FedMeS: Personalized Federated Continual Learning Leveraging Local Memory

Jin Xie[1] , **Chenqing Zhu**[1] , **Songze Li**[1,2]

[1]The Hong Kong University of Science and Technology (Guangzhou)

[2]The Hong Kong University of Science and Technology

{jxie171, czhu032}@connect.hkust-gz.edu.cn, songzeli@ust.hk

Abstract

We focus on the problem of Personalized Federated Continual Learning (PFCL): a group of distributed clients, each with a sequence of local tasks on arbitrary data distributions, collaborate through a central server to train a personalized model at each client, with the model expected to achieve good performance on all local tasks. We propose a novel PFCL framework called Federated Memory Strengthening (`FedMeS`) to address the challenges of client drift and catastrophic forgetting. In `FedMeS`, each client stores samples from previous tasks using a small amount of local memory, and leverages this information to both 1) calibrate gradient updates in local training; and 2) perform KNN-based Gaussian inference to facilitate local inference. `FedMeS` is designed to be *task-oblivious*, such that the same inference process is applied to samples from all tasks to achieve good performance. `FedMeS` is analyzed theoretically and evaluated experimentally. It is shown to outperform all baselines in average accuracy and forgetting rate, over various combinations of datasets, task distributions, and client numbers.

1 Introduction

Federated learning (FL) [McMahan *et al.*, 2017] is an emerging distributed learning framework that allows for collaborative training of a model across multiple clients while keeping their raw data locally stored. A typical FL process involves local training on each client and global model aggregation on a cloud server, with only model updates or gradients being shared between clients and server.

Data collected from different clients in an FL system often have drastically different distributions. As seen in Figure 1(a), this can lead to model parameter divergence and *client drift* [Venkatesha *et al.*, 2022], causing potentially poor performance for certain clients. The conventional way of training a single model is insufficient to fit all the non-IID data, and a personalized model needs to be trained for each participating client [Huang *et al.*, 2021; Fallah *et al.*, 2020], which is known as personalized FL.

Another key characteristic in real-world FL systems is that clients are continuously collecting new data (new task) which may exhibit different distributions from previous local data (tasks). Hence, it would be preferable to train a local model that is able to achieve consistently good performance in all local tasks. In an FL system, the problem is solved by federated continual learning (FCL) [Yao and Sun, 2020; Shoham *et al.*, 2019; Yoon *et al.*, 2021]. The phenomenon of a model failing to perform well on previously trained tasks is called *catastrophic forgetting* [Kirkpatrick *et al.*, 2017], which is illustrated in Figure. 1(b).

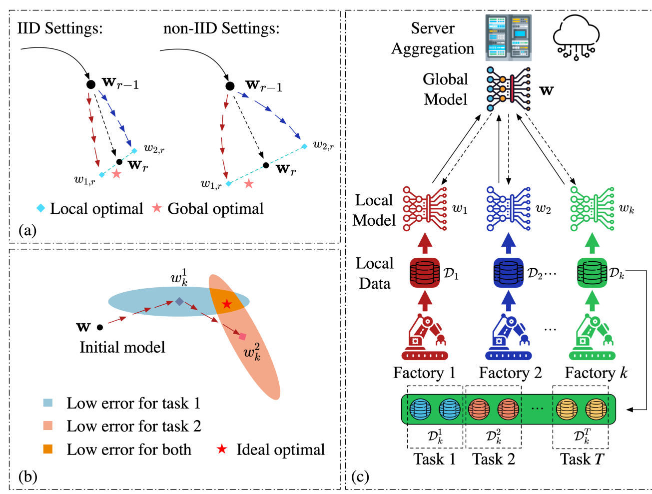


Figure 1: (a) Illustration of the model parameter divergence with non-IID datasets. (b) Illustration of catastrophic forgetting. (c) An overview of a PFCL system in IIoT scenario.

In practical FL systems, the data and task heterogeneity often exist both across clients and over time on a single client. For instance, as shown in Figure 1(c), in a IIoT scenario, multiple factories manufacturing different products would like to use FL for training defect detection models collaboratively. Other than the difference between the types of products, each factory may experience change of tasks over time due to e.g., change of recipe and upgrade of the production line. Aiming to address the challenges of client drift and catastrophic forgetting simultaneously, in the paper we focus on the personalized federated continual learning (PFCL) problem. In a PFCL system, each client observes a stream of arbitrarily different tasks, and would like to collaborate through the server to train a personalized model, which performs well on all local tasks.

While there have been prior attempts at tackling the FCL problem, like FedWeIT [Yoon *et al.*, 2021], where task-generic and task-specific knowledge are shared across clients to decompose the model parameters. However, as demonstrated in [Venkatesha *et al.*, 2022], FedWeIT struggles to handle the issue of client drift caused by data heterogeneity and does not address the scalability of tasks.

In this paper, we propose a novel federated learning PFCL framework called **F**ederated **M**emory **S**trengthening (FedMeS). FedMeS utilize small amount of local memory at each client to store information about previous tasks, and leverage this memory to assist both the training and inference processes. During training process, the gradients are constantly calibrated against the data samples from previous tasks to avoid catastrophic forgetting. A newly designed regularization term adjusted by a loss-based parameter is used to facilitate the training of personalized models using information from the global model. In the inference process, FedMeS directly leverages the memory information in training process to perform KNN-based Gaussian inference, further strengthening the model's personalization capability. Moreover, FedMeS exhibits a major advantage in being *task-oblivious*, meaning that the inference process for test samples from all tasks is identical, and all are expected to achieve high performance.

Through extensive experiments with various dataset combinations, task constructions, task distributions, and client numbers, we show that FedMeS uniformly outperforms all baselines in terms of accuracy metrics and forgetting rate. These results highlight the potential of FedMeS for real-world applications and as a basis for future research in the area of PFCL.

