pFedES: Generalized Proxy Feature Extractor Sharing for Model Heterogeneous Personalized Federated Learning

Liping Yi[1], Han Yu[2], Chao Ren[2], Gang Wang[1,*], Xiaoguang Liu[1,*], Xiaoxiao Li[2,3,4]

[1]College of Computer Science, TMCC, SysNet, DISSec, GTIISC, Nankai University, China
[2]College of Computing and Data Science, Nanyang Technological University, Singapore
[3]Department of Electrical and Computer Engineering, The University of British Columbia, Canada
[4]Vector Institute, Canada
{yiliping, wgzwp, liuxg}@nbjl.nankai.edu.cn
{han.yu, chao.ren}@ntu.edu.sg, xiaoxiao.li@ece.ubc.ca

Abstract

Federated learning (FL), as a privacy-preserving collaborative machine learning paradigm, has attracted significant interest from industry and academia. To allow each data owner (FL client) to train a heterogeneous and personalized local model based on its local data distribution, system resources and requirements on model structure, the field of model-heterogeneous personalized federated learning (MHPFL) has emerged. Existing MHPFL approaches either rely on the availability of a public dataset with special characteristics to facilitate knowledge transfer, incur high computational and communication costs, or face potential model leakage risks. To address these limitations, we propose a model-heterogeneous personalized Federated learning approach based on generalized proxy feature Extractor Sharing (pFedES) for supervised image classification tasks. (1) We devise a shared small proxy homogeneous feature extractor before each client's heterogeneous local model. (2) Clients train them via the proposed iterative learning to enable the exchange of global generalized knowledge and local personalized knowledge. (3) The small proxy local homogeneous extractors produced after local training are uploaded to the server for aggregation to facilitate knowledge fusion across clients. We theoretically prove pFedES converges with a non-convex convergence rate $\mathcal{O}(1/T)$. Experiments on 3 benchmark datasets against 9 baselines demonstrate that pFedES performs state-of-the-art model accuracy while maintaining efficient communication and computation.

Introduction

Federated learning (FL) (Goebel et al. 2023) is an emerging collaborative machine learning paradigm. It often relies on a central FL server to coordinate decentralized data owners (FL clients) to train a shared global FL model in a privacy-preserving manner (Kairouz et al. 2021).

In a traditional FL system, the server first broadcasts the global model to clients. Clients then train the received global model on their respective local data and upload the trained local models to the server. The server aggregates the received local models to update the global model. These steps are repeated until the global model converges. During the entire training process, only models are transmitted between the server and clients, while the data never leaves clients, thereby protecting data privacy.

The above prevailing mode of FL follows the model homogeneity assumption. It requires that all clients train local models with the same structures as the global model. Thus, it is still not equipped to address the following challenges often encountered in practice:

- **Data Heterogeneity**. Clients have non-independently and identically distributed (non-IID) data (Tan et al. 2022a; Yi et al. 2023c). Aggregating biased local models trained on such data might lead to a sub-optimal global model (Zhu et al. 2021a), which may perform worse than the local models trained solely on clients.
- **System Resource Heterogeneity**. FL clients are often devices (*e.g.*, mobile phones, autonomous vehicles) with divergent system resources (computational power, communication bandwidth, etc (Jiang et al. 2022; Yi et al. 2022)). The traditional FL approach limits all clients to train the smallest model supported by the lowest-end client, leading to model performance bottlenecks and wasted system resources of high-end devices.
- **Model Ownership Heterogeneity**. When FL clients are companies, they are often willing to fine-tune heterogeneous models from their internal repositories via FL. Due to intellectual property considerations (Ye et al. 2023), they are reluctant to expose model structures to others.

The field of model-heterogeneous personalized federated learning (MHPFL) has emerged to address these challenges. It enables each client to train a personalized and heterogeneous model based on its local data distribution, system resources, and model structure requirements (Yi et al. 2023a,b, 2024a,b).

Prior efforts for MHPFL can be divided into three main branches: Knowledge distillation-based MHPFL methods either depend on a public dataset which is not always available (Lin et al. 2020), or introduce heavy communication costs (Cheng et al. 2021), computational overheads (Huang et al. 2022b), and risks of privacy leakage (Takahashi et al. 2023; Tan et al. 2022b). Mutual learning-based MHPFL

*Corresponding authors

methods (Shen et al. 2020; Wu et al. 2022) train a local heterogeneous large model and a homogeneous small model on clients with a mutual learning approach and share the homogeneous ones across clients. Lacking discussions about the relationship between the two models in model capacity or structure leads to limited model performance. Model split-based MHPFL methods (Liang et al. 2020; Collins et al. 2021) split each client's local heterogeneous model into a homogeneous part and a heterogeneous part, and share the homogeneous parts across clients, leading to model performance bottleneck and leakage of the shared model structure.

To tackle the above limitations, we propose an efficient model-heterogeneous personalized Federated learning framework based on generalized proxy small homogeneous feature Extractor Sharing (pFedES) for supervised image classification tasks. (1) We design a proxy small homogeneous feature extractor before each client's local heterogeneous model. (2) Clients train them with the proposed iterative learning method to facilitate the bidirectional exchange of the globally generalized knowledge carried by the shared proxy homogeneous feature extractor and the locally personalized knowledge absorbed by the local heterogeneous model. (3) The updated local proxy homogeneous feature extractors are uploaded to the FL server for aggregation to facilitate knowledge sharing across heterogeneous clients. Only the proxy small homogeneous feature extractors are transmitted between the server and clients, pFedES incurs low communication costs and protects the privacy of local data and model structures. Only one small homogeneous feature extractor is being trained additionally by each client, introducing low extra computational overheads.

Our main contributions are summarized as follows:

- We design a proxy homogeneous feature extractor shared for information fusion. It generates enhanced data with generalized global knowledge for clients. The devised iterative training alternatively trains it with the heterogeneous client model for bidirectional knowledge transfer.
- Through theoretical analysis, we prove the non-convex convergence rate $\mathcal{O}(1/T)$ of pFedES and verify that it converges over wall-to-wall time.
- Extensive experiments on 3 benchmark datasets against 9 state-of-the-art MHPFL methods demonstrate the superior performance of pFedES, it achieves up to 1.29% higher accuracy, while reducing communication and computation costs by 99.6% and 82.9%.

