IFL-GAN: Improved Federated Learning Generative Adversarial Network With Maximum Mean Discrepancy Model Aggregation

Wei Li, *Member, IEEE*, Jinlin Chen, Zhenyu Wang, Zhidong Shen, Chao Ma, *Member, IEEE*, and Xiaohui Cui

***Abstract*— The generative adversarial network (GAN) is usually built from the centralized, independent identically distributed (i.i.d.) training data to generate realistic-like instances. In real-world applications, however, the data may be distributed over multiple clients and hard to be gathered due to bandwidth, departmental coordination, or storage concerns. Although existing works, such as federated learning GAN (FL-GAN), adopt different distributed strategies to train GAN models, there are still limitations when data are distributed in a non-i.i.d. manner. These studies suffer from convergence difficulty, producing generated data with low quality. Fortunately, we found that these challenges are often due to the use of a federated averaging strategy to aggregate local GAN models' updates. In this article, we propose an alternative approach to tackling this problem, which learns a globally shared GAN model by aggregating locally trained generators' updates with *maximum mean discrepancy* (MMD). In this way, we term our approach *improved FL-GAN* (IFL-GAN). The MMD score helps each local GAN hold different weights, making the global GAN in IFL-GAN getting converged more rapidly than federated averaging. Extensive experiments on MNIST, CIFAR10, and SVHN datasets demonstrate the significant improvement of our IFL-GAN in both achieving the highest *inception score* and producing high-quality instances.**

***Index Terms*— Federated learning, generative adversarial network (GAN), maximum mean discrepancy (MMD), non-independent identically distributed (i.i.d.) training data.**

Manuscript received 2 January 2020; revised 30 November 2020 and 2 January 2022; accepted 10 April 2022. Date of publication 26 April 2022; date of current version 1 December 2023. This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant JUSRP121073 and Grant JUSRP521004, in part by the 2021 Jiangsu Shuangchuang (Mass Innovation and Entrepreneurship) Talent Program under Grant JSSCBS20210827, and in part by the Open Foundation of Engineering Research Center of Cyberspace under Grant KJAQ202112014. *(Corresponding authors: Jinlin Chen; Xiaohui Cui.)*

Wei Li is with the Science Center for Future Foods, the School of Artificial Intelligence and Computer Science, and the Jiangsu Key Laboratory of Media Design and Software Technology, Jiangnan University, Wuxi, Jiangsu 214122, China (e-mail: cs_weili@jiangnan.edu.cn).

Jinlin Chen is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: csjlchen@comp.polyu.edu.hk).

Zhenyu Wang is with the School of Computer Science, Wuhan University, Wuhan 430072, China, and also with the Jiaxing Institute of Future Food, Jiaxing, Zhejiang 314005, China (e-mail: zhenyuwang@whu.edu.cn).

Zhidong Shen, Chao Ma, and Xiaohui Cui are with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China (e-mail: shenzd@whu.edu.cn; whmachao@ieee.org; xcui@whu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3167482.

Digital Object Identifier 10.1109/TNNLS.2022.3167482

I. Introduction

THE generative adversarial network (GAN) [6] has been demonstrated as a powerful generative model that casts generative modeling as a game between two networks: a discriminator D and a generator G. The discriminator D estimates a probability that a sample came from the training data rather than the generator G. In the training of GAN, G is viewed as a forger, which specializes in generating simulation data to fool D into accepting it as real. The GAN model sidesteps the difficulty of approximating many intractable probabilistic computations; it samples data from an easy-to-sample distribution, so the Markov chains [25] are never needed. Its gradients are tuned using backpropagation, which makes the training computationally inexpensive. Although GAN is successfully applied to many real-world applications [16]–[18], [28], training a GAN is still a challenge because training GAN with high capacity is usually built from the centralized, independent identically distributed (i.i.d.) training data. Such a scenario is not often the case in practice. For example, different departments in the same domain (e.g., The State Food and Drug Bureau and The Animal Husbandry and Veterinary Bureau) collect data that are of concern to their department. However, these data may be non-i.i.d. and hard to be gathered due to departmental coordination and bandwidth concerns such that the collected data are distributed over clients in a non-i.i.d. manner, rendering it difficult to train GAN under such a scenario. A non-i.i.d. example is shown in Fig. 1.

GAN variants, such as MD-GAN [9] and federated learning GAN (FL-GAN) [9], have been proposed to address this problem. FL-GAN trains several local GAN models with the federated averaging algorithm to learn a shared model by aggregating locally computed updates. By employing multiple GAN models and deploying each GAN (GAN_i) to Client_i, however, they still have limitations. MD-GAN (one single generator and multiple discriminators) is trained over multiple datacenters through the wide-area network (WAN) [11]. However, such a training strategy is very expensive and complicated. Moreover, the server usually generates k batches at each global iteration. All the N clients receive and compute the feedback on the same training batch if $k = 1$, and no feedback has a conflict on some concurrently processed data if $k = N$.

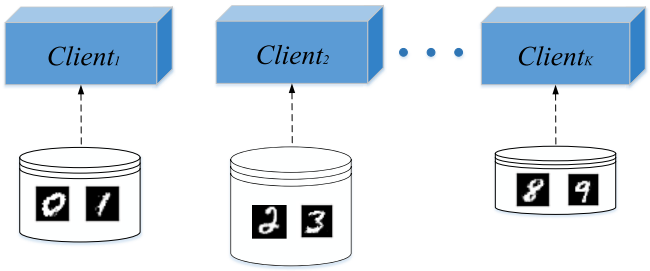


Fig. 1. Non-i.i.d. data are distributed over different clients. Each client Client_i holds few categories, while the quantity of data stored in Client_i may be different.

Thus, MD-GAN usually reduces the server workload by sacrificing the diversity of generated instances. FL-GAN, on the other hand, applies a federated learning strategy [21] to GAN models and trains all models with federated averaging. The federated averaging strategy produces promising results on i.i.d. training data [23]. However, it degrades the performance of models in the non-i.i.d. case [20], which produces generated data with low quality and suffers from a longer training time to get converged. A detailed discussion is shown in Section V.

To address this challenge, we propose *improved FL-GAN* (IFL-GAN), which learns a globally shared GAN model (global GAN) by aggregating locally trained generators' updates with *maximum mean discrepancy* (MMD) [8]. In scenarios where the federated averaging strategy is employed for learning a global GAN, all locally trained generators are treated as contributors with exactly the same weights, which is not always suitable in many real-world applications. For instance, we have K local GAN models denoted as $\text{GAN}_1 \sim \text{GAN}_K$ and $\text{GAN}_1 \sim \text{GAN}_i$, which reached the Nash equilibrium much earlier than the rest (i.e., $\text{GAN}_{i+1} \sim \text{GAN}_K$) mainly due to the non-i.i.d. property of the distributed training data. In the federated averaging strategy, it is obvious that this will cause a much longer time for the global GAN to get converged. This is because local models' $(\text{GAN}_1 \sim \text{GAN}_K)$ updates are triggered by global GAN, i.e., $\text{GAN}(x; \theta_{\text{GAN}_i}) = \text{GAN}(x; \theta_{\text{GAN}_{\text{global}}})$ after updating global GAN parameters with $\text{GAN}(x; \theta_{\text{GAN}_{\text{global}}}) = (1/K) \sum_1^K \text{GAN}(x; \theta_{\text{GAN}_i})$. It causes local GAN models that have reached the Nash equilibrium, jumping out of the equilibrium. Fortunately, this misery could be significantly alleviated by replacing the averaging strategy with MMD because it tends to assign larger weights to local GAN models, which are still far from convergence. In this manner, MMD makes all locally trained GAN models get converged more rapidly, which will greatly reduce the training time of the global GAN. In recent works of GAN research, MMD is introduced to calculate the supremum of the difference between source samples and target samples (a.k.a. the two-sample test). It is theoretically proved and empirically evaluated that the MMD score effectively reflects the performance of GAN models [15], [31]. A detailed discussion about MMD and the federated averaging method is presented in Section IV.

In summary, the major innovations and contributions of this study are described as follows.

1) This article proposes a novel approach, named IFL-GAN, to train GAN with MMD from decentralized, non-i.i.d. training data.
2) This article compares the federated averaging strategy with MMD from both theoretical and empirical perspectives, giving new insights into the success of IFL-GAN.
3) Through comprehensive experiments on three datasets with different resolutions, we demonstrate the effectiveness of IFL-GAN.

The rest part of this article is organized as follows. In Section II, existing works are discussed. The GAN model and federated learning are reviewed in Section III. Our IFL-GAN is introduced in detail in Section IV. In Section V, empirical evaluation is conducted. Section VI serves as our conclusion.

II. Related Work

Federatively training the neural networks is a thriving direction and attracts many researchers focusing on this topic. Here, we categorize the research works of federated learning according to the research that focuses on different aspects.

A. Improving Communication Efficiency Among Models

Since the data are distributed over a set of clients and the networks that these clients held need to communicate and exchange information, researchers focus on reducing the network latency and improving communication efficiency. Smith *et al.* [27] showed that multitask learning is naturally suited to handle the statistical challenge and proposed a new system-aware optimization approach, MOCHA. This is because unstable communication might cause devices to get offline, which makes federated learning more challenging. The goal of this approach is to achieve significant speedups for operating federated learning on separate devices. Bonawitz *et al.* [3] paid their attention to the device availability and unreliable device connectivity, as well as interrupted execution. They had built a scalable production system for federated learning on tens of millions of real-world devices. Although these studies have improved the communication efficiency among models, they pay less attention to improve the performance of models.