2 Related Work

2.1 Personalized Federated Learning

A lot of work has been done in personalized FL. A simple idea is by deploying a global model and fine tuning parameters through gradient descent on local clients [Cheng *et al.*, 2021; Yu *et al.*, 2020b; Zhang *et al.*, 2022]. Meta-learning based FL methods realize model personalization through hyperparameters [Khodak *et al.*, 2019]. PerKNN [Marfoq *et al.*, 2022] is a special case, where embeddings of training samples are stored for local memorization for KNN-based Gaussian inference. The mainstream design is to interpolate a global model and one local model per client, and the task-specific models are learned both globally and locally [Achituve *et al.*, 2021; Shen *et al.*, 2020]. Like using regularization terms on proximal models to help construct personalized information [Li *et al.*, 2021; Marfoq *et al.*, 2021].

2.2 Continual Learning

Memory replay methods are widely used in continual learning (CL) to maintain prediction accuracies of past tasks. Generally speaking, a memory buffer is used to store previous data which are replayed while learning a new task to alleviate forgetting [Wang *et al.*, 2022; Shim *et al.*, 2021]. Experience replay (ER) jointly optimizes the network parameters by interleaving the previous task exemplars with current task data [Riemer *et al.*, 2018; Isele and Cosgun, 2018]. An alternative solution is by constrained optimization. GEM [Lopez-Paz and Ranzato, 2017] and A-GEM [Chaudhry *et al.*, 2018] leverage episodic memory to compute previous task gradients to constrain the current update step. Besides replay methods, regularization-based methods [Yu *et al.*, 2020a; Shi *et al.*, 2021] and parameter isolation methods [De Lange *et al.*, 2021] have also been proposed for CL.

Federated Continual Learning. Although a lot of work has been done in CL, just a few works have tried to use CL in a federated setting. Besides the FedWeIT, other methods, like LFedCon2 [Casado *et al.*, 2020], use traditional classifiers instead of DNN and propose an algorithm dealing with a concept drift based on ensemble retraining. FLwF and FLwF-2T [Usmanova *et al.*, 2021] use a distillation-based approach dealing with catastrophic forgetting in FL scenario and focus on the class-incremental learning scenario.

3 Problem Definition

We consider an FL system that consists of m clients and a central server. Over time, each client k ($k = 1, \ldots, m$) continually receives private datasets from a sequence of T machine learning tasks. For each task t ($t = 1, \ldots, T$), the corresponding dataset at client k is denoted as $\mathcal{D}_k^t$. We focus on a general non-IID case, where $\mathcal{D}_k^t$ is drawn from some probability distribution $\mathcal{P}_k^t$, and no particular relationships for $\mathcal{P}_k^t$ across k and t are assumed.

The conventional FL problem corresponds to a single-task scenario. For a particular task t, the following objective function is optimized over the global model $\mathbf{w}$

$$\min_{\mathbf{w}} \mathcal{G}(F_1(\mathbf{w}; \mathcal{D}_1^t), \ldots, F_m(\mathbf{w}; \mathcal{D}_m^t)), \tag{1}$$

where $F_k(\cdot)$ is a local objective for client k over current dataset $\mathcal{D}_k^t$, and $\mathcal{G}(\cdot)$ is an aggregation function. In [McMahan *et al.*, 2017], $\mathcal{G} = \sum_{i=1}^{m} p_k F_k(\mathbf{w}, \mathcal{D}_k^t)$ is chosen as a weighted sum with $\sum p_k = 1$.

When task changes over time on each client, the client intends to obtain an evolving local model which maintains good performance on all its previous tasks. Motivated by this need, we introduce the concept of continual learning [Chaudhry *et al.*, 2018] into the personalized FL framework, and formally define the personalized federated continual learning (PFCL) problem. Specifically, for client k ($k = 1, ..., m$) with a sequence of local datasets $(\mathcal{D}_k^1, ..., \mathcal{D}_k^T)$, the personalized model w_k^t for task t ($t = 1, ..., T$) is obtained through

$$\begin{aligned} \min_{w_k^t} \; & G_k(w_k^t; \mathbf{w}^*) = \mathcal{L}\left(w_k^t; \mathcal{D}_k^t\right) + \frac{\lambda}{2} \left\| w_k^t - \mathbf{w}^* \right\|^2 \\ \text{s.t.} \; & \mathbf{w}^* \in \arg\min_{\mathbf{w}} \mathcal{G}\left(\mathcal{L}(\mathbf{w}; \mathcal{D}_1^t), ..., \mathcal{L}(\mathbf{w}; \mathcal{D}_m^t)\right) \\ & \mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \le \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right) \end{aligned} \tag{2}$$

Here $\mathcal{L}(w; \mathcal{D})$ is the empirical loss of w on dataset $\mathcal{D}$. $\mathcal{M}_k^t \subset \bigcup_{i=1}^{t-1} \mathcal{D}_k^i$ is the episodic memory on client k storing samples

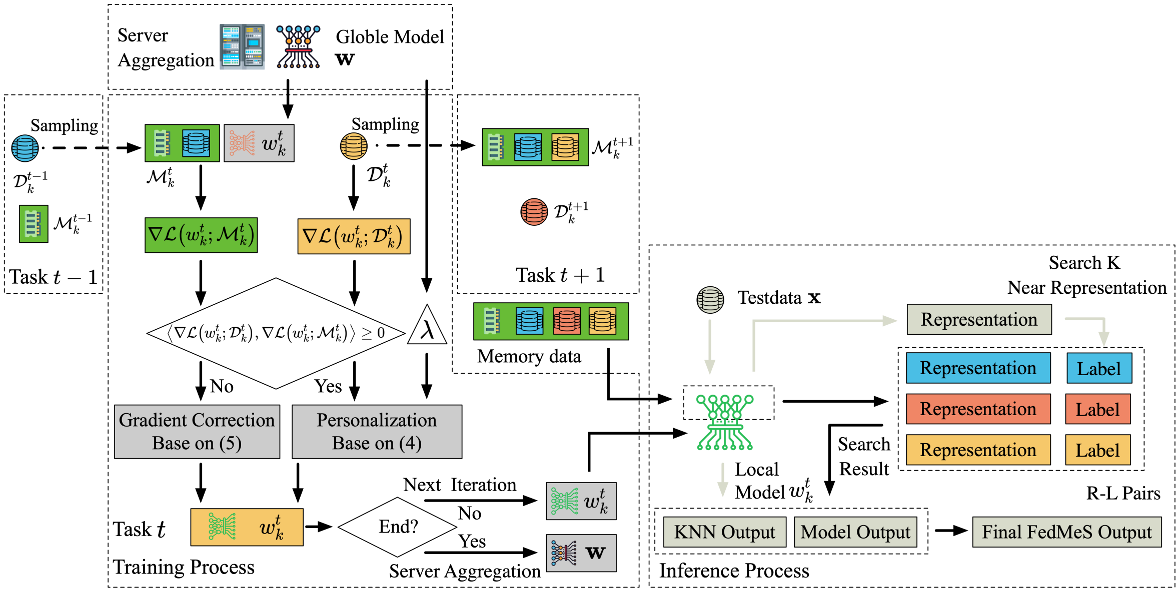


Figure 2: Overall workflow of `FedMeS`. Local episodic memory is utilized in both the training and inference processes.

