2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Personalized federated prototype learning in mixed heterogeneous data scenarios

Jiahao Zeng[1,2,*] , Wolong Xing[1,2,*], Liangtao Shi[3], Xin Huang[1,2], Jialin Wang[1,2], Zhile Cao[1,2] and Zhenkui Shi[1,2,†]

[1] Guangxi Normal University, Guilin, 541004, China
[2] Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education, GuangXi Normal University, Guilin, 541004, China
[3]Hefei University of Technology, Hefei, 230009, China
*These authors contributed equally to this work.
[†]Corresponding author
shizhenkui@gxnu.edu.cn

Abstract. Federated learning has received significant attention for its ability to s imultaneously protect customer privacy and leverage distributed data from multi ple devices for model training. However, conventional approaches often focus o n isolated heterogeneous scenarios, which results in skewed feature distributions or label distributions. Meanwhile, data heterogeneity is actually a key factor in improving model performance. To address this issue, we propose a new approac h called PFPL in mixed heterogeneous scenarios. The method provides richer d omain knowledge and unbiased convergence targets by constructing personalize d, unbiased prototypes for each client. Moreover, in the local update phase, we i ntroduce consistent regularization to align local instances with their personalize d prototypes, which significantly improves the convergence of the loss function. Experimental results on Digits and Office Caltech datasets validate the effectiv eness of our approach and successfully reduce the communication cost.

Keywords: Skewed label distribution, Skewed distribution of features, Personalized Federal Learning, data heterogeneity.

1 Introduction

The rapid proliferation of mobile phones, wearables, tablets, and smart home devices has led to exponential growth in the volume of data generated and retained by these de vices [1,2]. These data contain valuable insights for device owners. However, many us ers have become increasingly concerned about privacy, demanding that their data rem ain exclusively on local devices. Federated Learning (FL) [3] provides a privacy-prese rving distributed machine learning framework. In FL, a cloud server coordinates with distributed clients while ensuring data privacy through localized storage. The foundati onal FedAvg algorithm [4] iteratively aggregates client model parameters and distribu tes averaged global models to clients, enabling collaborative training without privacy disclosure. However, real-world scenarios involve data from heterogeneous sources w ith distinct characteristics, resulting in non-independent and identically distributed (non-IID) data [5,6]. Local client updates based on their data distributions often diverge from the global optimization trajectory. Personalized Federated Learning (PFL) has emerged as a prominent approach to developing client-specific models tailored to individual data distributions.

Personalized Federated Learning (PFL) addresses data heterogeneity by enabling clients (e.g., mobile devices or organizations) to develop customized models aligned with their unique data distributions [7]. Two fundamental challenges persist: (1) label distribution skew across clients, and (2) feature distribution divergence within identical label classes. Existing research predominantly focuses on single-mode heterogeneity (either label or feature skew), with limited exploration of cross-domain mixed heterogeneity where data originates from divergent domains with varying label distributions.

Under label skew conditions, the global model exhibits bias toward majority classes, leading to suboptimal generalization of personalized models on local client data. While hybrid local-global optimization [8] and model decoupling techniques [9] show efficacy in handling label skew, they fail to address feature distribution bias as the global model struggles to capture client-specific feature representations—even for data instances sharing identical labels across clients [10]. This feature space misalignment further hinders effective inter-client model collaboration. In practical cross-domain deployments with dual heterogeneity (concurrent label skew and feature divergence), these limitations not only degrade model performance but also hinder real-world applicability. Consequently, developing unified solutions for hybrid heterogeneous scenarios becomes imperative.

Real-world applications frequently exhibit dual heterogeneity scenarios combining label distribution skew and feature distribution divergence, as exemplified by cross-institutional CT image analysis where hospitals in different geographic regions possess distinct patient cohorts. This dual heterogeneity arises from two primary factors: (1) feature variations caused by discrepancies in medical imaging equipment specifications, and (2) label distribution skew stemming from demographic differences in disease prevalence across hospital populations.