Related Work

Existing MHPFL methods have two families: a) clients hold different subnets of the global model, and heterogeneous subnets can be aggregated on the server, such as FedRolex (Alam et al. 2022), FLASH (Babakniya et al. 2023), InCO (Chan et al. 2024), HeteroFL (Diao 2021), FjORD (Horváth 2021), HFL (Lu et al. 2022), Fed2 (Yu et al. 2021), FedResCuE (Zhu et al. 2022); b) clients hold models with completely different structures, which cannot be aggregated directly on the server, which includes:

Knowledge Distillation-based MHPFL. Some methods (Cronus (Chang et al. 2021), FedGEMS (Cheng et al. 2021), Fed-ET (Cho et al. 2022), FSFL (Huang et al. 2022a), FCCL (Huang et al. 2022b), DS-FL (Itahara et al. 2023), FedMD (Li and Wang 2019), FedKT (Li et al. 2021), FedDF (Lin et al. 2020), FedHeNN (Makhija et al. 2022), FedKEM (Nguyen et al. 2023), KRR-KD (Park et al. 2023), FedAUX (Sattler et al. 2021), CFD (Sattler et al. 2022), FedKEMF (Yu et al. 2022) and KT-pFL (Zhang et al. 2021)) allow the server to aggregate the output logits of local heterogeneous models on a public dataset to construct the global logits. However, the public dataset is not always accessible and should have the same distribution as private data. Data privacy may also be compromised by paired-logits inversion attacks (Takahashi et al. 2023). pFedHR (Wang et al. 2023) allows the server to split heterogeneous client models by layers and re-splice candidate models which are chosen for different clients by model similarity on a public dataset, also facing the above issues.

Some methods (FedIOD (Gong et al. 2024), DFRD (Luo et al. 2023), FedGD (Zhang et al. 2023), FedZKT (Zhang et al. 2022), FedGen (Zhu et al. 2021b)) introduce zero-shot knowledge distillation to FL for generating a shared dataset by training a generator, which is time-consuming. Other methods (HFD (Ahn et al. 2019, 2020), FedGKT (He et al. 2020), FD (Jeong et al. 2018), FedProto (Tan et al. 2022b), FedTGP (Zhang et al. 2024)) allow each client to upload the average logits or representations of local seen-class samples to the server for aggregation to produce the global class-logits or representations, which are sent back to clients and used to calculate the distillation loss with local logits for each local data sample, incurring high computational costs. The uploaded classes might leak privacy.

Mutual Learning-based MHPFL. In FML (Shen et al. 2020), FedKD (Wu et al. 2022), ProxyFL (Kalra et al. 2023) and FedAPEN (Qin et al. 2023), each client owns a small homogeneous model and a large heterogeneous model trained via mutual learning. The trained homogeneous models are aggregated by the server for knowledge fusion. However, they do not explore the relationship between the two models in model structure and parameter capacity, leading to limited model performance.

Model Split-based MHPFL. They split each client's local model into a feature extractor and a classifier, and only one part is shared. FedMatch (Chen et al. 2021), FedRep (Collins et al. 2021), FedBABU (Oh et al. 2022), FedAlt/FedSim (Pillutla et al. 2022) share the homogeneous feature extractor to enhance model generalization while preserving the personalized local classifier. In contrast, FedClassAvg (Jang et al. 2022), LG-FedAvg (Liang et al. 2020), FedGH (Yi et al. 2023a), and CHFL (Liu et al. 2022) share the homogeneous classifier to improve model classification while preserving the personalized local feature extractor. Since only part of the entire model is shared, the performance of local heterogeneous models faces bottlenecks. They are also prone to leaking the structure of the shared model part.

Our Insight. pFedES always keeps each client's local data and local heterogeneous model on clients. It adds a proxy shared global homogeneous feature extractor before the local heterogeneous model. Aggregating homogeneous

extractors implements knowledge fusion across heterogeneous clients. The shared global homogeneous extractor carries generalized global knowledge and the local heterogeneous model has personalized local knowledge, training them by the devised iterative training enhances the generalization and personalization of local models.

Preliminaries

`FedAvg` (McMahan et al. 2017) is a typical FL algorithm. It assumes that a FL system consists of one central FL server and N FL clients. In each communication round, the server randomly selects a fraction C of N clients (the selected client set is $\boldsymbol{S}$, $|\boldsymbol{S}| = \lfloor C \cdot N \rfloor = K$), and broadcasts the global model $\mathcal{F}(\omega)$ ($\mathcal{F}(\cdot)$ is model structure, ω are model parameters) to them. Client k trains the received global model $\mathcal{F}(\omega)$ on its local dataset D_k ($D_k \sim P_k$, D_k obeys distribution P_k, *i.e.*, local data from different clients are non-IID) to obtain an updated local model $\mathcal{F}(\omega_k)$ via gradient descent, *i.e.*, $\omega_k \leftarrow \omega - \eta \nabla \ell(\mathcal{F}(\boldsymbol{x}_i; \omega), y_i)$. $\ell(\mathcal{F}(\boldsymbol{x}_i; \omega), y_i)$ is the loss of the global model $\mathcal{F}(\omega)$ on the sample $(\boldsymbol{x}_i, y_i) \in D_k$. The updated local model $\mathcal{F}(\omega_k)$ is uploaded to the server. The server aggregates the received local models from K clients via weighted averaging to update the global model, *i.e.*, $\omega = \sum_{k \in \boldsymbol{S}} \frac{n_k}{n} \omega_k$ ($n_k = |D_k|$ is the data volume of client k, $n = \sum_{k=0}^{N-1} n_k$ is total data volume of all clients).

This typical FL algorithm requires all clients to train **homogeneous** models. Its training objective is to minimize the average loss of the global model $\mathcal{F}(\omega)$ on all client data,

$$\min_{\omega \in \mathbb{R}^d} \sum_{k=0}^{N-1} \frac{n_k}{n} \mathcal{L}_k(\mathcal{F}(\omega); D_k), \tag{1}$$

where the parameters of the global model ω are d-dimensional real numbers. $\mathcal{L}_k(\mathcal{F}(\omega); D_k)$ is the average loss of the global model $\mathcal{F}(\omega)$ on client k's local data D_k.

Our objective in this work is to study MHPFL in the context of supervised image classification tasks. We assume that all clients execute the same image classification tasks, and different clients hold **heterogeneous** local models with different structures, $\mathcal{F}_k(\omega_k)$ ($\mathcal{F}_k(\cdot)$ is the heterogeneous model structure, ω_k denotes personalized model parameters). `pFedES` aims to minimize the loss sum of all local heterogeneous personalized models on their local data,

$$\min_{\omega_0, \ldots, \omega_{N-1} \in \mathbb{R}^{d_0, \ldots, d_{N-1}}} \sum_{k=0}^{N-1} \mathcal{L}_k(\mathcal{F}_k(\omega_k); D_k), \tag{2}$$

where the parameters $\omega_0, \ldots, \omega_{N-1}$ of local heterogeneous models are $d_0, \ldots, d_{N-1}$-dimensional real numbers.

The Proposed pFedES Approach

To achieve the above objective, we devise a proxy sharing small homogeneous feature extractor $\mathcal{G}(\theta)$ ($\mathcal{G}(\cdot)$ is the extractor structure, θ denotes model parameters) before each FL client k's local heterogeneous model $\mathcal{F}_k(\omega_k)$ in Figure 1(a).

Proxy Homogeneous Feature Extractor Structure

It's practical to introduce low computational overhead consumed by training the extra proxy homogeneous feature extractors on clients while ensuring well-performed models. Therefore, we design a small CNN model consisting of two convolutional layers with *'padding=same'* as the proxy homogeneous feature extractor in Figure 1(b), which guarantees the dimensions of the input original data and the output enhanced data are the same. Other feature extractor structures that satisfy this dimension condition can also be applied into `pFedES`, and the user can tailor the homogeneous feature extractor structure according to practical requirements.

