FedRD: Personalized Federated Learning via Representation Distillation

Shuaishuai Zhang
School of Cyber Science and Engineering, Southeast University
Nanjing, China
sszhang@seu.edu.cn

Jie Huang
School of Cyber Science and Engineering,Southeast University
Purple Mountain Laboratories
Nanjing, China
jhuang@seu.edu.cn

Peihao Li
School of Cyber Science and Engineering,Southeast University
Nanjing, China
lipeihao@seu.edu.cn

***Abstract*—Existing Federated Learning (FL) methods are mostly based on the model-averaging scheme, in which the central server averages the model parameters shared by clients to obtain the latest global model. However, the performance of this traditional scheme will degrade seriously in non-i.i.d settings due to the update bias in local training. What's worse, transmitting model parameters leads to high communication overheads and privacy issues. Inspired by the dataset distillation (DD) methods, we propose a novel DD-based FL scheme called FedRD for addressing aforementioned problems. In FedRD, the whole model consists of the local projection model and the global task model. The client firstly uses the personalized projection model to transform local dataset into low-dimensional representations. Then, the synthetic set of data representations is learned via distribution matching. By sharing synthetic representations, clients can transmit the information of local dataset to the central server in a privacy-preserving and communication-efficient way. In the server side, the global task model is trained over the aggregation of synthetic representations. It enables FedRD to avoid the update bias in model-averaging schemes caused by data heterogeneity. Experiments are conducted over four image classification datasets, and the results demonstrate that FedRD outperforms compared methods significantly in terms of model accuracy and communication overhead.**

***Index Terms*—federated learning, dataset distillation, communication efficiency, privacy-preserving.**

I. Introduction

Federated Learning (FL) [1] is proposed as a distributed learning paradigm for training a global model collaboratively in a privacy-preserving way. A typical FL system usually consists of one central server and multiple clients. In each communication round, the clients train models on local private datasets and send the model parameters or updates to the server. The central server averages the messages shared by clients to obtain the latest global model.

However, the traditional FL methods [1]–[6] based on the model-averaging still face three challenges: (1)**Data heterogeneity**. In practice, the training data between clients are highly unbalanced and non-i.i.d. The data heterogeneity will lead to the bias of the model updates in local training and degrade the model performance seriously. Some advanced methods [2]–[6] are proposed to improve the model-averaging scheme by modifying the loss function [2] or using a normalized averaging method to eliminate objective inconsistency [3]. But these improved methods based on model-averaging can only weaken the biases in local model updates rather than avoiding them completely. (2)**Communication overhead**. The transmission of the whole model parameters for many rounds is a huge communication overhead for the devices in real world with limited communication bandwidth. This issue is more evident when taking more complex neural networks as the global model. (3)**Data privacy**. Recent works find that the model-averaging methods are vulnerable to the privacy attacks [7]–[9]. The attackers can steal the private information of the local dataset, even reconstruct the original data points by inferring model parameters shared by clients. It is desirable to propose a new FL scheme to avoid these issues in traditional model-averaging methods.

In this paper, we present a novel FL scheme, **FedRD**, referred to Personalized **Fed**erated Learning via **R**epresentation **D**istillation. Our work is inspired by the emerging dataset distillation (DD) [10]–[15] methods, which try to learn a condensed set of original dataset. The synthetic dataset consits of a fairly small amount of data (*e.g.,* 10 data points per class), but the model trained on the synthetic dataset performs similarly to the model trained on the whole dataset. FedRD aims to transmit local information from clients to the server by sharing synthetic sets, instead of the model parameters in model-averaging methods. The most related work to FedRD is another DD-based method FedDM [16], which performs DD on local devices and sends synthetic data to the server. However, the performance of FedDM is significantly affected by the amount of synthetic **d**ata **p**er **c**lass (dpc). Choosing a larger value of dpc for accuracy gains indicates higher communication overheads. Different from condensing the original dataset in FedDM, FedRD distills the feature representations with smaller dimensional size, which is called representation distillation. Under the same communication overhead, FedRD can achieve much higher accuracies than FedDM by sending more amounts of synthetic data. The representation distillation makes FedRD obtain a better trade-off between the model accuracy and communication overhead.

In detail, FedRD divides the model into two parts: the local projection model and the global task model. The local projection models are personalized for each client and

are trained in the client side. The local dataset is firstly transformed into low-dimensional feature representations by the projection model. Then, the condensed set of representations is synthesized by distribution matching [13] and sent to the server. The global task model is trained over all synthetic representations in the server side. By sharing synthetic representations, clients can transmit the information of local dataset to the central server in a privacy-preserving and communication-efficient way. Training models over all synthetic datasets enables FedRD to avoid the update bias caused by the data heterogeneity.

