FedDTG:Federated Data-Free Knowledge Distillation via Three-Player Generative Adversarial Networks

Abstract

Applying knowledge distillation to personalized cross-silo federated learning can well alleviate the problem of user heterogeneity. This approach, however, requires a proxy dataset, which is difficult to obtain in the real world. Moreover, the global model based on parameter averaging will lead to the leakage of user privacy. We introduce a distributed three-player GAN to implement data-free co-distillation between clients. This technique mitigates the user heterogeneity problem and better protects user privacy. We confirmed that the fake samples generated by GAN can make federated distillation more efficient and robust, and the co-distillation can achieve good performance for individual clients on the basis of obtaining global knowledge. Our extensive experiments on benchmark datasets demonstrate the superior generalization performance of the proposed methods, compared with the state-of-the-art.

1 Introduction

Federated Learning(FL) is an important machine learning approach that enables the server to use the computing resources from many clients to collaboratively train a centralized model without accepting their data. Classic federated learning, represented by FedAvg [Stephansen *et al.*, 2018], aggregates models from clients to achieve a global model so that the global model can be more general and capable. Non-iid data present a tough challenge for federated learning, which can cause the global model to converge slowly or even away from the global optima [Li *et al.*, 2020b] [Li *et al.*, 2021a]. There have been some studies trying to address the non-iid issue, which can be divided into two complementary perspectives.The first one focuses on stabilizing local training [Li *et al.*, 2020a] [Johnson and Zhang, 2013]. However the generalization performance of these approaches is debatable, which can be as bad as FedAvg in many cases [Li *et al.*, 2021c] . Another aims to improve the model aggregation, among which Knowledge Distillation(KD) [Hinton *et al.*, 2015] has emerged as an effective solution. Provided with an unlabeled dataset as the proxy, the server can efficiently refine the global model by distilling the knowledge from the aggregation model. Compared with the aggregation method of parameter average, federated distillation alleviates the model drift issue caused by user heterogeneity, and requires less communication overhead. However, this approach depands on a proxy dataset related to local dataset, which is impossible to satisfy in many industrial scenarios.

It is worth noting that all the above methods are implemented on the global model obtained by FedAvg. On one hand, different clients have personalized model structures, and it is not possible to directly average model parameters. On the other hand, the ensemble knowledge may in turn reduce the accuracy of the global model while the distribution of the proxy dataset is inconsistent with the global distribution. Most importantly, through the weight difference between the global model and local model uploaded by the user, the attacker can restore the local training dataset to a certain extent [Yin *et al.*, 2021], which will lead to the leakage of user privacy.

Unlike traditional federated learning which only needs to get an excellent global model, FedDTG allows each client to have a personalized model. Consider the case of personalized cross-silo federated learning, where the private data of clients may be different in size, class distributions and even the distribution of each class. FedDTG performs well in cross-silo federated learning, while the algorithm based on FedAvg without considering individual private data often cannot meets the personalized needs for individual clients. In FedDTG, each client can use the global generative model locally generates the missing data samples, which can regulate the local objective function. In addtion, when we need to different tasks on the same type of dataset, we do not need to retrain FedDTG. The fake data generated by FedDTG can be directly applied to all federated distillation algorithms.

To make federated learning more versatile and safer in realistic settings, we propose Federated Data-Free Knowledge Distillation via Three-Player Generative Adversarial Networks(FedDTG). We leverage the fake samples to enable federated distillation more efficient and robust. These fake samples are generated by a distributing GAN which can be trained simultaneously with federated distillation and local personalization tasks. Our main contributions are:

- Our proposed federated distillation approach FedDTG does not require a proxy dataset, allows for heterogeneous client models, alleviates the privacy risks, and reduces the accuracy difference between clients.

- FedDTG also learned a global generative model, which can be used to augment local dataset and prepare public datasets for future federated distillation.
- We experiment FedDTG on various image datasets with different degrees of non-iid and show that global generator can generate effective data to speed up the federated distillation and achieve higher accuracy compared with the state-of-the-art.