Building upon prototype learning foundations [9,11], we present Prototypical Federated Partial Learning (PFPL), a novel framework for hybrid heterogeneity scenarios. PFPL employs cross-domain Lipschitz-constrained prototype comparison to quantify domain-specific knowledge relevance. Through adaptive prototype aggregation weighted by inter-domain similarity metrics, it constructs client-specific prototypes that mitigate dominant domain bias. Furthermore, we introduce Personalized Prototype Alignment (PPA), a regularization mechanism that enforces consistency between local instance embeddings and client-specific prototypes through feature-space distance minimization, ensuring robustness under hybrid heterogeneity. The main contributions of this paper are summarized as follows:

- We propose a novel personalized prototype learning approach aimed at solving the problem of skewed label distribution and skewed feature distribution in hybrid heterogeneous scenarios.

- To cope with the label distribution imbalance problem, we introduce prototype learning to capture domain knowledge and propose a novel aggregation scheme to generate personalized prototypes for each client. Meanwhile, in the local update phase, we design personalized unbiased prototype consistency to provide fair and unbiased target signals by narrowing the feature distance between instance embeddings and personalized prototypes, thus effectively mitigating the impact of feature distribution imbalance on model performance.
- We conduct extensive experiments on Digits and PACS tasks. The experimental results show that our scheme outperforms some recent federated learning methods in call and heterogeneous scenarios.

2 Related work

2.1 Heterogeneous challenges in federated learning

Data heterogeneity in federated learning primarily manifests as two distinct types [12, 13]: label distribution skew and feature distribution shift [14]. To address label distribution skew, conventional methods often employ label-based dataset partitioning to construct pseudo-IID distributions, aiming to reduce training bias and enhance model generalization. pFedKT [14] achieves personalized-generalized balance through dual knowledge transfer: (1) local hypernetworks preserve historical personalized knowledge, and (2) contrastive learning propagates updated global knowledge. Other methods like FedProto [15] and FedProc [16] enforce feature-level consistency through prototype alignment and procedural feature matching, respectively. However, these methods predominantly focus on single-domain label skew scenarios while neglecting cross-domain feature shifts in real-world hybrid heterogeneity.

Feature distribution shift poses a distinct challenge, where cross-domain client data leads to suboptimal cross-domain generalization [17]. FedBN [12] addresses feature shift through client-specific batch normalization layers prior to model aggregation. ADCOL [18] and FCCL [19] impose substantial resource overhead, requiring adversarial discriminators and public datasets for cross-client alignment. While FPL [13] mitigates feature shift via prototype clustering, it prioritizes global model convergence over client personalization. These approaches primarily target isolated feature shift scenarios while neglecting concurrent label distribution skew. Our work addresses hybrid heterogeneity—simultaneous label skew and feature shift—by developing personalized models tailored to individual client data characteristics.

2.2 Personalized federated learning

Personalized federated learning is extensively employed to address the data heterogeneity issue in federated learning. This approach enables each client to customize and optimize the personalized model in accordance with the characteristics and requirements of its local data, thereby facilitating more precise localized model training and adaptation.

Personalized federated learning has evolved diverse architectural strategies to address data heterogeneity. Among parameter decoupling approaches, Filip Hanzely et al. [11] have proposed a method that generates personalized models for each client by mixing local and global models to balance the two. FedBABU [20] extends this paradigm through a three-stage process: local body training with fixed random-initialized heads, server-side body aggregation, and post-training head fine-tuning for personalization. FedRoD [21] innovates further with a dual-head architecture comprising a shared general head optimized via class-balanced loss and client-specific private heads trained with empirical loss, where only the body and general head participate in aggregation. Prototype-enhanced methods offer complementary solutions. FedNH [22] integrates prototype-semantic consistency learning to enhance feature discriminability while employing head regularization to prevent prototype collapse under class imbalance. This framework adopts alternating optimization: frozen-body head updates precede fixed-head body refinements.

However, most of the above personalization methods only consider the heterogeneous problem for a single scenario (skewed label distribution or skewed feature distribution). There are relatively few personalization methods for mixed scenarios of both, which limits the application of federated learning on more diverse non-IID data. Therefore, solving more diverse hybrid heterogeneous problems has become an important challenge for federated learning research.

3 Methodology

3.1 PRELIMINARY

Following typical federated learning [3], there are M participants, and the private data set of the participants is $D_m = \{x_i, y_i\}_{i=1}^{N_m}$, where N_m represents the client side data size. These private data follow different label distributions and come from different domains. For example, data sets D_i and D_j on two client sides i and j may have different label statistical distributions. This is common for photo classification apps installed on the mobile client side. The server needs to identify many classes $\mathbb{C} = \{\mathbb{C}(1), \cdots \mathbb{C}(k), \cdots\}$, while each client side only needs to identify a few classes that make up a subset of $\mathbb{C}$. The class sets may vary from client side to client side, although there is overlap. And these private data sets are derived from different domains, resulting in significant differences in the features of the data even if the categories are the same.