Overview

In communication round t, as depicted in Figure 1(a), `pFedES` performs the following steps:

1. The server randomly selects K clients among N clients and broadcasts the global proxy homogeneous feature extractor $\mathcal{G}(\theta^{t-1})$ to the selected clients $\boldsymbol{S}^t$.
2. Client $k \in \boldsymbol{S}^t$ trains the received global proxy homogeneous feature extractor $\mathcal{G}(\theta^{t-1})$ and local heterogeneous model $\mathcal{F}_k(\omega_k^{t-1})$ on local data D_k following the proposed iterative training method. Afterwards, the local proxy homogeneous feature extractor $\mathcal{G}(\theta_k^t)$ is uploaded to the FL server, while the heterogeneous local model $\mathcal{F}_k(\omega_k^t)$ remains in client k.
3. The server aggregates received homogeneous feature extractors $\mathcal{G}(\theta_k^t)$ to update the global proxy homogeneous feature extractor $\mathcal{G}(\theta^t)$.

The above steps are repeated until all clients' heterogeneous local models $\mathcal{F}_k(\omega_k)$ converge. Finally, only each client's personalized heterogeneous local model $\mathcal{F}_k(\omega_k)$ is used for **inference**. The details of `pFedES` are described in Algorithm 1 (Appendix A). [1]

Based on the above workflow, the training objective of `pFedES` in Eq. (2) can be re-expressed as:

$$\min_{\theta, \omega_0, \ldots, \omega_{N-1} \in \mathbb{R}^{d_0, \ldots, d_{N-1}}} \sum_{k=0}^{N-1} \mathcal{L}_k(\{\mathcal{G}(\theta), \mathcal{F}_k(\omega_k)\}; D_k). \tag{3}$$

Iterative Training and Model Aggregation

Motivation. It's intuitive to train the proxy homogeneous feature extractor and the local heterogeneous model by gradient descent simultaneously. However, this training manner faces two issues: 1) The complete global knowledge from the global proxy homogeneous feature extractor is only transferred to clients in the first training batch, and it fades in the remaining training batches. This incomplete global knowledge transfer might degrade model performance. 2) Training a larger model combined with the two models increases the memory burden, and it may be trained insufficiently on limited local data, which also may lead to degraded model performance.

[1] Appendices: https://github.com/LipingYi/pFedES

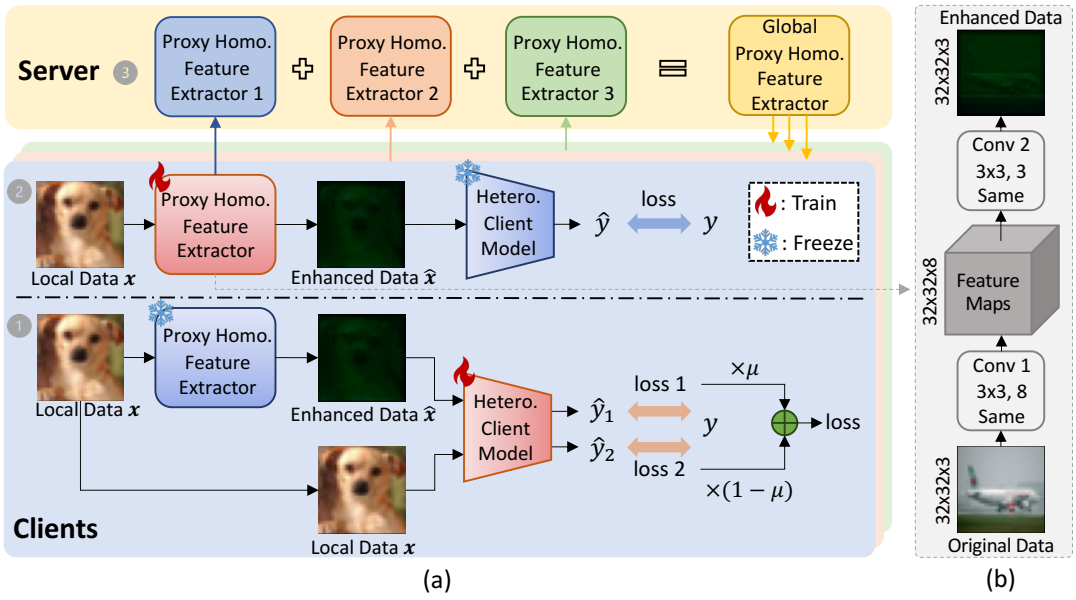

Figure 1: (a): Workflow of pFedES. (b): Proxy Homogeneous feature extractor structure.

To facilitate effective global and local complete knowledge transfer, we design a novel **iterative training** method to train each client's proxy homogeneous feature extractor and heterogeneous local model.

Freeze Proxy Extractor, Train Heterogeneous Model We first freeze the global proxy homogeneous feature extractor from the server and train the local heterogeneous model on local data, which transfers global generalized knowledge from the global proxy homogeneous feature extractor to clients.

As Step ① in Figure 1(a), in communication round t, client k freezes the global proxy homogeneous feature extractor $\mathcal{G}(\theta^{t-1})$ received from the server and trains the heterogeneous local model $\mathcal{F}_k(\omega_k^{t-1})$ with two inputs: 1) client k's original local data $(\boldsymbol{x}, y) \in D_k$ as prompt information being inputted into $\mathcal{G}(\theta^{t-1})$ to obtain the enhanced data $\hat{\boldsymbol{x}} = \mathcal{G}(\theta^{t-1}; \boldsymbol{x})$ ($\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$ are of the same dimension), which contains both global generalized knowledge and local personalized knowledge; 2) client k's original data $(\boldsymbol{x}, y) \in D_k$. They are input into $\mathcal{F}_k(\omega_k^{t-1})$ to output predictions:

$$\hat{y}_1 = \mathcal{F}_k(\omega_k^{t-1}; \hat{\boldsymbol{x}}); \hat{y}_2 = \mathcal{F}_k(\omega_k^{t-1}; \boldsymbol{x}). \tag{4}$$

Then, client k calculates the hard loss ℓ_1, ℓ_2 (*e.g.*, cross-entropy loss (Zhang and Sabuncu 2018)) between the predictions $\hat{y}_1$, $\hat{y}_2$ and the label y, respectively,

$$\ell_1 = \ell(\hat{y}_1, y), \ \ell_2 = \ell(\hat{y}_2, y). \tag{5}$$

In earlier communication rounds, the global proxy homogeneous feature extractor might be unstable. The enhanced data produced by a poor global proxy homogeneous feature extractor might be of low quality, thereby negatively impacting the performance of the heterogeneous local model. To balance the global generalized knowledge and the local personalized knowledge carried by the two types of input data, we compute a weighted sum of the hard loss ℓ_1, ℓ_2 of the two data inputs as the integrated loss ℓ_ω of the heterogeneous local model:

$$\ell_\omega = \mu \cdot \ell_1 + (1-\mu) \cdot \ell_2, \ \mu \in (0, 0.5]. \tag{6}$$

With the integrated loss ℓ_ω, the parameters of the heterogeneous local model are updated via gradient descent (*e.g.*, SGD (Ruder 2016)):

$$\omega_k^t \leftarrow \omega_k^{t-1} - \eta_\omega \nabla \ell_\omega. \tag{7}$$

η_ω is the learning rate of the heterogeneous local model. During this training step, the complete global knowledge from the frozen global proxy homogeneous feature extractor is transferred to the heterogeneous local model by enhanced data across all training batches, thereby improving local model generalization. The local personalized knowledge from the original local data is further incorporated into the heterogeneous local model, thereby enhancing local model personalization.

