FedCG: Leverage Conditional GAN for Protecting Privacy and Maintaining Competitive Performance in Federated Learning

Yuezhou Wu[1,2*], **Yan Kang**[2*†], **Jiahuan Luo**[2], **Yuanqin He**[2], **Lixin Fan**[2], **Rong Pan**[1†] and **Qiang Yang**[2,3]
[1]Sun Yat-sen University, China
[2]WeBank, China
[3]Hong Kong University of Science and Technology, China
wuyzh26@mail2.sysu.edu.cn, {yangkang, jiahuanluo, yuanqinhe, lixinfan}@webank.com,
panr@sysu.edu.cn, qyang@cse.ust.hk

Abstract

Federated learning (FL) aims to protect data privacy by enabling clients to build machine learning models collaboratively without sharing their private data. Recent works demonstrate that information exchanged during FL is subject to gradient-based privacy attacks and, consequently, a variety of privacy-preserving methods have been adopted to thwart such attacks. However, these defensive methods either introduce orders of magnitudes more computational and communication overheads (e.g., with homomorphic encryption) or incur substantial model performance losses in terms of prediction accuracy (e.g., with differential privacy). In this work, we propose FEDCG, a novel federated learning method that leverages conditional generative adversarial networks to achieve high-level privacy protection while still maintaining competitive model performance. FEDCG decomposes each client's local network into a private extractor and a public classifier and keeps the extractor local to protect privacy. Instead of exposing extractors, FEDCG shares clients' generators with the server for aggregating clients' shared knowledge aiming to enhance the performance of each client's local networks. Extensive experiments demonstrate that FEDCG can achieve competitive model performance compared with FL baselines, and privacy analysis shows that FEDCG has a high-level privacy-preserving capability.

1 Introduction

Deep neural networks (DNN) have achieved dramatic success in many areas, including computer vision, natural language processing, and recommendation systems. Their success largely depends upon the availability of a wealthy amount of training data. In many real-world applications, however, training data is typically distributed across different organizations, which are unwilling to share their data because of privacy and regulation concerns directly. To alleviate these concerns, federated learning (FL) [McMahan *et al.*, 2017] is proposed to enable multiple clients to collaboratively build DNN models without sharing clients' private data.

However, recent works have empirically demonstrated that the classic federated averaging method (FedAvg [McMahan *et al.*, 2017]) and its variants (e.g., FedProx [Li *et al.*, 2020]) are vulnerable to gradient-based privacy attacks such as the deep leakage from gradients (DLG) [Zhu and Han, 2020], which is able to reconstruct the original data of clients from publicly shared gradients and parameters. Varieties of technologies have been leveraged to further improve the privacy of FL, the most popular ones are homomorphic encryption (HE) [Gentry and others, 2009] and differential privacy (DP) [Dwork *et al.*, 2014]. HE provides a high-level security guarantee by encrypting exchanged information among clients. Nonetheless, its extremely high computation and communication cost make it unsuitable to DNN models. While DP imposes a low complexity on FL, it causes precision loss and still suffers from data recovery attacks. To prevent data leakage and still enjoy the benefits of FL, FEDSPLIT [Gu *et al.*, 2021] combining split learning [Gupta and Raskar, 2018] and FL proposes to split a client's network into private and public models, and protect privacy by hiding private model from the server. However, FEDSPLIT experiences a non-negligible performance drop because the private models are not engaged in the FL.

In this work, we propose FEDCG, a novel federated learning method that leverages conditional generative adversarial networks [Mirza and Osindero, 2014] to achieve high-level privacy protection resisting DLG attack while still maintaining competitive model performance compared with baseline FL methods. More specifically, FEDCG decomposes each client's local network into a *private* extractor and a *public* classifier, and keeps the extractor local to protect privacy. The novel part of FEDCG is that FEDCG shares clients' generators in the place of extractors with the server for aggregating clients' shared knowledge aiming to enhance model performance. This strategy has two immediate advantages. First, the possibility of clients' data leakage is significantly reduced because no model that directly contacts with original data is exposed, as compared to FL methods in which the server

*Co-first authors contributed equally
†Contact Author

has full access to clients' local networks (e.g., FEDAVG and FEDPROX). Second, the server can aggregate clients' generators and classifiers using knowledge distillation [Hinton *et al.*, 2015] without accessing any public data.

The main contributions of this work are:

- To the best of our knowledge, this work is the first attempt to integrate cGAN into FL aiming to protect clients' data privacy while enabling clients to have competitive model performance.
- Extensive experiments demonstrate that FEDCG can achieve competitive performance and provide privacy analysis to prove that FEDCG is superior in privacy protection compared with baseline FL methods.

