstrated in Section VII-B1, the efficiency of privacy-preserving generator selection depends on the number of users n and the label size κ. With the increase of n and κ, the growth of encrypted local parameters $[\![Y_i]\!]$ ($i \in [1, n]$) results in higher communication and computational overheads for secure cosine similarity. As the number n of users increases, $\mathcal{S}$ can leverage parallel computation to improve the efficiency of secure computations. This parallelization method allows for an efficient and privacy-preserving selection process with a larger number of users. Consequently, the scalability of FLGAN with respect to the number of users n is evident, making it a highly scalable solution. Besides, the overheads arising from encrypting local generator models $[\![W_{g_i}]\!]$ in FLGAN are recognized to be a concern. We also address optimizing computational and communication efficiency. One promising strategy is to compress local GNN models in FLGAN, Such a strategy has been explored in [42], which seeks to minimize both communication and computational overheads caused by model encryption while maintaining the high performance of GAN training in FLGAN.

For GAN training, FLGAN introduces extra training overhead compared to traditional GAN training since the model training is performed across multiple decentralized devices. Beyond the scope of the current work, we still focus on the issue of GNN acceleration on GPUs. There are several existing literature [43] that propose adaptive and efficient running time systems for accelerating GNNs on GPUs, which show the GNN acceleration approach outperforms the state-of-the-art GNN computing frameworks (up to $3.02\times$ faster). By leveraging enhanced computing resources and evolving GNN architectures, we can efficiently optimize the GAN training process in FLGAN. The increased computational power allows for faster GAN convergence and improved quality of generated samples, even for large datasets and a substantial number of users within the FLGAN framework.

B. Deployment in Real-World Application Scenarios

FLGAN can be deployed to serve one federation with multiple distributed data users, where different users (e.g., mobile devices, data institutions) preserve non-identical data distributions. It poses challenges for the model-tuning component of FL. This can result in accuracy loss and poor performance. It is significant to improve the model accuracy in FL. In non-IID FL, there is a trade-off between model efficiency and accuracy, that is, improving accuracy often requires increasing model complexity and computing resources of FLGAN, which in turn reduces model efficiency. Therefore, real-world application scenarios must be considered in the acceptable trade-off. For some application tasks with high accuracy requirements, such as medical analysis and financial prediction, etc., these applications may pay more attention to the accuracy of the FL model but can tolerate some loss of model efficiency.

Here are some examples of highly sensitive application scenarios where organizations are willing to sacrifice efficiency to ensure FL accuracy. For example, in medical applications, each medical institution may have its own patient population and data collection protocols, resulting in non-IID data distributions. The accuracy loss and poor performance of non-IID FL have severe consequences, including incorrect diagnoses and treatment decisions that negatively impact patient health. In financial applications, FL can be useful for financial institutions such as banks or credit card companies. Each institution may have different customer demographics and transaction patterns, resulting in non-IID data. the consequences of FL accuracy loss can have significant impacts on financial market and investment. If FL accuracy decreases, it can lead to incorrect risk assessments, investment decisions, and trade executions, resulting in financial losses.

To mitigate these consequences, FLGAN can be deployed in these applications, which involves federated GAN training to significantly improve the FL accuracy. The local users may be medical hospitals or finical institutions to train a federated GAN model in FLGAN, which can use multiple GPUs in parallel training to speed up the local GAN training.

IX. Conclusion and Future Work

In this paper, we presented our design of FLGAN, a GAN augmentation-driven federated learning framework that provides the GAN augmentation service for non-IID FL. To construct FLGAN, we first presented a federated GAN training under non-IID settings. Then, inspired by privacy guarantee, we proposed a privacy-preserving generator selection scheme using the CKKS cryptosystem to protect user-level privacy. Extensive experiments demonstrated the outstanding performance of FLGAN. Besides, compared with two state-of-the-art FL baselines (i.e., FedAvg and FedSGD), FLGAN had accelerated the convergence rate by up to 70% on the MNIST dataset, and had improved the test accuracy by up to 10% on MNIST, Fashion MNIST and USPS datasets, and up to 60% on CIFAR-10.

As another promising research focus, our future work can leverage a gradient-based image recovery approach to tackle the computational burden involved by local GAN training. This approach relies on inverting gradients, rather than training GAN, to recover input data, thereby reducing computational overhead. Meanwhile, we will resort to hardware-assisted trusted execution environments (TEEs) to create a secure environment for performing partial computation over plaintexts.