Freeze Heterogeneous Model, Train Proxy Extractor We freeze the local heterogeneous model updated in the first training step and train the proxy homogeneous feature extractor on local data to transfer local knowledge to global.

Following Step ② shown in Figure 1(a), the heterogeneous local model $\mathcal{F}_k(\omega_k^t)$ updated in Step ① is frozen and the global proxy homogeneous feature extractor $\mathcal{G}(\theta^{t-1})$ is trained. Client k inputs its local data $(\boldsymbol{x}, y) \in D_k$ into $\mathcal{G}(\theta^{t-1})$ to generate the enhanced data $\hat{\boldsymbol{x}} = \mathcal{G}(\theta^{t-1}; \boldsymbol{x})$. Then, it inputs $\hat{\boldsymbol{x}}$ into the frozen $\mathcal{F}_k(\omega_k^t)$ to obtain:

$$\hat{y} = \mathcal{F}_k(\omega_k^t; \hat{\boldsymbol{x}}). \tag{8}$$

Client k computes the loss ℓ_θ of prediction $\hat{y}$ and label y:

$$\ell_\theta = \ell(\hat{y}, y). \tag{9}$$

Client k updates the proxy extractor via gradient descent:

$$\theta_k^t \leftarrow \theta^{t-1} - \eta_\theta \nabla \ell_\theta. \tag{10}$$

η_θ is the learning rate of the proxy homogeneous feature extractor. During this step, personalized local knowledge is transferred into the updated proxy homogeneous local extractor, which is uploaded to the FL server for aggregation.

Proxy Homogeneous Feature Extractor Aggregation After receiving local proxy homogeneous feature extractors $\mathcal{G}(\theta_k^t)$ from clients, the server aggregates them to facilitate knowledge fusion across heterogeneous clients:

$$\theta^t = \sum_{k \in \boldsymbol{S}^t} \frac{n_k}{n} \theta_k^t. \tag{11}$$

Discussion

In this section, we discuss pFedES's following aspects:

Computational Cost. Besides training heterogeneous local models $\mathcal{F}_k(\omega_k)$, clients train small proxy homogeneous feature extractors $\mathcal{G}(\theta)$. Since we design a small CNN with two convolutional layers as the proxy feature extractor $\mathcal{G}(\theta)$, training it incurs low extra computational costs per round.

Communication Cost. Since only the small proxy homogeneous feature extractors $\mathcal{G}(\theta)$ are transmitted between the server and clients, pFedES incurs lower communication costs than transmitting complete models as in FedAvg.

Privacy Preservation. For data privacy, the client and the server only exchange the proxy homogeneous feature extractor $\mathcal{G}(\theta)$ while local data $\boldsymbol{x} \in D_k$ are always stored within clients, hindering data exploration outside the client. Only using proxy extractors $\mathcal{G}(\theta)$ while lacking enhanced data $\hat{\boldsymbol{x}}$ can not inversely infer original data $\boldsymbol{x}$. For model privacy, since only the proxy homogeneous feature extractor $\mathcal{G}(\theta)$ is transmitted, the client's heterogeneous local model $\mathcal{F}_k(\omega_k)$ never leaves the client, the server and the communication channel eavesdropper cannot steal client models $\mathcal{F}_k(\omega_k)$, protecting users' model intellectual property.

Convergence Analysis

We declare some notations. We use t to denote a communication round and $e \in \{0, 1, \ldots, E\}$ to denote the iterations of local training. $tE + e$ is the e-th iteration in the $(t+1)$-th round. $tE + 0$ indicates that in the $(t+1)$-th round, before local model training, clients receive the global extractor $\mathcal{G}(\theta^t)$ aggregated in the t-th round. $tE + E$ is the last iteration of local training, indicating the end of local training in the $(t+1)$-th round. We denote the combination of the frozen homogeneous feature extractor with $\mathcal{G}(\theta)$, and the training of the heterogeneous local model $\mathcal{F}_k(\omega_k)$ at the first branch of Step 1 during the iterative training process as $\mathcal{H}_k(\varphi_k) = \mathcal{G}(\theta) \circ \mathcal{F}_k(\omega_k)$. We assume $\mathcal{F}_k(\omega_k)$ and the combined model $\mathcal{H}_k(\varphi_k)$ use the same learning rate $\eta = \eta_\omega = \eta_\varphi$.

Assumption 1. ***Lipschitz Smoothness***. *The gradients of Client k's heterogeneous local model are $L1$–Lipschitz smooth, i.e.,*

$$\|\nabla \mathcal{L}_k^{t_1}(\omega_k^{t_1}; \boldsymbol{x}, y) - \nabla \mathcal{L}_k^{t_2}(\omega_k^{t_2}; \boldsymbol{x}, y)\| \leqslant L_1 \|\omega_k^{t_1} - \omega_k^{t_2}\|, \quad \forall t_1, t_2 > 0, k \in \{0, 1, \ldots, N-1\}, (\boldsymbol{x}, y) \in D_k. \tag{12}$$

The above formulation can be expressed as:

$$\mathcal{L}_k^{t_1} - \mathcal{L}_k^{t_2} \leqslant \langle \nabla \mathcal{L}_k^{t_2}, (\omega_k^{t_1} - \omega_k^{t_2}) \rangle + \frac{L_1}{2} \|\omega_k^{t_1} - \omega_k^{t_2}\|_2^2. \tag{13}$$

Assumption 2. ***Unbiased Gradient and Bounded Variance***. *Random gradient $g_{\omega,k}^t = \nabla \mathcal{L}_k^t(\omega_k^t; \mathcal{B}_k^t)$ ($\mathcal{B}$ is a batch of local data) of each client's heterogeneous local model $\mathcal{F}_k(\omega_k)$ is unbiased, and random gradient $g_{\varphi,k}^t = \nabla \mathcal{L}_k^t(\varphi_k^t; \mathcal{B}_k^t)$ of each client's combined model $\mathcal{H}_k(\varphi_k)$ is also unbiased, i.e.,*

$$\begin{aligned} \mathbb{E}_{\mathcal{B}_k^t \subseteq D_k}[g_{\omega,k}^t] &= \nabla \mathcal{L}_k^t(\omega_k^t), \\ \mathbb{E}_{\mathcal{B}_k^t \subseteq D_k}[g_{\varphi,k}^t] &= \nabla \mathcal{L}_k^t(\varphi_k^t), \end{aligned} \tag{14}$$

and the variance of random gradient $g_{\omega,k}^t$ and $g_{\varphi,k}^t$ is bounded by:

$$\begin{aligned} \mathbb{E}_{\mathcal{B}_k^t \subseteq D_k}[\|\nabla \mathcal{L}_k^t(\omega_k^t; \mathcal{B}_k^t) - \nabla \mathcal{L}_k^t(\omega_k^t)\|_2^2] &\leqslant \sigma^2, \\ \mathbb{E}_{\mathcal{B}_k^t \subseteq D_k}[\|\nabla \mathcal{L}_k^t(\varphi_k^t; \mathcal{B}_k^t) - \nabla \mathcal{L}_k^t(\varphi_k^t)\|_2^2] &\leqslant \delta^2. \end{aligned} \tag{15}$$

With these assumptions, we derive the following lemma and theorem (proofs can be found in Appendices C and D).