from all previous tasks (i.e., task 1 to $t-1$). Each client reserves an equal amount of memory to store some samples from each task. When learning the first task, $\mathcal{M}_k^1 = \emptyset$. Every time when client k finishes learning task t, examples are randomly sampled from $\mathcal{D}_k^t$ and stored in the allocated space. The newly stored examples, together with current $\mathcal{M}_k^t$, constitute $\mathcal{M}_k^{t+1}$ as the episodic memory used for next task. The constraint $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ ensures that the model obtained from the new task t has a lower loss than the previous model over the samples of past datasets. This effectively alleviates the forgetting of previous tasks on current models.

4 `FedMeS`

We propose an algorithm called **F**ederated **Me**mory **S**trengthening (`FedMeS`) to solve the PFCL problem defined in (2). The key idea of `FedMeS` is to flexibly use the local memory on the samples of previous tasks in both the local training and inference processes. In training process, the local memory is used for gradient correction to avoid catastrophic forgetting; in inference process, a KNN algorithm based on the representations of local samples helps to improve the accuracy of the personalized model. The overall workflow of `FedMeS` is illustrated in Figure 2.

4.1 Training Process of `FedMeS`

As in a conventional FL setting, the training of task t ($t = 1, \ldots, T$) proceeds over multiple global iterations between the server and the clients. In each global iteration, the server broadcasts the global model $\mathbf{w}$ to the clients, waits for clients to upload personalized models w_k^t, and aggregates them to update the global model $\mathbf{w} \leftarrow \frac{1}{m}\sum_{k=1}^{m} w_k^t$.

During local training process, each client k needs to run multiple local iterations to update w_k^t. We focus on describing parameter updates in a single local iteration.

As shown in (2), PFCL is a constraint minimizing problem, for which traditional Stochastic Gradient Descent does not directly apply. As shown in [Lopez-Paz and Ranzato, 2017], the constraint $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ is equivalent to the following condition on the inner product of gradients on the current and previous tasks:

$$\left\langle \nabla\mathcal{L}\left(w_k^t; \mathcal{D}_k^t\right), \nabla\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right)\right\rangle \geq 0. \tag{3}$$

By this transformation, it is not necessary to store the old parameters w_k^{t-1} and compute loss on $\mathcal{M}_k^t$ in every iteration; only the inner product needs to be computed and compared. If the inequality in (3) is satisfied, it means that the updates on current task t and the local memory are roughly in the same direction, so the optimization on current task would not negatively impact the performance of past tasks, and it is safe to update the model along the gradient of current task as follows, for some learning rate η_1:

$$w_k^t = w_k^t - \eta_1 \left(\nabla\mathcal{L}(w_k^t; \mathcal{D}_k^t) + \lambda \|w_k^t - \mathbf{w}\|\right). \tag{4}$$

When (3) does not hold, client k first adjusts its local weights w_k^t to avoid forgetting, through the following gradient correction step, for some learning rate η_2:

$$\begin{aligned} w_k^t = w_k^t - \eta_2 \Bigg(&\nabla\mathcal{L}(w_k^t; \mathcal{D}_k^t) \\ &- \frac{\nabla\mathcal{L}(w_k^t; \mathcal{D}_k^t)^\top \nabla\mathcal{L}(w_k^t; \mathcal{M}_k^t)}{\nabla\mathcal{L}(w_k^t; \mathcal{M}_k^t)^\top \nabla\mathcal{L}(w_k^t; \mathcal{M}_k^t)} \nabla\mathcal{L}(w_k^t; \mathcal{M}_k^t)\Bigg) \end{aligned} \tag{5}$$

This gradient correction occurs *within* the local weights and does not involve global weights (the reason why we do not need term $\lambda\|w_k^t - \mathbf{w}\|$ in (5)).

We note that *only one* of (4) and (5) would be executed in every local training iteration: we update the gradients based on the local objective function in (2) only when (3) holds, which means that the gradient update does not lead to catastrophic forgetting. When (3) is not satisfied, as demonstrated in [Chaudhry *et al.*, 2018], updating w_k^t

according to (5) multiple times allows the inner product $\langle \nabla \mathcal{L}\left(w_k^t; \mathcal{D}_k^t\right), \nabla \mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right)\rangle$ (which is < 0 currently) to gradually approach and eventually exceed zero.

In FedMeS, rather than fixing regularization parameter λ, we propose a loss-based approach for dynamically adjusting λ. Specifically, we set the value of λ as:

$$\lambda = 2 \cdot \text{sigmoid}\left(\frac{1}{\mathcal{L}(\mathbf{w}, \mathcal{D}_k^t)}\right) \tag{6}$$

Intuitively, when $\mathcal{L}(\mathbf{w}, \mathcal{D}_k^t)$ is relatively large, it means the global model $\mathbf{w}$ performs poorly on the current task of client k, and the personalized model w_k^t should deviate from the global model by decreasing λ. On the other hand, a small $\mathcal{L}(\mathbf{w}, \mathcal{D}_k^t)$ would encourage w_k^t to approach the global model $\mathbf{w}$ which correspond to a lager λ. Here the sigmoid function is used to limit the value of λ within $[0, 2]$.

