Regularized Mutual Learning for Personalized Federated Learning

Ruihong Yang[1] 11930391@mail.sustech.edu.cn
Junchao Tian[1,*] tianjc2019@mail.sustech.edu.cn
Yu Zhang[1,2,†] yu.zhang.ust@gmail.com
[1] *Department of Computer Science and Engineering, Southern University of Science and Technology*
[2] *Peng Cheng Laboratory*

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

Federated Learning (FL) is a privacy-protected learning paradigm, which allows many clients to jointly train a model under the coordination of a server without the local data leakage. In real-world scenarios, data in different clients usually cannot satisfy the independent and identically distributed (i.i.d.) assumption adopted widely in machine learning. Traditionally training a single global model may cause performance degradation and difficulty in ensuring convergence in such a non-i.i.d. case. To handle this case, various models can be trained for each client to capture the personalization in each client. In this paper, we propose a new personalized FL framework, called Personalized Federated Mutual Learning (PFML), to use the non-i.i.d. characteristics to generate specific models for clients. Specifically, the PFML method integrates mutual learning into the local update process in each client to not only improve the performance of both the global and personalized models but also speed up the convergence compared with state-of-the-art methods. Moreover, the proposed PFML method can help maintain the heterogeneity of client models and protect the information of personalized models. Experiments on benchmark datasets show the effectiveness of the proposed PFML model.

Keywords: Federated learning, non-i.i.d. data, personalization

1. Introduction

The success of machine learning, especially deep learning, depends on a large amount of data. Deep learning usually conducts centralized training on the data collected from different sources, hence there exist potential risks of data privacy and security. On the other hand, due to confidentiality, data cannot be shared among enterprises, which causes the problem of data islands. To solve the above two problems, Federated Learning (FL) (McMahan et al., 2017; Yang et al., 2019; Kairouz et al., 2019) came into being. FL is a distributed machine learning paradigm with privacy protection. It allows many clients (such as mobile devices) to jointly train a global model under the coordination of a central server while ensuring that the training data stays locally on clients.

Most of existing works of FL such as FedAvg (McMahan et al., 2017) focus on training a single model for all clients. However, in real-world applications, the data in all the clients

[*] The first two authors contributed equally.
[†] Corresponding author.

usually have different distributions so that a single global model may not perform well on all clients. Therefore, the personalized FL framework is proposed to train local customized models for each client to address this non-i.i.d. problem. There are several methods to customize client models, such as fine-tuning, meta learning, mixture methods, and multi-task learning. As a fine-tuning method, FedPer (Arivazhagan et al., 2019) requires that clients share some base layers with the server and train their own personalized layers by adapting to their local data. The pFedMe model (Dinh et al., 2020) uses Moreau envelopes as clients' regularized objective to fine-tune personalized models in several gradient descent steps. Integrated with meta learning methods such as Model Agnostic Meta Learning (MAML) (Finn et al., 2017), Per-FedAvg (Fallah et al., 2020) attempts to build a well generalized global model and fine-tune personalized models on local data for better performance. As a mixture method, APFL (Deng et al., 2020) trains a personalized model for each client by incorporating a part of the global model with some mixing weights. PFL-MoE (Guo et al., 2020) trains personalized models by mixing the outputs of the local model and the global model via a gating network. By formulating FL as a multi-task learning problem, MOCHA (Smith et al., 2017) regards each client as an individual task and trains their personalized models via a primal-dual optimization method.

Although the aforementioned models achieve good performance for personalized FL, they have their own limitations. For example, the Per-FedAvg, APFL and pFedMe models assume that the global model and the client models have the same model architecture, which may not be satisfied in real-world applications as clients may use heterogeneous models with different architectures. The FedPer model also cannot handle the model heterogeneity among clients as it requires that all the clients have similar model architectures with differences only lying in classification layers. FedPer model is not so good on non-iid datasets because of the limitations of fine-tuning based personalized methods(Huang et al., 2021). In the MOCHA model, communication and computation costs are high when the number of clients is large since each client is required to participate in every training epoch.

To learn a global model and personalized models with good performance at the same time, we propose a Personalized Federated Mutual Learning (PFML) model. We combine mutual learning into the model learning of each client so that the proposed PFML can handle the model heterogeneity in clients. Specifically, each client has two models, including the local model and the auxiliary model. The local model minimizes the local loss regularized by minimizing the distance from the global model to make it more conducive to the generalization of the global model, and the auxiliary model minimizes the local loss regularized by minimizing the distance from the model in the previous round, which contains personalized information, to generate a personalized model. Experiments conducted on several benchmark datasets validate the effectiveness of the proposed PFML method. The contributions of this paper include:

(1) **Good Global Model**. Due to the non-i.i.d. nature of data in all the clients, personalized models of clients usually differ from each other. When such different personalized models are pursued, the performance of the global model is usually impaired as shown in our experiments. The proposed PFML is to train a global model and personalized models of clients that perform well simultaneously. As shown in the experiments, the global model learned in the PFML method has better performance than those in the FedAvg and pFedMe methods.