Lemma 1. *Based on Assumptions 1 and 2, during local iterations $\{0, 1, ..., E\}$ in the $(t+1)$-th FL training round, the loss of an arbitrary client's heterogeneous local model is bounded by:*

$$\mathbb{E}[\mathcal{L}_{(t+1)E}] \leq \mathcal{L}_{tE+0} + (\frac{L_1 \eta^2 \tilde{\mu}^2}{2} - \eta\tilde{\mu}) \sum_{e=0}^{E-1} \|\nabla \mathcal{L}_{tE+e}\|_2^2 + \frac{L_1 \eta^2 (\sigma^2 + \delta^2)}{2}. \tag{16}$$

where $\tilde{\mu} = 1 - \mu$, $\mu \in (0, 0.5]$, $\tilde{\mu} \in [0.5, 1)$.

Theorem 1. ***Non-convex convergence rate of pFedES***. *Based on the above assumptions and lemma, for an arbitrary client and any $\epsilon > 0$, the following inequality holds:*

$$\begin{aligned} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{e=0}^{E-1} \|\nabla \mathcal{L}_{tE+e}\|_2^2 &\leq \frac{\frac{1}{T} \sum_{t=0}^{T-1} (\mathcal{L}_{tE+0} - \mathbb{E}[\mathcal{L}_{(t+1)E}])}{\eta\tilde{\mu} - \frac{L_1 \eta^2 \tilde{\mu}^2}{2}} \\ &\quad + \frac{\frac{L_1 \eta^2 (\sigma^2 + \delta^2)}{2}}{\eta\tilde{\mu} - \frac{L_1 \eta^2 \tilde{\mu}^2}{2}} < \epsilon, \\ s.t. \eta &< \frac{2\epsilon\tilde{\mu}}{L_1(\sigma^2 + \delta^2 + \tilde{\mu}^2 \epsilon)}. \end{aligned} \tag{17}$$

Hence, under pFedES, a client's heterogeneous local model converges at a non-convex rate of $\epsilon \sim \mathcal{O}(\frac{1}{T})$.

Experimental Evaluation

To evaluate pFedES, we compare it against 9 state-of-the-art MHPFL approaches on 3 benchmark datasets. The experiments are conducted with Pytorch on 4 NVIDIA GeForce RTX 3090 GPUs with 24G memory.[2]

[2]Codes: https://github.com/LipingYi/pFedES

Method	N=10, C=100%			N=50, C=20%			N=100, C=10%		
	MNIST	CIFAR-10	CIFAR-100	MNIST	CIFAR-10	CIFAR-100	MNIST	CIFAR-10	CIFAR-100
Standalone	99.93±0.74	96.35±0.56	74.32±0.89	99.95±0.23	95.25±0.67	62.38±0.41	99.81±0.92	92.58±0.35	54.93±0.78
LG-FedAvg	99.90±0.50	96.47±0.87	73.43±0.42	99.90±0.95	94.20±0.28	61.77±0.63	99.29±0.54	90.25±0.79	46.64±0.36
FedGH	99.93±0.27	96.51±0.23	74.39±0.05	99.96±0.37	95.28±0.20	62.61±0.95	99.82±0.56	92.59±0.67	54.95±0.72
FML	99.93±0.91	94.83±0.48	70.02±0.22	99.91±0.67	93.18±0.80	57.56±0.53	99.42±0.45	87.93±0.29	46.20±0.77
FedKD	99.71±0.34	94.77±0.61	70.04±0.88	99.75±0.52	92.93±0.43	57.56±0.98	99.31±0.30	90.23±0.71	50.99±0.55
FedAPEN	99.94±0.84	95.38±0.26	71.48±0.69	99.95±0.73	93.31±0.67	57.62±0.85	99.40±0.32	87.97±0.91	46.85±0.47
FD	99.87±0.56	96.30±0.74	-	-	-	-	-	-	-
FedProto	99.91±0.29	95.83±0.88	72.79±0.64	99.95±0.43	95.10±0.73	62.55±0.51	99.51±0.39	91.19±0.82	54.01±0.28
FedTGP	99.93±0.63	96.14±0.85	72.79±0.28	99.95±0.46	94.93±0.32	62.60±0.75	99.50±0.10	92.22±0.77	54.24±0.57
pFedES	**99.95**±0.10	**96.68**±0.31	**74.42**±0.23	**100.00**±0.16	**95.74**±0.24	**63.55**±0.39	**99.93**±0.34	**92.89**±0.12	**55.15**±0.04

Note: N: number of clients. C: fraction of participating clients. '-' denotes failure to converge. " ": best method. " ": best baseline.

Table 1: Average test accuracy (%) for model-homogeneous FL.

Method	N=10, C=100%			N=50, C=20%			N=100, C=10%		
	MNIST	CIFAR-10	CIFAR-100	MNIST	CIFAR-10	CIFAR-100	MNIST	CIFAR-10	CIFAR-100
Standalone	99.94±0.45	96.53±0.68	72.53±0.29	99.95±0.83	95.14±0.76	62.71±0.52	99.60±0.91	91.97±0.37	53.04±0.24
LG-FedAvg	99.88±0.64	96.30±0.58	72.20±0.19	99.86±0.80	94.83±0.93	60.95±0.47	99.07±0.72	91.27±0.25	45.83±0.67
FedGH	99.95±0.26	96.55±0.62	72.60±0.94	99.96±0.28	95.59±0.51	63.29±0.15	99.60±0.05	92.51±0.18	53.69±0.30
FML	53.20±0.84	-	-	53.21±0.50	-	-	-	-	-
FedKD	55.36±0.95	80.20±0.30	53.23±0.77	54.94±0.42	77.37±0.61	44.27±0.85	55.69±0.22	73.21±0.56	37.21±0.98
FedAPEN	53.54±0.34	-	-	53.92±0.70	-	-	46.97±0.53	-	-
FD	99.93±0.89	96.21±0.46	-	59.08±0.12	-	-	-	-	-
FedProto	99.95±0.81	96.51±0.73	72.59±0.60	99.95±0.28	95.48±0.63	62.69±0.94	99.49±0.55	92.49±0.26	53.67±0.79
FedTGP	99.91±0.50	96.43±0.62	72.36±0.81	99.91±0.38	95.53±0.45	63.28±0.07	99.49±0.64	92.50±0.31	53.20±0.78
pFedES	**99.96**±0.14	**96.70**±0.09	**73.89**±0.26	**99.98**±0.19	**95.79**±0.02	**64.32**±0.11	**99.62**±0.23	**92.72**±0.07	**54.40**±0.15

Note: N: number of clients. C: fraction of participating clients. '-' denotes failure to converge. " ": best method. " ": best baseline.