In summary, our contributions are as follows:

- We present FedRD, a novel FL scheme based on representation distillation, to tackle the data heterogeneity problem. By sharing synthetic data representations, clients in FedRD can transmit the information of local dataset to the central server in a privacy-preserving and communication-efficient way.
- We first propose to synthesize the condensed set of low-dimensional representations of local dataset. By reducing the dimensional size, FedRD can transmit larger amounts of synthetic data, resulting in a significant increase in accuracy without increasing communication overheads.
- We provide the analysis of the communication cost and the privacy of FedRD compared with other FL methods.
- A comprehensive set of experiments are conducted over four standard datasets to validate FedRD. Experiment results demonstrate that FedRD outperforms compared methods significantly on non-i.i.d setting, in terms of the model accuracy and communication overhead.

This paper is organized as follows. In section II, we review the related works. Section III introduces the motivation and the details of our FedRD. In section IV, we analyze the communication cost and the privacy of FedRD. Section V reports the experiment results of FedRD and compared methods. Finally, we make an conclusion in section VI.

II. Related Work

A. Federated Learning

Federated Learning is a decentralized training paradigm for learning a global model collaboratively. Most existing FL methods are based on FedAvg [1], in which the clients conduct model training on local datast and the central server averages the model parameters shared by clients to update the global model. Due to the serious data heterogeneity, the local training can only obtain the minimization of local loss landscape, instead of the global loss landscape. It degrades the performance of model-averaging schemes on non-i.i.d settings seriously. Many works are proposed to improve the local training [2], [5] or global aggregation [3], [17]to learn a better global model. Sharing model parameters also poses high communication overhead and privacy issues [18]. The attackers can conduct privacy attacks [7]–[9] on shared model parameters to infer the private information of local dataset. Applying differential privacy (DP) [19]–[22] is the most common way to preserve privacy in FL.

B. Dataset Distillation

The idea of dataset distillation (DD) is firstly proposed by Wang *et al.* [10]. DD methods aim to synthesize a small number of data points that do not exist in the original dataset, but will, when given to the learning algorithm as training data, approximate the model trained on the original data. After that, much works have been done to improve the effectiveness of the synthetic dataset, such as acquiring soft labels [23], matching the gradients [11] or distribution [13] between the real and fake datasets, incorporating augmentations [12], longrange trajectory matching [14], learning a generative model to synthesize training data [15]. Wang *et al.* [24] also investigates the privacy issues of DD and demonstrates that DD can provide a empirical DP guarantee.

III. Method

A. Problem Statement

We consider a typical personalized FL system, which involves a central server and N clients. The learning model $q_k(\cdot) = \phi_\theta(\varphi_{w_k}(\cdot))$ of client $k \in [N]$ contains two part: the personalized local projection model φ_{w_k} parameterized with w_k and the global task model ϕ_θ parameterized with θ. The parameters w_k of the projection model are different between clients, while all clients share the same parameters θ of the global task model. Each client owns a local dataset $\mathcal{D}_k = \{\boldsymbol{x}, y\}$ and the datasets between clients are non-independently and identically distributed (non-i.i.d). We focus on the image classification tasks in this work. The central server coordinates clients to conduct collaborative learning over $\mathcal{D} \triangleq \cup_{i\in[N]}\mathcal{D}_k$ without exchanging the raw data. The generic form of the personalized FL can be written as

$$\min \frac{1}{N}\sum_{k=1}^{N} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_k}\left[l(q_k(\boldsymbol{x}), y)\right], \tag{1}$$

where the learning model $q_k : \mathbb{R}^d \to \mathcal{Y}$ of the i-th client maps the input data $\boldsymbol{x} \in \mathbb{R}^d$ to predicted labels $q_k(\boldsymbol{x}) \in \mathcal{Y}$ and the loss function l penalizes the distance of $q_k(\boldsymbol{x})$ from its true label y.

B. FedRD Algorithm

1) The overview of FedRD: The overall framework of our FedRD is demonstrated in Fig. 1 and the detailed procedures are showed in Algorithm 1. There are five steps in each round of learning:

Step 1: The server delivers the latest task model ϕ_{θ_r} to all clients. The global task models can be the backbone models used for image tasks (e.g., ConvNet, VGG, ResNet).

Step 2: The clients load the latest task model ϕ_{θ_r} and update the local projection model φ_{w_k} on their private dataset $\mathcal{D}_k$ by minimizing the loss

$$\mathcal{L}_\varphi = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_k}\left[l(\phi_{\theta_r}(\varphi_{w_k}(\boldsymbol{x})), y)\right], \tag{2}$$

where l is the cross-entroy loss function. During the training procedures, the parameters θ_r are freezed in the client side. We keep optimizing the local projection model in each