- **Mixed heterogeneous scenarios in federated learning:** $P_i(x \mid y) \neq P_j(x \mid y)$, $(P_i(y) \neq P_n(y))$. There is a skewed feature distribution and a skewed label distribution between private data. Specifically, the label distribution before different client sides is different, and the data comes from different domains, presenting a unique feature distribution despite the overlap between domains.

In addition, participants agree to share models with the same architecture. We treat the model as two modules:

Feature Extraction Module ϕ (i.e., the embeddings function) transforms the input from the raw feature space to the embedding space, $f(\phi, x) \to h \in R^d$, the sample x coding d-dimensional feature vector $h = f(\phi, x) \in R^d$. Decision Module φ makes classification decisions for the given learning task. $g:(\varphi, h) \to \hat{y} \in \mathbb{C}$ maps feature h to the logits output $\hat{y} = g(\varphi, h) \in \mathbb{C}$. So, the label function can be written as: $F(\phi, \varphi) = f(\phi) \cdot g(\varphi)$, and we use ω to represent (ϕ, φ) for short.

Prototypes: Each prototype is the average of the feature vectors of the same class

$$C^{(k)} = \frac{1}{|D^k|} \sum_{(x,y) \in D^k} f(\phi; x). \quad (1)$$

where D^k represents the data instances label K, and $|D^k|$ represents the number of data instances label K.

Local Prototypes: We define a feature $C^{(k)}$ to represent the k-th class in $\mathbb{C}$. For the i-th client, $C^{(k)}$ is the average of the features obtained by inputting samples with the label k into the feature extraction module.

$$C_i^{(k)} = \frac{1}{|D_i^k|} \sum_{(x,y) \in D_i^k} f_i(\phi_i; x). \quad (2)$$

where D_i^k represents the data samples of class K in the i-th client.

Global Prototypes: For a given class j, the server receives locally computed features with class label j from a group of clients. These local features with labels j are aggregated by taking their average to generate the global feature $\bar{C}^{(j)}$ for class j.

$$\bar{C}^{(k)} = \frac{1}{\aleph^k} \sum_{i \in \aleph^k} \frac{|D_i^k|}{N^k} C_i^{(k)}. \quad (3)$$

where $C_i^{(k)}$ represents the local features of class K from the i-th client, $\aleph^k$ represents the set of clients that have class K, and N^k represents the total number of data instances of all client-side classes label K.

However, the global prototype is not suitable for the mixed heterogeneous scenario in this paper, which mainly has two problems: **1.** A single global prototype blurs the difference between different domains, and it is difficult to learn special knowledge between different domains. **2.** Since the weight parameter of the global prototype is determined by the amount of data in the category sample, the final global prototype is biased towards the dominant user with a large amount of data, which makes it difficult for the client side with few data instances to learn. A simple approach is to build an unbiased prototype; that is, the prototype weight of each client is the same, but this approach still faces the challenge of problem 1, and this approach makes the client side with few er sample instances benefit but hurts the client side with more sample instances to participate in federated learning.

3.2 Personalized federated prototyping learning

We propose a solution for hybrid heterogeneous FL. This paper uses a prototype as the main component for exchanging information at the client side and server level. The framework is shown in Fig.1. The central server receives local prototype sets $\{\Theta(1),\Theta(2),\cdots,\Theta(m)\}$ from m local client sides and then clusters prototypes $\{\Phi(1),\Phi(2),\cdots,\Phi(k),\cdots\}$ of the same category. In a hybrid heterogeneous FL setup, these prototype sets overlap but are not the same. Take the handwritten digit data set as an example. The first client side is the recognition numbers 2, 3, 4, from the MNIST data set, while the other client side is the recognition numbers 4, 5, from the SYN data set. These are two sets of handwritten digits from different domains, with different sample categories, albeit overlapping. For the prototype category of each client in the cluster, by assigning weights with the L2 distance of the other client-side prototypes, aggregation generates a personalized prototype specific to the client.