2 Related Work

2.1 Federated Learning

As a classic algorithm for Federated Learning(FL), FedAvg [Stephansen *et al.*, 2018] first sends the global model to the clients participating in this communication round of federation. Then the client updates the received global model using averaged stochastic gradient descent (SGD) [Polyak and Juditsky, 2006] on its own local dataset and sends the updated local model to a central server. Finally, the server performs weighted parameter average over the received client models and produce a new global model. To improve FedAvg on non-IID data, there have been some studies trying to stabilize local training. FedProx [Li *et al.*, 2020a] proposes a proximal term to improve local training. The proximal term is used to measure the difference between the local model and the current global model, which is computed based on the '2-norm distance'. By addding proximal term to local objective, the update of the local model will be restricted by the global model. Similarly, SCAFFOLD [Karimireddy *et al.*, 2020] uses the difference between the local control variates and the global control variates to correct the local updates. However the generalization of these methods remains to be verified. As shown in the [Yin *et al.*, 2021], those studies have little or even no advantage over FedAvg under certain non-iid settings.

2.2 Federated Distillation

Knowledge distillation is first introduced to compress knowledge from one or more teacher models into a small student model. Hinton [Hinton *et al.*, 2015] defined the knowledge of the model as the softened logits and the student model mimic the knowledge of teacher model to acquire its abilities. Most methods of knowledge distillation only need the outputs of hidden layer or output layer. Compared to directly exchanging the model parameters, federated distillation only requires lower communication costs to achieve better aggregation results. The FedMD proposed by [Li and Wang, 2019] uses transfer learning to combine knowledge distillation and federated learning. They let each client participating in this communication round calculate its own class scores on the public dataset, and transmits the result to a central server. Then the server will average the received class scores to get the updated consensus. This consensus represents the knowledge of all participating client models and each client will train its model to approach the consensus on the public dataset. In this way, each client model can learn the knowledge of other client models without sharing its own private data or models. However this method will bring a loss of accuracy and huge differences between clients in the case of heterogeneous users. [Lin *et al.*, 2020] proposed FedDF to improve the efficacy of model aggregation. They use the global model in FedAvg as the student model and do ensemble distillation to get the knowledge from all client teacher models. The ensemble knowledge is represented by the average logit outputs of all parties on an unlabeled dataset from other domains. However only refining the global model can not completely take advantage of distillation to slove non-iid problem, which may even weaken the effect of knowledge distillation. The FedGen proposed by [Zhu *et al.*, 2021] is the first to combine data-free distillation with federated learning. FedGen only needs to share a lightweight generator model and the prediction layer of the local model to the server for averaging. The global generator outputs feature representations, which is the input of the prediction layer, to reinforce the local learning with a better generalization performance. However, due to the over dependence on the global generator, which conveys the all ensemble user information, it can not give full play to the advantages of knowledge distillation.

2.3 Federated GANs

Generative adversarial networks (GAN) [Goodfellow *et al.*, 2014] is a generative model, which provide a way to learn deep representations without extensively annotated training dataset. Distributed GANs are proposed to train a generator over datasets from multiple works. The MD-GAN proposed by [Hardy *et al.*, 2019] consists of a single global generator and distributed discriminators. The global generator plays a game with the ensemble of all participant distributed Discriminators. The server in MD-GAN sends two distinct batches, which are composed of the data generated by the global generator, to each party and the client will performs some learning iterations on its discriminator. Then the client sends the error feedback to the server. Finally, the server computers the average of all feedbacks to update its parameters. During the training process, clients will iteratively exchange their discriminators with each other to avoid overfitting. With similar settings, the global generator in [Yonetani *et al.*, 2019] only needs to fool the weakest individual discriminator. However, the communication cost caused by the transmission of the generated data is too high. There are also some works directly apply FedAvg to generators and discriminators. Our method adds a classifier for a tripartite confrontation. The classifier can help the generator learn the data distribution faster, and also serve as a personalized task for the client.

3 Method

Before we start, here are some notations and typical FL settings to better illustrate our method. Each local client $Local_k$ has a Generator $\mathbf{G}_k$,a Discriminator $\mathbf{D}_k$ and a Classifier $\mathbf{C}_k$ which represents the local personalized training task. $Local_k$ owns a labeled private dataset $Data_k$. Let p_{data} be the global true data distribution, which is non-observable by conventional FL, and $p_{\mathbf{G}}$ be the generated sample distribution.

The goal for FedDTG is to train N different classifiers that that perform well on the corresponding user tasks and a global generator $\mathbf{G}$ to make p_{data} and $p_{\mathbf{G}}$ as similar as possible. The training process can be divided into three steps: Local Adversarial Training, Server Aggregation and Federated Distillation.