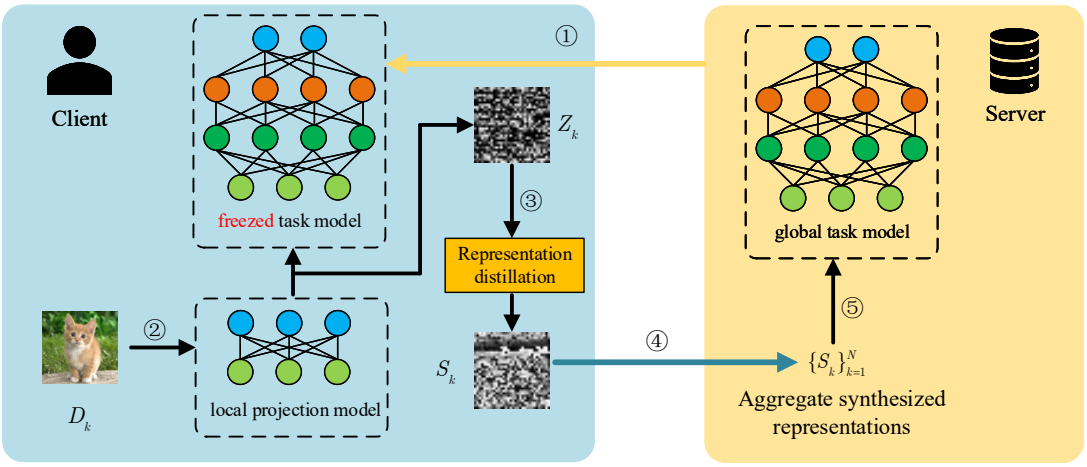


Fig. 1: An overview of our proposed FedRD.

learning round according to the latest task model. It aims to find the representations $\mathcal{Z}_k = \{\varphi_{w_k}(\boldsymbol{x})|\boldsymbol{x} \in \mathcal{D}_k\}$ that are optimal for the succeeding task model. The output dimension of the projection model is d', *i.e.*, $\varphi_{w_k}(\boldsymbol{x}) \in \mathbb{R}^{d'}$, $\boldsymbol{x} \in \mathbb{R}^d$ and $d' \ll d$. For convenience, we resize the representations $\varphi_{w_k}(\boldsymbol{x})$ as an image with the channel, width, height of 1, $\sqrt{d'}$ and $\sqrt{d'}$. Then, the resized representation can be directly input to the succeeding task models.

Step 3: The client conducts representation distillation to synthesize the condensed set of representations $\mathcal{S}_k$, which has the similar distribution to the real representations $\mathcal{Z}_k$. The synthetic representation $\boldsymbol{s} \in \mathcal{S}_k$ has the same dimension to the resized representation, which is $1 \times \sqrt{d'} \times \sqrt{d'}$. The amount of synthetic representations per class is usually set as 10, 50 or 100 [13]. Thus, the data amount $|\mathcal{S}_k|$ is much smaller than $|\mathcal{Z}_k|$. The details of representation distillation will be introduced in part III-B2.

Step 4: Each client sends the synthetic representations $\mathcal{S}_k$ to the central server.

Step 5: The central server collects all synthetic representations and trains the global task model ϕ_{θ_r} over the aggregated dataset $\{\mathcal{S}_k\}_{k=1}^N$.

2) Representation distillation: Inspired by the dataset distillation [10], [11], [13] techniques, we propose representation distillation to synthesize condensed representations of the local private dataset in FL. By sharing synthetic representations, we can transfer the information of local dataset from clients to the central server in a privacy-preserving and communication-efficient way.

Firstly, we exploit the local projection model φ_{w_k} to transform the original data $\boldsymbol{x} \in \mathbb{R}^d$ to the data representations $\boldsymbol{z} = \varphi_{w_k}(\boldsymbol{x}) \in \mathbb{R}^{d'}$, where $d' \ll d$. Similar to the method DM [13], we synthesize the condensed representations $\mathcal{S}$ by matching the distributions. The maximum mean discrepancy (MMD) [25] is commonly applied to estimate the real data distribution in the latent space with a lower dimension:

$$\sup_{\|\psi_\vartheta\|_{\mathcal{H}} \leq 1} (\mathbb{E}[\psi_\vartheta(\mathcal{Z}) - \mathbb{E}[\psi_\vartheta(\mathcal{S})]), \tag{3}$$

where $\mathcal{H}$ is reproducing kernel Hilbert space and ψ_ϑ is the embedding function that maps the input into the laten space. As we have no access to the ground-truth data distributions, we use the empirical estimate of the MMD:

$$\mathbb{E}_{\vartheta \sim P_\vartheta} \| \frac{1}{|\mathcal{D}_k|} \sum_{(\boldsymbol{x},y) \in \mathcal{D}_k} \psi_\vartheta(\varphi_{w_k}(\boldsymbol{x})) - \frac{1}{|\mathcal{S}_k|} \sum_{(\boldsymbol{s},y) \in \mathcal{S}_k} \psi_\vartheta(\boldsymbol{s}) \|^2, \tag{4}$$

where the embedding function ψ_ϑ can be a neural network parameterized with ϑ, which is sampled from the distribution P_ϑ. For computational efficiency, we sample a mini-batch $B_c^{\mathcal{D}_k} \sim \mathcal{D}_k$ and $B_c^{\mathcal{S}_k} \sim \mathcal{S}_k$ for class $c \in [C]$ randomly in each iteration of learning synthetic representations. Following to the work [12], we apply the differentiable Siamese augmentation $\mathcal{A}(\cdot)$ to real data in training. Then, the local projection model φ_{w_k} takes $\mathcal{A}(B_c^{\mathcal{D}_k})$ as input to obtain the mini-batch of real representations. The discrepancy between the distributions of synthetic representations and the real ones is computed for each class c in various embedding spaces by sampling ϑ. We learn the synthetic representation $\mathcal{S}_k$ by minimizing the loss $\mathcal{L}_k$ as follows:

$$\begin{aligned} \mathcal{L}_k = \sum_{c=0}^{C-1} \mathbb{E}_{\vartheta \sim P_\vartheta} \| & \frac{1}{|B_c^{\mathcal{D}_k}|} \sum_{(\boldsymbol{x},y) \in B_c^{\mathcal{D}_k}} \psi_\vartheta(\varphi_{w_k}(\mathcal{A}(\boldsymbol{x}))) \\ & - \frac{1}{|B_c^{\mathcal{S}_k}|} \sum_{(\boldsymbol{s},y) \in B_c^{\mathcal{S}_k}} \psi_\vartheta(\boldsymbol{s}) \|^2. \end{aligned} \tag{5}$$

IV. Analysis

A. Communication costs

Compared with model-averaging methods, transmitting synthetic representations in FedRD can save very much communication costs. For example, we conduct FedRD, FedDM and FedAvg on CIFAR10 dataset. The dimensional size of the original image is $d = 3 * 32 * 32$. We set the data amount of synthetic representations per class as 10 for FedRD and FedDM. The dimensional size of representations $d' = 1 * 32 * 32$ in FedRD. Thus, the total size

Algorithm 1 FedRD: Personalized Federated Learning via Representation Distillation

Require: ϕ_θ: the global task model ϕ_θ parameterized with θ, η: learning rate

Server executes:
for each round $r = 1, \cdots, R$ **do**
  for each client $k = 1, \cdots, K$ **do**
    $\mathcal{S}_k \leftarrow$ClientUpdate(k, θ_r)
  end for
  Update the global task model ϕ_{θ_r} on aggregated data $\{\mathcal{S}_k\}_{k=1}^K$ by SGD with the learning rate η_g
end for

ClientUpdate(k, θ_r):
Update the local projection model φ_{w_k} on local data $\mathcal{D}_k$ by minimizing the loss $\mathcal{L}_\varphi$ in Eq. (2)
Initialize $\mathcal{S}_k$ from random noise or real examples
for each round $t = 1, \cdots, T$ **do**
  Sample $\vartheta \sim P_\vartheta$
  Sample mini-batch pairs $B_c^{\mathcal{D}_k} \sim \mathcal{D}_k$ and $B_c^{\mathcal{S}_k} \sim \mathcal{S}_k$ for every class c
  Compute the loss $\mathcal{L}_k$ in Eq. (5)
  Update $\mathcal{S}_k \leftarrow \mathcal{S}_k - \eta_d \nabla_{\mathcal{S}_k} \mathcal{L}_k$
end for

of the synthetic representations of each client in FedRD is $10 * 10 * 1 * 32 * 32 \approx 1 \times 10^5$. For FedDM, the dimensional size of the synthetic data is $3 * 32 * 32$ and the total size of synthetic set is $10 * 10 * 3 * 32 * 32 \approx 3 \times 10^5$. For the model-averaging method FedAvg, the shared messages are all model parameters. The total size of parameters is approximately 9.2×10^6 for VGG11 and 1.1×10^7. We can find that DD-based methods can save considerable communication costs compared to model-averaging methods. Our proposed FedRD can further reduce the communication cost than FedDM by decreasing the dimensional size of synthetic data.

B. Privacy

FedRD protects the privacy of local data from two aspects: For one thing, the synthetic representations are not real instances of the local dataset. For another, the attackers cannot infer from the representations $\varphi_{w_k}(\boldsymbol{x})$ to obtain the original data $\boldsymbol{x}$, as the local projection model φ_{w_k} is kept local. The adversaries have no access to the architectures and parameters of the projection model. It is an empirical analysis of the privacy-preserving of FedRD.

We can also incorporate DP-SGD [20] into the training procedures to achieve an rigrious (ϵ, δ)-DP guarantee of FedRD, which is defined as follows:

***Definition** 1 ((ϵ, δ)-DP guarantee [19].):* Given two privacy parameters $\epsilon > 0$ and $0 \leq \delta < 1$, a randomized algorithm $\mathcal{M}$ satisfies (ϵ, δ)-DP, if for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$ differing by at most one record, we have

$$\mathbb{P}\left[\mathcal{M}(\mathcal{D}) \in \mathcal{S}\right] \leq e^\epsilon \cdot \mathbb{P}\left[\mathcal{M}(\mathcal{D}') \in \mathcal{S}\right] + \delta \tag{6}$$

for all subset of outputs $\mathcal{S} \in Range(\mathcal{M})$.
Specifically, we add Gaussian noise into the clipped gradients in each iteration of optimizing synthetic data. Between the line 15 and line 16 in Algorithm 1, we process the original gradients as follows:

$$\begin{aligned} \nabla_{\mathcal{S}_k^c} \mathcal{L}_k^c &\leftarrow \nabla_{\mathcal{S}_k^c} \mathcal{L}_k^c / \max(1, \frac{\|\nabla_{\mathcal{S}_k^c} \mathcal{L}_k^c\|_2}{\mathcal{C}}) \\ \nabla_{\mathcal{S}_k^c} \mathcal{L}_k^c &\leftarrow \nabla_{\mathcal{S}_k^c} \mathcal{L}_k^c + \frac{1}{|B_c^{\mathcal{D}_k}|} \mathcal{N}(0, \sigma^2 \mathcal{C}^2 \boldsymbol{I}), \end{aligned} \tag{7}$$

where $\mathcal{C}$ is the clipping threshold and σ is the noise scale. Following FedDM [16], the (ϵ, δ)-DP guarantee of FedRD can be achieved by choosing the values of σ according to Theorem 1.