B. Integrating Federated Learning With Other Schemes

Some researchers try to integrate the federated learning approach (e.g., federated learning [19], [34]) with other deep learning models and apply the integrated models to the real-world applications. Zhuo *et al.* [34] integrated the reinforcement learning with federated learning. This is because different departments' decision policies are private and hard to be shared with each other. On the other hand, building individual decision policies with high quality is a nontrivial task. Therefore, it is necessary to federatively learn decision policies. Liu *et al.* [19] applied federated learning to robots' communication and made robots fuse and transfer their experience so that robots can quickly adapt to the new environment. In this way, they termed their new approach

lifelong federated reinforcement learning (LFRL). LFRL is consistent with human cognitive science and fits well in cloud robotic systems. Different from those federated learning approaches, the following studies explore the integration of federated learning with GAN.

MD-GAN [9] is the recent approach that trains GAN models in a federated fashion. MD-GAN employs multiple discriminators and only one generator to reduce computing costs. Note that the generator is on the server, while these discriminators are on the local clients. The generator produces simulation data and sends them to each local client simultaneously. The discriminator is still used to distinguish the simulation data from real data. Note that each discriminator has $(1/(n-1))$ probability to be exchanged with another discriminator, which is randomly selected from $n-1$ discriminators, given that there are n discriminators. The exchanging approach adopts the gossip algorithm [10]. Although swapping the parameters between two discriminators can avoid overfitting problems, swapped models need to be retrained. We still take Fig. 1 as the example. Discriminator 1 has learned the knowledge of figures "0" and "1," and the knowledge of figures "2" and "3" has been learned by discriminator 2. If we swap the parameters of the two discriminators, both of them need to be retrained to understand which figures are "0" or "2," resulting in more training costs. In this way, MD-GAN needs to determine a tradeoff between training complexity and data diversity, reducing the diversity of generated data.

FL-GAN [9] aims to train a set of GAN models with the federated averaging method. Specifically, each client holds a vanilla GAN model, and a shared model is learned by aggregating locally computed updates with iterative federated averaging. The federated averaging method shows promising results when facing i.i.d. training data. However, it degrades the performance of models in the non-i.i.d. case [20].

III. Preliminaries

A. Generative Adversarial Network

The GAN was proposed by Goodfellow *et al.* [6] as a novel generative model to simultaneously train a generator G and a discriminator D using the following function:

$$\min_G \max_D \; V(G,D) = \mathbb{E}_{x\sim p_r(x)}\big[\log D(x)\big] + \mathbb{E}_{z\sim p_z(z)}\big[\log(1-D(G(z)))\big] \quad (1)$$

where x comes from a distribution $p_r(x)$ underlying the original dataset and z comes from a predefined noise distribution $p_z(z)$, which is usually an easy-to-sample distribution, e.g., uniform distribution with (-1, 1) or Gaussian distribution with (0, 1). G starts with sampling input z from $p_z(z)$ and then maps z to data space $G(z;\theta_G)$ through a differentiable network. On the other hand, D aims to recognize whether an instance is from training data or from G. In general, D strives to minimize the score it assigns to the generated data $G(z)$ by minimizing $D(G(z))$ and maximize the score it assigns to the original data x by maximizing $D(x)$. In this way, D and G are alternatively optimized, and the Jensen–Shannon (JS) divergence is utilized to measure the difference between $p_r(x)$ and $p_G(x)$. JS divergence reaches its lowest value as D and G reach the Nash equilibrium [4], where $D(G(z)) = D(x) = 0.5$. The GAN model gets converged under such a scenario.

B. Federated Learning

Both the centralized and decentralized architectures usually assume the balanced and i.i.d. training data [20]. Federated learning [21] evolves around a scenario where the training data are stored locally in multiple clients (e.g., mobile devices). Hence, each particular local dataset could not be representative of the overall distribution. Usually, the machine learning models the clients held are neural networks (e.g., the convolutional neural network). Thus, the algorithm adopted by the federated learning is applicable to any finite-sum objective, which is shown as follows:

$$\min_{\omega\in\mathcal{R}^d} f(\omega) \quad \text{where} \quad f(\omega) \stackrel{def}{=} \frac{1}{n}\sum_{i=1}^{n} f_i(\omega) \quad (2)$$

where $f_i(\omega) = \ell(x_i, y_i, \omega)$ indicates the loss of the prediction on the sample (x_i, y_i) with parameters ω. Assuming that there are K clients and the data samples are partitioned into K subsets, with $\mathcal{P}_k$ $(n_k = |\mathcal{P}_k|)$, the set of indexes of data samples in client k. Thus, (2) can be rewritten as follows:

$$f(\omega) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(\omega) \quad \text{where} \quad F_k(\omega) = \frac{1}{n_k}\sum_{i\in\mathcal{P}_k} f_i(\omega) \quad (3)$$

where $\mathcal{P}_k$ refers to a partition. Since F_k could be an arbitrarily bad approximation to f, the setting of federated learning can be extrapolated to non-i.i.d. However, Nilsson *et al.* [23] showed that the federated averaging achieves the best performance for federated learning and is practically equivalent to the centralized architecture only when training data are i.i.d. In the non-i.i.d. case, the centralized approach performs better than the federated averaging [20].

C. Federated Learning GAN

FL-GAN focuses on integrating federated learning with GAN models [9]. We still take Fig. 1 as the example. Assume that there are K clients, and each client holds a regular GAN. Each GAN builds a mapping function that can map a random noise z from randomized space $\mathcal{Z}$ into the data space χ_i, $\sum_{i=1}^{K}\chi_i = \mathcal{X}$, given that χ_i is not representative of the overall data space $\mathcal{X}$ but a local representative. Therefore, the data distribution, $p_{r_i}(x)$, within a client is a part of overall distribution $p_r(x)$, and $\sum_{i=1}^{K} p_{r_i}(x) = p_r(x)$. We define a prior distribution on $\mathcal{Z}$ as $p_z(z)$ in each client and sample noise from $p_z(z)$. In this way, the objective function of FL-GAN can be defined as follows:

$$\min_G \max_D \; V(G,D) = \frac{1}{K}\sum_{i=1}^{K}\big(E_{x\sim p_{r_i}(x)}\log D_i(x) + E_{z\sim p_z(z)}\big[\log(1-D_i(G_i(z)))\big]\big) \quad (4)$$

where $p_z(z)$ follows an easy-to-sample distribution [e.g., the Gaussian distribution (0, 1)], and i indicates the ith GAN

in the ith client. From (4), we can see that the FL-GAN adopts model averaging to aggregate local GAN models' updates to calculate the parameters of the global GAN, i.e., $G_{\text{glb}} = (1/K)\sum_{i=1}^{K} G(x;\theta_{g_i})$. After that, the parameters of each local GAN would be replaced by the parameters of global GAN, and this is formulated by $G(x;\theta_{g_i}) = G_{\text{glb}}, i \in [1, K]$. This process would be repeated until all local GAN models reach the equilibrium. However, such an aggregation may cause convergence difficulty or longer time to get converged, resulting in simulation data with low diversity [9]. This is because federated averaging method implies that all local GAN models either simultaneously reach equilibrium or are far away from the equilibrium. Even though there is a specific local GAN (GAN_i) reaching the equilibrium, it would be replaced by G_{glb} in next iteration, rendering GAN_i jumping out of its own optimal status. In Section IV, we propose the IFL-GAN and demonstrate how to address this issue with IFL-GAN.

IV. Improved Federated Learning GAN

In this section, we propose IFL-GAN by aggregating GAN models with MMD. Specifically, we discuss two important issues in our new approach: 1) how to train IFL-GAN on decentralized and non-i.i.d. data with MMD and 2) comparing MMD with the federated averaging method.

A. Training IFL-GAN With MMD

For each client, it still holds one single GAN model, and the generator within each GAN model builds a mapping function that maps noise code z from randomized space $\mathcal{Z}$ into data space χ_i. χ_i is a subspace of data space $\mathcal{X}$. However, the weight of each GAN in IFL-GAN is derived from the MMD score rather than the weighted average score. In this way, the objective function of IFL-GAN can be defined as follows:

$$\begin{aligned}
&\min_G \max_D \; V(G,D) \\
&\quad = \alpha_1\big(E_{x\sim p_{r_1}(x)}\log D_1(x) \\
&\qquad\quad + E_{z\sim p_z(z)}\big[\log(1-D_1(G_1(z)))\big]\big) \\
&\qquad + \alpha_2\big(E_{x\sim p_{r_2}(x)}\log D_2(x) \\
&\qquad\quad + E_{z\sim p_z(z)}\big[\log(1-D_2(G_2(z)))\big]\big) + \cdots \\
&\qquad + \alpha_K\big(E_{x\sim p_{r_K}(x)}\log D_K(x) \\
&\qquad\quad + E_{z\sim p_z(z)}\big[\log(1-D_K(G_K(z)))\big]\big) \\
&\text{s.t.} \sum_{i=1}^{K}\alpha_i = 1
\end{aligned} \tag{5}$$

where α_i $(i = 1, 2, \ldots, K)$ indicates the weight of the ith GAN model and its value is derived from the MMD score [see (6)]. In (6), f usually refers to the Gaussian kernel function that maps data into the reproducing kernel Hilbert space (RKHS). x refers to the training data with minibatch size, while $G(z)$ indicates the generated data with minibatch size. The MMD score is determined by calculating the supremum of expectations [i.e., $E(*)$] of $f(x)$ and that of $f(G(z))$

$$\begin{aligned}
&\text{mmd}_i = \sup_{||f||\leqslant 1} ||E[f(x)] - E[f(G(z))]||^2 \\
&\text{s.t. } \alpha_i = \frac{e^{\text{mmd}_i}}{\sum_{j=1}^{K} e^{\text{mmd}_j}}.
\end{aligned} \tag{6}$$