(2) **Good Personalized Models**. Existing personalized FL models usually use the global model as a base model to learn personalized models through fine-tuning or one-step update. We think these trained personalized models contain much information from the global model. Different from those methods, the proposed PFML method can improve the performance of personalized models by using mutual learning in clients and the historical information from the auxiliary models that do not directly participate in the server aggregation.

(3) **Tolerance of Model Heterogeneity**. Many personalized FL models require that the personalized models of clients have the same model architecture as the global model, which places a restriction on the use of FL in many real-world applications. By introducing deep mutual learning into the local training of clients, the proposed PFML model can allow clients to have heterogeneous model architectures. Hence, the proposed PFML model is more flexible.

2. Related Work

Challenges of Federated Learning. When McMahan et al. (2017) first propose the FedAvg algorithm to address a large-scale next-word prediction task and demonstrated its effectiveness and data privacy protection, FL gradually began to attract the attention of machine learning researchers. However, Li et al. (2020) note that FL is currently facing some challenges, including the expensive communication cost between the server and clients, system heterogeneity in clients, non-i.i.d. data distribution among clients, and privacy concerns. A large amount of research has shown that the non-i.i.d. distribution of data will result in the degradation of the performance of the learned model. Facing the challenge of non-i.i.d. data, different types of approaches have been proposed. Some methods (Tuor et al., 2020; Yoshida et al., 2019) use simple strategies like data sharing, which means clients share part of their own data with the server. Some aim to improve the robustness of the global model against non-i.i.d. data. For example, the FedProx method (Li et al., 2018) adds a regularized term in the local training objective to guarantee that local models are not far away from the global model. However, these methods have their own weakness. Data sharing based methods may violate the privacy requirement of FL. A robust global model may still not be suitable for all the clients.

Model Distillation. Model distillation (Hinton et al., 2015), which is also known as knowledge distillation, uses a pre-trained large network as a teacher to provide additional knowledge (e.g., smoothed probability estimation) for a small student network, and empirical studies show that the optimization process of the student network becomes easier and its performance is comparable to or even better than that of the teacher network. Different from one-way knowledge transfer between the teacher and student networks, Deep Mutual Learning (DML) (Zhang et al., 2018) allows two networks to learn from each other through their predicted probability distributions during the training process. Specifically, DML uses the Kullback-Leibler divergence to match the probability estimates of the two networks involved. Currently, some works such as FedMD (Li and Wang, 2019) have applied model distillation into federated learning. The main idea of FedMD is that the server collects clients' predictions on a public dataset and computes a consensus for clients' next training round. After fitting on the public dataset, clients will finetune on their private data to improve the performance further. However, a public dataset is required in FedMD, which cannot always be satisfied in real-world applications. The proposed PFML framework, which combines the idea of deep mutual learning into clients' local updates, does not have such requirements. The PFML model not only protects the information of local models but also retains the heterogeneity of local model architectures.

3. Problem Definition

In vanilla horizontal federated learning, there is a server and N clients. The server initiates the global model with random parameters. Then the server randomly selects a set of active clients and sends the global model to them. Each client will make a copy of the global model and then train the model with their local data. After finishing the training, each client returns the local model or the gradients back to the server. The server aggregates these local models or gradients to update the global model. The process is repeated until the global model converges or the number of communication rounds reaches a threshold. All the clients jointly optimize the following objective function as

$$\min_{w \in \mathbb{R}^d} \left\{ F(w) = \frac{1}{N} \sum_{i=1}^{N} f_i(w) \right\}. \tag{1}$$

Each client has a labeled dataset and f_i represents the loss of the the ith client as

$$f_i(w) = \frac{1}{|\mathcal{D}_i|} \sum_{(x,y) \in \mathcal{D}_i} l_i(w; x, y), \tag{2}$$

where $\mathcal{D}_i$ denotes the labeled dataset in the ith client, $|\mathcal{D}_i|$ denotes the size of $\mathcal{D}_i$, and $l_i(w; x, y)$ measures the loss between the true label y and the prediction by the model w on a local data point x. After solving problem (1), an optimal global model w^* will be learned.

However, according to existing research, a single global model is usually not well generalized when data are heterogeneous across clients. To better address the impact of the non-i.i.d. data challenge, it is natural to customize client models locally. Different from problem (1) to find a single solution for all the clients, a personalized FL method usually optimizes the following objective as

$$\{\theta_1^*, \ldots, \theta_N^*\} = \arg\min \sum_{i=1}^{N} f_i\left(\theta_i\right), \tag{3}$$

which aims to learn optimal personalized models for each client.

4. The PFML Model

In this section, we present the proposed PFML model.

4.1. Overview

PFML incorporates the idea of deep mutual learning into the learning process of clients to generate personalized models. The framework of the PFML model is shown in Figure 1.

In the PFML model, there are two models (i.e., the local model and the auxiliary model) in each client, and they are trained using the same local data. The local model will be transmitted to the server for the model aggregation, and the auxiliary model is used to generate a personalized model with good performance on local data. The following sections will introduce the details of the PFML model.