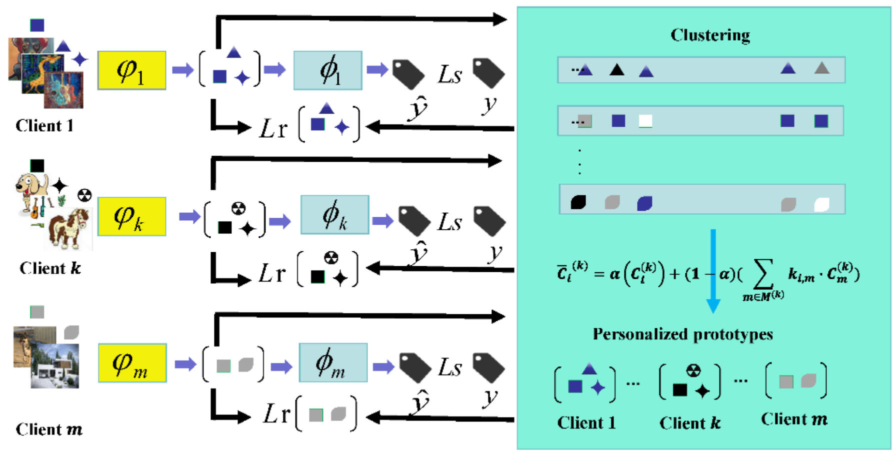


Fig. 1. Architecture of personalized Federated Prototype Learning (PFPL).

Personalized prototype: The server level collects the prototype set of the client and clusters it for the client side prototype category. Taking the prototype with the category K of client side i as an example, its personalized prototype is as follows:

$$\bar{C}_i^{(k)} = \alpha(C_i^{(k)}) + (1-\alpha)(\sum_{m\in M^{(k)}} k_{i,m}\cdot C_m^{(k)}) \tag{4}$$

where α represents a hyperparameter that controls the degree of personalization, $M^{(k)}$ represents a client side cluster with a K-class label prototype, C_m^k represents the K-class label prototype uploaded by the m client side, and $k_{i,m}$ represents the weight coefficie nt of the client side m to the i client. The calculation formula is determined by comparing the L2 distance of the two client side K-class prototypes, as follows:

$$k_{i,m} = \frac{\| C_i^{(k)}, C_m^{(k)} \|_2}{\sum_{m \in \Phi(k)} \| C_i^{(k)}, C_m^{(k)} \|_2} \tag{5}$$

where $\Phi(k)$ represents the cluster prototype with the server level label K, and the L2 distance between prototypes is calculated as,:

$$\| C_i^{(k)}, C_m^{(k)} \|_2 = \sum_k d(C_i^{(k)} - C_m^{(k)})^2 \tag{6}$$

where d is the locally generated distance metrics of the prototype $C_i^{(k)}$ with label k and prototype $C_m^{(k)}$ with the same label on the other client side m. Distance measures can take many forms, such as L1 distance, L2 distance, and bulldozer distance. Here we use L2 distance metrics.

We generate its personalized prototype for each client side. In simple terms, personalized prototypes with the same label on different clients are affected by domain migration differently. When assigning weight, prototypes from the same domain will assign more weight, while the weight assigned from different domains will be less, making personalized prototypes tend to be more knowledge of the same domain and stay away from the influence of different domains, thus effectively solving the above problem **1** and our weight is determined according to the L2 distance between different client side prototypes, and is not determined by the amount of data on the client side, so it will not be affected by the dominant domain. Although the amount of data on individual prototypes is small, it can also learn more from a large number of prototypes of the same domain, thus solving the problem **2**.

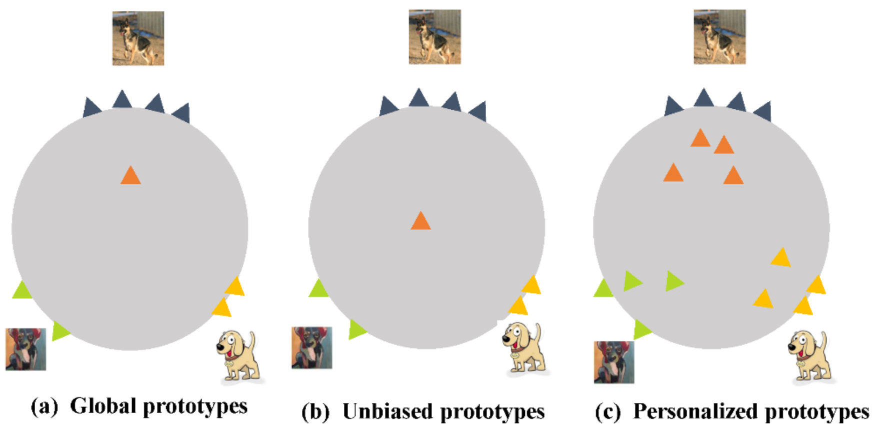


Fig. 2. Description of different prototypes.