2 Related Work

2.1 Federated Learning

Federated learning (FL) is a distributed machine learning paradigm that enables clients (devices or organizations) to train a machine learning model collaboratively without exposing clients' data. The concept of FL was first proposed by [McMahan *et al.*, 2017]. Further, many serious challenges emerge with the development of FL. Particularly, the "naked" FL methods without any privacy protection mechanism are proven to be vulnerable to data recovery attacks such as deep leakage [Zhu and Han, 2020] and model inversion [Fredrikson *et al.*, 2015]. Therefore, a wealth of technologies have been proposed to improve the privacy of FL. The most popular ones are homomorphic encryption (HE) and differential privacy (DP). However, HE is extremely computationally expensive, while DP suffers from non-negligible precision loss. Another school of FL [Gupta and Raskar, 2018; Gu *et al.*, 2021] tries to strike a balance between privacy and efficiency by splitting a neural network into private and public models and sharing only the public one.

2.2 DLG in Federated Learning

Federated learning is proposed to protect data privacy by keeping private data localized and sharing only model gradients or parameters. However, recent research on *Deep Leakage from Gradients* (DLG) demonstrates [Zhu and Han, 2020] that shared gradients can actually leak private training data. Particularly, DLG can achieve pixel-wise level data recovery without any assistance information. The follow-up work [Geiping *et al.*, 2020] further extends DLG to more realistic settings where gradients are averaged over several iterations or several images and shows that users' privacy is not protected in these settings. GRNN [Ren *et al.*, 2021] improves DLG by recovering fake data and labels through a generative model instead of regressing them directly from random initialization.

2.3 GAN In Federated Learning

Recent research works that utilize GAN in the FL setting focus mainly on two lines of works: One is leveraging GAN to perform data recovery attacks. [Hitaj *et al.*, 2017] assumes a malicious client that utilizes the shared model as the discriminator to train the generator in a GAN. Then, the trained generator is used to mimic the training samples of the victim client. Another is to train high-quality GAN across distributed data sources under privacy, efficiency, or heterogeneity constraints. MD-GAN [Hardy *et al.*, 2019] proposes a system that the server hosts the generator while each client hosts a discriminator. The discriminators communicate with each other in a peer-to-peer fashion for improving computational efficiency. FedGAN [Rasouli *et al.*, 2020] trains a GAN across Non-IID data sources in a communication efficient way, but it may produce biased data.

3 Proposed Method

3.1 Problem Formulation

In this work, we consider typical federated learning (FL) setting that includes a central server and N clients holding private datasets $\{\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_n\}$. These private datasets share the same feature space but have different sample spaces.

Each client i has a classification network parameterized by $\theta_{E_i,C_i} := [\theta_{E_i}; \theta_{C_i}]$ consists of a feature extractor $E_i : \mathcal{X}_i \to \mathbb{R}^d$ parameterized by θ_{E_i}, and a classifier $C_i : \mathbb{R}^d \to \mathbb{R}^c$ parameterized by θ_{C_i}, where d is the feature dimension and c is the number of classes. For protecting privacy and maintaining competitive performance, each client is provided with a conditional GAN (cGAN) consisting of a generator $G_i : \mathcal{Z} \to \mathbb{R}^d$ parameterized by θ_{G_i}, and a discriminator $D_i : \mathbb{R}^d \to \mathcal{I}$ parameterized by θ_{D_i}, where $\mathcal{Z}$ is the Gaussian distribution and $\mathcal{I}$ indicates a single scalar in the range of [0, 1]. The training procedure of the cGAN is performed locally aiming to train the generator G_i to approximate the extractor E_i such that $G_i(z, y)$ captures the distribution of features extracted by $E_i(x|y)$.

As illustrated in Figure 1, the workflow of FEDCG goes as follows: in each FL communication round, each client i uploads its G_i and C_i to the server once the local training is completed while keeps the E_i and D_i local to strengthen privacy protection. Then, the server applies knowledge distillation to build a global generator G_g and a global classifier C_g. Next, clients download G_g and C_g to replace their corresponding local models and start the next training iteration.

In our FEDCG, clients collaboratively train the global generator and classifier with the help of the central server, while

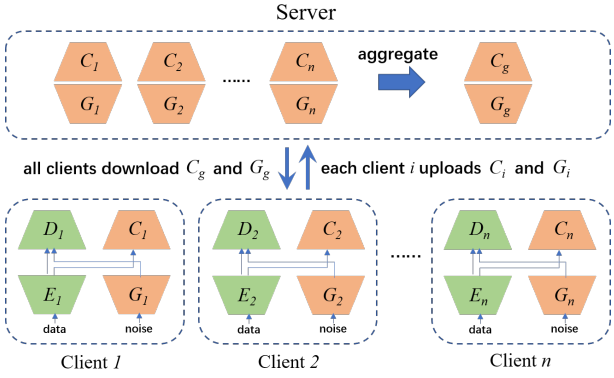


Figure 1: **Overview of FEDCG.**

each client leverages the global generator and global classifier to build a personalized local classification network that can perform well on its local test dataset. We will elaborate on this in the following two sections.