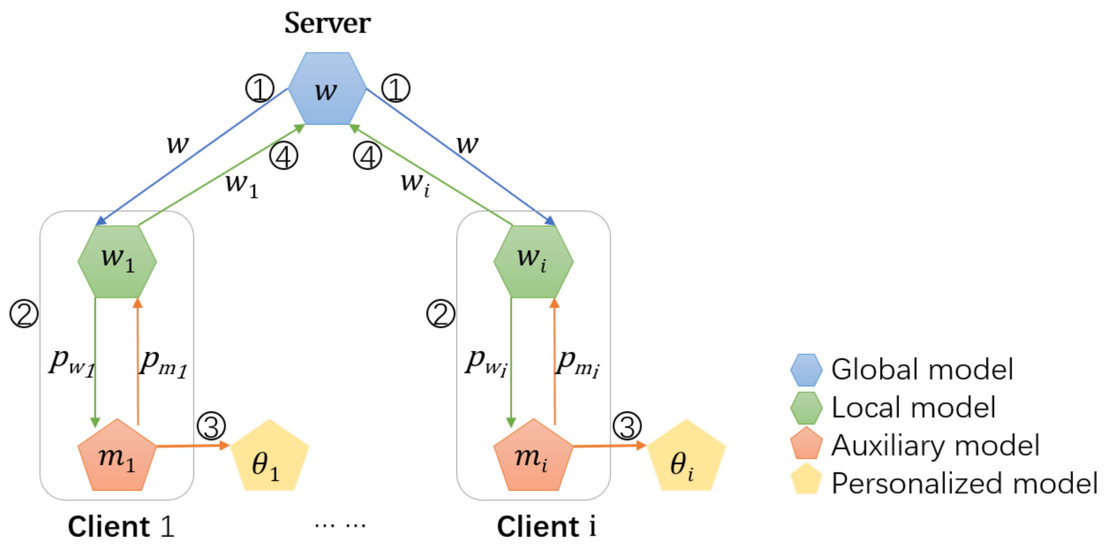


Figure 1: The framework of the proposed PFML model. The training process is described as follows. A server selects active clients and broadcasts a global model with parameters w to them as their local models w_i. Clients then initialize their auxiliary models m_i and conduct regularized mutual learning between local models and auxiliary models. After mutual training, auxiliary models will generate personalized models with parameters θ_i through the one-step update corresponding to the personalization step. Then the ith client updates w_i and m_i, and sends w_i to the server for aggregation. Finally, the server updates its global model and continues until convergence.

4.2. The Entire Model

Client Mutual Learning. Deep mutual learning has shown its success by co-training two models through two-way knowledge transfer, where the knowledge is represented as the predicted class-conditional probability of the two models on the local data. In the PFML model, each client has two models, which are called the local model and the auxiliary model, respectively. Similar to deep mutual learning, in the PFML model, we design the local update in a client by combining this two-way knowledge transfer in the mutual learning of the local model and the auxiliary model. Specifically, the two models (one with global information and the other with local information) regularize the output of their peers so as to achieve the purpose of mutual learning. Mathematically, based on Eq. (2), the loss functions of two models in the ith client can be formulated by

$$L_i(w_i) = f_i(w_i) + D_{KL}\left(p_{m_i}, p_{w_i}\right) \tag{4}$$

$$L_i(m_i) = f_i(m_i) + D_{KL}\left(p_{w_i}, p_{m_i}\right), \tag{5}$$

where w_i and m_i denote the sets of parameters in the local model and the auxiliary model of the the ith client, respectively, $D_{KL}(\cdot,\cdot)$ denotes the Kullback-Leibler divergence, and p_{m_i} (p_{w_i}) denotes the predicted class-conditional probability of the local (auxiliary) model. As shown in Eqs. (4) and (5), for the ith client, the objectives of its two models (i.e., the local model w_i and the auxiliary model m_i) are composed of two parts: one is the classification loss, and another is a mimicry loss in terms of predicted class-conditional probabilities of the two models on the same local data. Here we use the Kullback-Leibler divergence to define the mimicry loss.

Personalized Step. Most of the existing personalized FL methods try to learn personalized models through the one-step update based on the global model w as an initial model. For instance, the Per-FedAvg method (Fallah et al., 2020) considers that the ith client can run one or more gradient decent steps w.r.t its own loss function (i.e., $\theta_i^* = w - \alpha \nabla f_i(w)$ with α as the learning rate) as one-step update and then obtain the personalized model θ_i^*. The pFedMe model minimizes $f_i\left(\theta_i\right) + \frac{\lambda}{2}\left\|\theta_i - w\right\|^2$ as one-step update and obtains the optimal personalized model θ_i^*. In the proposed PFML model, the auxiliary model m_i in the ith client is designed to generate the personalized model θ_i^* because it contains more local information than the local model which is closer to the global model. Moreover, as we expect θ_i^* to learn some historical knowledge from the auxiliary model obtained in the last round, we add a regularization term to minimize the distance between it and the previous auxiliary model m_i^{prev} and the objective based on Eq. (5) is formulated as

$$h_i(m_i; m_i^{prev}) = L_i(m_i) + \frac{\lambda}{2}\left\|m_i^{prev} - m_i\right\|^2, \tag{6}$$

where the hyperparameter λ controls the distance between two models. The optimal personalized model θ_i^* for the ith client is to minimize $h_i(m_i; m^{prev})$ as a one-step update. That is,

$$\theta_i^* = m_i^* = \arg\min_{m_i \in \mathbb{R}^d} h_i(m_i; m_i^{prev}). \tag{7}$$