3.1 Local Adversarial Training

A regular GAN comprises a min-max game, played between the generator $\mathbf{G}$ and discriminator $\mathbf{D}$, with the objective function shown in Equation 1

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

As shown in Fig 1, we add the local classifier $\mathbf{C}_k$ to the training of GAN, which is parameterized by θ_c^k that minimizes the risk on the personalized user task. On one hand, $\mathbf{C}_k$ can help $\mathbf{G}_k$ learning the local data distribution p_{data}^k faster; on the other hand, $\mathbf{G}_k$ can help $\mathbf{C}_k$ alleviate the model drift caused by unbalanced local data during local training.

Like the traditional GAN, $\mathbf{D}_k$ is a binary classifier, which needs to classify the real samples as true and the generated samples as false. $\mathbf{D}_k$ needs to maximize the following objective function:

区分真假样本

$$\mathcal{L}_{\mathbf{D}_k} = \frac{1}{\|data_k\|}[\sum_{x \sim p_{data_k}} log\mathbf{D}_k(x) + \sum_{z \sim p_z(z), \hat{y} \sim p_{\hat{y}}(\hat{y})} log(1 - \mathbf{D}_k(\mathbf{G}(z, \hat{y})))] \quad (2)$$

$\mathbf{G}_k$ takes noise vector $z \sim \mathcal{N}(0,1)$ and label vector $\hat{y} \sim \mathcal{U}(1,n)$ as inputs and outputs fake samples. The generated fake sample needs to be correctly classified by $\mathbf{C}_k$ and classified as true by the $\mathbf{D}_k$. With the help of $\mathbf{C}_k$, $\mathbf{G}_k$ can correspond the generated fake samples to the input labels and learn the local data distribution faster. $\mathbf{G}_k$ needs to minimize the following objective function:

鉴别器无法区分，且分类器分类正确

$$\mathcal{L}_{\mathbf{G}_k} = \frac{1}{\|data_k\|} \sum_{z \sim p_z(z), \hat{y} \sim p_{\hat{y}}(\hat{y})} [1 - log(\mathbf{D}_k(\mathbf{G}(z, \hat{y}))) + CE(\mathbf{C}(\mathbf{G}(z, \hat{y})), \hat{y})] \quad (3)$$

where CE stands for cross entropy. $\mathbf{C}_k$ should not only correctly classify the local data, but also correctly classify the fake samples generated by $\mathbf{G}_k$. With the help of $\mathbf{G}_k$, $\mathbf{C}_k$ will not overfit the local training dataset, resulting in model drift, especially in some extreme non-iid cases (for example, each client has only one category of data). $\mathbf{C}_k$ needs to minimize the following objective function:

对真假样本都能正确分类

$$\mathcal{L}_{\mathbf{C}_k} = \frac{1}{\|data_k\|}[\sum_{(x,y) \sim p_{data_k}} CE(\mathbf{C}_k(x), y) + \sum_{(x_g, \hat{y}) \sim p_{\mathbf{G}_k}} CE(\mathbf{C}_k(x_g), \hat{y})] \quad (4)$$

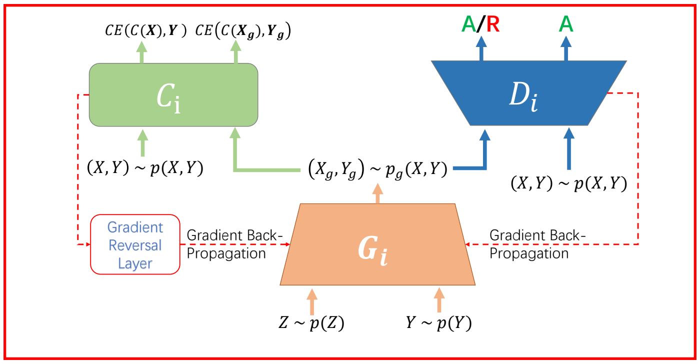
$CE(C(\boldsymbol{X}), \boldsymbol{Y})$
$CE(C(\boldsymbol{X_g}), \boldsymbol{Y_g})$
A/R
A
C_i
D_i
$(X,Y) \sim p(X,Y)$
$(X_g, Y_g) \sim p_g(X,Y)$
$(X,Y) \sim p(X,Y)$
Gradient Reversal Layer
Gradient Back-Propagation
G_i
Gradient Back-Propagation
$Z \sim p(Z)$
$Y \sim p(Y)$

Figure 1: An illustration of the the local GAN. Different colors represent different models, i denoting the client index, 'R' denoting rejection, 'A' denoting acceptance and 'CE' denoting the cross entropy loss.

Our local three-player GAN training is similar to [Vandenhende *et al.*, 2019]. Howeer their generator is used to output hard samples to help the classifier train faster, while our $\mathbf{G}_k$ is used to help the $\mathbf{C}_k$ learn the global objective function better.