Since the MMD score (mmd_i) may be larger than 1, we utilize the *Softmax* function to normalize MMD scores, obtaining normalized MMD score α_i, s.t. $\sum_{i=1}^{K}\alpha_i = 1$. Similar to vanilla GAN, each discriminator D_i of IFL-GAN pursues maximizing the probability of assigning the correct label to both training and generated samples, and each generator G_i tries to fool D_i into accepting its outputs as real data by maximizing $D_i(G_i(z))$. Here, we assume that each discriminator D_i and generator G_i have enough capacity. Thus, for each generator G_i fixed, each optimal discriminator D_i^* is shown as follows:

$$D_i^*(x) = \frac{p_{r_i}}{p_{r_i} + p_{G_i}}. \tag{7}$$

For $p_{r_i} = p_{G_i}$, $D_i^*(x) = (1/2)$. In this way, each optimal generator G_i is shown as follows:

$$\begin{aligned}
V(G_i) = -2\log 2 + KL\left(p_{r_i}\middle|\frac{p_{r_i}+p_{G_i}}{2}\right) \\
+ KL\left(p_{G_i}\middle|\frac{p_{r_i}+p_{G_i}}{2}\right).
\end{aligned} \tag{8}$$

The proofs for optimal discriminator and generator are referred to in the vanilla GAN study [6]. When both discriminator and generator in a specific client reach the optimal status (i.e., $p_{r_i} = p_{G_i}$), the local generator G_i within the ith client has successfully learned the knowledge of data stored in this client. Our goal is that all local generators have to learn the knowledge of data stored in different clients, rendering global generators producing all modes of data. Accordingly, the formula of global generator G_{glb} is shown as follows:

$$G_{\text{glb}}\big(x;\theta_{G_{\text{glb}}}\big) = \sum_{i=1}^{n}\alpha_i G_i\big(x;\theta_{G_i}\big) \tag{9}$$

where θ_{G_i} indicates the ith generator's parameters and $\theta_{G_{\text{glb}}}$ indicates the parameters of global generator.

After aggregating the parameters of $G_{1:K}$, the global generator, G_{glb}, has learned the knowledge of all data samples, and the global minimum of the virtual training criterion $V(G_{\text{glb}})$ is achieved if and only if $p_{G_{\text{glb}}} = p_r$, given $p_r = \sum_{i=1}^{K} p_{G_i}$ and $\sum_{i=1}^{K} p_{G_i} = p_{G_{\text{glb}}}$. At that point, $V(G_{\text{glb}})$ achieves the optimal value $-2\log 2$. After that, the parameters of each generator, $G_i(x;\theta_{G_i})$, is replaced by $G_{\text{glb}}(x;\theta_{G_{\text{glb}}})$, and the formula is shown in the following equation:

$$G_i\big(x;\theta_{G_i}\big) = G_{\text{glb}}\big(x;\theta_{G_{\text{glb}}}\big), \quad i = 1, 2, \ldots, K. \tag{10}$$

Note that our IFL-GAN is also better than FL-GAN on imbalanced setting of distributed data. All local GAN models are treated as exactly the same contributors in federated learning. This strategy reduces the impact on global generator for such a GAN model (e.g., GAN_1), which is trained on imbalanced data (the detailed explanation could refer to Section V), given $G_{\text{glb}} = (1/K)\sum_{i=1}^{K} G(x;\theta_{g_i})$ and $G(x;\theta_{g_i}) = G_{\text{glb}}$, $i \in [1, K]$. If GAN is trained on imbalanced data, generator needs longer time to learn data distribution. Hence, the MMD score would be enlarged for this case. According to (9) and (10), IFL-GAN can preserve the parameters of GAN_1 to larger extent, reducing training time.

There is one more thing worth to be noted. Each local GAN may reach or approach the Nash equilibrium at different

Algorithm 1 IFL-GAN

Input: Original dataset, noise z, $p_z(z)$.
Output: Simulation data.
K GAN models and each one holds the same hyperparameters and the same function, i.e., *Adam* [14] optimizer with learning
rate 0.0002 and *Binary Cross Entropy* loss function. The index is indicated by i.
for *each epoch t = 1, 2,...* **do**
  Clients execute:
  for *each client i ∈ [1, K]* **do**
    Updating the discriminator's (D_i) parameters by ascending its stochastic gradient;
    $\nabla_{\theta_D} \frac{1}{m} \sum_1^m \{\log D_i(x^{(j)}) + \log(1 - D_i(G_i(z^{(j)})))\}$;
    Updating the local generator's (G_i) parameters by descending its stochastic gradient;
    $\nabla_{\theta_G} \frac{1}{m} \sum_1^m \{\log(1 - D_i(G_i(z^{(j)})))\}$, m indicates noise samples $z^1, \ldots, z^m$;
  end
  Calculating and normalizing MMD score mmd_i produced by x_i and $G_i(z)$ with Eq.(6);
  Applying *Softmax* function to all MMD scores to obtain α_i and setting the threshold c_{a_i};
  Server executes:
    $G_{glb}(x; \theta_{G_{glb}}) = \sum_{i=1}^K \alpha_i G_i(x; \theta_{G_i})$;
  Clients update:
  for *each client i ∈ [1, K]* **do**
    if $mmd_i > c_{a_i}$ **then**
      $G_i(x; \theta_{G_i}) = G_{glb}(x; \theta_{G_{glb}})$;
    end
    else
      continue;
    end
  end
end

epochs due to model initialization and non-i.i.d. training data, directly replacing these local GAN models with G_{glb} results in them jumping into the nonoptimal status from the Nash equilibrium [29]. In our proposed IFL-GAN, the MMD score would be regarded as an indicator. In general, we utilize finite samples from two distributions to measure the MMD score [15]. In our work, we assume that the distributions refer to generated data distribution and original data distribution, and finite samples are sampled from both distributions. Because of sampling variance, the MMD score may not be zero even though GAN has reached the Nash equilibrium [15]. Since the MMD estimator implicitly involves a threshold, c_α, for distinguishing distributions by finite samples, we usually conduct a null hypothesis $\mathbb{H}_0 : \mathbb{P} = \mathbb{Q}$ [8]. If the MMD score is greater than c_α, we should reject such a hypothesis; otherwise, we should accept this hypothesis. In this study, we set the threshold with the minimal MMD score. This is because the smaller the MMD score is, the better matching the two distributions are. A more detailed discussion is shown in Section V.

The pseudocode of IFL-GAN is formally presented in Algorithm 1. The generator and discriminator architectures of each GAN model in Algorithm 1 are based on regular DCGAN [1]. The details of components conform to Conv-BatchNorm-ReLu [24] (generator G) or Conv-BatchNorm-LeakyReLu [30] (discriminator D).

In Algorithm 1, each client holds a specific GAN and uses subscript i to indicate it, i.e., G_i and $i \in [1, K]$. For each client, we train the local GAN model and send its parameters (θ_{G_i}) and the corresponding softmax MMD score α_i to the server to learn a global generator (G_{glb}) by aggregating local GAN models and MMD scores. After that, the generator of each local GAN is replaced with G_{glb} if GAN_i does not reach the Nash equilibrium. This process is repeated until all GAN models reach the Nash equilibrium ($p_{G_1} = p_{r_1}$, $p_{G_2} = p_{r_2}, \ldots, p_{G_K} = p_{r_K}$). As a consequence, the global generator has learned the knowledge of all data samples, and the simulation data generated by global generator hold all features of decentralized data according to (9). Fig. 2 shows the architecture of our proposed IFL-GAN.

B. Avg Versus MMD

Since the generator in GAN specializes in producing simulation data, we compare the federated averaging method with MMD by targeting the generator. In the MMD, we assume that the generated instances, $G(z)$, are defined via sampling from the generated data distribution $\mathbb{P}$; the original samples, x,