We use the gradient descent method to learn this optimal solution. The gradient $\nabla h_i(m_i; m_i^{prev})$ of Eq. (6) can be computed as

$$\nabla h_i(m_i; m_i^{prev}) = \nabla L_i(m_i) + \lambda(m_i^{prev} - m_i). \tag{8}$$

A gradient descent operation is called a personalization step. After K personalization steps where K is a predefined hyperparameter, we will get the personalized model. Because of non-i.i.d. data across clients, some methods such as FedProx have considered adding the parameters of the global model as a part of the regularization term. Inspired by this idea, we also add a regularization term to keep the local model w_i not too far from the global model w^* and the corresponding objective is formulated based on Eq. (4) as

$$g_i(w_i; w^*) = L_i(w_i) + \frac{\lambda}{2}\left\|w^* - w_i\right\|^2. \tag{9}$$

Similar to the auxiliary model, we minimize $g_i(w_i; w^*)$ within K steps and get the optimal solution $\hat{w}_i$. Both θ_i^* and $\hat{w}_i$ will be used in the next step to update the auxiliary model and the local model. And the server will aggregate the sampled client's local models to update the global model and then distribute it to clients to start the next round of federated training.

4.3. The Objective of PFML

In summary, the PFML model can learn the local model and the auxiliary model in clients and the global model in the sever via solving the following three problems as

$$\begin{cases} \theta_i^* = \arg\min\limits_{m_i}\ h_i(m_i; m_i^{prev}) = \arg\min\limits_{m_i}\ L_i(m_i) + \dfrac{\lambda}{2}\left\|m_i^{prev} - m_i\right\|^2 \\ \hat{w}_i = \arg\min\limits_{w_i}\ g_i(w_i; w^*) = \arg\min\limits_{w_i}\ L_i(w_i) + \dfrac{\lambda}{2}\left\|w^* - w_i\right\|^2 \\ w^* = \arg\min\limits_{w \in \mathbb{R}^d} \left\{ f(w) = \dfrac{1}{N}\sum\limits_{i=1}^{N} f_i(w) \right\} \end{cases} \tag{10}$$

There are two types of objectives, including the global objective that is to obtain the global model (w^*), and the local objective that is to learn personalized models (i.e., $\theta_1^*, \ldots, \theta_N^*$) using only their own data. We can iteratively optimize these two problems in the following two steps.

(i) **Update the global model**. At first, the global model is initialized randomly, and later it is updated by aggregating the sampled client's local model. The process of updating the global model is almost the same as the FedAvg method by performing the average operation on model parameters of sampled clients' local models. Here, to speed up the convergence, a hyperparameter β is introduced similar to (Karimireddy et al., 2020; Reddi et al., 2021). Specifically, we update the global model by aggregating models of sampled clients as

$$w^{t+1} = (1-\beta)w^t + \frac{\beta}{|\mathcal{U}^t|}\sum_{t \in \mathcal{U}^t} w_i^t, \tag{11}$$

where w^{t+1} and w^t denote the global models in the $(t+1)$th and tth round respectively, $\mathcal{U}^t$ denotes the client set sampled in the tth round, $|\mathcal{U}^t|$ denotes the number of sampled clients, and w_i^t denotes the local model of the ith client. When β is set to 1, the aggregation process will be the same as FedAvg.

(ii) **Update the personalized models**. When the ith client receives the global model, it is passed to the local model and the auxiliary model. When the architecture of the personalized model, which is just the auxiliary model, is different from that of the global model, the auxiliary model is initialized randomly. For the auxiliary model, we perform K steps of gradient update in Eq. (6) and get the solution as a personalized model θ_i^*. We do the same for the local model and get a solution $\hat{w}_i$. After the personalization step, each client will send the local model to the sever for the aggregation.

The whole algorithm for the PFML model is shown in Algorithm 1.

5. Experiments

In this section, we empirically evaluate the proposed PFML model.

5.1. Experimental Settings

Three benckmark datasets, including MNIST, Synthetic (Li et al., 2018), and CIFAR-10, are used in our experiments.