3.2 Server Aggregation

After local training, each client participating in this communication round needs to send $\mathbf{G}_k$ and $\mathbf{D}_k$ to the server. Then, the server will use parameter averaging to aggregate the global generator and discriminator, parameterized by θ_g and θ_d which are calculated by the following functions:

$$\theta_g = \sum_{k=1}^{M} \frac{1}{M} \theta_g^k, \theta_d = \sum_{k=1}^{M} \frac{1}{M} \theta_d^k$$

It is worth noting that the local generator $\mathbf{G}_k$ does not directly touch the real local dataset during the entire training process, and the generatered dataset is determined by noise vector $z \sim \mathcal{N}(0,1)$ and label vector $\hat{y} \sim \mathcal{U}(1,n)$, which is different from the local label distribution. For $\mathbf{D}_k$, it's a binary classifier and can not provide the probability information of each class. So we cannot use the uploaded $\mathbf{G}_k$ and $\mathbf{D}_k$ to get the local label distribution and directly restore the local training dataset. Finally the server sends the new global $\mathbf{G}$ and $\mathbf{D}$ to the client. The whole process of server aggregation is shown in A on the left side of Fig 2.

At this stage, the new global $\mathbf{G}$ can obtain the outputs of all local $\mathbf{C}_k$ on the generated fake samples. We have tried to use these outputs to further optimize the global $\mathbf{G}$. This can indeed reduce the number of communication rounds and achieve better performance. However, the frequently changing parameters of global $\mathbf{G}$ and generated fake samples will greatly increase the communication cost of each round. Therefore, in the server aggregation stage, we only use the parameter averaging method.

3.3 Federated Distillation

When the client receives the models sent by the server, it will replace the $\mathbf{G}_k$ and $\mathbf{D}_k$ with the global $\mathbf{G}$ and $\mathbf{D}$. The server will specify the batch size and noise vector $\hat{z} \sim \mathcal{N}(0,1)$ for the federal distillation stage. For each batch, the client will generate the same number of label vector $\hat{y}$ for each class and input them into tho $\mathbf{G}_k$ together with $\hat{z}$. Since the inputs and model parameters are consistent, all clients of this communication round will output the same fake samples. In this way, the communication cost caused by transmitting a large amount of generated data is avoided.

Input the fake samples into $\mathbf{C}_k$ to get the local soft labels, and send it to the server. The server will calculate the average soft label of all clients except the current one, and send the result y_{dis}^k representing the ensemble knowledge back. Finally the client calculates the KL-divergence between the output of $\mathbf{C}_k$ and y_{dis}^k as the distillation loss, and plus the loss of classifying the input fake samples. So, $\mathbf{C}_k$ needs to minimize the following objective function at this stage:

$$\mathcal{L}_{dis} = \frac{1}{\|data_{dis}\|}[\sum_{(x_g, \hat{y}) \sim p_{\mathbf{G}_k}} (1 - \alpha) CE(\mathbf{C}_k(x_g), \hat{y}) + \alpha KL(\mathbf{C}_k(x_g), y_{dis}^k)] \quad (5)$$

在蒸馏阶段，每个客户机产生相同的假样本，本地的预测结果和其他客户机的预测结果，计算KL loss，集成知识

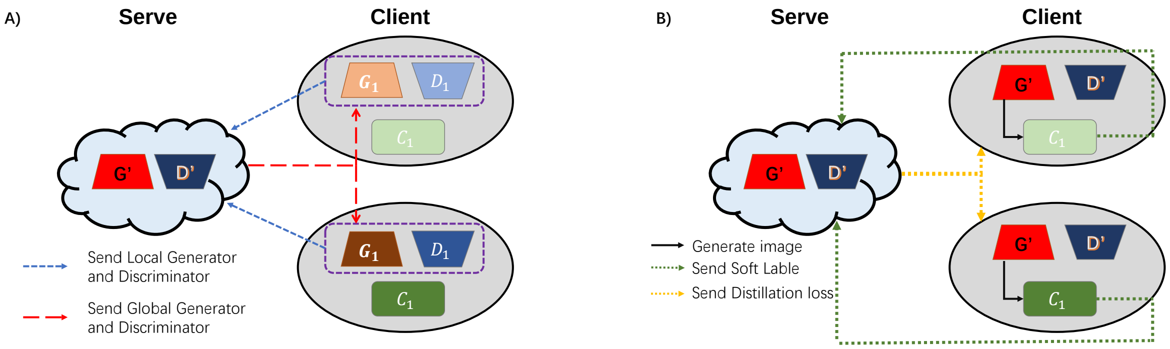

Figure 2: The last two steps of FedDTG training:A) Server Aggregation ; B) Federated Distillation.The shades of the same color represent the personalization models of different clinets and different kinds of lines represent different data transmitted.

where α stands for the weight of knowledge distillation. Here, we prefer to choose a large α value, because the classification generated fake samples has been included in the process of local adversarial trainging.