Local Model Update: The client side needs to update the local model to generate c onsistent client-side functionality. We also introduce unbiased personalized prototype consistency, which allows the local prototype to approximate its personalized prototyp e through regularization in local updates. Specifically, the loss function is defined as f ollows:

$$\ell(D_i, w_i) = \ell_S(F(w_i; x_i), y_i) + \lambda \cdot \ell_R\left(\bar{C}_i^{(k)}, C_i^{(k)}\right) \tag{7}$$

where D_i stands for data from the i-th client, λ is an important parameter for regulariz ation.

Algorithm 1: PFPL

Input: D_i , w_i , $i = 1, \cdots m$

Output: The final personalization model $\{w_i\}$, $i = 1, \cdots, m$

Server executes: ($\Theta(i), \bar{C}_i^{(k)}$)
Initialize w for all clines
for each round $T = 1, 2, \cdots$ **do**
  for each client i in parallel **do**
    $\{\Theta(i)\} \longleftarrow$ LocalUpdate $(i, \bar{C}^{(k)})$
  end for
  Clustering prototype sets uploaded by clines
  Update personalized prototype sets by Eq.(4):
  Send the personalized prototype to the corresponding
  client side
end for
LocalUpdate $(i, \bar{C}_i^{(k)})$ **:**
for each local epoch **do**
  for batch $(x_i, y_i) \in D_i$ **do**
    Compute local features by Eq.(2)
    Compute loss by Eq.(7) using local prototype
    Update local model according to the loss
    Update local prototype sets $\Theta(i)$ with personalized
    prototypes in $\{\bar{C}_i^{(k)}\}$
  end for
end for
return $\Theta(i), i = 1, \ldots m$

Discussion: We further explain the differences between the three prototypes in Fig. 2 The global prototype inherently confuses the knowledge of different domains and s hows a skewed feature space towards the potentially dominant domain in heterogeneo us federated learning. Unbiased prototypes also have the problem of confusing the kno wledge of different domains, and giving the same weight to the client side with a large number of instance samples is itself an unfair allocation, which reduces the enthusiasm of the client side with a large data instance to participate in federated learning. Our personalized prototype solves the above two problems at the same time. Specifically, for problem 1, our personalized prototype is generated for different client side aggregates rather than a single global prototype or unbiased prototype, which effectively solves the problem that the client side data comes from different domains. For problem 2, our weight coefficient is mainly determined by hyperparameter a and K, a guarantees the degree to which our personalized prototype is biased towards the local prototype. The calculation formula of K guarantees that other client-side prototypes from the same domain assign more weight, while client-side prototypes from different domains assign less weight, so that the final personalized prototype is biased towards the real domain and deviates from other domains. Compared with the traditional model gradient parameter, the dimension of the prototype is much smaller than that of the overall model, which brings less computational cost to the participants. In addition, prototype uploads are privacy-safe because they are one-dimensional vectors generated by averaging low-dimensional representations from the same class of samples, which is an irreversible process. Second, an attacker cannot rebuild the original data source from the prototype without accessing the local model. Therefore, prototype not only provides lower computational cost, but also a privacy-preserving scheme in heterogeneous federated learning.

3.3 Optimization Objective

The goal of PFPL is to solve joint optimization problems on distributed networks. PFPL applies prototype-based communication, which allows a local model to align its local prototype with its personalized prototype while minimizing the sum of losses for all client side local learning tasks. The learning goal of personalized federated prototypes across heterogeneous clients can be expressed as:

$$\arg\min_{\phi,\varphi} \sum_{i=1}^{m} \frac{D_i}{N} \ell_S\left(f(w_i; x_i), y_i\right) + \lambda \cdot \sum_{k=1}^{|\mathbb{C}|} \sum_{i=1}^{m} \ell_R\left(\bar{C}_i^{(k)}, C_i^{(k)}\right). \tag{8}$$