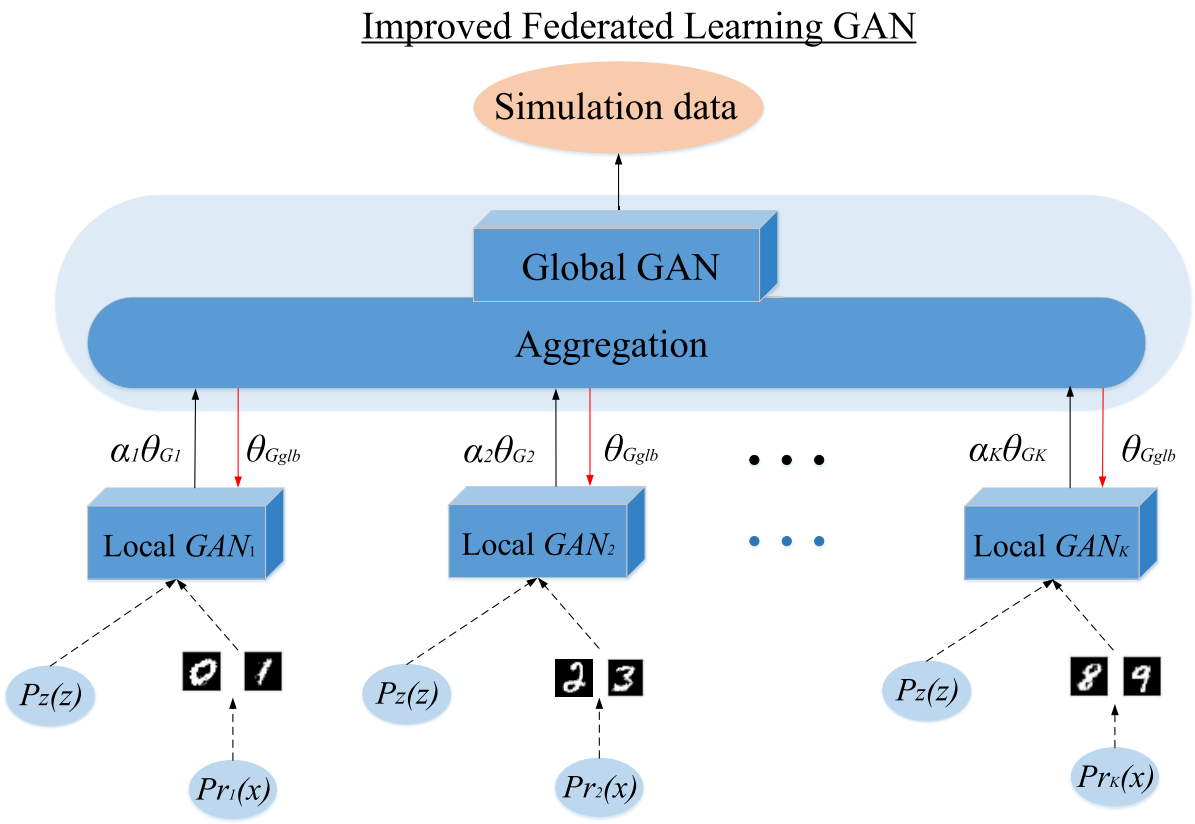


Fig. 2. Dotted arrows indicate sampling from a specific distribution. θ_{G_1} indicates the parameters of the first generator, which is in the first client. The parameters of all local generators $(\theta_{G_1}, \theta_{G_1}, \ldots, \theta_{G_K})$ and their MMD scores normalized by *Softmax* function $(\alpha_1, \alpha_2, \ldots, \alpha_K)$ are uploaded to the global generator to calculate its parameters with aggregation [see (9)]. After that, each local GAN (G_1 to G_K) is replaced with aggregated GAN ($\theta_{G_{\text{glb}}}$), which is indicated by the red line.

are defined via sampling from the original data distribution $\mathbb{Q}$. Those samples are mapped into the RKHS with the Gaussian Kernel function. After that, we utilize the Euclidean distance to reexpress (6) in RKHS. Hereby, the estimator of MMD $(\mathbb{P}, \mathbb{Q})$ is defined as follows:

$$\text{MMD}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{\mathbb{P}}(x, x') - 2\mathbb{E}_{\mathbb{P},\mathbb{Q}}(x, G(z)) + \mathbb{E}_{\mathbb{Q}}(G(z), G(z)'). \tag{11}$$

The detailed deduction of the MMD estimator is shown in study [8]. Note that the MMD estimator implicitly involves a threshold for distinguishing distributions by finite samples. In other words, we usually conduct a hypothesis test with a null hypothesis: $\mathbb{H}_0 : \mathbb{P} = \mathbb{Q}$. In general, judging whether this equation is to be held or not relies on comparing the test statistic MMD$[x, G(z)]$ with a particular threshold c_a. If c_a is exceeded, then the test rejects the hypothesis; otherwise, the test should accept the hypothesis.

Since the data are distributed over different clients in a non-i.i.d. manner, it is a nontrivial task to know the data details, such as sample quantity and category that a client holds. This causes it to be hard to perceive the learning status of each local model, e.g., whether a local GAN model reaches the Nash equilibrium or not. The traditional federated averaging method cannot loyally reflect such different status because it views all local models with the same contribution. In opposite, the MMD score can loyally reflect such a difference, i.e., the smaller MMD score indicates a better status, while the larger MMD score refers to a worse status [8], given that the MMD score reflects the matching degree of two distributions. Hence, the MMD is more suitable than the averaging method for aggregating locally computed updates on federated learning. Next, we would like to discuss the supreme advantages of MMD according to the learning status. For convenient demonstration, here, we assume that $K = 2$: two local GAN models: GAN_1 and GAN_2, and one global GAN model: GAN_{glb}

$$\begin{cases} G_{\text{glb}_{\text{avg}}} = \dfrac{1}{2} G_1(z; \theta_{G1}) + \dfrac{1}{2} G_2(z; \theta_{G_2}) \\ G_{\text{glb}_{\text{mmd}}} = \alpha_1 \times G_1(z; \theta_{G1}) + \alpha_2 \times G_2(z; \theta_{G_2}) \end{cases} \tag{12}$$

where θ_{Gi} indicates the parameters of generator in the ith GAN. $G_{\text{glb}_{\text{avg}}}$ indicates that the parameters of global generator are calculated by federated averaging method, while our approach adopts MMD to aggregate the parameters of local generators, and here, we term $G_{\text{glb}_{\text{mmd}}}$. Given that the training data are distributed over different clients in the non-i.i.d. manner, we could observe $0 < \alpha_1 \neq \alpha_2 < 1$ after training some epochs, and they refer to MMD scores. From (12), an intuitive scenario could be observed that $G_{\text{glb}_{\text{mmd}}}$ within (12) becomes a weighted equation, and $G_{\text{glb}_{\text{avg}}}$ is an absolute arithmetic averaging equation. Compared with the averaging method, the benefits of the weighted method include: 1) averaging the averages breaks the fundamental rules of math [13] such

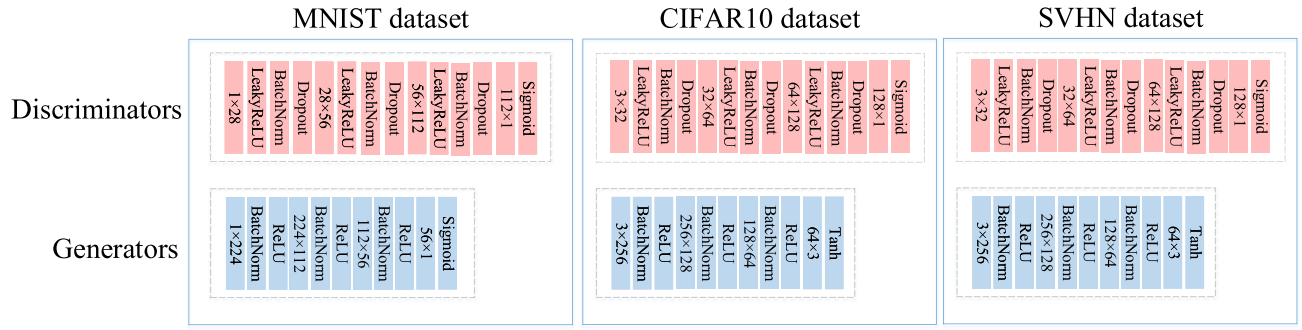


Fig. 3. Architectural details of IFL-GAN for MNIST, CIFAR10, and SVHN datasets.

that some studies insist that averaging method is derived by the "wrong math" but believe that the weighted method is based upon the "correct math" and should be applied accordingly [13] and 2) the weighted method allows the final number to quantitatively reflect the relative importance of each local model that is being averaged in federated averaging [32]. However, the federated averaging strategy only works well when all local models are equally important [33]. Yet, this is not often the case in practice, especially for that the data are distributed over clients in the non-i.i.d. manner.

In general, we view the process of training a model as the process of optimizing the model's parameters [5]. With the training increasing, the updated parameters guide the model toward the optimal status with backpropagation. Here, we assume that G_1 is more optimal than G_2, which means that both the distribution of synthetic data produced by G_1 and that of original data hold a smaller MMD score than G_2. Hence, we have $0 < \alpha_1 < \alpha_2 < 1$. With this in place, to make the global model converge rapidly, an intuitive common sense is that the parameters of G_1 need smaller updating, while the parameters of G_2 require larger updating. However, the averaging method views them with the same contribution, which may easily suffer from the problem that G_1 has skipped the optimal status when G_2 is far away from this goal. Hence, $G_{\text{glb}_{\text{mmd}}}$ gets converged more rapidly than $G_{\text{glb}_{\text{avg}}}$. We now state this properly in the following theorem.

Theorem: We assume that $\mathcal{M}_1$ denotes a model that adopts the averaging method to aggregate locally computed updates; $\mathcal{M}_2$ denotes a model that adopts the MMD method to learn a shared model by aggregating locally computed updates. For each $\mathcal{M}_i$, we set $K = 2$ (i.e., two local GAN models GAN_1 and GAN_2 and a global GAN model GAN_{glb}). Moreover, we assume that α_i represents the performance (the matching degree between the generated data distribution and the original data distribution) of G_i, given that the generator outputs realistic-like data. By optimizing the parameters of the model, the matching degree between the two distributions tends to be enhanced. Also, we utilize the same components and parameters to initialize both $\mathcal{M}_1$ and $\mathcal{M}_2$ and employ the same hyperparameters to train the two models. In addition, the training data are distributed over different clients in the non-i.i.d. manner, which means that $\alpha_1 \neq \alpha_2$. If we suppose that $\alpha_1 < \alpha_2$, then $\mathcal{M}_2$ would get converged more rapidly than $\mathcal{M}_1$.