Algorithm 1 PFML

```
input : $w^0$: initial global model weight
        $T$: total number of communication rounds
        $E$: the number of local updates
        $\eta$: the learning rate of local updating
        $\lambda$: the weight of the regularization term
        $K$: the number of one-step update for obtaining personalized models
        $D$: the training batchsize
        $\beta$: the coefficient to speed up the convergence

Server aggregation:
for $t = 0$ to $T-1$ do
    Server selects a subset of active clients $\mathcal{U}^t$ randomly and sends $w^t$ to them
    for each client $i \in \mathcal{U}^t$ do
        $w_i^t$ = Client Update($w^t$)
    end
    Aggregate $w_i^t$ ($i \in \mathcal{U}^t$) to update global model $w^{t+1}$: $w^{t+1} = (1-\beta)w^t + \frac{\beta}{|\mathcal{U}^t|}\sum_{i\in\mathcal{U}^t} w_i^t$
end

Client Update:
for each client $i \in \mathcal{U}^t$ do
    $w_{i,0}^t = w^t$ // local model
    $m_{i,0}^t = m_i^{prev} = w^t$ // auxiliary model
    for $e = 0$ to $E-1$ do
        Sample a batch of data with the batch size $D$
        for $k = 0$ to $K-1$ do
            // Personalization Steps:
            For $m_{i,e}^t$, minimize $L_i\left(m_{i,e}^t\right) + \frac{\lambda}{2}\left\|m_i^{prev} - m_{i,e}^t\right\|^2$ and get optimal personalized model $\theta_i^*$
            For $w_{i,e}^t$, minimize $L_i\left(w_{i,e}^t\right) + \frac{\lambda}{2}\left\|w_t - w_{i,e}^t\right\|^2$ and get $\hat{w}_i$
        end
        Update $m_{i,e}^t$ : $m_{i,e+1}^t = m_{i,e}^t - \eta\nabla_{L_i\left(m_{i,e}^t\right)} - \eta\lambda\left(m_{i,e}^t - \theta_i^*\right)$
        Update $w_{i,e}^t$ : $w_{i,e+1}^t = w_{i,e}^t - \eta\nabla_{L_i\left(w_{i,e}^t\right)} - \eta\lambda\left(w_{i,e}^t - \hat{w}_i\right)$
    end
end
Return $w_i^t$ ($i \in \mathcal{U}^t$) to Server
```

The MNIST dataset contains 10 classes and 70,000 instances. We divide the dataset into 20 parts for 20 clients. Each client has 2 out of the 10 classes as its local data. The classes owned by each client come from the set $\{(0,1),(2,3),(4,5),(6,7),(8,9)\}$. For the other two datasets, we adopt the non-i.i.d. setting of (Dinh et al., 2020). For the Synthetic dataset, we generate the data using two parameters $\bar{\alpha} = 0.5$ and $\bar{\beta} = 0.5$ to control the degree of difference among datasets of clients divide it into 100 parts for 100 clients. For the CIFAR-10 dataset, we consider 20 clients and each of them has only 3 out of the 10 classes. All the datasets are split randomly with 75% and 25% for training and testing, respectively.

We consider both convex objectives and non-convex objectives in FL. For simplicity, in the μ-strongly convex setting, we consider a l_2-regularized multinomial logistic regression (MLR) model, which is a fully connected layer following the log-softmax activation. In the non-convex setting, we use a two-layer deep neural network (DNN) with a hidden layer whose size is 100 for the MNIST dataset and 20 for the Synthetic dataset. We use the LeNet5 (denoted by CNN) for the CIFAR-10 dataset.

5.2. Experimental Results

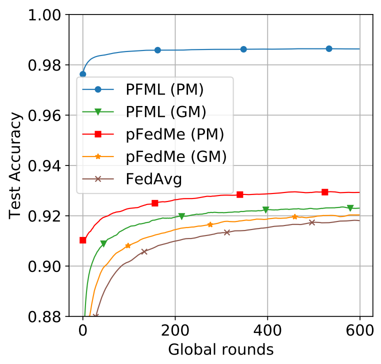
We conduct experiments on the benchmark datasets and analyze experimental results as follows.

(1) The μ-strongly convex setting: We use the MLR model for the MNIST and Synthetic datasets. The results shown in Figure 2 and Table 1 report the test accuracy and training loss of the FedAvg, pFedMe, and PFML models. For the pFedMe and PFML, the suffixes 'GM' and 'PM' stand for the global model and the personalized model, respectively. The accuracy of GM is measured on the testing datasets of all clients and the accuracy of PM is measured on the local dataset of each client. According to the results, we can see that the performance of the personalized models trained by PFML has a significant improvement than other baselines. For instance, for the MNIST dataset, PFML's personalized models have 5.5% and 6.8% improvement when compared with pFedMe and FedAvg. Moreover, the global model of the PFML model behaves better than FedAvg and pFedMe. For example, on the Synthetic dataset, PFML's global model is 1.1% and 1.6% better than pFedMe and FedAvg.

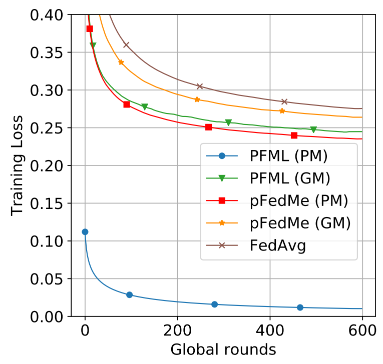
Table 1: The classification accuracy of FedAvg, pFedMe, and PFML on the MNIST and Synthetic datasets.