4 Experiments

4.1 Setup

Dataset: We experiment competing approaches on three classic datasets for deep learning: MNIST [Lecun and Cortes, 2010], EMNIST [Cohen *et al.*, 2017] and Fashion-MNIST [Xiao *et al.*, 2017]. MNIST dataset is composed of 28 x 28 pixels grayscale images with 6,000 training images and 1,000 testing images per number. EMNIST dataset is for character image classification of the same size with 145,600 characters in total. The picture size, number of trainging and testing samples and number of class of FashionMNIST are exactly the same as MNIST, and each class represents different clothing types.

Configurations: Unless otherwise mentioned, we creat 20 user models in total with an active-user ratio $frac = 50\%$. For each local adversarial training, we use a mini batch with the size B = 32. For each federated distillation in FedDTG, we use 10000 generated images. For fair comparison and reflecting the federal environment lacking local data, we use at most $r = 25\%$ training dataset and distribute it to user models to simulate the non-iid federated setting, and use all testing dataset for performance evaluation.

User heterogeneity: We use a Dirichlet distribution **Dir**(α) as model non-iid data distribution, which follow prior arts [Li *et al.*, 2021b]. In Dirichlet distribution, the value of α represents the degree of non-iid.The smaller α, the higher the degree of data heterogeneity, which is different in size, class distributions and the distribution of each class.

Baselines: FedDTG is designed for safer model fusion and more effective non-iid resolution, considering the accuracy of each local model on the test dataset. Thus, in addition to FedAvg [Stephansen *et al.*, 2018], we omit the comparisons to methods for stabilizing local trainging and effective model aggregation. **FedProx** [Li *et al.*, 2020a] limits the local model updates with a proximal term for better local training under heterogeneous systems. **FedDF** is designed for effective model fusion, which is a data-based KD approach [Lin *et al.*, 2020]. We provide part of the training samples without labels as the proxy dataset.

Algorithm 1 FedDTG

Input: local model $\mathbf{G}_k, \mathbf{D}_k, \mathbf{C}_k$
Output: global $\mathbf{G}$ and N local $\mathbf{C}_k$

```
initialization;
for each communication round t = 1,....,T do
    $S_t \leftarrow$ random subset(C fraction) of the K clients.
    for each client $k \in S_t$ in parallel do
        $\theta_g^k \leftarrow \theta_g, \theta_d^k \leftarrow \theta_d$
        for n steps do
            calculate the GAN loss $\mathcal{L}_{\mathbf{G}_k}, \mathcal{L}_{\mathbf{D}_k}, \mathcal{L}_{\mathbf{C}_k}$
            $\theta_g^k \leftarrow \theta_g^k - \nabla\mathcal{L}_{\mathbf{G}_k}$,
            $\theta_d^k \leftarrow \theta_d^k - \nabla\mathcal{L}_{\mathbf{D}_k}$,
            $\theta_c^k \leftarrow \theta_c^k - \nabla\mathcal{L}_{\mathbf{C}_k}$
        end for
        User $Local_k$ sends $\theta_g^k$ and $\theta_d^k$ back to serve.
    end for
    Server updatas $\theta_g \leftarrow \frac{1}{|S_t|}\sum_{k\in S_t}\theta_g^k, \theta_d \leftarrow \frac{1}{|S_t|}\sum_{k\in S_t}\theta_d^k$ and specify noise vector $\hat{z} \sim \mathcal{N}(0,1)$
    for each client $k \in S_t$ in parallel do
        $\theta_g^k \leftarrow \theta_g, \theta_d^k \leftarrow \theta_d$
        generate label vector $\hat{y} \sim \mathcal{U}(1,n)$
        input $\hat{y}$ and $\hat{z}$ to $\mathbf{G}_k$ to generate fake samples $x_g$
        input $x_g$ to $\mathbf{C}_k$ to get soft label $y_c^k$
        send $y_c^k$ back to the serve
    end for
    Server calculates the average soft label for eack client $y_{dis}^k = \frac{1}{|S_t|}\sum_{i\in S_t}^{i\neq k} y_c^i$
    for each client $k \in S_t$ in parallel do
        calculate the distillation loss $L_{dis}$
        $\theta_g^k \leftarrow \theta_g^k - \nabla\mathcal{L}_{dis}$
    end for
end for
return solution
```

本地训练生成器，鉴别器，分类器

聚合生成器和鉴别器

生成虚拟数据，以及各个本地模型对其的预测结果

知识蒸馏

Table 1: Performance overview given different data settings. A smaller α indicates higher heterogeneity and r denotes the sampling ratio of all trainging dataset.