***Theorem** 1 (DP of FedRD):* Given the synthetic representations $\mathcal{S}$ is initialized from random noise, FedRD trained with DP-SGD can guarantee (ϵ, δ)-DP in FL, with $\sigma \geq \sqrt{\frac{\log(\delta)}{Tq^2 - \epsilon}}$ or $\sigma \geq \sqrt{\frac{2\log(1/\delta)}{\epsilon}}$ if $Tq^2 \leq \frac{\epsilon}{2}$ in each communication round, where T is the total iterations of learning and q is the sampling ratio of the batch data.

V. Experiments

A. Experiments Setup

1) Dataset: The experiments are conducted over four standard image datasets, including Fashion MNIST, SVHN, CIFAR10 and CIFAR100. We simulate the non-i.i.d. setting with Dirichlet distribution [26] by changing the parameter α. A smaller value of α indicates a higher level of the data heterogeneity. By default, we split the whole dataset with $\alpha = 0.1$ for 10 clients. Each client randomly divides the local subset into the training set and the test set in the ratio of 8:2. We report the average accuracies of all clients on their local test set.

2) Baseline methods: We compare FedRD with three model-aggregation based methods: FedAvg [1], FedProx [2] and FedNova [3], as well as a method based on data distillation, FedDM [16].

3) Models: The global task model is a ConvNet, consisting of three convolution layers and one fully connected layer. The three convolution layers all have 128 filters with the kernel size of 3 and are followed with one Batchnorm layer, one ReLu activaion layer and one MaxPooling layer with the kernel size of 2. The local projection model only consists of two convolution. The first layer has 8 filters and the second one has 16 filters. The output dimensions of the local projection model $d' = w * h$, where w and h are the width and height of the input data. The mapping function used in computing MMD takes the same architecture to the task model, except for the last fully connected layer.

4) Hyperparameters and models: The total communication round $R = 20$ for all compared methods. For **DD-based methods**, FedRD and FedDM, the clients conduct $T = 1000$ iterations to learn the synthetic representations in each communication round with a learning rate $\eta_s = 1$. The batch size is 256 for real and synthetic data. The amount

TABLE I: The accuracies of compared methods on four different datasets.

	F-MNIST	SVHN	CIFAR10	CIFAR100
FedAvg	82.25±0.53	85.56±0.52	58.13±1.01	25.67±0.21
FedProx	84.27±0.58	86.26±0.59	56.94±0.48	25.64±0.32
FedNova	81.15±1.38	84.34±0.73	60.28±2.47	24.49±0.91
FedDM	91.35±0.51	85.55±0.92	66.98±0.55	34.93±0.24
FedRD	96.16±0.13	87.86±0.35	78.07±0.82	40.46±0.51

of synthetic data per class of FedDM is 10 for all four datasets. The amount of synthetic representations per class of FedRD is 10 for FashionMNIST and 30 for other three datasets. This setting ensures FedRD and FedDM spend the same communication cost on sending synthetic data. The local projection models in FedRD are trained with the batch size 256 for 10 epochs by Adam with the learning rate $\eta_l = 0.01$. In the server side, the global task model is trained with the batch size 256 for 500 epochs by Adam with the learning rate $\eta_l = 0.01$. For **Model-averaging methods**, FedAvg, FedProx and FedNova, the hyperparameters are chosen for the optimal performance.

B. Model accuracies and communication overhead

Firstly, we evaluate our method FedRD in terms of the model accuracy and the communication overhead. The communication overhead is the total size of messages shared by each client. The mean and variance of the model accuracy are listed in Table I. As we can see, FedRD performs the best among all five methods on four different datasets. For example, FedRD can achieve the accuracy of 78.07±0.82% on CIFAR10, which is 11.09% higher than the DD-based method FedDM and 17.79% higher than the model-averaging method FedNova. Our FedRD and FedDM both outperforms the model-averaging methods FedAvg, FedProx and FedNova. It is because learning the global model over the aggregated synthetic datasets can capture the global information of all clients' datasets. In model-averaging methods, the global models are trained in the client side and can only see the local dataset. The model updates shared from clients are biased seriously, especially in non-i.i.d settings.