3.2 Two-stage Client Update

The client's local training procedure involves two stages: classification network update and generative network update. Figure 2 illustrates the two stages while Algorithm 1 describes the detailed procedure.

In the classification network update stage (Algo 1, lines 3-9), each client i optimizes its extractor θ_{E_i} and classifier θ_{C_i} by minimizing the classification loss:

$$\mathcal{L}^{cls} = \mathbb{E}_{x,y\sim\mathcal{X}_i}\ \Omega\big(C_i(E_i(x|y;\theta_{E_i});\theta_{C_i}), y\big), \tag{1}$$

where Ω is cross-entropy function. In addition, client i also wants to integrate the shared knowledge embedded in the global generator (aggregated in the previous round) into its local extractor. To this end, it freezes the global generator θ_{G_g} and optimizes its local extractor θ_{E_i} by minimizing the mean-square-error loss as follows:

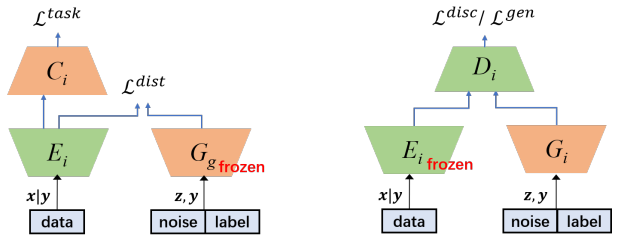
$$\mathcal{L}^{dist} = \mathbb{E}_{x,y\sim\mathcal{X}_i}\mathbb{E}_{z\sim\mathcal{Z}}||E_i(x|y;\theta_{E_i}) - G_g(z,y;\theta_{G_g})||^2. \tag{2}$$

Then, we have the task loss $\mathcal{L}^{task}$, which is the combination of $\mathcal{L}^{cls}$ and $\mathcal{L}^{dist}$, and is formalized by:

$$\mathcal{L}^{task} = \mathcal{L}^{cls} + \gamma\,\mathcal{L}^{dist}, \tag{3}$$

where the γ is a non-negative hyperparameter that adjusts the balance between the two loss terms. In this work, γ gradually increases from 0 to 1 as the global generator becomes more accurate in producing feature representations during training.

In the generative network update stage (Algo 1, lines 10-18), each client i wants to approximate its local generator's output to the feature representations extracted by its local extractor. To this end, it freezes the parameters θ_{E_i} of extractor E_i and conducts a cGAN training procedure to train the generator. More specifically, it samples a mini-batch of training data (x, y) and feeds x to the E_i to obtain the "ground-truth" feature representations h. Then, it randomly generates Gaussian noises z with the same batch size and feeds (z, y) to generator G_i to generate estimated feature representations $\hat{h}$. Next, it feeds h and $\hat{h}$ to discriminator D_i to calculate discriminator loss $\mathcal{L}^{disc}$ and generator loss $\mathcal{L}^{gen}$ according to



(a) Classification Network Update (b) Generative Network Update

Figure 2: Two-stage client update. (a) Classification network update. (b) Generative network update.

Algorithm 1 Two-stage Client Update

Input: clients' datasets $\{\mathcal{X}_i\}_{i=1}^n$; clients' extractors, classifiers, generators and discriminators: $\{E_i(\cdot;\theta_{E_i}), C_i(\cdot;\theta_{C_i}), G_i(\cdot;\theta_{G_i}), D_i(\cdot;\theta_{D_i})\}_{i=1}^n$; global generator $\{G_g(\cdot;\theta_{G_g})$ and global classifier $C_g(\cdot;\theta_{C_g}\})$; learning rate η_1, η_2; local training epoch T

1: Clients receive θ_{G_g} and θ_{C_g} from the server.
2: **for each** client $i = 1, ..., N$ in parallel **do**
3: $\quad\theta_{C_i} \leftarrow \theta_{C_g}$;
4: $\quad$**for** $t \in \{1, ..., T\}$ **do**
5: $\quad\quad$**for all** $x, y \in \mathcal{X}_i$ **do**
6: $\quad\quad\quad$sample z from $N(0, 1)$
7: $\quad\quad\quad\theta_{E_i,C_i} \leftarrow \theta_{E_i,C_i} - \eta_1\nabla_{\theta_{E_i,C_i}}\mathcal{L}^{task}(x, y, z)$
8: $\quad\quad$**end for**
9: $\quad$**end for**
10: $\quad\theta_{G_i} \leftarrow \theta_{G_g}$;
11: $\quad$**for** $t \in \{1, ..., T\}$ **do**
12: $\quad\quad$**for all** $x, y \in \mathcal{X}_i$ **do**
13: $\quad\quad\quad$sample z from $N(0, 1)$
14: $\quad\quad\quad\theta_{D_i} \leftarrow \theta_{D_i} - \eta_2\nabla_{\theta_{D_i}}\mathcal{L}^{disc}(x, y, z)$
15: $\quad\quad\quad\theta_{G_i} \leftarrow \theta_{G_i} - \eta_2\nabla_{\theta_{G_i}}\mathcal{L}^{gen}(y, z)$
16: $\quad\quad$**end for**
17: $\quad$**end for**
18: **end for each**
19: Client i sends θ_{G_i} and θ_{C_i} to server.