		Average Test Accuracy				
Dataset	Setting	FedAvg	FedProx	FedDF	FedGen	FedDTG
MNIST $r = 25\%$	α =0.05	87.19 ± 1.55	88.64 ± 1.42	88.63 ± 0.70	91.67 ± 0.87	$\mathbf{94.97 \pm 0.98}$
	α =0.1	89.84 ± 0.47	89.79 ± 0.48	89.82 ± 0.33	93.11 ± 0.42	$\mathbf{95.77 \pm 0.43}$
	α =0.4	92.38 ± 0.26	92.83 ± 0.14	93.17 ± 0.11	94.85 ± 0.25	$\mathbf{96.02 \pm 0.11}$
MNIST $r = 10\%$	α =0.05	85.86 ± 2.46	87.38 ± 2.37	88.69 ± 0.64	90.44 ± 0.44	$\mathbf{93.89 \pm 0.99}$
	α =0.1	87.19 ± 1.49	88.63 ± 1.07	88.86 ± 0.51	92.88 ± 0.19	$\mathbf{95.02 \pm 0.47}$
	α =0.4	91.27 ± 0.28	91.52 ± 0.26	92.33 ± 0.17	93.87 ± 0.13	$\mathbf{95.27 \pm 0.24}$
EMNIST $r = 10\%$	α =0.05	62.43 ± 1.07	61.92 ± 1.48	63.57 ± 1.14	66.26 ± 1.36	$\mathbf{73.50 \pm 0.20}$
	α =0.1	66.70 ± 0.96	65.62 ± 0.72	67.01 ± 0.87	70.84 ± 0.67	$\mathbf{74.44 \pm 0.28}$
	α =0.4	71.11 ± 0.32	71.36 ± 0.31	72.68 ± 0.37	75.31 ± 0.52	$\mathbf{75.46 \pm 0.19}$
FashionMNIST $r = 10\%$	α =0.05	68.68 ± 2.12	69.60 ± 1.52	66.92 ± 0.81	69.40 ± 1.03	$\mathbf{77.94 \pm 0.32}$
	α =0.1	71.98 ± 1.04	71.99 ± 1.01	69.15 ± 0.92	73.86 ± 0.33	$\mathbf{79.90 \pm 0.29}$
	α =0.4	78.90 ± 0.67	78.87 ± 0.66	78.51 ± 0.38	78.58 ± 0.36	$\mathbf{80.31 \pm 0.24}$

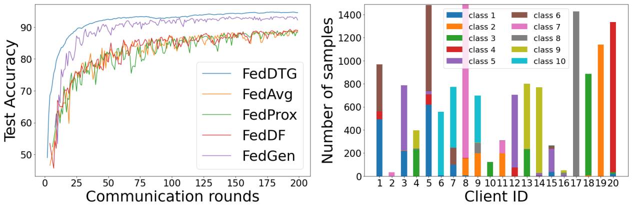

Figure 3: Performance on MNIST under $\alpha = 0.05, r = 25\%$ and Illustration of the number of samples per class allocated to each client

FedGen is a data-free federated distillation method with flexible parameter sharing [Zhu *et al.*, 2021].

4.2 Performance Overview

Impacts of data heterogeneity: As shown in Table 1, FedDTG achieves optimal and stable results under different levels of user heterogeneity. It is worth noting that FedProx and FedDF are optimized based on the average of participant model parameters by FedAvg, and FedGen also uses part of the parameters sharing. However, in FedDTG algorithm, the knowledge of other models is transferred in the co-distillation between clients, and there is no direct parameter averaging of the classsifier model. So, this task is more difficult than other algorithms. FedDTG still performs better compared with other methods, especially when the value of α is small, which represents a higher heterogeneous data distributions. This result is in line with our motivation. With the help of distributed GAN, the generated fake samples limits the local model so that its local objective function does not deviate excessively from the global objective function, and mitigates the discrepancy of latent distributionsacross users together. The use of soft labels on fake samples for knowledge distillation will further alleviate the problem of heterogeneous data distribution. This knowledge is otherwise not accessible by baselines such as FedAvg and FedProx.