4.2 Inference Process of FedMeS

As shown in Figure 2, FedMeS utilizes local memory not only to mitigate catastrophic forgetting during training, but also to improve the inference performance on test samples. Specifically, to perform an inference task after learning task t, a client k first generates a set of representation-label pairs (R-L pairs) from the current local memory as

$$\left\{\left(P_{w_k^t}\left(\mathbf{m}_k^i\right), y_{\mathbf{m}_k^i}\right) : \left(\mathbf{m}_k^i, y_{\mathbf{m}_k^i}\right) \in \mathcal{M}_k^t\right\} \tag{7}$$

Here $\mathbf{m}_k^i, i = 1, \ldots, |\mathcal{M}_k^t|$, is the input of the i-th sample in $\mathcal{M}_k^t$, and $y_{\mathbf{m}_k^i}$ is its label.

Function $P_{w_k^t}(\mathbf{m}_k^i)$ generates an embedding representation of $\mathbf{m}_k^i$, which for example, could be the output of the last convolutional layer in the case of CNNs, or the output of an arbitary self-attention layer in the case of transformers. Then, for a test sample $\mathbf{x}$ (from unknown task), we first find the K nearest neighbors of $\mathbf{x}$ from the formed R-L pairs:

$$\mathcal{K}^{(K)}(\mathbf{x}) = \left\{\left(P_{w_k^t}(\mathbf{m}_k^j), y_{\mathbf{m}_k^j}\right) : 1 \le j \le K\right\} \tag{8}$$

which satisfy

$$dist(P_{w_k^t}(\mathbf{x}), P_{w_k^t}(\mathbf{m}_k^j)) \le dist(P_{w_k^t}(\mathbf{x}), P_{w_k^t}(\mathbf{m}_k^{j+1})) \tag{9}$$

Here $dist(\cdot, \cdot)$ could be any distance metric, and in FedMeS the Euclidean distance is used. Denote $b_{w_k^t}(\mathbf{x})$ as the local estimate of the conditional probability $\mathcal{P}_{\mathcal{M}_k^t}(y|\mathbf{x})$, where $\mathcal{P}_{\mathcal{M}_k^t}$ is the probability distribution of $\mathcal{M}_k^t$. Then the K nearest neighbours found in $\mathcal{K}^{(K)}(\mathbf{x})$ are used to compute $[b_{w_k^t}(\mathbf{x})]_y$ with a Gaussian kernel:

$$\begin{aligned}[b_{w_k^t}(\mathbf{x})]_y \propto & \sum_{j=1}^{K} \mathbf{1}_{y = y_{\mathbf{m}_k^j}} \times \\ & \exp\{-dist(P_{w_k^t}(\mathbf{x}), P_{w_k^t}(\mathbf{m}_k^j))\}\end{aligned} \tag{10}$$

Finally, the FedMeS prediction result of $\mathbf{x}$ on client k is obtain by the following distribution

$$\text{FedMeS}_k(\mathbf{x}) \triangleq \theta_k \cdot b_{w_k^t}(\mathbf{x}) + (1 - \theta_k) h_{w_k^t}(\mathbf{x}). \tag{11}$$

Here $h_{w_k^t}$ is the personalized local model parameterized by w_k^t, and $\theta_k \in (0, 1)$ is a hyperparameter which can be tuned through a local validation or cross-validation.

Remark 1. *As proved in [Khandelwal* et al.*, 2019], augmenting the model inference with a memorization mechanism (KNN in this case) helps to improve the performance. In [Marfoq* et al.*, 2022], local memorization through KNN has been applied to improve the accuracies of local models in personalized FL, for a single task.* FedMeS *extends the application of this technique on episodic memorization over* an arbitrary sequence of tasks*, via utilizing a subset of samples from each task. Also, this inference enhancement of* FedMeS *comes* for free*, as this memory is readily available from the preceeding training process.*

Remark 2. *Another major advantage of* FedMeS *is that it is* task-oblivious*. That is, the same inference process is applied for* all *test samples, and no prior knowledge is needed about which task the sample belongs to. This also reflects the robustness of* FedMeS*: regardless of the original task, a good inference performance is always guaranteed by a unified* FedMeS *inference process. This is, however, not the case for other task-incremental learning CL methods like in [Delange* et al.*, 2021].*

4.3 Convergence Analysis

In this section, we analyze the convergence performance of FedMeS. All proofs are omitted due to page limit. Following assumptions are made to facilitate the analysis. For each communication round r on client k when solving task t, denote $w_k^{(r)}, \mathbf{w}^{(r)}$ respectively as the value of $w_k^t, \mathbf{w}$ at round r.

Assumption 1. *The loss function $\mathcal{L}(w_k^{(r)})$ is c-strongly convex and L-smooth for $k = 1, ..., m$.*

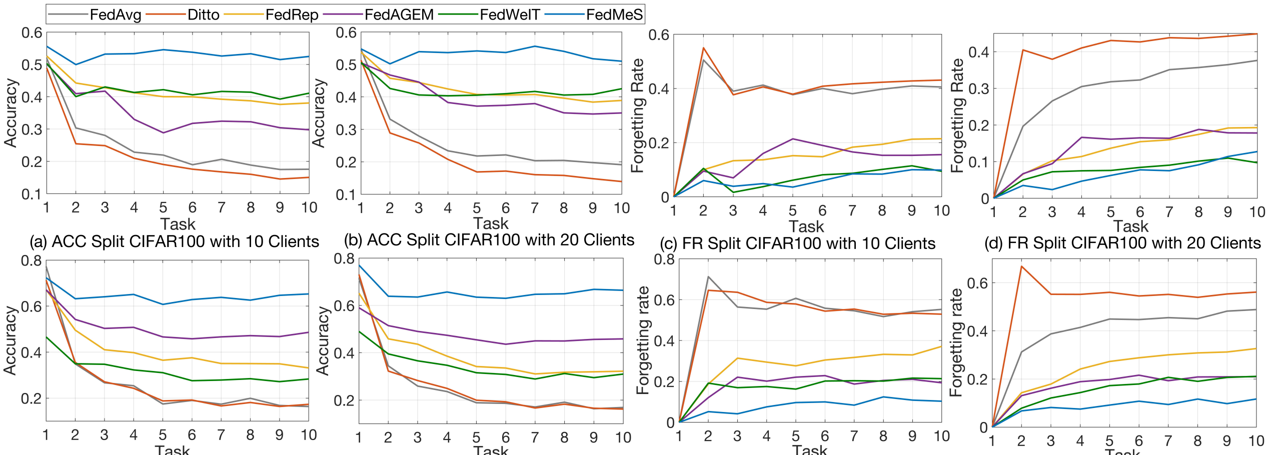
Assumption 2. *The expectation of stochastic gradients of the loss function $\mathcal{L}(w_k^{(r)})$ is uniformly bounded at all devices and all iterations, i.e.:*

$$\mathbb{E}[\|\nabla \mathcal{L}\left(w_k^{(r)}, \xi_k^r\right)\|^2] \le \sigma^2 \tag{12}$$