References

[1] K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 1175–1191.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, PMLR, 2017, pp. 1273–1282.

[3] M. Song et al., "Analyzing user-level privacy attack against federated learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2430–2444, Oct. 2020.

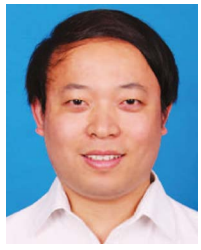
[4] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 1–21.

[5] Y. Huang et al., "Personalized cross-silo federated learning on non-iid data," in *Proc. AAAI Conf. Artif. Intell.*, AAAI Press, 2021, pp. 7865–7873.

[6] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," in *Proc. Int. Conf. Learn. Representations*, 2021.
[7] Q. Meng, W. Chen, Y. Wang, Z.-M. Ma, and T.-Y. Liu, "Convergence analysis of distributed stochastic gradient descent with shuffling," *Neurocomputing*, vol. 337, pp. 46–57, 2019.
[8] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1698–1707.
[9] M. Rasouli, T. Sun, and R. Rajagopal, "FedGAN: Federated generative adversarial networks for distributed data," 2020, *arXiv: 2006.07228*.
[10] B. Xin, W. Yang, Y. Geng, S. Chen, S. Wang, and L. Huang, "Private FL-GAN: Differential privacy synthetic data generation based on federated learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 2927–2931.
[11] C. Hardy, E. L. Merrer, and B. Sericola, "MD- GAN: Multi-discriminator generative adversarial networks for distributed datasets," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2019, pp. 866–877.
[12] Y. Zhang, H. Qu, Q. Chang, H. Liu, D. Metaxas, and C. Chen, "Training federated GANs with theoretical guarantees: A universal aggregation approach," 2021, *arXiv:2102.04655*.
[13] R. Yonetani, T. Takahashi, A. Hashimoto, and Y. Ushiku, "Decentralized learning of generative adversarial networks from non-iid data," 2019, *arXiv: 1905.09684*.
[14] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 603–618.
[15] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1310–1321.
[16] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *Proc. Int. Conf. Learn. Representations*, 2020.
[17] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu, "FedPD: A federated learning framework with optimal rates and adaptivity to non-iid data," 2020, *arXiv: 2005.11418*.
[18] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst.*, 2020, pp. 429–450.
[19] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 1–34.
[20] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 1–28.
[21] H. Wang, M. Yurochkin, Y. Sun, D. S. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–13.
[22] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*.
[23] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[24] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Springer, 2017, pp. 409–437.
[25] H. Chen, I. Chillotti, and Y. Song, "Improved bootstrapping for approximate homomorphic encryption," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, Springer, 2019, pp. 34–54.
[26] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, pp. 1–40, 2009.
[27] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Secur. Privacy*, 2019, pp. 691–706.
[28] Q. Wang et al., "Privacy-preserving collaborative model learning: The case of word vector training," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2381–2393, Dec. 2018.
[29] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Proc. Eur. Symp. Res. Comput. Secur.*, Springer, 2008, pp. 192–206.
[30] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," 2016, *arXiv:1611.01673*.
[31] X. Jiang, M. Kim, K. E. Lauter, and Y. Song, "Secure outsourced matrix computation and application to neural networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, D. Lie, M. Mannan, M. Backes, and X. Wang, Eds., 2018, pp. 1209–1222.
[32] T. Van Erven and P. Harremos, "Rényi divergence and Kullback-Leibler divergence," *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 3797–3820, Jul. 2014.
[33] B. Fuglede and F. Topsoe, "Jensen-Shannon divergence and Hilbert space embedding," in *Proc. Int. Symp. Inf. Theory*, 2004, pp. 31.
[34] "SEAL Microsoft (release 3.6)," microsoft Research, 2020. [Online]. Available: https://github.com/Microsoft/SEAL
[35] Y. LeCun, C. Cortes, and C. J. Burges, "MNIST database," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist
[36] Zalando, "Fashion mnist database," 1998. [Online]. Available: https://github.com/zalandoresearch/fashion-mnist
[37] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," Tech. Rep., 2009.
[38] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[40] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.
[41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[42] X. He, Q. Liu, and Y. Yang, "MV-GNN: Multi-view graph neural network for compression artifacts reduction," *IEEE Trans. Image Process.*, vol. 29, pp. 6829–6840, 2020.
[43] Y. Wang et al., "GNNAdvisor: An adaptive and efficient runtime system for GNN acceleration on GPUs," in *Proc USENIX Symp. Operating Syst. Des. Implementation*, 2021, pp. 515–531.