Algorithm	Model	MNIST ($\|\mathcal{U}^t\|$=10)			Synthetic ($\|\mathcal{U}^t\|$=10)		
		λ	β	Accuracy (%)	λ	β	Accuracy (%)
FedAvg	MLR		1.0	91.87 ± 0.02		1.0	77.97 ± 0.28
pFedMe-GM	MLR	15	2.0	92.13 ± 0.02	20	2.0	78.48 ± 0.42
pFedMe-PM	MLR	15	2.0	93.06 ± 0.03	20	2.0	82.81 ± 0.03
PFML-GM (ours)	MLR	15	2.0	92.45 ± 0.02	20	2.0	79.58 ± 0.18
PFML-PM (ours)	MLR	15	2.0	$\mathbf{98.65} \pm 0.01$	20	2.0	$\mathbf{90.74} \pm 0.01$
FedAvg	DNN		1.0	94.86 ± 0.05		1.0	81.12 ± 0.35
pFedMe-GM	DNN	30	2.0	95.85 ± 0.05	30	2.0	83.13 ± 0.37
pFedMe-PM	DNN	30	2.0	96.72 ± 0.01	30	2.0	84.81 ± 0.23
PFML-GM (ours)	DNN	30	2.0	96.70 ± 0.04	20	2.0	84.78 ± 0.32
PFML-PM (ours)	DNN	30	2.0	$\mathbf{99.07} \pm 0.01$	20	2.0	$\mathbf{94.58} \pm 0.06$

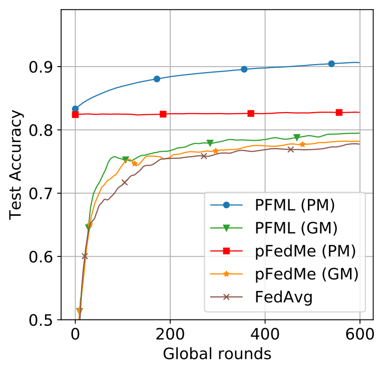
(2) The non-convex setting: We use DNN model for the MNIST and the Synthetic dataset classification tasks and CNN for the CIFAR-10 classification task. The results show in Figures 3 and 4 correspond to the experimental results with 5 and 10 active clients, respectively. On the three datasets, the performance of PFML's personalized models is the best, and its global model performs better than pFedMe and FedAvg. On the MNIST and Synthetic datasets, PFML's personalized models have a significant improvement than the other two algorithms, and even the performance of its global model is close to that of pFedMe's personalized models. On the CIFAR-10 dataset, the performance of the global model of pFedMe is worst and the performance of the global model of PFML is 8.7% and 4.4% better than pFedMe and FedAvg. Detailed results can refer to Tables 1 and 2.