Table 2: Average test accuracy (%) for model-heterogeneous FL.

Experiment Setup

Datasets. We evaluate `pFedES` and baselines on 3 image classification datasets: MNIST [3] (LeCun et al. 1998), CIFAR-10 and CIFAR-100 [4] (Krizhevsky et al. 2009). They are divided into non-IID datasets following the method specified in (Shamsian et al. 2021). For MNIST and CIFAR-10, we assign only data from 2 out of the 10 classes to each client (non-IID: 2/10). For CIFAR-100, we assign only data from 10 out of the 100 classes to each client (non-IID: 10/100). Then, each client's local data are further divided into the training set and the testing set following the ratio of 8:2. The testing set is stored by each client and follows the same distribution as the training set.

Models. As shown in Tables 3 and 4 (Appendix B), each client trains a CNN model. In model-homogeneous settings, each client has the same CNN-1 model. In model-heterogeneous settings, clients are assigned with {CNN-1,. . ., CNN-5} with uniform probability. For both model-homogeneous and model-heterogeneous settings on CIFAR, the structure of the proxy homogeneous feature extractor in `pFedES` is as shown in Figure 1(b). For MNIST, the number of filters is 1 in the second convolutional layer of the proxy homogeneous feature extractor in Figure 1(b). For `FML`, `FedKD` and `FedAPEN` in model-heterogeneous settings, the smallest 'CNN-5' model is used as the homogeneous model.

Baselines. We compare `pFedES` with 9 best baselines belonging to the three categories of fully model-heterogeneous FL in the Related Work Section. `Standalone`, clients train local models independently. Model split: `LG-FedAvg` (Liang et al. 2020) and `FedGH` (Yi et al. 2023a). Mutual learning: `FML` (Shen et al. 2020), `FedKD` (Wu et al. 2022), and `FedAPEN` (Qin et al. 2023). Public-data independent knowledge distillation: `FD` (Jeong et al. 2018), `FedProto` (Tan et al. 2022b) and `FedTGP` (Zhang et al. 2024).

Evaluation Metrics. We measure `pFedES` and baselines with: **(1) Accuracy**: we measure the individual test accuracy of each client's local model and calculate the average test accuracy. **(2) Communication Cost**: We trace the number of transmitted parameters when the average model accuracy reaches the given target accuracy. **(3) Computation Cost**: We track the computational FLOPs consumed when the average model accuracy reaches the given target accuracy.

Training Strategy. We tune the optimal FL settings for all methods via grid search. The epochs of local model training $E \in \{1, 10\}$ and the batch size $B \in \{64, 128, 256, 512\}$. The optimizer is SGD with learning rate $\eta = \eta_\omega = \eta_\theta = 0.01$. We also tune special hyperparameters for baselines and report optimal results. We adjust two hyperparameters (the loss weight μ and training epoch E_{fe} of the proxy homogeneous feature extractor) for `pFedES` [5]. To compare with

[3] http://yann.lecun.com/exdb/mnist/

[4] https://www.cs.toronto.edu/\%7Ekriz/cifar.html

[5] More details are given in Appendix B which also describes more experiment results of average test accuracy curves, visualized personalization, visualized enhanced data, robustness to client participant rates, sensitivity to homogeneous feature extractor structures, and ablation study.

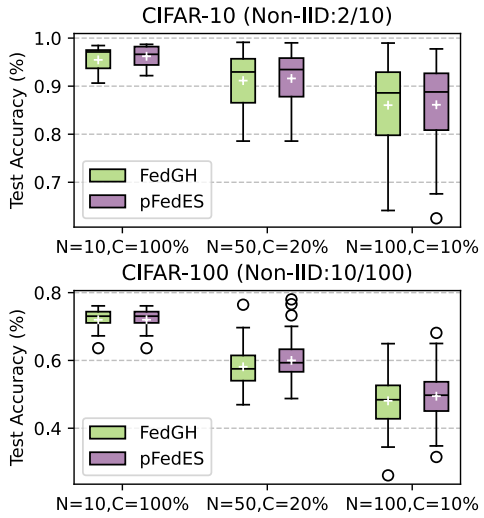

Figure 2: Client individual accuracy distribution.

the baselines fairly, we set the total number of communication rounds $T \in \{100, 500\}$ to ensure that all MHPFL algorithms converge. Each experiment is tested with 3 trails and recorded the average result.

Comparisons Results

We compare `pFedES` with baselines under model-homogeneous and -heterogeneous FL with different N, C.

Average Accuracy. Tables 1 and 2 show that the average accuracy of `pFedES` surpasses other baselines under both model-homogeneous and model-heterogeneous settings, by up to 1.29%. Since `pFedES` performs slight accuracy variances with best baselines on MNIST, we only depict the average accuracy of `pFedES` and the best baseline - `FedGH` varies as communication rounds on CIFAR-10 and CIFAR-100 in model-heterogeneous FL. Figure 4 (Appendix B) shows that `pFedES` converges to a higher average accuracy with faster or similar convergence speeds to `FedGH`. The subsequent experiments are conducted on more complicated CIFAR datasets in model-heterogeneous FL scenarios.

Individual Accuracy. Figure 2 presents client individual accuracy distribution of `pFedES` and the best baseline - `FedGH`. '+' denotes the average accuracy of all clients, and a shorter box bounded by the upper and lower quartile indicates smaller accuracy variances across clients. `pFedES` has higher average accuracy and a smaller variance than `FedGH` in most settings, further validating its effectiveness.