Zhuoran Ma received the PhD degree from the Department of Cyber Engineering, Xidian University, in 2022. She is currently a lecturer with the Department of Cyber Engineering, Xidian university, Xi'an, China. Her current research interests include data security, secure computation outsourcing, and privacy-preserving machine learning.

Yang Liu received the BS degree in computer science and technology from Xidian University, in 2017. He is currently working toward the master's degree with the School of Cyber Engineering, Xidian University. His research interests cover formal analysis of authentication protocols and deep learning neural network in cyber security.



Yinbin Miao (Member, IEEE) received the BE degree from the Department of Telecommunication Engineering, Jilin University, Changchun, China, in 2011, and the PhD degree from the Department of Telecommunication Engineering, Xidian University, Xi'an, China, in 2016. He is currently a lecturer with the Department of Cyber Engineering, Xidian University, Xi'an, China. His research interests include information security and applied cryptography.

Guowen Xu (Member, IEEE) received the PhD degree from the University of Electronic Science and Technology of China, in 2020. He is currently a research fellow with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. As the first author, he has published more than 15 papers in reputable venues, including ACM ACSAC, ACM ASIACCS, *IEEE Transactions on Information Forensics and Security*, and *IEEE Transactions on Dependable and Secure Computing*. His research interests include applied cryptography, and AI security and privacy issues. He was a recipient of the Best Paper Award of ICPADS 2020 and the INFOCOM 2021 Student Conference Award.

Ximeng Liu (Senior Member, IEEE) received the BSc degree in electronic engineering from Xidian University, Xi'an, China, in 2010, and the PhD degree in cryptography from Xidian University, China, in 2015. Currently, he is the full professor with the College of Mathematics and Computer Science, Fuzhou University. Also, he was a research fellow with the School of Information System, Singapore Management University, Singapore. He has published more than 200 papers on the topics of cloud security and Big Data security including papers in *IEEE Transactions on Computers*, *IEEE Transactions on Industrial Informatics*, *IEEE Transactions on Dependable and Secure Computing*, *IEEE Transactions on Services Computing*, *IEEE Internet of Things Journal*, and so on. He was awarded "Minjiang Scholars" distinguished professor, "Qishan Scholars" in Fuzhou University, and ACM SIGSAC China Rising Star Award (2018). His research interests include cloud security, applied cryptography and Big Data security. He is a member of ACM, CCF.

Jianfeng Ma (Member, IEEE) received the BS degree in mathematics from Shaanxi Normal University, Xi'an, China, in 1985, and the MS and PhD degrees in computer software and telecommunication engineering from Xidian University, Xi'an, China, in 1988 and 1995, respectively. From 1999 to 2001, he was a research fellow with the Nanyang Technological University of Singapore. He is currently a professor and a PhD supervisor with the Department of Computer Science and Technology, Xidian University, Xi'an, China. He is also the director of the Shaanxi Key Laboratory of Network and System Security. His current research interests include information and network security, wireless and mobile computing systems, and computer networks.

Robert H. Deng (Fellow, IEEE) has been a professor with the School of Information Systems, Singapore Management University, since 2004. His research interests include data security and privacy, multimedia security, and network and system security. He was an associate editor of *IEEE Transactions on Information Forensics and Security* from 2009 to 2012. He is currently an associate editor of *IEEE Transactions on Dependable and Secure Computing* and *Security and Communication Networks* (John Wiley). He is the co-chair of the Steering Committee of the ACM Symposium on Information, Computer and Communications Security. He received the University Outstanding Researcher Award from the National University of Singapore in 1999 and the Lee Kuan Yew Fellow for Research Excellence from Singapore Management University in 2006. He was named Community Service Star and Showcased Senior Information Security Professional by $(ISC)^2$ under its Asia-Pacific Information Security Leadership Achievements Program, in 2010.