Compared with DD-based method FedDM, the additional accuracy gain of FedRD comes from the larger number of synthetic representations shared by clients. In this part, the dimensional size of original images $d = c \times h \times w$ and we set the dimensional size of synthetic representations $d' = 1 \times h \times w$, where c, h and w are the channels, width and height of original images, respectively. The number of channels $c = 3$ on dataset SVHN, CIFAR10 and CIFAR100, making the dimensional size $d = 3d'$. When the communication overhead stay the same, the data amount in the synthetic dataset in FedRD is three times of that in FedDM. A larger amount of data points in the synthetic dataset can better represent the distribution of original dataset. It is the key to improve the model performance trained over synthetic dataset, which has been discussed in previous works [10], [11], [13], [16].

FedRD also converges faster than other compared methods, as showed in Fig. 2. In the first 5 communication rounds, FedRD has already converged well and obtained the highest model accuracy. On CIFAR10, FedRD can achieve the accuracy of 65% after the first communication round, while the accuracies of model-averaging methods are all lower than 30%. The advantages of FedRD are more evident when considering the communication overhead. The model accuracies along with the communication overhead are demonstrated in Fig. 3. After 20 communication rounds, the communication overhead per client in the model-averaging methods is 25MB, which is almost 2.5 times of that in FedRD and FedDM. For FedRD, the model accuracy can hit 75% with only 2.5MB of communication overhead.

C. FedRD with DP guarantee

In this part, we evaluate the performance of FedRD and compared methods under different levels of DP guarantee. Experiments are conducted on CIFAR10 and other settings follow the part V-B. For FedRD and FedDM, we guarantee the (ϵ, δ)-DP with $\sigma \geq \sqrt{\frac{2\log(1/\delta)}{\epsilon}}$ if $Tq^2 \leq \frac{\epsilon}{2}$, which is a loose bound. We set the value of noise scale $\sigma = 1, 3, 5$ and the corresponding privacy budget $\epsilon = 10, 1.11, 0.4$ with $\delta = 10^{-5}$. For model-averaging methods FedAvg, FedProx and FedNova, we use the library [27] to compute the privacy budgets when using DP-SGD. When $\sigma = 0.5, 1.5, 2.5$, the model-averaging methods can achieve similar privacy budgets to FedRD and FedDM. The clipping threshold $\mathcal{C}$ is tuned between $1.0, 2.0, 4.0$. As we can see in Table II, FedRD can outperform other compared methods under different privacy budgets. A smaller value of ϵ indicates a higher level of DP guarantee, while it comes at the cost of more accuracy degradation. Compared with model-averaging methods, the losses of accuracies in DD-based methods are smaller when ϵ decreases from 10 to 0.1. It demonstrates that the DD-based methods are more robustness against the DP noise.

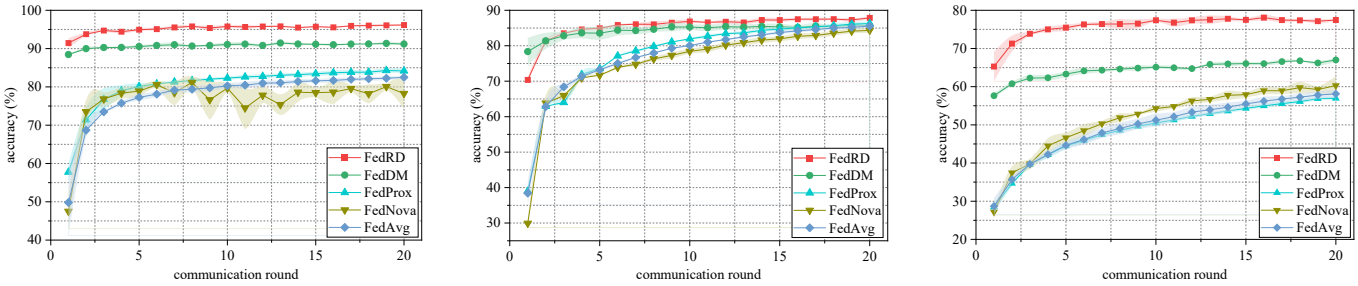
TABLE II: The accuracies of compared methods under different privacy budget ϵ on CIFAR10.

privacy budget	$\epsilon = 10$	$\epsilon = 1.1$	$\epsilon = 0.4$
FedAvg	46.06	40.93	38.98
FedProx	46.53	41.31	37.47
FedNova	50.09	42.91	40.66
FedDM	54.51	53.02	52.82
FedRD	60.96	55.51	54.88

D. Analysis of FedRD

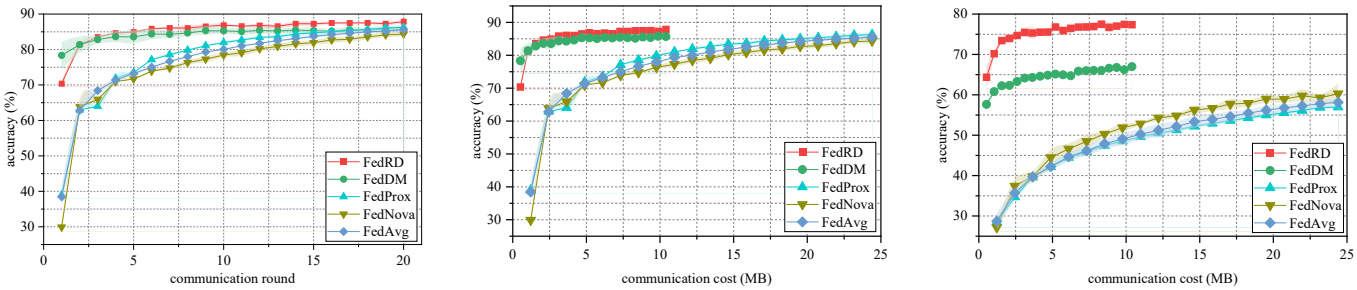
In this subsection, we investigate the model performance of FedRD with different settings, including the initialization of the synthetic representations, the amount of synthetic data per class and the structure of local projection models.