FedDF is distilled on the global model obtained by FedAvg, so with the increase of the degree of non-iid, it will have a certain effect, but the improvement is not great. Different from FedDF, the performance gain of our approach is significant compared with FedAvg. This discrepancy shows that our proposed approach of directly doing co-distillation between the clinet models is more effective than the method od fine tuning the global model which is based on FedAvg.
As one of the most competitive baseline, FedGen achieves good results in most cases, but it does not make full use of the method of knowledge distillation. As we can see, our proposed approach outperformances FedGen in most cases, especially when the degree of data heterogeneity is high. In FedDTG, the local client model not only needs to correctly classify the generated images, but also needs to distill the knowledge of other clients, which is more stable than FedGen completely relying on the generator to transmit information.

In addition, compared with other methods, the reduction of training data has little impact on FedDTG. This is because the generated image not only expands the local training data, but also limits the local objective function from excessively deviating from the global objective function.
Learning efficiency: Since FedDTG does not have a global classifier, we use the average accuracy of all client models on the test dataset as the final result. As show in Figure 3, FedDTG has the most rapid and stable learning curves to reach a performance and outperformances other baselines on MNIST under $\alpha = 0.05, r = 25\%$. We show the number of samples per class allocated to each client under this federated setting on the right side of Figure 3. We can see that in this extreme non-iid case, most clients have only one class of images, and the client 2 even has only 32 local training images, which will greatly increase the difficulty of federated training. Although FedGen enjoys a learning effciency higher than other baselines under certain data setting, with the help of classified generated images and co-distillation, our approach has less fluctuation in accuracy during traing.

Table 2: Ablation experiment

	test acc
local training	29.30
w/o co-distillation	92.33
w/o global generator	93.78
FedDTG	**94.97**

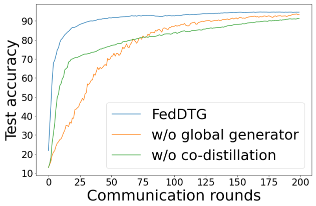

Figure 4: Ablation experiment

4.3 Ablation experiment

We tested the effects of different parts of FedDTG on the final experimental results. In FedDTG, the transmission of information between clients mainly comes from two parts: The first part is the global generator **G**, including the correct classification of the generated images in the Local Adversarial Training stage and the Federated Distillation stage; another part is the average of the soft labels of other clients received from the server, that is, the co-distillation in the Federated Distillation stage.

As shown in Table 2, if these two items are removed together, there will be no interaction between clients, and the final result depends entirely on the quality of the Local data. If co-distillation is used alone, the whole method is similar to the traditional federated distillation after removing the global teacher model. If the global generator **G** is used alone, it is equivalent to directly expanding the local dataset with the generated images. As shown in Figure 4, adding the global generator can reduce the accuracy fluctuation during training because of the expanded local data and adding co-distillation can accelerate the convergence of the whole algorithm. Both of them have significantly improved the results.

4.4 Personalization performance

To evaluate the personalization performance of all methods in detail, we also distribute the testing dataset to clients with the same **Dir**$(\alpha = 0.05)$ for local performance evaluation. We analyze the testing accuracy of the personalized model owned by each client in Figure 5. For FedDTG, even the worst client achieves an accuracy of 92.15% on local test dataset which is a huge improvement compared with other algorithms. In addition, the test accuracy of all clients in FedDTG is relatively centralized, which once again proves the advantages of global generator and co-distillation.

4.5 Generalization study

We tested the generalization performance of FedDTG under two extremely unbalanced label distribution and quantity distribution. We use 10 users and $r = 10\%$ MNIST training data. In the first case, each user has only one class of data. In the second case, 5 users have 10 samples, and the other 5 users have 600 samples, and each user has the same number of samples belonging to each class.