Assumption 3. *The global model converges with rate $g(r)$, i.e., there exists $g(r)$ such that $lim_{r\to\infty} g(r) = 0$, $\mathbb{E}[\|\mathbf{w}^{(r)} - \mathbf{w}^*\|^2] \le g(r)$.*

First, we discuss the situation where the constraint $\mathcal{L}(w_k^t; \mathcal{M}_k^t) \le \mathcal{L}(w_k^{t-1}; \mathcal{M}_k^t)$ in (2) is not satisfied, which corresponds to $\langle \nabla \mathcal{L}(w_k^t; \mathcal{D}_k^t), \nabla \mathcal{L}(w_k^t; \mathcal{M}_k^t)\rangle < 0$ in our algorithm. Under this circumstance, FedMeS starts to execute (5) to perform gradient correction. We denote the iteration index of repeating (5) as $s(s = 1, 2, ...)$, and $g(w_k^{(s)}, \xi_k^s)$ as the stochastic gradient of $\mathcal{L}(w_k^{(s)}; \mathcal{M}_k^t)$ at iteration s. Following *First and second moment limits* assumptions in [Bottou *et al.*, 2018], we make two assumptions below,

Assumption 4. *There exists scalars $\mu_G \ge \mu > 0$ such that for all $s \in \mathbb{N}$,*



(a) ACC Split CIFAR100 with 10 Clients (b) ACC Split CIFAR100 with 20 Clients (c) FR Split CIFAR100 with 10 Clients (d) FR Split CIFAR100 with 20 Clients

(e) ACC Split MiniImageNet with 10 Clients (f) ACC Split MiniImageNet with 20 Clients (g) FR Split MiniImageNet with 10 Clients (h) FR Split MiniImageNet with 20 Clients

Figure 3: (a - d) Average accuracy and Average forgetting rate among all clients in all learned tasks at x-th task on Split CIFAR-100 with 10 clients and 20 clients. (e - h) Average accuracy and Average forgetting rate among all clients in all learned tasks at x-th task on Split MiniImageNet with 10 clients and 20 clients.

$$\nabla \mathcal{L}\left(w_k^{(s)}; \mathcal{M}_k^t\right)^\top \mathbb{E}_{\xi_k^s}[g\left(w_k^{(s)}, \xi_k^s\right)] \geq \mu \|\nabla \mathcal{L}\left(w_k^{(s)}; \mathcal{M}_k^t\right)\|_2^2, \tag{13}$$

$$\|\mathbb{E}_{\xi_k^s}[g\left(w_k^{(s)}, \xi_k^s\right)]\|_2 \leq \mu_G \|\nabla \mathcal{L}\left(w_k^{(s)}; \mathcal{M}_k^t\right)\|_2$$

Assumption 5. *There exists scalars $M \geq 0$ and $M_V \geq 0$ such that, for all $s \in \mathbb{N}$,*

$$\mathbb{V}_{\xi_k^s}[g(w_k^{(s)}, \xi_k^s)] \leq M + M_V \|\nabla \mathcal{L}(w_k^{(s)}; \mathcal{M}_k^t)\|_2^2 \tag{14}$$

Theorem 6. *Under the assumptions above and with the updating rule of* (5)*, when $s \geq 2$, we have,*

$$\mathbb{E}[\mathcal{L}(w_k^{(s)}; \mathcal{M}_k^t) - \mathcal{L}(w_k^*; \mathcal{M}_k^t)] \leq \frac{LM}{2c^2\mu^2} \tag{15}$$

Using Theorem 6, as long as the constraint $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ is violated, the updating rule of (5) on w_k ensures that $\mathcal{L}(w_k^t; \mathcal{M}_k^t)$ converges to its local optimum. Therefore, after a certain number of iterations $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right)$ would be less than $\mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ with high probability, satisfying the constraint again.

Then, we analyze the situation where the inequality constraint of $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ is satisfied on client k. In this case, the PFCL objective for `FedMeS` can be simplified as (2). As is proved in Theorem 1 in [Li *et al.*, 2021] for (2), the following theorem holds.

Theorem 7. *With Assumptions 1, 2 and 3, there exists a constant C such that for $\lambda \in \mathbb{R}$, w_k^r converges to $w_k^* := \arg\min_{w_k^t} G_k\left(w_k^t; \mathbf{w}^*\right)$ with rate $Cg(r)$.*

By Theorem 6, even if the violation of $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ unfortunately occurs, as we only optimize on $\mathcal{D}_k^t$ and neglect $\mathcal{M}_k^t$ under the rule of (4), the off-track w_k can always be corrected to meet the constraint. By Theorem 7, the local model w_k^t would enjoy the same convergence rate with the global model $\mathbf{w}$ with a constant multiple gap when $\mathcal{L}\left(w_k^t; \mathcal{M}_k^t\right) \leq \mathcal{L}\left(w_k^{t-1}; \mathcal{M}_k^t\right)$ is satisfied. To sum up, while the gradient correction in (5) may occur several times during the training, the corrected local model would always converge to its optimum.