1) Initialization with random noise: We conduct experiments on CIFAR10 to evaluate the model accuracies of FedRD and FedDM on different initializations of the synthetic data. FedDM(real) samples instances from the original dataset



(a) FashionMNIST (b) SVHN (c) CIFAR10

Fig. 2: Test accuracies of compared methods along with the number of communication rounds.



(a) FashionMNIST (b) SVHN (c) CIFAR10

Fig. 3: Test accuracies of compared methods along with the communication overhead.

and FedRD(real) samples instances from the real representations of original dataset through the projection model. FedDM(random) and FedRD(random) both initialize synthetic data based on the standard normal distribution $\mathcal{N}(0,1)$. The experiment results are listed in Table III. We can observe that the *real* settings all perform better than the *random* settings. It is because the real samples are more representive than the synthetic data for the original data distribution. However, applying *random* setting with DP-SGD can preserve the differential privacy of FedRD. In Table III, the *random* settings can outperform the baseline method FedAvg on three datasets, except for SVHN. It demonstrates that FedRD with random initialization can not only protect the clients' privacy, but also improve the model accuracies and reduce the communication overhead significantly.

TABLE III: The accuracies of FedDM and FedRD with random and real initialization on synthetic dataset.

	FashionMNIST	SVHN	CIFAR10	CIFAR100
FedAvg	82.25	85.56	58.13	25.67
FedDM(random)	89.76	80.28	58.04	31.44
FedDM(real)	91.35	85.55	66.98	34.93
FedRD(random)	95.98	82.91	69.84	38.37
FedRD(real)	96.16	87.86	78.07	40.46

2) The amount of synthetic data per class: Experiments are conducted on CIFAR10 to investigate the impact of the amount of synthetic **d**ata **p**er **c**lass (dpc) on the performance of FedRD and FedDM. It can be observed in Fig. 4, the accuracies of FedDM and FedRD both increase as the dpc gets larger. The dimensional size of synthetic data $d = 3\times h\times w$ in FedDM and the dimensional size of synthetic representations $d' = 1\times h\times w$ in FedRD. Thus, the communication overheads of FedRD in cases of $dpc = 9, 15, 60$ are the same to those of FedDM in cases of $dpc = 3, 5, 20$, respectively. When the communication overhead stay same, FedRD can all achieve higher model accuracies than FedDM. Even if FedDM applies $dpc = 20$, the model accuracy is still lower than FedRD($dpc = 9$), while the communication overhead of FedDM is approximately 6.6 times as much as that of FedRD. It confirms the advantages of our proposed FedRD in terms of the model accuracy and the communication efficiency.

3) The dimensional size of representations: In this part, we report the model accuracies and communication costs along with different dimensions d' of the synthetic representations. We change the output dimensions d' of the two-layer projection model from 64 to 2304 (the second row in Table IV). The synthetic representation with dimension d' is resized as an image with channel, width, height of 1, $\sqrt{d'}$ and $\sqrt{d'}$. Other settings follows the part V-B and the dataset is CIFAR10. As we can see in Table IV, the model accuracies increase with the dimensions get larger, while it results in more communication overheads. When $\sqrt{d'}$ gets larger from 8 to 32, the improvement in accuracy is very remarkable. When $\sqrt{d'}$ gets larger from 32 to 48, there is only an accuracy gain of 2%. Thus, setting the value of $d' = 32$ can obtain a good tradeoff between the model accuracy and the communication overhead for CIFAR10 dataset.

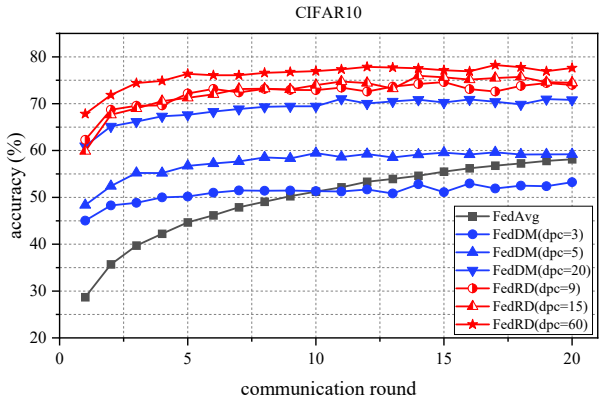


Fig. 4: Test accuracies of FedDM and FedRD with different amounts of synthetic data per class (dpc).