where loss $\ell_S(F(wi; x_i), y_i)$ represents the objective loss for the i-th client, and we use the standard cross-entropy loss as the objective loss function. N represents the sum of all client-side instance data, $|\mathbb{C}|$ represents the number of classes for the labels, and ℓ_R is the regularization term used to measure distance, with its expression as follows:

$$\ell_R\left(\bar{C}_i^{(k)}, C_i^{(k)}\right) = \| \bar{C}_i^{(k)}, C_i^{(k)} \|_2 \tag{9}$$

where ℓ_R is the distance metrics of the locally generated prototype $C_i^{(k)}$ and the globally aggregated personalized prototype $\bar{C}_i^{(k)}$. Here we use the L2 distance to measure the difference between the two. The specific algorithm is shown in Algorithm 1.

4 Convergence analysis

We use the first-level model (decision module) as our objective loss function.

Assumption 1. (Lipschitz Smooth). It is assumed that each local objective function is L_1-Lipschitz Smooth, which also implies that the gradient of the local objective functi on is L_1-Lipschitz continuous.

$$\left\|\nabla \ell_{t_2} - \nabla \ell_{t_2}\right\|_2 \le L_1 \left\|w_{i,t_1} - w_{i,t_2}\right\|_2, \forall t_1, t_2 > 0, i \in \{1, 2, \cdots, m\}. \tag{10}$$

This also implies the following quadratic bound:

$$\ell_{t_1} - \ell_{t_2} \le \left\langle \nabla \ell_{t_2}, (w_{i,t_1} - w_{i,t_2}) \right\rangle + \frac{L_1}{2} \left\|w_{i,t_1} - w_{i,t_2}\right\|_2^2, \forall t_1, t_2 > 0, i \in \{1, 2, \cdots, m\}. \tag{11}$$

Assumption 2. (Unbiased Gradient and Bounded Variance)The stochastic gradient $g_{i,t} = \ell(w_{i,t})$ is an unbiased estimator of the local gradient for each client. Assuming th at its expectation satisfies the following equation:

$$E_{\xi_i} \sim_{D_i} \left[g_{i,t} \right] = \nabla \ell(w_{i,t}) = \nabla \ell_t, \forall i \in \{1, 2, \cdots, m\}, \tag{12}$$

and its variance is bounded by σ^2:

$$E\left[\left\|g_{i,t} - \nabla \ell_{(w_{i,t})}\right\|_2^2\right] \le \sigma^2. \tag{13}$$

Assumption 3. (Bounded Expectation of Euclidean norm of Stochastic Gradients). Th e expectation of the random gradient is bounded by G:

$$E\left[\left\|g_{i,t}\right\|_2\right] \le G, \forall i \in \{1, 2, \cdots, m\}. \tag{14}$$

Assumption 4. The functions of each feature extraction module, commonly known as embedding functions, are L_2-Lipschitz continuous.

$$\left\| f_i(\phi_{i,t_1}) - f_i(\phi_{i,t_2}) \right\| \le L_2 \left\|\phi_{i,t_1} - \phi_{i,t_2}\right\|_2, \forall t_1, t_2 > 0, i \in \{1, 2, \cdots, m\}. \tag{15}$$

We can obtain theoretical results for non-convex problems if the above assumption holds. In Theorem 1, we provide the expected decrease in each round. We use $e \in \{\frac{1}{2}, 1, 2, \cdots, E\}$ to represent local iterations and t to represent global communication rounds. Here, tE represents the time step before global features aggregation, and $tE + \frac{1}{2}$ represents the time step between global features aggregation and the first iterati on of this round.

Theorem 1. (One-round deviation) Let Assumption 1 to 4 hold. For an arbitrary client, after every communication round, we have,

$$\left[\left\|\ell_{(t+1)E+\frac{1}{2}}\right\|\right] \le \ell_{tE+\frac{1}{2}} - \left(\eta - \frac{L_1\eta^2}{2}\right)\sum_{e=\frac{1}{2}}^{E-1}\left\|\nabla \ell_{tE+e}\right\|_2^2 + \frac{L_1E\eta^2}{2}\sigma^2 + \lambda L_2\eta EG. \quad \mathbf{(16)}$$

Theorem 1 indicates the deviation bound of the local objective function for an arbitrary client after each communication round. Convergence can be guaranteed when there is a certain expected one-round decrease, which can be achieved by choosing appropriate η and λ .