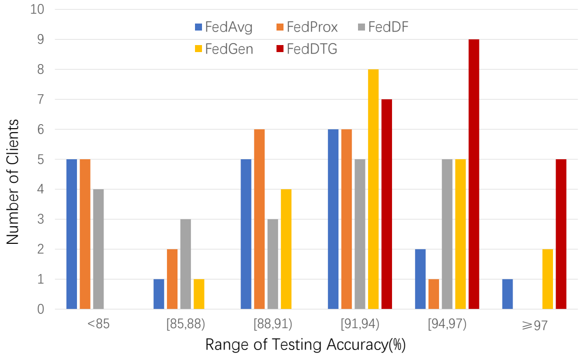

Figure 5: The distribution of the testing accuracy of all clients with MNIST under $\alpha = 0.05, r = 25\%$

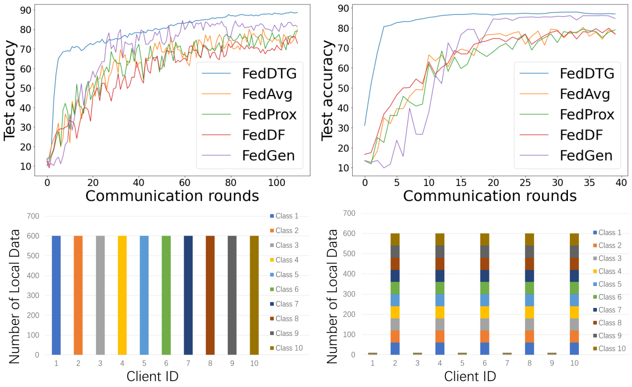

Figure 6: Performance on two extremely unbalanced label distribution and quantity distribution

As shown in Figure 6, the stability of FedDTG has obvious advantages over other algorithms, which once again verifies the role of the global generator **G**. Moreover, the co-distillation between clients can make them obtain the global aggregation knowledge faster which make our method reach the convergence state faster.

5 Conclusion

In this paper, we tackle the challenging problem of personalized cross-silo federated learning and develop FedDTG that introduce a novel distributed GAN to help fedrated co-distillation between clients without infringing their data privacy. We empirically demonstrate that this mechanism can significantly get better performance faster and more stable.

References

[Cohen *et al.*, 2017] G. Cohen, S. Afshar, J. Tapson, and A. V. Schaik. Emnist: Extending mnist to handwritten letters. In *International Joint Conference on Neural Networks*, 2017.

[Goodfellow *et al.*, 2014] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Advances in Neural Information Processing Systems*, 3:2672–2680, 2014.

[Hardy *et al.*, 2019] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. MD-GAN: multi-discriminator generative adversarial networks for distributed datasets. In *2019 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2019, Rio de Janeiro, Brazil, May 20-24, 2019*, pages 866–877. IEEE, 2019.

[Hinton *et al.*, 2015] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[Johnson and Zhang, 2013] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *News in Physiological Sciences*, 1(3):315–323, 2013.

[Karimireddy *et al.*, 2020] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR, 2020.

[Lecun and Cortes, 2010] Y. Lecun and C. Cortes. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 2010.

[Li and Wang, 2019] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *CoRR*, abs/1910.03581, 2019.

[Li *et al.*, 2020a] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In Inderjit S. Dhillon, Dimitris S. Papailiopoulos, and Vivienne Sze, editors, *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*. mlsys.org, 2020.

[Li *et al.*, 2020b] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[Li *et al.*, 2021a] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *CoRR*, abs/2102.02079, 2021.

[Li *et al.*, 2021b] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *CoRR*, abs/2102.02079, 2021.

[Li *et al.*, 2021c] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10713–10722. Computer Vision Foundation / IEEE, 2021.

[Lin *et al.*, 2020] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[Polyak and Juditsky, 2006] B. T. Polyak and AB Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 2006.

[Stephansen *et al.*, 2018] J. B. Stephansen, A. N. Olesen, M. Olsen, A. Ambati, Eileen B. Leary, Hyatt E. Moore, Oscar Carrillo, Ling Lin, Fang Han, and Han Yan. Neural network analysis of sleep stages enables efficient diagnosis of narcolepsy. *Nature Communications*, 9(1), 2018.

[Vandenhende *et al.*, 2019] Simon Vandenhende, Bert De Brabandere, Davy Neven, and Luc Van Gool. A three-player GAN: generating hard samples to improve classification networks. In *16th International Conference on Machine Vision Applications, MVA 2019, Tokyo, Japan, May 27-31, 2019*, pages 1–6. IEEE, 2019.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Yin *et al.*, 2021] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 16337–16346. Computer Vision Foundation / IEEE, 2021.

[Yonetani *et al.*, 2019] Ryo Yonetani, Tomohiro Takahashi, Atsushi Hashimoto, and Yoshitaka Ushiku. Decentralized learning of generative adversarial networks from multi-client non-iid data. *CoRR*, abs/1905.09684, 2019.

[Zhu *et al.*, 2021] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12878–12889. PMLR, 2021.