Proof: Let the current performance of G_2 be α_2, and α_2' represents the performance of G_2 in $\mathcal{M}_1$; α_2'' indicates that of G_2 in $\mathcal{M}_2$ after performing (13). Since $\alpha_1 < \alpha_2$ and $\sum \alpha_i = 1$ [see (5)], we have $\alpha_2 >$ const (i.e., 0.5). In this way, $G_{\text{glb}_{\text{avg}}}$ certainly reduces the performance of G_2 because the averaging method enforces the weight of G_2 (i.e., α_2) to be small. Since the parameters of G_2 are kept to the greatest extent in $G_{\text{glb}_{\text{mmd}}}$ [see (12)], we obtain $\alpha_2'' > \alpha_2'$ according to (13). Hence, G_2 in $\mathcal{M}_2$ gets larger updating than that in $\mathcal{M}_1$. As to G_1, we naturally hold $\alpha_1 <$ const. Let α_1' represent the performance of G_1 in $\mathcal{M}_1$, and α_1'' indicates that of G_1 in $\mathcal{M}_2$ after performing (13); we get $\alpha_1'' < \alpha_1'$. In this way, G_1 in $\mathcal{M}_2$ gets smaller updating than that in $\mathcal{M}_1$. Since the updating strategy in $\mathcal{M}_2$ is more practical than $\mathcal{M}_1$, $\mathcal{M}_2$ gets converged more rapidly than $\mathcal{M}_1$. As illustrated above, this theorem could be easily extended to the cases that $K > 2$. In addition, the scenario of $\alpha_1 > \alpha_2$ is similar to $\alpha_1 < \alpha_2$. Next, we would like to discuss the specific implementations of $\mathcal{M}_1$ and $\mathcal{M}_2$ through FL-GAN and our proposed IFL-GAN

$$\begin{cases} G_1(z;\theta_{G_1}) = G_2(z;\theta_{G_2}) = G_{\text{glb}_{\text{avg}}} \\ G_1(z;\theta_{G_1}) = G_2(z;\theta_{G_2}) = G_{\text{glb}_{\text{mmd}}}. \end{cases} \tag{13}$$

As the training process of the two models proceeds, a scenario would have arrived where GAN_1 reaches or nears the Nash equilibrium, while GAN_2 is far away from this status. According to the **Clients update** of Algorithm 1, the local model GAN_i gets updated only when its MMD score is larger than the threshold c_{a_i}; otherwise, the local model GAN_i refuses the updating if its MMD score is equivalent to or less than c_{a_i}. In this way, the parameters of GAN_1 would not be changed, and only GAN_2 gets updated in IFL-GAN. Unfortunately, the operation of $G_1(z;\theta_{G_1}) = G_{\text{glb}_{\text{avg}}}$ changes the parameters of GAN_1 in FL-GAN, resulting in that GAN_1 jumps out of the Nash equilibrium. Therefore, our proposed approach enables the GAN model to get converged faster than averaging method.

V. Experiments

For the experiments, the implementation details of our proposed IFL-GAN are shown in Fig. 3. Note that each generator and discriminator in IFL-GAN are initialized by the same hyperparameters (normal (0.0, 0.02) and biases are 0.0), and the noise is sampled from the standard

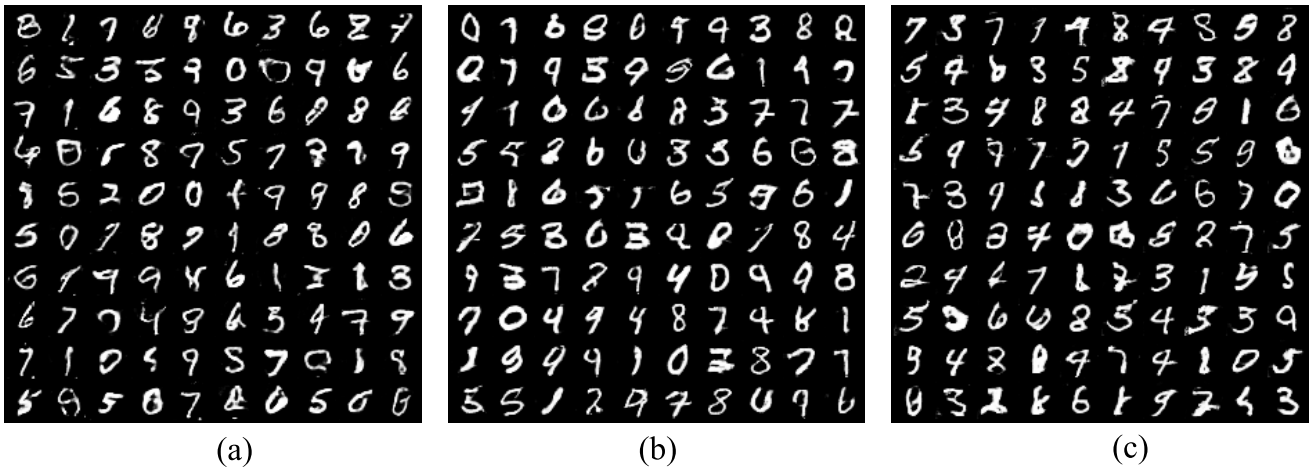


Fig. 4. Synthetic images produced by the three candidate models on balanced data setting of MNIST with $K = 2$. The first client holds figures "0"–"4" with 10 000 samples, while the second client holds figures "5"–"9" with 10 000 samples. MD-GAN does not capture figure "4," while FL-GAN does not capture figure "2." Only our IFL-GAN generates all figures from "0" to "9." (a) MD-GAN. (b) FL-GAN. (c) IFL-GAN.

Gaussian (0, 1) [$p_z(z)$]. Two recently popular GAN variants are employed as baselines, which are **MD-GAN** [9] and FL-GAN [9], [21]. They are representatives of distributed GAN. For making a fair comparison, we adopt the same components and hyperparameters (e.g., Epoch $=$ 200 and learning rate $=$ 0.0002) and implementation details for baselines and IFL-GAN. All models are implemented in the *Pytorch* framework.

Three commonly used public image datasets, the MNIST, CIFAR-10, and SVHN, are studied. The MNIST dataset consists of a gray-scale image with a size of 28 $\times$ 28. Both the CIFAR-10 dataset and the SVHN dataset contain RGB images, and their sizes are set to 3 $\times$ 32 $\times$ 32. We conduct experiments with a number of models $K \in [2, 5, 10]$. For model split, $K = 2$ (5 or 10) indicates two (five or ten) local GAN models, one global GAN for FL-GAN, our proposed IFL-GAN, K discriminators, and one single generator are employed for MD-GAN. For training data split (taking MNIST as the example) to K clients, $K = 2$ indicates that each client holds five categories. The first client holds figures "0"–"4," while the second client holds figures "5"–"9." Each client holds 10 000 training samples, and we call it "balanced data." The first client holds 10 000 data samples, while the second client holds 1000 data samples, and we call it "imbalanced data." When the second client holds 100 data samples, we term it "extremely imbalanced data." For $K = 5$, each client holds two categories and has 10 000 data samples in the case of balanced data. The first three clients hold 15 000 data samples, and each one has 5000 samples, while the last two clients hold 2000 data samples, and each one has 1000 samples in the imbalanced case. Each client holds 100 data samples for the last two clients in the extremely imbalanced case. For $K = 10$, each client just holds one category and has 5000 data samples in the balanced case. In the imbalanced case, each client within the first five clients holds 5000 samples, while each client within the last five clients holds 1000 samples. In the extremely imbalanced case, each client within the last five clients holds 100 samples.

A. MNIST

We first apply IFL-GAN and baselines to the MNIST dataset in the case of $K = 2$ with balanced data. For FL-GAN and IFL-GAN, the generated images are produced by a global generator. For MD-GAN, there is only one single generator. Since our proposed approach is to utilize the MMD to assign the weights for corresponding local models, and determining whether the empirical MMD shows a statistically significant difference is achieved by comparing the test statistic with a particular threshold [8], we utilize the minimal MMD score to set the threshold for each mmd_i during training. This is because the smaller MMD score implicitly indicates that one distribution is better matching another distribution. In this case, the threshold c_{a_1} of mmd_1 is 0.11027 at epoch $=$ 182, and the threshold c_{a_2} of mmd_2 is 0.11463 at epoch $=$ 184. In other words, if mmd_1 (mmd_2) score is larger than 0.11027 (0.11463), we should reject the hypothesis that assuming the two distributions match with each other; otherwise, we should accept the hypothesis. Once we accept the two hypotheses from the two clients, the global GAN generates the simulation data. The generated images of the three candidate models in the same case are shown in Fig. 4, and the corresponding loss values of two generators for FL-GAN and IFL-GAN are shown in Fig. 5, which is plotted by *exponential moving average* method. From Fig. 4, we can observe that the MD-GAN does not capture figure "4," while FL-GAN does not hold figure "2." Only our proposed IFL-GAN generates all figures, which demonstrates the effectiveness of IFL-GAN. From Fig. 5, we can observe that IFL-GAN outperforms FL-GAN in terms of smaller fluctuation and faster convergence.