Corollary 1. (Non-convex pFedPM convergence). The loss function ℓ of an arbitrary client monotonously decreases in every communication round when

$$\eta_{e'} < \frac{2(\sum_{e=\frac{1}{2}}^{e'}\left\|\nabla \ell_{tE+e}\right\|_2^2 - \lambda L_2EG)}{L_2EG}, \quad (17)$$

where $e' = \{\frac{1}{2}, 1, 2, \cdots, E\}$ and

$$\lambda_t < \frac{\left\|\nabla \ell_{tE+\frac{1}{2}}\right\|_2^2}{L_2EG}. \quad (18)$$

Thus, the loss function converges. Corollary 1 guarantees that the expected bias of the loss function is negative, ensuring the convergence of the loss function. We can further ensure the convergence of the algorithm by choosing appropriate learning rates η and importance weights λ .

5 Experiments

Datasets: We evaluate our method on three classification tasks:

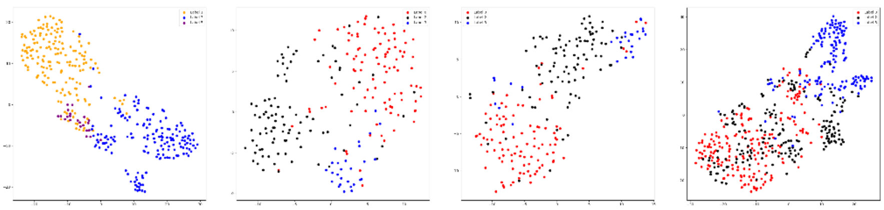
- Digits [23] includes four domains: MNIST (M), USPS (U), SVHN (SV), and SYN (SY) with 10 categories (digit numbers from 0 to 9).
- PACS [24] data set is a domain adaptive image dataset, including 4 domains: photos, art paintings, cartoons, and sketches. Each field contains 7 categories.

Local models: For these three classification tasks, we use the classical ResNet18 [25] model as our base model, and all methods use the same network architecture to make fair comparisons across different tasks.

Baselines of FL: We investigate the performance of our method PFPL under mixed heterogeneous conditions and compare it with baselines, including FedAvg, Local. I n addition, some FL methods in single-domain scenarios are also included. Feature distribution skewed: FedBN [12], FRaug [17]. Label distribution skewed: FedProto [15], Ditto [26], APFL [27], FedRod [21], FedKD [28].

Mixed heterogeneous setting: This paper considers a heterogeneous scenario where the label distribution is skewed and the feature distribution is skewed. We borrow the concept of n-way, k-short from less sample learning, which n controls the number of classes on the client side and k controls the number of training instances per class. To simulate the label distribution skewed, we stochastic change the values of n and k for each client side. For the feature distribution skewed, we stochastic assign instance data from different domains to the client side. The final client-side data only has data for individual category labels and is sourced from different domains, albeit with overlap.

Implementation Details: We implement the comparison of PFPL and general baseline methods in PyTorch. We use 20 client sides for all data sets. For Digits and PAPC Dataset, the average number of categories n for local clients is set to 3, 4, 5, and for Office-31 [29] Dataset, the average number of categories n for local clients is set to 10, 15, 20. And the number of each class in each client side is initially set to 100%. To make a fair comparison, we follow the same settings. For all methods, we use an SGD optimizer with a learning rate of $lr = 0.01$. The corresponding weight decay is e^{-5} and the momentum is 0.9. The training batch size is 4, and we communicate epoch for $E = 100$ and the local update wheel $T = 1$.



(a) (b) (c) (d)

Fig. 3. t-SNE visualization of the prototype generated by the PFPL method. We consider clients from four different domains in the PACS dataset, corresponding to the (a),(b),(c) and (d) in the picture, and the number of classes for each client is uniformly set to n = 3.

PFPL under varying α : As shown in Equation (4), in the server-level personalized prototype aggregation stage, a controls the weight of the local prototype. As shown in Fig.3, in the range of 0-1, the optimal values of three different data sets a are 0.3, 0.5, and 0.6.

The model performance under the number of classes n of different clients: Table 1 reports the average test accuracy for all clients. It can be seen that PFPL has the highest accuracy in most cases among FL under different n controls.