Trade-off among Accuracy, Computational & Communication Costs. We compare `pFedES` and the state-of-the-art baseline `FedGH` in terms of model accuracy, computational costs and communication costs. The target accuracy set for $N = \{10, 50, 100\}$ on CIFAR-10 dataset is 90% and that set for $N = \{10, 50, 100\}$ on CIFAR-100 dataset are $\{70\%, 60\%, 50\%\}$. As shown in Figure 3, `pFedES` consistently achieves the highest model accuracy with far lower communication costs than `FedGH`, while incurring similar computational costs. This indicates that `pFedES` strikes

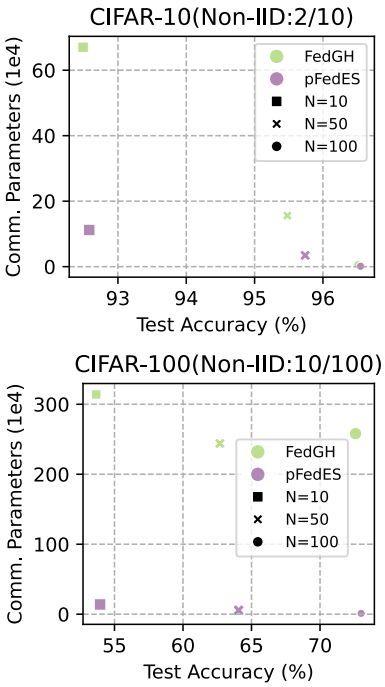

Figure 3: Trade-off among test accuracy, computational and communication costs. The sizes of markers (dots of varying sizes) reflect the computational FLOPs (1e9).

the best trade-off among the three metrics. Compared with `FedGH`, `pFedES` incurs $1/224$ communication and $1/5.85$ computational costs (*i.e.*, 99.6% communication and 82.9% computational cost savings), due to its faster convergence.

Conclusion

This paper proposed a novel model-heterogeneous personalized federated learning approach, `pFedES`, based on sharing proxy homogeneous feature extractors with efficient privacy preservation, and communication and computation cost savings. It enables each client to alternatively train a proxy homogeneous feature extractor and heterogeneous local model to exchange global and local knowledge. Aggregating the proxy homogeneous local feature extractors from clients fuses knowledge across heterogeneous clients. Theoretical analysis and experiments demonstrate its effectiveness and efficiency in communication and computation.

Acknowledgements

Xiaoguang Liu is supported by the National Science Foundation of China under Grant 62272252 & 62272253, and the Fundamental Research Funds for the Central Universities. Han Yu and Xiaoxiao Li are supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 1. This research is also supported, in part, by the RIE2025 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) (Award I2301E0026), administered by A*STAR, as well as supported by Alibaba

Group and NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL); and National Research Foundation, Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-019).

References

Ahn, J.; et al. 2019. Wireless Federated Distillation for Distributed Edge Learning with Heterogeneous Data. In *Proc. PIMRC*, 1–6. Istanbul, Turkey: IEEE.

Ahn, J.; et al. 2020. Cooperative Learning VIA Federated Distillation OVER Fading Channels. In *Proc. ICASSP*, 8856–8860. Barcelona, Spain: IEEE.

Alam, S.; et al. 2022. FedRolex: Model-Heterogeneous Federated Learning with Rolling Sub-Model Extraction. In *Proc. NeurIPS*. virtual: .

Babakniya, S.; et al. 2023. Revisiting Sparsity Hunting in Federated Learning: Why does Sparsity Consensus Matter? *Transactions on Machine Learning Research*, 1(1): 1.

Bank, D.; Koenigstein, N.; and Giryes, R. 2021. Autoencoders. arXiv:2003.05991.

Chan, Y.-H.; et al. 2024. Internal Cross-layer Gradients for Extending Homogeneity to Heterogeneity in Federated Learning. In *Proc. ICLR*, 1. Vienna, Austria: OpenReview.

Chang, H.; et al. 2021. Cronus: Robust and Heterogeneous Collaborative Learning with Black-Box Knowledge Transfer. In *Proc. NeurIPS Workshop*. virtual: .

Chen, J.; et al. 2021. FedMatch: Federated Learning Over Heterogeneous Question Answering Data. In *Proc. CIKM*, 181–190. virtual: ACM.

Cheng, S.; et al. 2021. FedGEMS: Federated Learning of Larger Server Models via Selective Knowledge Fusion. *CoRR*, abs/2110.11027.

Cho, Y. J.; et al. 2022. Heterogeneous Ensemble Knowledge Transfer for Training Large Models in Federated Learning. In *Proc. IJCAI*, 2881–2887. virtual: ijcai.org.

Collins, L.; et al. 2021. Exploiting Shared Representations for Personalized Federated Learning. In *Proc. ICML*, volume 139, 2089–2099. virtual: PMLR.

Diao, E. 2021. HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *Proc. ICLR*, 1. Austria: OpenReview.net.

Goebel, R.; Yu, H.; Faltings, B.; Fan, L.; and Xiong, Z. 2023. *Trustworthy Federated Learning*, volume 13448. 1: Springer, Cham.

Gong, X.; et al. 2024. Federated Learning via Input-Output Collaborative Distillation. In *Proc. AAAI*, 22058–22066. Vancouver, Canada: AAAI Press.

He, C.; et al. 2020. Group Knowledge Transfer: Federated Learning of Large CNNs at the Edge. In *Proc. NeurIPS*. virtual: .

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proc. (CVPR)*, 770–778. 1: 1.

Horváth, S. 2021. FjORD: Fair and Accurate Federated Learning under heterogeneous targets with Ordered Dropout. In *Proc. NIPS*, 12876–12889. Virtual: OpenReview.net.

Huang, W.; et al. 2022a. Few-Shot Model Agnostic Federated Learning. In *Proc. MM*, 7309–7316. Lisboa, Portugal: ACM.

Huang, W.; et al. 2022b. Learn from Others and Be Yourself in Heterogeneous Federated Learning. In *Proc. CVPR*, 10133–10143. virtual: IEEE.

Itahara, S.; et al. 2023. Distillation-Based Semi-Supervised Federated Learning for Communication-Efficient Collaborative Training With Non-IID Private Data. *IEEE Trans. Mob. Comput.*, 22(1): 191–205.

Jang, J.; et al. 2022. FedClassAvg: Local Representation Learning for Personalized Federated Learning on Heterogeneous Neural Networks. In *Proc. ICPP*, 76:1–76:10. virtual: ACM.

Jeong, E.; et al. 2018. Communication-Efficient On-Device Machine Learning: Federated Distillation and Augmentation under Non-IID Private Data. In *Proc. NeurIPS Workshop on Machine Learning on the Phone and other Consumer Devices*. virtual: .

Jiang, Y.; et al. 2022. Model Pruning Enables Efficient Federated Learning on Edge Devices. *TNNLS*, 1(1): 1.

Kairouz, P.; et al. 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends in Machine Learning*, 14(1–2): 1–210.

Kalra, S.; et al. 2023. Decentralized federated learning through proxy model sharing. *Nature communications*, 14(1): 2899.

Krizhevsky, A.; et al. 2009. *Learning multiple layers of features from tiny images*. : Toronto, ON, Canada.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11): 2278–2324.

Li, D.; and Wang, J. 2019. FedMD: Heterogenous Federated Learning via Model Distillation. In *Proc. NeurIPS Workshop*. virtual: .

Li, Q.; et al. 2021. Practical One-Shot Federated Learning for Cross-Silo Setting. In *Proc. IJCAI*, 1484–1490. virtual: ijcai.org.

Liang, P. P.; et al. 2020. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 1(1).

Lin, T.; et al. 2020. Ensemble Distillation for Robust Model Fusion in Federated Learning. In *Proc. NeurIPS*. virtual: .

Liu, C.; et al. 2022. Completely Heterogeneous Federated Learning. *CoRR*, abs/2210.15865.