5 Experiments

5.1 Setup

Datasets and models. We select five commonly used public datasets: CIFAR-100 [Krizhevsky *et al.*, 2009], EMINIST (Extended-MNIST) [Cohen *et al.*, 2017], CORe50 [Lomonaco and Maltoni, 2017], MiniImageNet-100 [Vinyals *et al.*, 2016] and TinyImageNet-200 [Le and Yang, 2015]. For the purpose of PFCL evaluation, following the dataset splitting method proposed in [Rebuffi *et al.*, 2017] we split these datasets into multiple tasks forming four *cross-class datasets*: **Split CIFAR-100**: CIFAR-100 consists of 100 classes, we split them into 10 tasks with 10 classes each. **Split EMINIST**: We utilize 60 of the 62 categories in the original dataset, and split them into 10 tasks with 6 classes each. A total of 120,000 imagines are used. **Split CORe50**: CORe50 is specifically designed for assessing continual learning techniques and has 50 objects collected in 11 different sessions. We naturally split it into 11 tasks with 50 classes each. **Split MiniImageNet**: MiniImageNet-100 is commonly used in few-shot learning benchmarks, which consists of 50,000 data points and 10,000 testing points from 100 classes. We split this dataset into 10 tasks with 10 classes each.

Besides, we also design the *cross-domain datasets* to evaluate the cross domain performance for `FedMeS`. **Fusion Tasks-40**: This benchmark combines images from three distinct datasets: CIFAR-100, MiniImageNet-100, and TinyImageNet-200, resulting in a total of 400 classes. These classes are then divided into 40 non-IID tasks, with each task comprising 10 disjoint classes from the other tasks. This dataset is substantial, with 200,000 images from the three heterogeneous datasets.

We use 6-layer CNNs for the Split CIFAR-100 and Split CORe50, 2-layer CNNs for the Split EMINIST, and ResNet-18 [He *et al.*, 2016] for Split MiniImageNet and Fusion Tasks-40. For the task and dataset distributions, each client is assigned a unique task sequence, in which each task consists of randomly selected subset of 2-5 classes, with the goal of ensuring data heterogeneity.

Table 1: Acc_ALL(Acc) and average forgetting rate (FR) over all clients and all learned tasks.

DATASETS	SPLIT CIFAR-100				SPLIT EMNIST			
CLIENT NUMBER	10		20		10		20	
METHODS	ACC	FR	ACC	FR	ACC	FR	ACC	FR
FEDAVG	.249 ±.02	.35 ±.05	.263 ±.03	.29 ±.03	.450 ±.02	.49 ±.02	.465 ±.03	.48 ±.02
DITTO	.219 ±.02	.37 ±.03	.221 ±.02	.38 ±.06	.388 ±.03	.71 ±.04	.381 ±.02	.72 ±.03
FEDREP	.415 ±.04	.13 ±.02	.425 ±.06	.13 ±.02	.723 ±.04	.23 ±.04	.723 ±.06	.24 ±.06
FEDAGEM	.351 ±.04	.14 ±.03	.398 ±.05	.14 ±.03	.817 ±.04	.09 ±.05	.828 ±.04	.16 ±.04
FEDWEIT	.421 ±.05	.06 ±.03	.432 ±.05	.08 ±.03	.867 ±.03	.03 ±.01	.857 ±.03	.03 ±.02
FEDMES	**.530 ±.05**	**.06 ±.01**	**.533 ±.04**	**.06 ±.02**	**.935 ±.01**	**.01 ±.01**	**.964 ±.01**	**.01 ±.01**
DATASETS	SPLIT CORE50				SPLIT MINIIMAGENET			
CLIENT NUMBER	10		20		10		20	
METHODS	ACC	FR	ACC	FR	ACC	FR	ACC	FR
FEDAVG	.303 ±.01	.60 ±.02	.311 ±.01	.59 ±.02	.271 ±.02	.51 ±.04	.262 ±.02	.39 ±.05
DITTO	.266 ±.01	.79 ±.02	.267 ±.01	.81 ±.03	.264 ±.03	.49 ±.05	.265 ±.01	.51 ±.04
FEDREP	.547 ±.03	.34 ±.02	.551 ±.04	.35 ±.03	.410 ±.03	.30 ±.05	.388 ±.03	.24 ±.02
FEDAGEM	.731 ±.04	.18 ±.03	.741 ±.04	.20 ±.03	.504 ±.05	.18 ±.03	.477 ±.05	.17 ±.05
FEDWEIT	.595 ±.04	.17 ±.04	.589 ±.05	.19 ±.04	.319 ±.04	.17 ±.03	.343 ±.04	.15 ±.03
FEDMES	**.877 ±.04**	**.04 ±.01**	**.891 ±.04**	**.03 ±.01**	**.645 ±.05**	**.08 ±.02**	**.659 ±.03**	**.08 ±.02**

Metrics. There are mainly two kinds of metrics considered in this paper following [Chaudhry *et al.*, 2018].

- **Average Accuracy:** We apply four different kinds of average accuracy to evaluate the performance, we define the averaged accuracy of client k among all learned t tasks after the training of task t: $A_{k,t} = \frac{1}{t}\sum_{i=1}^{t} a_{t,i}^{k}$ as accuracy of client k at task t, where $a_{t,i}^{k}$ $(i < t)$ is the test accuracy of task i after the training of task t in client k; averaged accuracy of client k after training all T tasks $Acc\_Client_k = \frac{1}{T}\sum_{i=1}^{T} A_{k,i}$; average accuracy among all m clients at t-th task: $Acc\_Task_t = \frac{1}{m}\sum_{j=1}^{m} A_{j,t}$; average accuracy among all m clients in all learned T tasks after completing the training process of all tasks: $Acc\_ALL = \frac{1}{m}\frac{1}{T}\sum_{j=1}^{m}\sum_{i=1}^{T} A_{j,t}$.
- **Forgetting rate:** The forgetting rate is the averaged disparity between minimum task accuracy during continuous training, it can measure the performance preventing catastrophic forgetting. For the forgetting rate F_t^k of client k at t-th task, it is defined as $F_t^k = \frac{1}{t-1}\sum_{i=1}^{t-1} \max_{j\in\{1,\ldots,t-1\}}(a_{j,i}^{k} - a_{t,i}^{k})$.

Baselines. Since there is no particular algorithm for PFCL problems, we compare our proposed `FedMeS` with other personalized FL and FCL techniques.