(4), and alternatively minimize the two losses to optimize the generator θ_{G_i} and discriminator θ_{D_i}.

$$\begin{aligned}\mathcal{L}^{disc} = \mathbb{E}_{x,y\sim\mathcal{X}_i}\mathbb{E}_{z\sim\mathcal{Z}}\big[&\log\left(1 - D_i(E_i(x|y;\theta_{E_i});\theta_{D_i})\right)\\ &+ \log D_i(G_i(z,y;\theta_{G_i});\theta_{D_i})\big],\\ \mathcal{L}^{gen} = \mathbb{E}_{x,y\sim\mathcal{X}_i}\mathbb{E}_{z\sim\mathcal{Z}}&\log\left(1 - D_i(G_i(z,y;\theta_{G_i});\theta_{D_i})\right).\end{aligned} \tag{4}$$

Once the local training is completed, each client i sends its generator θ_{G_i} and classifier θ_{C_i} to the server for aggregation.

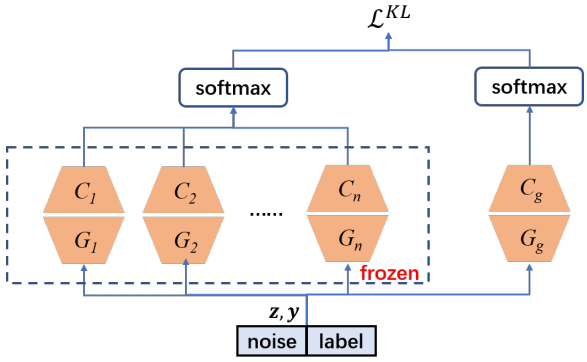


Figure 3: Server aggregation. The server generates Gaussian noise z and class label y as the inputs of clients' $\{G_i\}_{i=1}^n$ and global G_g, and it optimizes θ_{C_g} and θ_{C_g} by minimizing the KL divergence between the distribution ensembled from $\{C_i\}_{i=1}^n$ and the one outputted from C_g.

Algorithm 2 Server Aggregation

Input: size of data set $|\mathcal{X}_i|_{i=1}^n$; clients' generators and classifiers $\{G_i(\cdot;\theta_{G_i}), C_i(\cdot;\theta_{C_i})\}_{i=1}^n$; learning rate η_3; training iteration T, sample batch size B.

1: Server receives $\{\theta_{G_i}\}_{i=1}^n$ and $\{\theta_{C_i}\}_{i=1}^n$ from all clients.
2: $\{\theta_{G_g}, \theta_{C_g}\} = \sum_{i=1}^n \frac{|\mathcal{X}_i|}{\sum_{i=1}^n |\mathcal{X}_i|}\{\theta_{G_i}, \theta_{C_i}\}$
3: **for** $t \in \{1, ..., T\}$ **do**
4: Sample (z, y), where $z \sim N(0, 1)$ and $y \sim \mathcal{U}(0, c)$.
5: $\{\theta_{G_g}, \theta_{C_g}\} \leftarrow \{\theta_{G_g}, \theta_{C_g}\} - \eta_3 \nabla_{\theta_{G_g, C_g}} \mathcal{L}^{\text{KL}}(y, z)$
6: **end for**
7: Server sends θ_{G_g} and θ_{C_g} to all clients.

3.3 Server Aggregation

The server utilizes knowledge distillation (KD) [Lin *et al.*, 2020] to perform the aggregation. One major advantage of FEDCG over existing KD methods in FL is that FEDCG does not require the server to access any public data in order to perform distillation.

Figure 3 illustrates the server aggregation while Algorithm 2 describes the detailed procedure. When the server receives generators $\{G_i\}_{i=1}^n$ and classifiers $\{C_i\}_{i=1}^n$ uploaded by clients, it initializes parameters of the global generator θ_{G_g} and global classifier θ_{C_g} by weighted averaging $\{\theta_{G_i}\}_{i=1}^n$ and $\{\theta_{C_i}\}_{i=1}^n$. For distillation (Algo 2, lines 3-6), the server first generates a mini-batch of training data (z, y), where labels y are sampled from uniform distribution $\mathcal{U}(0, c)$ and noises z are sampled from Gaussian distribution $N(0, 1)$. Then, according to (5) the server feeds (z, y) to all generators and calculates class probability distributions $\mathcal{P}_c(y, z)$ and $\mathcal{P}_s(y, z)$, the former is ensembled from clients' classifiers while the latter is from the global classifier. Next, the server optimizes the global classifier θ_{C_g} and generator θ_{G_g} by minimizing the KL divergence between two distributions, according to (6).