Lu, X.; et al. 2022. Heterogeneous Model Fusion Federated Learning Mechanism Based on Model Mapping. *IEEE Internet Things J.*, 9(8): 6058–6068.

Luo, K.; Wang, S.; Fu, Y.; Li, X.; Lan, Y.; and Gao, M. 2023. DFRD: Data-Free Robustness Distillation for Heterogeneous Federated Learning. In *Proc. NeurIPS*. New Orleans, LA, USA.

Makhija, D.; et al. 2022. Architecture Agnostic Federated Learning for Neural Networks. In *Proc. ICML*, volume 162, 14860–14870. virtual: PMLR.

McMahan, B.; et al. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. AISTATS*, volume 54, 1273–1282. USA: PMLR.

Nguyen, D. P.; et al. 2023. Enhancing Heterogeneous Federated Learning with Knowledge Extraction and Multi-Model Fusion. In *Proc. SC Workshop*, 36–43. Denver, CO, USA: ACM.

Oh, J.; et al. 2022. FedBABU: Toward Enhanced Representation for Federated Image Classification. In *Proc. ICLR*. virtual: OpenReview.net.

Park, S.; et al. 2023. Towards Understanding Ensemble Distillation in Federated Learning. In *Proc. ICML*, volume 202, 27132–27187. Honolulu, Hawaii, USA: PMLR.

Pillutla, K.; et al. 2022. Federated Learning with Partial Model Personalization. In *Proc. ICML*, volume 162, 17716–17758. virtual: PMLR.

Qin, Z.; et al. 2023. FedAPEN: Personalized Cross-silo Federated Learning with Adaptability to Statistical Heterogeneity. In *Proc. KDD*, 1954–1964. Long Beach, CA, USA: ACM.

Ruder, S. 2016. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747: 1.

Sattler, F.; et al. 2021. FEDAUX: Leveraging Unlabeled Auxiliary Data in Federated Learning. *IEEE Trans. Neural Networks Learn. Syst.*, 1(1): 1–13.

Sattler, F.; et al. 2022. CFD: Communication-Efficient Federated Distillation via Soft-Label Quantization and Delta Coding. *IEEE Trans. Netw. Sci. Eng.*, 9(4): 2025–2038.

Shamsian, A.; et al. 2021. Personalized Federated Learning using Hypernetworks. In *Proc. ICML*, volume 139, 9489–9502. virtual: PMLR.

Shen, T.; et al. 2020. Federated Mutual Learning. *CoRR*, abs/2006.16765.

Takahashi, H.; et al. 2023. Breaching FedMD: Image Recovery via Paired-Logits Inversion Attack. In *Proc. CVPR*, 12198–12207. Vancouver, BC, Canada: IEEE.

Tan, A. Z.; et al. 2022a. Towards Personalized Federated Learning. *IEEE Trans. Neural Networks Learn. Syst.*, 1(1): 1–17.

Tan, Y.; et al. 2022b. FedProto: Federated Prototype Learning across Heterogeneous Clients. In *Proc. AAAI*, 8432–8440. virtual: AAAI Press.

van der Maaten, L.; and Hinton, G. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86): 2579–2605.

Wang, J.; Yang, X.; Cui, S.; Che, L.; Lyu, L.; Xu, D.; and Ma, F. 2023. Towards Personalized Federated Learning via Heterogeneous Model Reassembly. In *Proc. NeurIPS*. New Orleans, LA, USA.

Wu, C.; et al. 2022. Communication-efficient federated learning via knowledge distillation. *Nature Communications*, 13(1): 2032.

Ye, M.; et al. 2023. Heterogeneous Federated Learning: State-of-the-art and Research Challenges. *CoRR*, abs/2307.10616: 1.

Yi, L.; Wang, G.; Liu, X.; Shi, Z.; and Yu, H. 2023a. FedGH: Heterogeneous Federated Learning with Generalized Global Header. In *Proc. ACM MM*, 11. Canada: ACM.

Yi, L.; Yu, H.; Ren, C.; Wang, G.; Liu, X.; and Li, X. 2024a. Federated Model Heterogeneous Matryoshka Representation Learning. In *NeurIPS*.

Yi, L.; Yu, H.; Shi, Z.; Wang, G.; Liu, X.; Cui, L.; and Li, X. 2024b. FedSSA: Semantic Similarity-based Aggregation for Efficient Model-Heterogeneous Personalized Federated Learning. In *IJCAI*.

Yi, L.; Yu, H.; Wang, G.; and Liu, X. 2023b. FedLoRA: Model-Heterogeneous Personalized Federated Learning with LoRA Tuning. *CoRR*, abs/2310.13283.

Yi, L.; et al. 2022. QSFL: A Two-Level Uplink Communication Optimization Framework for Federated Learning. In *Proc. ICML*, volume 162, 25501–25513. online: PMLR.

Yi, L.; et al. 2023c. pFedLHNs: Personalized Federated Learning via Local Hypernetworks. In *Proc. ICANN*, 516–528. Springer, Crete, Greece: Springer.

Yu, F.; et al. 2021. Fed2: Feature-Aligned Federated Learning. In *Proc. KDD*, 2066–2074. virtual: ACM.

Yu, S.; et al. 2022. Resource-aware Federated Learning using Knowledge Extraction and Multi-model Fusion. *CoRR*, abs/2208.07978.

Zhang, J.; Guo, S.; Guo, J.; Zeng, D.; Zhou, J.; and Zomaya, A. Y. 2023. Towards Data-Independent Knowledge Transfer in Model-Heterogeneous Federated Learning. *IEEE Trans. Computers*, 72(10): 2888–2901.

Zhang, J.; Liu, Y.; Hua, Y.; and Cao, J. 2024. FedTGP: Trainable Global Prototypes with Adaptive-Margin-Enhanced Contrastive Learning for Data and Model Heterogeneity in Federated Learning. In *Proc. AAAI*, 16768–16776. Vancouver, Canada: AAAI Press.

Zhang, J.; et al. 2021. Parameterized Knowledge Transfer for Personalized Federated Learning. In *Proc. NeurIPS*, 10092–10104. virtual: OpenReview.net.

Zhang, L.; et al. 2022. FedZKT: Zero-Shot Knowledge Transfer towards Resource-Constrained Federated Learning with Heterogeneous On-Device Models. In *Proc. ICDCS*, 928–938. virtual: IEEE.

Zhang, Z.; and Sabuncu, M. R. 2018. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In *Proc. NeurIPS*, 8792–8802. Montréal, Canada: Curran Associates Inc.

Zhu, H.; et al. 2021a. Federated learning on non-IID data: A survey. *Neurocomputing*, 465: 371–390.

Zhu, Z.; et al. 2021b. Data-Free Knowledge Distillation for Heterogeneous Federated Learning. In *Proc. ICML*, volume 139, 12878–12889. virtual: PMLR.

Zhu, Z.; et al. 2022. Resilient and Communication Efficient Learning for Heterogeneous Federated Systems. In *Proc. ICML*, volume 162, 27504–27526. virtual: PMLR.