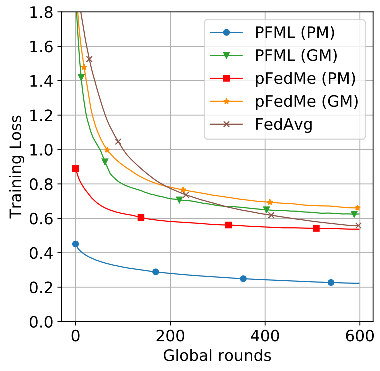
(a) test accuracy on the MNIST dataset



(b) training loss on the MNIST dataset

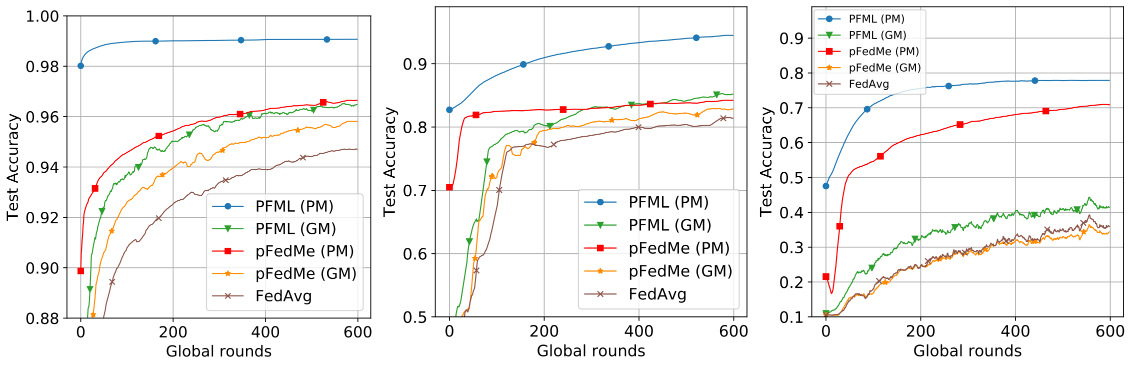


(c) test accuracy on the Synthetic dataset



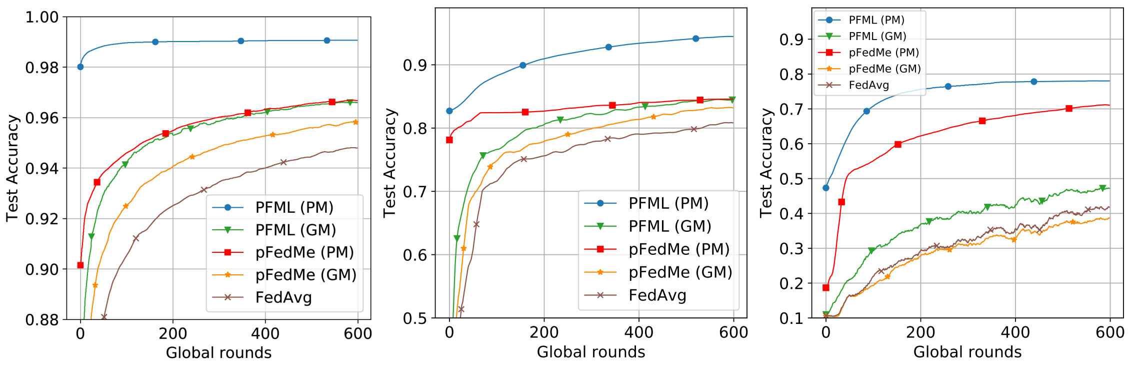
(d) training loss on the Synthetic dataset

Figure 2: The classification accuracy of various models on two non-i.i.d. datasets based on the MLR model, where $|\mathcal{U}^t| = 10$, $E = 10$, $\eta = 0.01$, $K = 3$, $\beta = 2$, and $D = 200$.



(a) MNIST/DNN (b) Synthetic/DNN (c) CIFAR-10/CNN

Figure 3: The classification accuracy of various models on three non-i.i.d. datasets based on the DNN model, where $|\mathcal{U}^t| = 5$, $E = 10$, $\eta = 0.01$, $K = 3$, and $D = 200$.



(a) MNIST/DNN (b) Synthetic/DNN (c) CIFAR-10/CNN

Figure 4: The classification accuracy of various models on three non-i.i.d. datasets based on the DNN model, where $|\mathcal{U}^t| = 10$, $E = 10$, $\eta = 0.01$, $K = 3$, and $D = 200$.

Table 2: The classification accuracy of various models on the CIFAR-10 dataset, where $E = 10$, $\eta = 0.01$, $\lambda = 15$, $K = 3$, $\beta = 1$, and $D = 200$.

Algorithm	Model	CIFAR-10 ($|\mathcal{U}^t|$=5)	CIFAR-10 ($|\mathcal{U}^t|$=10)
FedAvg	CNN	43.54 ± 0.05	45.08 ± 0.06
pFedMe-GM	CNN	39.73 ± 0.02	40.76 ± 0.02
pFedMe-PM	CNN	71.19 ± 0.03	71.38 ± 0.01
PFML-GM (ours)	CNN	49.24 ± 0.12	49.48 ± 0.24
PFML-PM (ours)	CNN	$\mathbf{78.20} \pm 0.06$	$\mathbf{78.24} \pm 0.07$

5.3. Results on Client Model Heterogeneity

Most existing works in FL require the global model and local models to have the same model architecture. However, because of the system heterogeneity, clients may require different model architectures for better performance (Li and Wang, 2019). PFML can support the personalized model, that is the auxiliary model, to have a different model architecture from the global model. To validate this ability of PFML, we use DNN for auxiliary models in clients and MLR for both the local models in clients and the global model in the server. Then we make a small modification on Algorithm 1 about the initialization of the auxiliary model and the local model. That is, we initialize the local model by the global MLR model and initialize the auxiliary model (DNN) randomly. We compare this modified algorithm with the original PFML that only uses a model for both the auxiliary model and the local model. In the experiment, we use two values of λ (i.e., 20 and 30) for the modified algorithm, and the original PFML uses the best value of λ. According to the results showed in Table 3, when comparing with the original PFML that only uses MLR, the performance of the global model (denoted by GM) for this modified algorithm is comparable. But the performance of the personalized model (denoted by PM) for this modified algorithm is better because DNN has a better capacity than MLR. When comparing with the original PFML that only uses DNN, the modified algorithm performs slightly worse for the global model due to the MLR used but performs comparable for the personalized model. According to the results, we can see that the proposed PFML model is flexible to handle the model heterogeneity without much performance degradation, which is an additional advantage of the proposed PFML model.

Table 3: The classification accuracy on the Synthetic dataset for the analysis of the model heterogeneity, where $E = 10$, $\eta = 0.01$, $K = 3$, $\beta = 2$, and $D = 200$.

	MLR (GM) + DNN (PM)		MLR	DNN
λ	20	30	20	30
GM	80.16%	80.59%	80.44%	84.71%
PM	94.40%	94.58%	90.71%	94.61%

5.4. Sensitivity Analysis

In this section, we conduct experiments to test the sensitivity of the performance with respect to some hyperparameters in the proposed PFML model, including K, λ, and β.

In Algorithm 1, K denotes the number of the personalization steps. The results in Table 4 show that the performance is not very sensitive with respect to K. One reason is that while K determines the local update in clients, if the global epoch or local epoch is long enough, the entire algorithm will converge no matter what the value of K is.