$$\begin{aligned} \mathcal{P}_c(y, z) &= \sigma\big(\sum_{i=1}^n \frac{|\mathcal{X}_i|}{\sum_{i=1}^n |\mathcal{X}_i|} C_i(G_i(y, z; \theta_{G_i}); \theta_{C_i})\big), \\ \mathcal{P}_s(y, z) &= \sigma\big(C_g(G_g(y, z; \theta_{G_g}); \theta_{C_g})\big), \end{aligned} \tag{5}$$

$$\mathcal{L}^{\text{KL}} = \mathbb{E}_{y\sim\mathcal{U}} \mathbb{E}_{z\sim\mathcal{Z}} \text{KL}(\mathcal{P}_c(y, z), \mathcal{P}_s(y, z)). \tag{6}$$

where $|\cdot|$ denotes the size of data set, KL indicates Kullback–Leibler divergence and σ is the softmax function. Once the server aggregation is completed, the server dispatches the global generator θ_{G_g} and global classifier θ_{C_g} to all clients.

3.4 Privacy Analysis

In the literature, there are a variety of data recovery methods, among which DLG [Zhu and Han, 2020] is able to achieve exact pixel-wise level data revealing. In this work, we consider the *semi-honest* scenario where the server follows the FL protocol but utilizes DLG to recover original data from a victim client. The sizes of private data can be sent to the server as hyperparameters as assumed in the conventional FL. We evaluate the privacy-preserving capability of FEDCG by comparing the quality of image data recovered in FEDCG, FEDAVG and FEDSPLIT. (7) shows the DLG loss $\mathcal{L}^{dlg}$ for the semi-honest server recovering the real data of victim client i.

$$\mathcal{L}^{dlg} = ||\nabla_\theta \mathcal{L}^{cls}(x_i) - \nabla_\theta \mathcal{L}^{cls}(\tilde{x})||^2, \tag{7}$$

where x_i is the real data of victim client i while $\tilde{x}$ is the variable to be trained to approximate x_i by minimizing DLG loss $\mathcal{L}^{dlg}$, which is the distance between $\nabla_\theta \mathcal{L}^{cls}(x_i)$ and $\nabla_\theta \mathcal{L}^{cls}(\tilde{x})$. The former is observed gradients of $\mathcal{L}^{cls}$ (see (1)) w.r.t. model parameters θ for the real data x_i, while the latter is estimated gradients for $\tilde{x}$. For FEDAVG, the server knows the full network of client i, thus $\theta := \theta_{E_i, C_i}$. While for FEDSPLIT, the server only knows the classifier, and thus $\theta := \theta_{C_i}$. Although subsequent research works on DLG generally employs cosine similarity as the loss function, the image reconstruction quality of MSE is more satisfactory for LeNet networks [Geiping *et al.*, 2020].

Similar to FEDSPLIT, FEDCG hides private extractors from the server for protecting privacy. Nonetheless, FEDCG shares clients' generators with the server for aggregating shared knowledge. Thus, FEDCG has auxiliary information on extractors' output distribution estimated by generators. We define the DLG loss for FEDCG as follows:

$$\begin{aligned} \mathcal{L}^{dlg}_{\text{FEDCG}} = ||\nabla_{\theta_{C_i}} \mathcal{L}^{cls}(x_i) - \nabla_{\theta_{C_i}} \mathcal{L}^{cls}(\tilde{x})||^2 + \alpha \sum_c \\ (||mean(\tilde{E}(\tilde{x})^c) - \mu^c||^2 + ||var(\tilde{E}(\tilde{x})^c) - \sigma^c||^2), \end{aligned} \tag{8}$$

where $\tilde{E}$ is the estimated extractor whose parameters are not known by the server, and thus it is randomly initialized. The second optimization term of $\mathcal{L}^{dlg}_{\text{FEDCG}}$ aligns the statistical information of features between the estimated extractor $\tilde{E}$ of the server and the shared generator of victim client G_i. μ^c and σ^c are the mean and standard deviation on each channel c of features generated by G_i. We will quantitatively measure privacy-preserving capabilities of FEDAVG, FEDSPLIT and FEDCG according to (7) and (8) in the next section.

4 Experiments

In this section, we compare the performance of our proposed FEDCG against FL baselines and evaluate the privacy-preserving capability of FEDCG.