Furthermore, we continue to quantitatively evaluate the performance of our proposed IFL-GAN in different cases (i.e., $K = 2, 5, 10$ on "balanced data," "imbalanced data," and "extremely imbalanced data" settings of the MNIST dataset). The MNIST score [9] (the higher the better), similar to the inceptions score, is employed as the quantitative metric in our study. The MNIST scores achieved by our proposed IFL-GAN and baselines are shown in Table I. Fig. 6 shows the

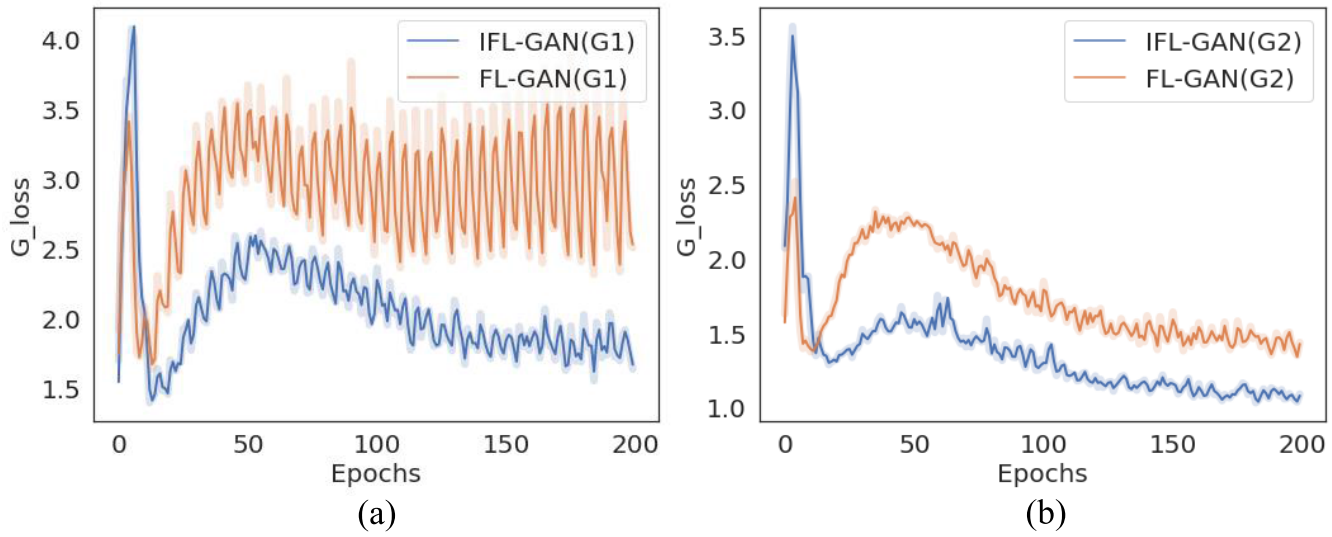


Fig. 5. In the case of $K = 2$ on the balanced data setting, (a) indicates the first generator loss for our proposed IFL-GAN and FL-GAN and (b) indicates the second generator loss for the two models. In each subfigure, the red curve indicates the generator loss of FL-GAN, while the blue curve indicates that of IFL-GAN.

TABLE I

MNIST SCORES ON BALANCED, IMBALANCED, AND EXTREMELY IMBALANCED SETTINGS OF THE MNIST DATASET. IT IS OBVIOUS THAT OUR PROPOSED IFL-GAN ACHIEVES THE BEST PERFORMANCE AND OBTAINS THE HIGHEST MNIST SCORES IN DIFFERENT CASES

Models	Balanced data	Imbalanced data	Extremely Imbalanced data
MD-GAN (K=2)	6.009	6.108	5.174
MD-GAN (K=5)	5.946	5.862	5.687
MD-GAN (K=10)	5.944	5.761	4.408
FL-GAN (K=2)	6.711	5.831	3.613
FL-GAN (K=5)	5.005	5.434	5.638
FL-GAN (K=10)	4.906	3.621	2.499
IFL-GAN (K=2)	**6.329**	**7.083**	**6.017**
IFL-GAN (K=5)	**6.127**	**5.948**	**5.802**
IFL-GAN (K=10)	**6.094**	**5.787**	**4.701**

Fig. 6. Synthetic images produced by the three candidate models on extremely imbalanced data setting of MNIST in the case of $K = 2$ in which the client$_1$ holds figures from "0" to "4" with 10 000 samples, while the second client$_2$ holds figures from "5" to "9" with 100 samples. In this case, MD-GAN basically focuses on the figures "0"–"4," while the FL-GAN holds the worse quality among the three models. Only our proposed IFL-GAN produces diverse generated images. (a) MD-GAN. (b) FL-GAN. (c) IFL-GAN.

generated images from all the three models in the case of $K = 2$ on the extremely imbalanced setting of MNIST. It is obvious that our proposed IFL-GAN produces more diverse simulation images than the other two GAN variants. For example, Fig. 6(c) shows the figures "6" (row "4" and col "8"), "7" (row "1" and col "6"), "8" (row "4" and col "6"), and

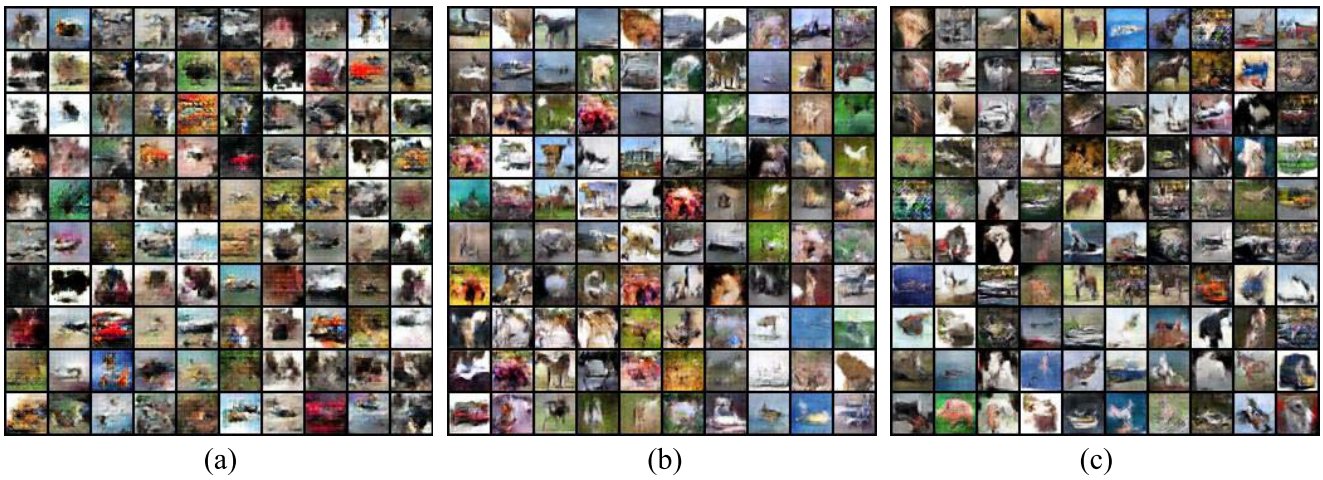


Fig. 7. Synthetic images produced by three candidate models in the case of $K = 2$ on the balanced data setting of CIFAR10 in which the first client holds categories "airplane," "automobile," "bird," "cat," and "deer," while the second client holds categories "dog," "frog," "horse," "ship," and "truck." (a) MD-GAN. (b) FL-GAN. (c) IFL-GAN.

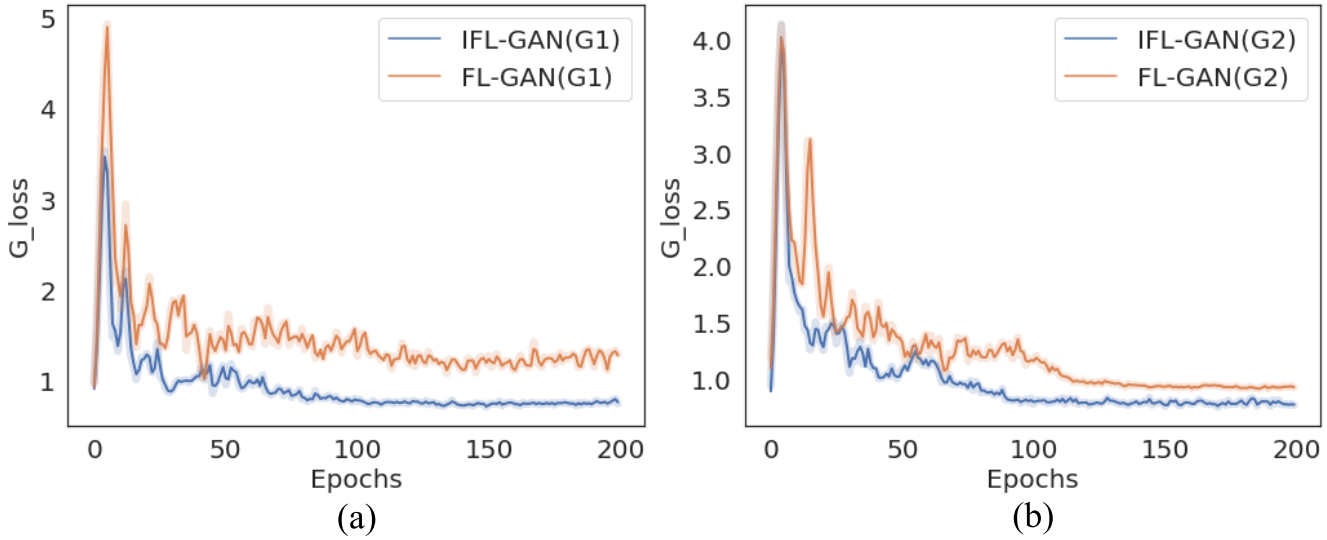


Fig. 8. In the case of $K = 2$ on the balanced data setting of CIFAR10, (a) indicates the first generator loss for our proposed IFL-GAN and FL-GAN, and (b) indicates the second generator loss for the two models. In each subfigure, the red curve indicates the generator loss of FL-GAN, while the blue curve indicates that of IFL-GAN.