- `FedAvg`: A classical FL method which the server aggregates the models for all clients according to a weighted averaging of model parameters in each clients.
- `Ditto`: A simple personalized FL method that utilizes a regularization term addressing the accuracy, robustness and fairness in FL while optimizing communication efficiency.
- `FedRep`: A personalized FL method that learns a divided model with global representation and personalized heads. Only the global representation is communicated between the server and clients, while each client adapts its personalized head locally.
- `FedAGEM`: This can be seen as a simple federated continual learning method that combines the conventional `A-GEM` method with `FedAvg`, achieved by applying `A-GEM` as the local training process on the client side.
- `FedWeIT`: state-of-the-art FCL approach based on parameter isolation, which uses masks to divide the model parameter into base parameters and task-adaptive parameters. The server averages the base parameters and broadcasts the task-adaptive parameters from all clients. Each client then trains all the task-adaptive parameters with the new task's weights based on a regularized objective.

All the experiments were conducted using PyTorch version 1.9 on a single machine equipped with two Intel Xeon 6226R CPUs, 384GB of memory, and four NVIDIA 3090 GPUs. The operating system utilized was Ubuntu 20.04.4. Each experiment is repeated for 5 times. The averages and standard deviations of the above metrics are reported.

5.2 Results

Cross-class Performance

Tables 1 presents Acc_ALL and forgetting rate in four cross-class datasets. For every cross-class datasets, our proposed `FedMeS` method outperforms all baselines in terms of average accuracy forgetting rate. It is observed that `FedWeIT` experienced a significant decline in performance when applied to the Split MiniImageNet. This is primarily due to its requirement of modifying the model structure to decompose the model parameters individually. Specifically, the downsample layers in ResNet-18 contain a relatively small number of essential parameters, and decomposing these layers negatively impacted the model's accuracy. Additionally, this modification process significantly increases the complexity of implementation. In contrast, `FedMeS` does not require such modifications, thus highlighting another advantage of `FedMeS` in terms of efficient implementation. Figure 3 and Figure 4 respectively presents the Acc_Task and Acc_Client for the Split CIFAR-100 datasets. For every task `FedMeS` achieves highest Acc_Task and lowest forgetting rate, and for each client `FedMeS` achieves the best Acc_Client. According to these results, we further make the following observations.

Catastrophic forgetting. Shown in Figure 3, catastrophic forgetting causes serious limitation for `FedAvg` and `Ditto`, as they do not incorporate previous task information in training. As a result, their model accuracies are inferior to other methods with much higher forgetting rates. `FedRep` exhibits certain robustness against heterogeneity over tasks and clients. However, without a designed mechanism to address catastrophic forgetting, it still subjects to gradual decay in average accuracy as new tasks arrive. The isolation method of `FedWeIT` to obtain adaptive weights on the clients cannot well maintain the knowledge from the previous tasks, resulting in a lower accuracy than `FedMeS` in every dataset. `FedMeS` is less affected by catastrophic forgetting due to its use of episodic memory to replay knowledge from previous tasks.

Client drift. `FedAGEM` and `FedWeIT` fail to effectively address data heterogeneity, resulting in inferior model performance compared to `FedMeS`. `FedWeIT` relies on the stored knowledge of all tasks at the server, which may dilute the impact of individual tasks of each client. In contrast, `FedMeS` achieves the highest accuracy in all settings thanks to its reliable personalization mechanism and local inference process. The advantage of `FedMeS` in adapting client drift is more evident from Figure 4, where `FedMeS` is shown to achieve the highest accuracy performance for all clients. Also, `FedMeS` has the narrowest shade area over all clients, indicating its ability to obtain consistent performance across all clients and all tasks.

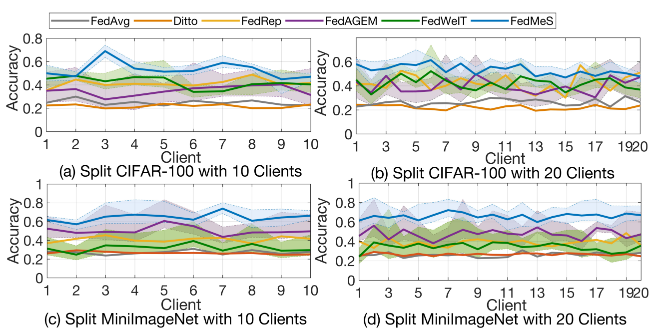


Figure 4: (a) and (b) Average accuracy of client x among all learned tasks on Split CIFAR-100 with 10 clients and 20 clients. (c) and (d) Average accuracy of client x among all learned tasks on Split MiniImageNet with 10 clients and 20 clients. The shade area is the accuracy range of tasks in each client.

Cross-domain Performance

Figure 5 presents the averaged test accuracy of each client across all tasks in the Fusion Tasks-40 dataset. The results indicate that the proposed `FedMeS` method outperforms the `FedWeIT` method, with higher average accuracy and lower forgetting rate among all clients and tasks. The poor performance of certain clients has a significant impact on other clients, and the isolation method employed by `FedWeIT` to mitigate catastrophic forgetting proves to be ineffective in this experiment with a large number of tasks. In contrast, the proposed `FedMeS` method demonstrates consistent performance across all tasks and clients, providing strong evidence for its effectiveness in addressing the catastrophic forgetting and client drift issues in the PFCL problem.