4.1 Experiment Settings

Model Architecture

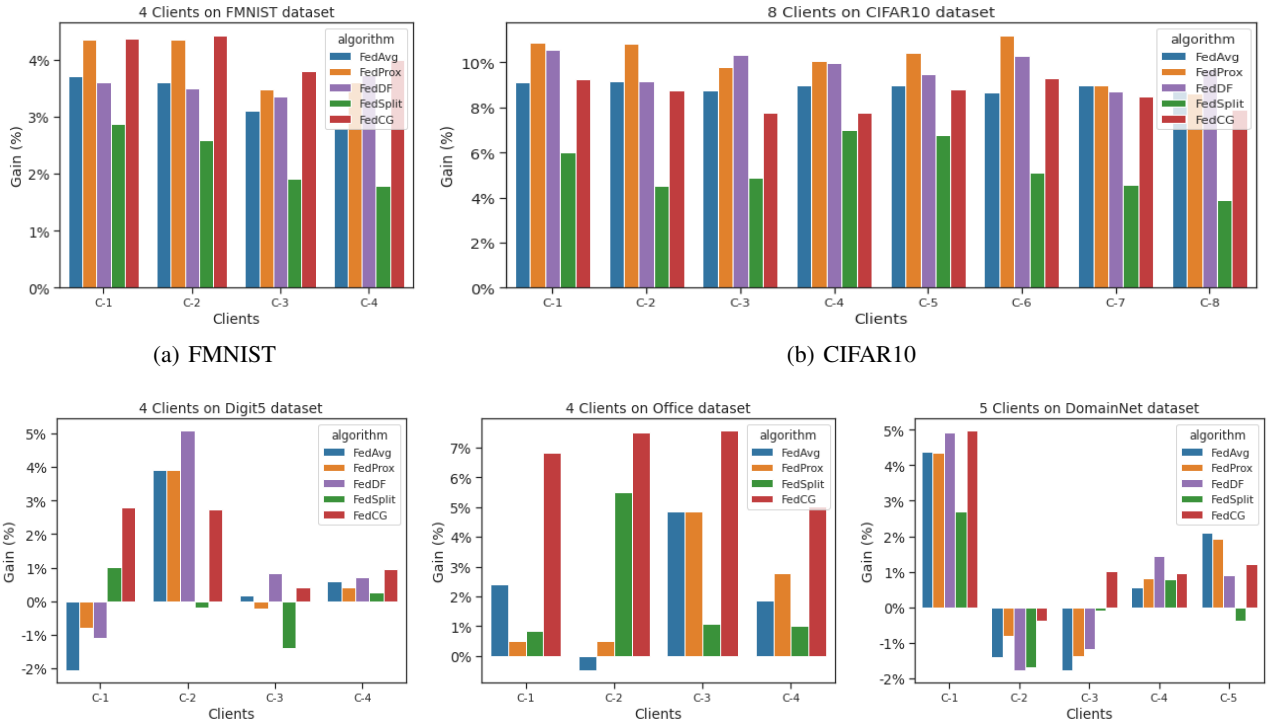
LeNet5 [LeCun *et al.*, 1998] is used as the backbone network for image classification tasks in FL system. The first 2 convolutional layers of LeNet are regarded as *private* extractor, while the latter 3 fully connected layers are regarded as *public* classifier. Our CGAN architecture is a modified version of DCGAN [Radford *et al.*, 2015], in which the size and stride of convolution kernels are adjusted to match the output dimensions of the extractor and generator.

Datasets

We evaluate model performance of clients on 5 image datasets: FMNIST, CIFAR10, Digit5 [Peng *et al.*, 2019], Office-Caltech10 [Gong *et al.*, 2012], and DomainNet [Peng *et al.*, 2019]. MNIST and CIFAR10 simulate the **IID setting**

Method	IID			non-IID		
	FMNIST (4)	CIFAR10 (4)	CIFAR10 (8)	Digit5 (4)	Office (4)	Domainnet (5)
LOCAL	79.66±0.55	40.90±0.92	40.65±0.84	83.09±0.42	60.61±0.78	48.33±0.27
FEDAVG	82.97±0.67	47.23±0.25	49.61±0.65	83.73±0.55	62.76±0.76	49.11±0.47
FEDPROX	83.61±0.64	**49.19**±0.89	**50.76**±0.26	83.92±0.85	62.99±1.07	49.32±0.53
FEDDF	83.20±0.57	47.88±0.62	50.43±0.49	84.48±0.28	*	49.21±0.42
FEDSPLIT	81.95±0.43	44.67±0.64	46.00±0.78	82.96±0.42	62.71±0.88	48.61±0.49
FEDGEN	81.66±0.46	44.98±0.49	45.57±0.59	82.55±0.65	62.70±1.05	47.86±0.64
FEDCG (ours)	**83.81**±0.28	47.52±0.68	49.15±0.48	**84.82**±0.40	**67.34**±0.83	**49.90**±0.18

Table 1: Comparison of FEDCG with baselines in terms of top-1 test accuracy. Results reported in bold are the best performance. * indicates no results is measured. The number in the parentheses indicates the number of clients.



(a) FMNIST (b) CIFAR10

(c) Digit5 (d) Office (e) DomainNet

Figure 4: Accuracy gains achieved by FEDAVG, FEDPROX, FEDDF, FEDSPLIT, and FEDCG (red) over LOCAL of each client on all 5 datasets. The vertical axis is the performance difference in terms of accuracy (%). A positive (negative) gain means FL methods achieves better (worse) than the LOCAL model.