"9" (row "1" and col "7"). Fig. 6(a) basically focuses on the figures "0"–"4," while Fig. 6(b) shows the worse quality of generated images among the three models. For the imbalanced case, the generated data quality is similar to Fig. 6. As is visually predictable from the outputs, the MNIST score of our proposed IFL-GAN is higher than other GAN variants in different cases, demonstrating the effectiveness of IFL-GAN.

B. CIFAR10

We continue to apply our proposed IFL-GAN and baselines to the CIFAR10 dataset in different cases: $K = 2, 5, 10$ on the "balanced data," "imbalanced data," and "extremely imbalanced data" settings. Still, we utilize the minimal MMD score to set the threshold during training. Here, we demonstrate the case of $K = 2$ with balanced data for convenient observation. In this case, the threshold c_{a_1} of mmd_1 is 0.09416 at epoch = 171, and the threshold c_{a_2} of mmd_2 is 0.08366 at epoch = 199. The threshold can help us reject the hypothesis (generated data distribution matching to original data distribution) if the MMD score is larger than the threshold; otherwise, we should accept the hypothesis. When we accept the two hypotheses, the global GAN generates simulation data. The generated images, in this case, are shown in Fig. 7, and the corresponding generator loss values are shown in Fig. 8. It also indicates that our proposed IFL-GAN outperforms FL-GAN in terms of faster convergence and smaller fluctuation, especially for the first generator G_1. To further evaluate the performance of our proposed IFL-GAN, we adopt the inception score (the higher the better) [2], [26] as the metric. The inception score involves using a pretrained Inception v3 network model for image classification to predict the class probabilities for each generated image, and these probabilities are conditional

TABLE II

INCEPTION SCORES ON BALANCED, IMBALANCED, AND EXTREMELY IMBALANCED SETTINGS OF THE CIFAR10 DATASET. OUR PROPOSED IFL-GAN STILL ACHIEVES THE BEST PERFORMANCE AND OBTAINS THE HIGHEST INCEPTION SCORES IN DIFFERENT CASES

Models	Balanced data	Imbalanced data	Extremely Imbalanced data
MD-GAN (K=2)	3.466	3.088	3.183
MD-GAN (K=5)	3.172	1.912	1.987
MD-GAN (K=10)	2.042	2.008	1.936
FL-GAN (K=2)	4.152	3.128	3.188
FL-GAN (K=5)	3.185	1.095	1.173
FL-GAN (K=10)	3.166	1.239	1.154
IFL-GAN (K=2)	**4.811**	**3.459**	**3.759**
IFL-GAN (K=5)	**3.731**	**2.936**	**3.272**
IFL-GAN (K=10)	**3.516**	**3.143**	**2.519**

(a) (b) (c)

Fig. 9. Synthetic images produced by three candidate models in the case of $K = 2$ on the balanced data setting of SVHN in which the first client holds the categories from "0" to "4," while the second client holds the categories from "5" to "9." (a) MD-GAN. (b) FL-GAN. (c) IFL-GAN.

probabilities because images that contain meaningful objects should have a conditional label distribution $p(y|x)$ with low entropy [26]. The inception score of generated images is reported in Table II. It is obvious that our proposed IFL-GAN still outperforms the baselines in different cases, which further validates the effectiveness of IFL-GAN.

C. SVHN

The street view house numbers (SVHNs) [7] are constructed from real-world house numbers in Google Street View images [22]. It can be seen as similar in flavor to MNIST (e.g., the images are of small cropped digits) but incorporates an order of magnitude more labeled data (over 600 000 digit images). We continue to apply our proposed IFL-GAN and the baselines to this dataset. We still set the threshold with the minimal MMD score during training. In the case of balanced data with $K = 2$, the threshold c_{a_1} of mmd_1 is 0.05169 at epoch $= 169$, and the threshold c_{a_2} of mmd_2 is 0.0475 at epoch $= 152$. We would reject the hypothesis if the MMD score is larger than the threshold and accept this hypothesis if the MMD score is smaller and equal to the threshold. After accepting the two hypotheses, the global GAN generates the simulation data. The corresponding generated images of this case are shown in Fig. 9, and the generator loss values are shown in Fig. 10, which illustrates that IFL-GAN gets converged faster than FL-GAN. In addition, we develop the SVHN score (the higher the better), which is calculated in a similar way as the MNIST score to evaluate the performance of the three candidate models on balanced data, imbalanced data, and extremely imbalanced data with $K = 2, 5, 10$. To obtain the SVHN score, the training set (i.e., 73 257 digits) is employed to train a classifier, and 100 generated images are viewed as the test set. The SVHN scores of generated images from the three candidate models are reported in Table III. From Table III, we can observe that our proposed IFL-GAN still outperforms baselines in achieving the highest SVHN score on the SVHN dataset.

D. Impact of Different Data Division Methods

Note that, although the data division in our study is artificial, it could reflect the real scene to some extent. Here, we discuss the data division methods of both our study and [12], which is a representative data division method in federated learning. The study [12] splits data samples according to the Dirichlet distribution. It draws $q \sim \mathbf{Dir}(\alpha p)$ samples from a Dirichlet distribution in which p characterizes a prior class distribution and α denotes the concentration parameter. Note that the prior class distribution is a uniform distribution in [12]. When α is a small value (e.g., $\alpha \leq 1$), each client holds a few or even one category with q images and holds most categories when

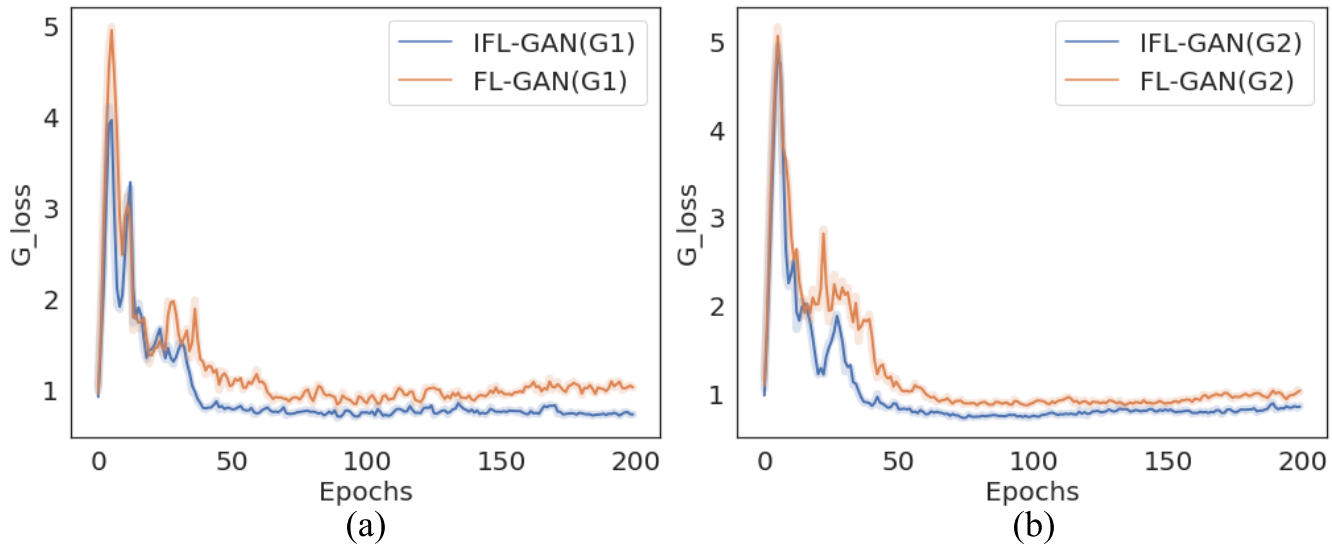


Fig. 10. In the case of $K = 2$ on the balanced data setting of SVHN, (a) indicates the first generator loss for our proposed IFL-GAN and FL-GAN, and (b) indicates the second generator loss for the two models.

TABLE III

SVHN SCORES ON BALANCED, IMBALANCED, AND EXTREMELY IMBALANCED SETTINGS OF THE SVHN DATASET. THE EXPERIMENTAL RESULTS FURTHER DEMONSTRATE THE EFFECTIVENESS OF OUR PROPOSED IFL-GAN

Models	Balanced data	Imbalanced data	Extremely Imbalanced data
MD-GAN (K=2)	4.409	3.916	3.996
MD-GAN (K=5)	3.747	3.937	4.247
MD-GAN (K=10)	3.884	3.439	3.657
FL-GAN (K=2)	4.394	4.174	3.940
FL-GAN (K=5)	4.336	1.256	1.259
FL-GAN (K=10)	3.794	1.994	1.128
IFL-GAN (K=2)	**4.525**	**4.487**	**4.149**
IFL-GAN (K=5)	**4.905**	**4.197**	**4.456**
IFL-GAN (K=10)	**4.117**	**3.956**	**3.984**

α has a larger value (e.g., $\alpha \to \infty$). In our study, the class distribution still follows a uniform distribution. Moreover, each client holds a few or just one category when $K = 5$ or $K = 10$ and holds more categories when $K = 2$. In other words, the division method in our study is similar to that in [12] to a certain degree. We also conduct experiments with $\alpha = 20.0$ and $\alpha = 2.0$ on CIFAR10. Here, we assume two clients and divide samples according to the Dirichlet distribution. For the case of $\alpha = 20.0$, client 1 holds categories "airplane," "automobile," "frog," "ship," and "truck," while client 2 holds categories "bird," "cat," "deer," "dog," and "horse." This corresponds to the case of $K = 2$ in our study where client 1 holds categories "airplane," "automobile," "bird," "cat," and "deer," while client 2 holds categories "dog," "frog," "horse," "ship," and "truck." The corresponding inception scores are similar to Table II. For the case of $\alpha = 2.0$, client 1 holds categories "automobile," "deer," and "dog" with 10 000 images, while client 2 holds categories "airplane," "bird," "cat," "frog," "horse," "ship," and "truck" with 10 000 images (i.e., balanced data) or 100 images (i.e., extremely imbalanced data). Both FedAvg and MMD are utilized to update the parameters of the model, respectively. The inception scores are shown in Table IV. It is obvious that our approach still outperforms the traditional FedAvg method when utilizing Dirichlet to divide data samples.