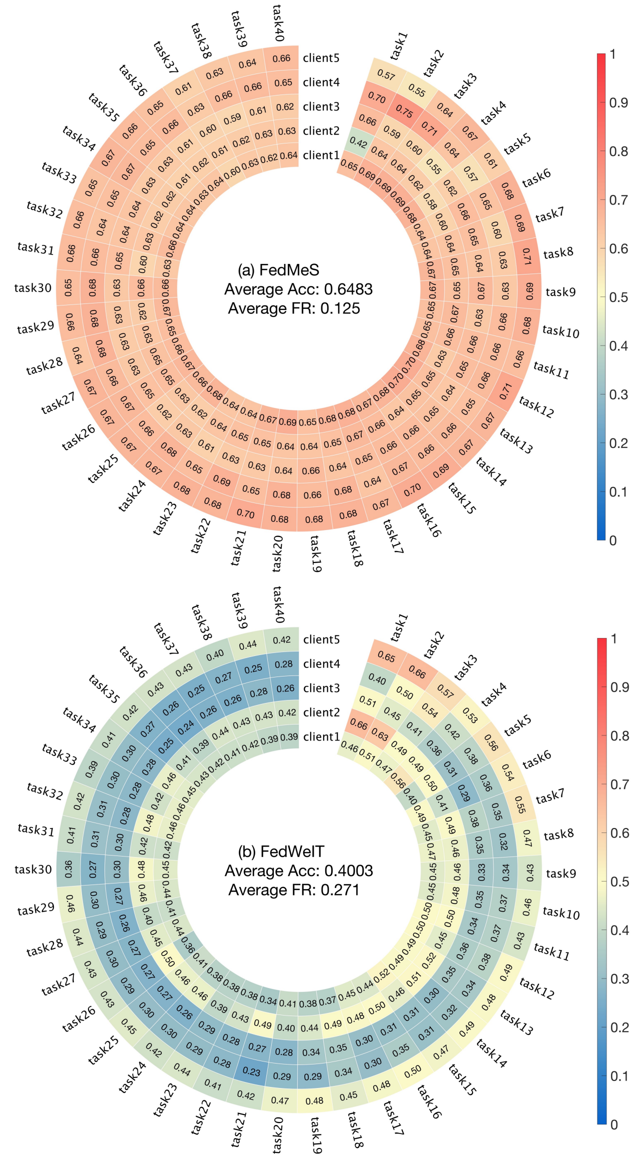


Figure 5: Averaged test accuracy of each client among all learned t tasks after the training of task t on Fusion Tasks-40.

6 Conclusion

This paper has presented an in-depth examination of the challenges associated with catastrophic forgetting and client drift in PFCL and proposed the `FedMeS` framework as a solution to these issues. `FedMeS` leverages a small reference memory in the local training process to replay knowledge from previous tasks to alleviate forgetting; and the same memory is also used for the inference process by applying KNN-based Gaussian inference to further improve model personalization capability. We thoroughly analyzed the convergence behavior of `FedMeS`, and performed extensive experiments over various PFCL tasks. For all experiments, `FedMeS` uniformly outperforms existing techniques in terms of prediction accuracy and forgetting rate.

Acknowledgement

This work is in part supported by the National Nature Science Foundation of China (NSFC) Grant 62106057, Guangzhou Municipal Science and Technology Guangzhou-HKUST(GZ) Joint Project 2023A03J0151 and Project 2023A03J0011, Foshan HKUST Projects FSUST20-FYTRI04B, and Guangdong Provincial Key Lab of Integrated Communication, Sensing and Computation for Ubiquitous Internet of Things.

References

[Achituve *et al.*, 2021] Idan Achituve, Aviv Shamsian, Aviv Navon, Gal Chechik, and Ethan Fetaya. Personalized federated learning with gaussian processes. *Advances in Neural Information Processing Systems*, 34:8392–8406, 2021.

[Bottou *et al.*, 2018] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[Casado *et al.*, 2020] Fernando E Casado, Dylan Lema, Roberto Iglesias, Carlos V Regueiro, and Senén Barro. Federated and continual learning for classification tasks in a society of devices. *arXiv preprint arXiv:2006.07129*, 2020.

[Chaudhry *et al.*, 2018] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

[Cheng *et al.*, 2021] Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is fine in federated learning. *arXiv preprint arXiv:2108.07313*, 2021.

[Cohen *et al.*, 2017] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[De Lange *et al.*, 2021] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[Delange *et al.*, 2021] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[Fallah *et al.*, 2020] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Huang *et al.*, 2021] 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 *AAAI*, pages 7865–7873, 2021.

[Isele and Cosgun, 2018] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[Khandelwal *et al.*, 2019] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2019.

[Khodak *et al.*, 2019] Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. *Advances in Neural Information Processing Systems*, 32, 2019.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[Le and Yang, 2015] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[Li *et al.*, 2021] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[Lomonaco and Maltoni, 2017] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26. PMLR, 2017.

[Lopez-Paz and Ranzato, 2017] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[Marfoq *et al.*, 2021] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

[Marfoq *et al.*, 2022] Othmane Marfoq, Giovanni Neglia, Richard Vidal, and Laetitia Kameni. Personalized federated learning through local memorization. In *International Conference on Machine Learning*, pages 15070–15092. PMLR, 2022.

[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. PMLR, 2017.

[Rebuffi *et al.*, 2017] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[Riemer *et al.*, 2018] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

[Shen *et al.*, 2020] Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Kun Kuang, Fei Wu, and Chao Wu. Federated mutual learning. *arXiv preprint arXiv:2006.16765*, 2020.

[Shi *et al.*, 2021] Yujun Shi, Li Yuan, Yunpeng Chen, and Jiashi Feng. Continual learning via bit-level information preserving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16674–16683, 2021.

[Shim *et al.*, 2021] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9630–9638, 2021.

[Shoham *et al.*, 2019] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019.

[Usmanova *et al.*, 2021] Anastasiia Usmanova, François Portet, Philippe Lalanda, and German Vega. A distillation-based approach integrating continual learning and federated learning for pervasive services. *arXiv preprint arXiv:2109.04197*, 2021.

[Venkatesha *et al.*, 2022] Yeshwanth Venkatesha, Youngeun Kim, Hyoungseob Park, Yuhang Li, and Priyadarshini Panda. Addressing client drift in federated continual learning with adaptive optimization. *Available at SSRN 4188586*, 2022.

[Vinyals *et al.*, 2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[Wang *et al.*, 2022] Liyuan Wang, Xingxing Zhang, Kuo Yang, Longhui Yu, Chongxuan Li, Lanqing Hong, Shifeng Zhang, Zhenguo Li, Yi Zhong, and Jun Zhu. Memory replay with data compression for continual learning. *arXiv preprint arXiv:2202.06592*, 2022.

[Yao and Sun, 2020] Xin Yao and Lifeng Sun. Continual local training for better initialization of federated models. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1736–1740. IEEE, 2020.

[Yoon *et al.*, 2021] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pages 12073–12086. PMLR, 2021.

[Yu *et al.*, 2020a] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020.

[Yu *et al.*, 2020b] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.

[Zhang *et al.*, 2022] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10174–10183, 2022.