(independent identically distributed). While Digit5, Office-Caltech10, and DomainNet comprise data from multiple domains, and thus they naturally form the **Non-IID setting**. Digit5 is a collection of 5 benchmarks for digit recognition, namely *MNIST*, *Synthetic Digits*, *MNIST-M*, *SVHN*, and *USPS*. Office-Caltech10 contains 10 Office Supplies from 4 domains: *Amazon*, *DSLR*, *Webcam*, and *Caltech*. DomainNet comprises of 6 domains: *Painting*, *Clipart*, *Infograph*, *Quickdraw*, *Real* and *Sketch*.

Baselines

We chose 5 FL baselines from two categories of FL methods. The first category includes FEDAVG [McMahan *et al.*, 2017], FEDPROX [Li *et al.*, 2020] and FEDDF [Lin *et al.*, 2020], in which clients share their full networks with the server. The second one includes FEDSPLIT [Gu *et al.*, 2021] and FEDGEN [Zhu *et al.*, 2021], in which clients share only their public classifiers. FEDAVG is the most widely used FL method. FEDPROX introduces a proximal term in the local objective to regularize the local model training. FEDDF utilizes knowledge distillation to aggregate local models on the server leveraging unlabeled public data. FEDSPLIT shares only the public classifier of the local network to protect privacy. FEDGEN employs knowledge distillation to train a global generator, which in turn helps clients train their local networks. In this work, we implement FEDGEN based on FEDSPLIT, in which only the public classifiers of local networks are shared. We also consider the client's local network trained solely based

Dataset	Metric	FEDAVG	FEDAVG (0.001)	FEDAVG (0.1)	FEDSPLIT	FEDCG
CIFAR10	Accuracy(%)	47.23±0.43	37.47±1.19	35.43±0.62	44.67±0.64	**47.52**±0.68
	PSNR(dB)	24.14±0.22	20.30±0.27	6.81±0.11	**6.23**±0.02	8.99±0.05
DIGIT	Accuracy(%)	83.73±0.55	68.55±1.45	64.34±1.47	82.96±0.42	**84.82**±0.40
	PSNR(dB)	26.82±0.33	20.20±0.89	6.32±0.07	**5.47**±0.12	7.85±0.11
OFFICE	Accuracy(%)	62.76±0.76	40.58±2.02	32.72±0.71	62.71±0.88	**67.34**±0.83
	PSNR(dB)	23.14±0.24	18.78±0.08	6.38±0.07	**5.61**±0.10	7.57±0.11

Table 2: Comparison of FEDAVG, FEDSPLIT and FEDCG in terms of model performance and privacy-preserving capability. The numerical number in the parentheses indicates the noise level σ^2.

on local data as a baseline and denote it as LOCAL.

Configurations

We perform 100 global communication rounds and 20 local epochs with a batch size of 16. All experiments use the Adam optimizer with a learning rate of 3e-4 and a weight decay of 1e-4. For FEDPROX, we tried proximal term factor in the range of $\{0.001, 0.01, 0.1, 1\}$ and picked the best one. FEDGEN, FEDDF and FEDCG perform 2000 iterations in the server for model fusion with a batch size of 16.

We consider each domain as a client for Digit5, Office-Caltech10 and DomainNet except that *MNIST* and *Painting* are held out as distillation data for FEDDF. We consider 4- and 8-client scenarios for CIFAR10 and consider a 4-client scenario for FMNIST. For FMNIST, CIFAR10, and Digit5, we randomly sample 2000 images for each client as the local training set. For Office-Caltech10 and DomainNet, 50% of the original data of each domain is used as the local training set. We use validation and test datasets on clients to report the best test accuracy over 5 different random seeds. We also leverage *diversity* loss from [Mao *et al.*, 2019] to improve the stability of the generator.

4.2 Experiment Results

Performance Evaluation

We evaluate FEDCG's performance by first comparing its averaged clients' accuracy with those of 6 baselines on 5 datasets for both IID and non-IID settings. Table 1 shows that FEDCG achieves the best accuracy in 4 out of 6 scenarios, demonstrating its competitive performance. Specifically, FEDCG achieves the best accuracy in all 3 non-IID scenarios. In particular, it outperforms the-next-best-performing FEDPROX by 4.35% on Office. In IID scenarios, FEDAVG, FEDPROX and FEDDF have an edge in that they aggregate full local networks while there is no negative transfer effect caused by data heterogeneity. As a result, they perform better than FEDCG overall, on CIFAR10 particularly. However, they are vulnerable to DLG attacks, discussed in the next section.