TABLE IV

INCEPTION SCORES ON DIRICHLET DATA DIVISION. THE RESULTS STILL SHOW THE EFFECTIVENESS OF OUR IFL-GAN

Models	Balanced data	Extremely Imbalanced data
FL-GAN	4.36	3.99
IFL-GAN	**4.95**	**4.74**

E. Discussion

Although MD-GAN and FL-GAN can handle the distributed data, they have limitations if data are distributed over multiple clients in a non-i.i.d. manner (see Fig. 6). This is because the federated averaging method views each local model with the same contribution to train the global model. Data distributed over clients in the i.i.d. manner is not often the case in practice, so each local model inevitably holds different contributions to the global model. To this end, we propose IFL-GAN, which takes the MMD score as each local model's contribution weight to drive the learning of a globally shared model by

aggregating locally computed updates with those weights. Our IFL-GAN is more generalizable and achieves faster and more stable convergence (see Figs. 5, 8, and 10) than baselines (i.e., FL-GAN and MD-GAN) on decentralized and non-i.i.d. data (see Figs. 4, 7, and 9) and still produces diverse instances when facing extremely imbalanced data (see Fig. 6).

VI. Conclusion

In this study, to deal with the challenge of training GAN models on distributed data in a non-i.i.d. manner, we propose IFL-GAN. Comprehensive experiments demonstrate the following capabilities of our IFL-GAN: 1) achieving higher MNIST scores, inception scores, and SVHN scores than FL-GAN and MD-GAN on decentralized and non-i.i.d. data; 2) generating more diverse and appealing recognizable images; and 3) converging faster and more stable than FL-GAN.

References

[1] L. M. A. Radford and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.

[2] S. Barratt and R. Sharma, "A note on the inception score," 2018, *arXiv:1801.01973*.

[3] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*.

[4] C. Daskalakis, W. P. Goldberg, and H. C. Papadimitriou, "The complexity of computing a Nash equilibrium," *SIAM J. Comput.*, vol. 39, no. 1, pp. 195–259, 2009.

[5] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.

[6] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[7] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," 2013, *arXiv:1312.6082*.

[8] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 723–773, Mar. 2012.

[9] C. Hardy, E. Le Merrer, and B. Sericola, "MD-GAN: Multi-discriminator generative adversarial networks for distributed datasets," 2018, *arXiv:1811.03850*.

[10] I. Hegedűs, G. Danner, and M. Jelasity, "Gossip learning as a decentralized alternative to federated learning," in *Proc. IFIP Int. Conf. Distrib. Appl. Interoperable Syst.* Cham, Switzerland: Springer, 2019, pp. 74–90.

[11] K. Hsieh *et al.*, "Gaia: Geo-distributed machine learning approaching LAN speeds," in *Proc. 14th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2017, pp. 629–647.

[12] T.-M. Harry Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv:1909.06335*.

[13] S.-P. Hu, "Simple mean, weighted mean, or geometric mean," in *Proc. ISPA/SCEA Int. Conf.*, San Diego, CA, USA, 2010, pp. 1–24.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[15] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos, "MMD GAN: Towards deeper understanding of moment matching network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2203–2213.

[16] W. Li, W. Ding, R. Sadasivam, X. Cui, and P. Chen, "His-GAN: A histogram-based GAN model to improve data generation quality," *Neural Netw.*, vol. 119, pp. 31–45, Nov. 2019.

[17] W. Li, L. Fan, Z. Wang, C. Ma, and X. Cui, "Tackling mode collapse in multi-generator GANs with orthogonal vectors," *Pattern Recognit.*, vol. 110, Feb. 2021, Art. no. 107646.

[18] W. Li, Z. Liang, P. Ma, R. Wang, X. Cui, and P. Chen, "Hausdorff GAN: Improving GAN generation quality with Hausdorff metric," *IEEE Trans. Cybern.*, early access, Mar. 18, 2021, doi: 10.1109/TCYB.2021.3062396.

[19] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," 2019, *arXiv:1901.06455*.

[20] R. Mayer and H.-A. Jacobsen, "Scalable deep learning on distributed infrastructures: Challenges, techniques and tools," 2019, *arXiv:1903.11314*.

[21] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

[22] Y. Netzer, T. Wang, A. Coates, A. Bissacco, Bo Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011.

[23] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proc. 2nd Workshop Distrib. Infrastruct. Deep Learn.*, Dec. 2018, pp. 1–8.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[25] J. R. Norris, *Markov chains*, no. 2. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[26] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.

[27] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4424–4434.

[28] X. Wang and A. Gupta, "Generative image modeling using style and structure adversarial networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 318–335.

[29] C. Xiao, P. Zhong, and C. Zheng, "BourGAN: Generative networks with metric embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2269–2280.

[30] B. Xu, N. Wang, C. Naiyan, and T. Mu Li, "Empirical Evaluation of Rectified Activations in Convolutional Network," 2015, *arXiv:1505.00853*.

[31] Q. Xu *et al.*, "An empirical study on evaluation metrics of generative adversarial networks," 2018, *arXiv:1806.07755*.

[32] R. R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *IEEE Trans. Syst., Man, Cybern.*, vol. 18, no. 1, pp. 183–190, Jan./Feb. 1988.

[33] R. R. Yager and J. Kacprzyk, *The Ordered Weighted Averaging Operators: Theory and Applications*. Berlin, Germany: Springer, Sci. Bus. Media, 2012.

[34] H. Hankui Zhuo, W. Feng, Y. Lin, Q. Xu, and Q. Yang, "Federated deep reinforcement learning," 2019, *arXiv:1901.08277*.

Wei Li (Member, IEEE) received the bachelor's degree from the School of Mathematics and Statistics, Wuhan University, Wuhan, China, in 2008, and the Ph.D. degree from the School of Cyber Science and Engineering, Wuhan University, in 2019.

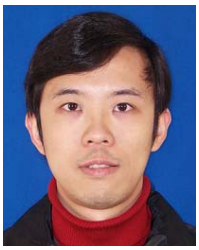
He was a Visiting Student with the University of Massachusetts Boston, Boston, MA, USA, and had visited The Hong Kong Polytechnic University, Hong Kong, as a Research Assistant. He is currently an Associate Professor with the School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi, China. His research areas include data mining, machine learning, and artificial intelligence.

Jinlin Chen received the master's degree from The Hong Kong Polytechnic University, Hong Kong, in 2016, where he is currently pursuing the Ph.D. degree with the Department of Computing.

From 2017 to 2018, he was a Machine Learning Engineer with The Hong Kong Applied Science and Technology Research Institute, Hong Kong. His research interests include machine learning security, robotics control, and multiagent reinforcement learning.

Zhenyu Wang received the bachelor's degree in software engineering from Wuhan University, Wuhan, China, in 2010, and the Ph.D. degree from the School of Remote Sensing Information Engineering, Wuhan University, in 2018.

He held a post-doctoral research position at the School of Computer Science, Wuhan University. He is currently a Researcher with the Jiaxing Institute of Future Food, Jiaxing, China. His research areas include data mining, big data on food safety, and artificial intelligence.



Zhidong Shen received the M.A. and Ph.D. degrees in computer science from Wuhan University, Wuhan, China, in 2003 and 2006.

He did his visiting research at the Queensland University of Technology, Brisbane, QLD, Australia, in 2010, and the University of Victoria, Victoria, BC, Canada, in 2014. He is currently an Associate Professor with the School of Cyber Science and Engineering, Wuhan University. His research interests include cyberspace security, trusted computing, machine learning, and big data.

Dr. Shen is also a member of the Chinese Computer Federation (CCF).

Chao Ma (Member, IEEE) is currently an Assistant Professor with the School of Cyber Science and Engineering, Wuhan University, Wuhan, China. He has published over 30 academic papers in major international journals and conference proceedings. His research interests include time-series analytics, representation learning, deep learning, explainable AI, and big data analytics.

Dr. Ma is also a Professional Member of the Chinese Computer Federation (CCF).

Xiaohui Cui received the B.S. degree in photoelectric technology and the M.S. degree in computer science from Wuhan University, Wuhan, China, in 1996 and 2000, and the Ph.D. degree in computer science and engineering from the University of Louisville, Louisville, KY, USA, in 2004.

He is currently a Professor with the School of Cyber Science and Engineering, Wuhan University. He has published over 160 papers in major artificial intelligence journals and conferences. His research areas include data mining, machine learning, and artificial intelligence.

Dr. Cui received multiple national grants.