TABLE IV: The accuracies and communication cost (com-cost) of FedDM along with different dimensions d' of representations.

width $\sqrt{d'}$	8	16	24	32	40	48
dimensions d'	64	256	576	1024	1600	2304
com-cost(MB)	1.46	5.84	13.14	23.36	36.5	52.56
accuracy(%)	51.66	71.96	75.06	78.07	78.69	80.05

4) Different local projection models: In this part, we investigate the impact of different local projection models on the performance of FedRD. Experiments are conducted on 3 different datasets, including SVHN, CIFAR10 and CIFAR100. We change the number of convolution layers in the local projection models from 1 to 4. For different cases, the dimensional sizes of the synthetic represents d' always equal to $h \times w$. In Fig. 5, we can observe that the accuracies all increase as the layers get larger. The accuracy gains from the layers are more evident in complex datasets. For example, the model accuracy in the case of 4 layers is approximately 20% higher than that in the case of 1 layer on CIFAR100. Applying a local projection model with more convolution layers can improve the effectiveness of the data representations, while increasing the computing overhead in the client side. Thus, we need to trade off the model performance against the local computing overhead.

VI. Conclusion

In this paper, we propose a novel FL scheme FedRD to tackle the challenges of data heterogeneity, communication overheads and privacy. Different from model-averaging methods, FedRD synthesizes condensed set of data representations on local and send it to the server. The task model is trained on the aggregation of synthetic data to obtain a global view of all clients' dataset. The synthetic representations have a lower dimension size than the original data, which furthermore reduces the communication cost. The experiments results show that FedRD can obtain a better performance in terms of the model accuracy and the communication overhead.

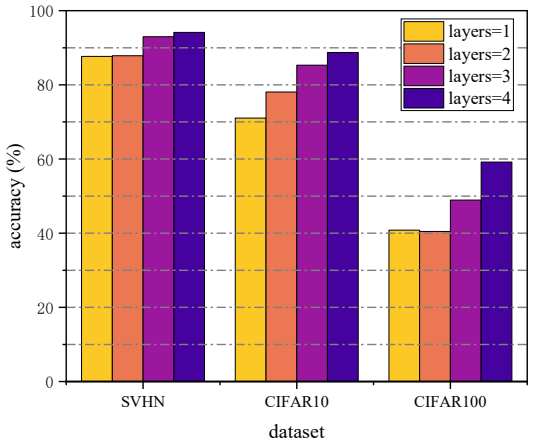


Fig. 5: Test accuracies of FedRD with different numbers of convolution layers in local projection models.

Acknowledgment

This work was supported by the Purple Mountain Laboratories for Network and Communication Security. We thank Southeast University and Purple Mountain Laboratories. Thanks the Big Data Computing Center of Southeast University for supporting computing resources.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[2] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[3] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[4] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for on-device federated learning," *arXiv preprint arXiv:1910.06378*, vol. 2, no. 6, 2019.

[5] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 10 713–10 722.

[6] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *International conference on machine learning*. PMLR, 2021, pp. 2089–2099.

[7] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting Unintended Feature Leakage in Collaborative Learning," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 691–706.

[8] L. Zhu and S. Han, "Deep Leakage from Gradients," in *Federated Learning: Privacy and Incentive*, ser. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 17–31.

[9] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 337–16 346.

[10] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, "Dataset distillation," *arXiv preprint arXiv:1811.10959*, 2018.

[11] B. Zhao, K. R. Mopuri, and H. Bilen, "Dataset condensation with gradient matching," *arXiv preprint arXiv:2006.05929*, 2020.

[12] B. Zhao and H. Bilen, "Dataset condensation with differentiable siamese augmentation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 674–12 685.

[13] ——, "Dataset condensation with distribution matching," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 6514–6523.
[14] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu, "Dataset distillation by matching training trajectories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4750–4759.
[15] ——, "Generalizing dataset distillation via deep generative prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3739–3748.
[16] Y. Xiong, R. Wang, M. Cheng, F. Yu, and C.-J. Hsieh, "Feddm: Iterative distribution matching for communication-efficient federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 323–16 332.
[17] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 7252–7261.
[18] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.
[19] C. Dwork, "Differential privacy," in *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*. Springer, 2006, pp. 1–12.
[20] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.
[21] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," *arXiv preprint arXiv:1710.06963*, 2017.
[22] R. Hu, Y. Guo, and Y. Gong, "Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy," *IEEE Transactions on Mobile Computing*, 2023.
[23] I. Sucholutsky and M. Schonlau, "Soft-label dataset distillation and text dataset distillation," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.
[24] T. Dong, B. Zhao, and L. Lyu, "Privacy for free: How does dataset condensation help privacy?" in *International Conference on Machine Learning*. PMLR, 2022, pp. 5378–5396.
[25] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.
[26] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.
[27] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Ghosh, A. Bharadwaj, J. Zhao, G. Cormode, and I. Mironov, "Opacus: User-friendly differential privacy library in PyTorch," *arXiv preprint arXiv:2109.12298*, 2021.