Table 4: The classification accuracy when varying K on the MNIST Dataset , where λ is set to 30 and 15, respectively, when using the DNN or MLR model, $E = 10$, $\eta = 0.01$, $K = 3$, $\beta = 2$, and $D = 200$.

$\|\mathcal{U}^t\|$	5		10	
	DNN		MLR	
K	GM	PM	GM	PM
1	96.77%	99.08%	92.48%	98.64%
3	96.67%	99.07%	92.44%	98.65%
5	96.65%	99.07%	92.48%	98.65%
7	96.68%	99.07%	92.44%	98.65%

λ is a regularization parameter to weigh the training loss and the regularizer in both the local model and the auxiliary model for a client. According to the results shown in Table 5, the performance of the global and personalized model changes slightly. Specifically, when λ increases, the performance of the personalized model (denoted by PM) becomes slightly better, but the classification accuracy of the global model (denoted by GM) first increases slightly and then decreases slightly. To balance the performance of the two models, the value of λ can be set to 20 or 30, which is the default setting of λ in all the experiments.

Table 5: The classification accuracy on the Synthetic dataset when varying λ, where $E = 10$, $\eta = 0.01$, $K = 3$, $\beta = 2$, and $D = 200$.

$\|\mathcal{U}^t\|$	5		10	
	MLR		DNN	
λ	GM	PM	GM	PM
10	79.88%	90.45%	84.71%	94.16%
20	80.44%	90.71%	85.00%	94.42%
30	79.99%	90.81%	84.68%	94.61%
40	80.16%	90.94%	84.76%	94.80%

β has an important connection with the server aggregation process. When its value is relatively large, the pace of the update in the global model will be relatively large, and when its value is close to 1, it will be close to the simple weighted averaging in FedAvg. According to the results in Table 6, a suitable value of β can be set to 2 or 3, under which the performance of both the global and personalized models is better than other cases. This result may imply that the pace of updating the global model should be appropriate so that the performance of the global and personlized models can be good simultaneously.

Table 6: The classification accuracy on the Synthetic dataset when varying β, where λ is set to 30 and 20, respectively, when using DNN and MLR, $E = 10$, $\eta = 0.01$, $K = 3$, and $D = 200$.

$\|\mathcal{U}^t\|$	5		10	
	DNN		MLR	
β	GM	PM	GM	PM
1	81.78%	94.63%	78.96%	90.61%
2	84.71%	94.61%	80.36%	90.66%
3	85.36%	94.58%	80.29%	90.74%
4	82.46%	94.49%	79.95%	90.76%

6. Conclusion

In this paper, we propose the PFML method for personalized FL. Empirical studies show that the PFML method is competitive with state-of-the-art personalized FL models. While improving the performance of the personalized models, the PFML model maintains good generalization performance of the global model as well as the heterogeneity of client models and protects the information of personalized models. In future work, we will extend the PFML model to other FL settings.

Acknowledgments

This work is supported by NSFC 62076118.

References

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *CoRR*, abs/1912.00818, 2019. URL http://arxiv.org/abs/1912.00818.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *CoRR*, abs/2003.13461, 2020. URL https://arxiv.org/abs/2003.13461.

Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. 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. URL https://proceedings.neurips.cc/paper/2020/hash/f4f1f13c8289ac1b1ee0ff176b56fc60-Abstract.html.

Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. 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. URL https://proceedings.neurips.cc/paper/2020/hash/24389bfe4fe2eba8bf9aa9203a44cdad-Abstract.html.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 2017. URL http://proceedings.mlr.press/v70/finn17a.html.

Binbin Guo, Yuan Mei, Danyang Xiao, Weigang Wu, Ye Yin, and Hongli Chang. Pfl-moe: Personalized federated learning based on mixture of experts. *CoRR*, abs/2012.15589, 2020. URL https://arxiv.org/abs/2012.15589.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 7865–7873. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/16960.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *CoRR*, abs/1912.04977, 2019. URL http://arxiv.org/abs/1912.04977.

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. URL http://proceedings.mlr.press/v119/karimireddy20a.html.

Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *CoRR*, abs/1910.03581, 2019. URL http://arxiv.org/abs/1910.03581.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3): 50–60, 2020.

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. PMLR, 2017.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=LkFG3lB13U5.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. Federated multi-task learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4424–4434, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6211080fa89981f66b1a0c9d55c61d0f-Abstract.html.

Tiffany Tuor, Shiqiang Wang, Bong-Jun Ko, Changchang Liu, and Kin K. Leung. Overcoming noisy and irrelevant data in federated learning. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 5020–5027. IEEE, 2020. doi: 10.1109/ICPR48806.2021.9412599. URL https://doi.org/10.1109/ICPR48806.2021.9412599.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2):12:1–12:19, 2019. doi: 10.1145/3298981. URL https://doi.org/10.1145/3298981.

Naoya Yoshida, Takayuki Nishio, Masahiro Morikura, Koji Yamamoto, and Ryo Yonetani. Hybrid-fl: Cooperative learning mechanism using non-iid data in wireless networks. *CoRR*, abs/1905.07210, 2019. URL http://arxiv.org/abs/1905.07210.

Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.