Because the goal of FEDCG is to improve the performance of each client's personalized local network validating on local test data, we further compare the accuracy gains between FEDCG and 5 FL baselines over the LOCAL. In IID scenarios, all FL methods outperform LOCAL on all clients by large margins, as shown in Figures 4(a) and 4(b). Particularly, FEDCG performs best on FMNIST(4) while FEDPROX performs best on CIFAR10(8). In non-IID scenarios, while no FL method can beat LOCAL on every client across all 3 non-IID datasets, FEDCG achieves the best result such that it outperforms LOCAL on 12 out of 13 clients, as shown in Figure 4(c), 4(d) and 4(e). FEDAVG, FEDPROX and FEDDF are not excel in non-IID scenarios as they are in IID scenarios because the average-based global model may be far from client's local optima [Li *et al.*, 2021]. Besides, the distillation dataset leveraged by FEDDF is from a different domain than those of clients, which may have negative impacts on the performance of aggregated global model.

Privacy Evaluation

We calculate Peak Signal-to-Noise Ratio (PSNR) to measure the similarity between original images and images recovered from DLG. PSNR is an objective standard for image evaluation, and it is defined as the logarithm of the ratio of the squared maximum value of RGB image fluctuation over MSE between two images. The higher the PSNR score, the higher the similarity between the two images. We also apply differential privacy (DP) to FEDAVG by adding Gaussian noises to shared gradients. We experiment with two noise levels, $\sigma^2 = 0.1$ and $\sigma^2 = 0.001$.

Table 2 compares FEDCG with FEDSPLIT and FEDAVG in terms of model performance measured by accuracy and privacy-preserving capability measured by PSNR between the ground-truth image and the recovered one using DLG. Compared to FEDAVG and FEDSPLIT, FEDCG achieves competitive accuracies for all three datasets. However, FEDAVG has much higher risk of leaking data information (high PSNR value). Although the data privacy could be better protected by introducing DP to FEDAVG, there is a significant drop (over 20%) in the model performance. On the other hand, FEDSPLIT and FEDCG effectively protect the data privacy (low PSNR value). While achieving similar privacy protection level, FEDCG demonstrates better model accuracies than those of FEDSPLIT for all three datasets by at least 2%.

5 Conclusion

We propose FEDCG, a novel federated learning method that leverages conditional GAN to protect data privacy while maintaining competitive model performance. FEDCG decomposes each client's local network into a private extractor and a public classifier, and keeps the extractor local to protect privacy. It shares clients' generators with the server to aggregate shared knowledge aiming to enhance the performance of clients' local networks. Experiments show that FEDCG has a high-level privacy-preserving capability and can achieve competitive model performance.

Acknowledgments

This work is partially supported by the National Key Research and Development Program of China under grant [2018AAA0101100], the Special Funds for Central Government Guiding Development of Local Science & Technology under grant [2020B1515310019] and the Basic Research Project of Guangzhou, China under grant [202002030300].

References

[Dwork *et al.*, 2014] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9:211–407, 2014.

[Fredrikson *et al.*, 2015] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[Geiping *et al.*, 2020] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients - how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, volume 33, pages 16937–16947. Curran Associates, Inc., 2020.

[Gentry and others, 2009] Craig Gentry et al. *A fully homomorphic encryption scheme*, volume 20. Stanford university Stanford, 2009.

[Gong *et al.*, 2012] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012.

[Gu *et al.*, 2021] Hanlin Gu, Lixin Fan, Bowen Li, Yan Kang, Yuan Yao, and Qiang Yang. Federated deep learning with bayesian privacy. *arXiv preprint arXiv:2109.13012*, 2021.

[Gupta and Raskar, 2018] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[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)*, pages 866–877. IEEE, 2019.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS 2014 Deep Learning Workshop*, 2015.

[Hitaj *et al.*, 2017] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618, 2017.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998.

[Li *et al.*, 2020] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450, 2020.

[Li *et al.*, 2021] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *arXiv preprint arXiv:2102.02079*, 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 *Advances in Neural Information Processing Systems*, volume 33, pages 2351–2363. Curran Associates, Inc., 2020.

[Mao *et al.*, 2019] Qi Mao, Hsin-Ying Lee, Hung-Yu Tseng, Siwei Ma, and Ming-Hsuan Yang. Mode seeking generative adversarial networks for diverse image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282, 2017.

[Mirza and Osindero, 2014] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[Peng *et al.*, 2019] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1406–1415, 2019.

[Radford *et al.*, 2015] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[Rasouli *et al.*, 2020] Mohammad Rasouli, Tao Sun, and Ram Rajagopal. Fedgan: Federated generative adversarial networks for distributed data. *arXiv preprint arXiv:2006.07228*, 2020.

[Ren *et al.*, 2021] Hanchi Ren, Jingjing Deng, and Xianghua Xie. Grnn: Generative regression neural network–a data leakage attack for federated learning. *arXiv preprint arXiv:2105.00529*, 2021.

[Zhu and Han, 2020] Ligeng Zhu and Song Han. Deep leakage from gradients. In *Federated Learning*, pages 17–31. Springer, 2020.

[Zhu *et al.*, 2021] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. *arXiv preprint arXiv:2105.10056*, 2021.