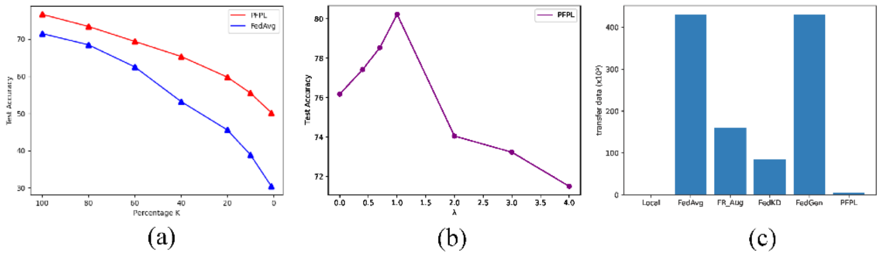


Fig. 4. (a) Average test accuracy of PFPL and FedAvg on PACS with varying numbers of samples in each class. (b) Model accuracy corresponding to different λ . (c) Comparison of the number of parameters transferred in each round of global iteration.

Table 1. Results for the Digits dataset on different algorithms.

Algo-rithms	Acc n=3					Acc n=4	Acc n=5	Rounds
	MNIST	USPS	SVHN	SYN	AVG	AVG	AVG	
Local	98.32	93.64	85.42	53.18	91.15	92.17	92.86	0
FedAvg	98.64	92.17	86.35	54.16	87.56	88.42	88.75	200
Ditto	96.35	90.86	85.44	53.12	86.74	87.14	87.56	200
APFL	98.42	91.24	86.14	53.42	87.98	88.58	89.43	200
FedRod	95.58	90.33	85.18	52.47	85.61	86.47	86.21	200
FedKD	97.74	92.58	83.22	54.16	86.32	87.36	88.22	200
FedGen	96.35	91.42	82.96	54.88	85.48	86.44	87.64	200
FedBN	98.56	93.10	86.37	53.35	87.54	88.63	89.48	200
FRAug	98.17	92.58	84.51	52.67	93.64	94.51	95.16	200
FedProto	98.42	93.17	**88.15**	54.42	93.16	94.16	94.23	200
PFPL	**98.68**	**93.94**	87.63	**60.28**	**94.75**	**95.84**	**96.17**	200

Scalability of PFPL on varying number of samples: Fig.4a shows that PFPL can scale to scenarios with fewer samples available on clients. The test accuracy consiste ntly decreases when there are fewer samples for training, but PFPL drops more slowly than FedAvg as a result of its adaptability and scalability on various data sizes.

PFPL under varying λ :Fig.4b shows the change in performance at different valu es of λ in Equation (7). We specify the initial range of λ at [0,4] and extract a set of va lues from it. We record the average test accuracy of the PAPC data set, K = 100%, n = 4, and the distance loss of the prototype. In this case, as λ increases, the original dista nce loss (regularizer) decreases, while the average test accuracy decreases sharply afte r $\lambda = 1$, and finally we take the optimal value of λ as 1.

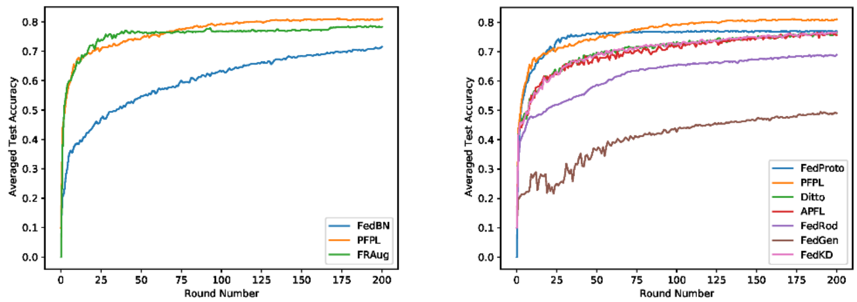


Fig. 5. Precision comparison between PFPL and a single heterogeneous personalized federated learning method in mixed heterogeneous scenarios. (a) indicates the skewed feature distribution scenario, and (b) indicates the skewed label distribution scenario.

PFPL communication efficiency comparison: Fig.4c depicts the number of para meters to be transmitted by each client in each communication round. In comparison with the classical approach, our method transmits the minimal number of parameters i n each round, thereby effectively minimizing the volume of communication throughou t the entire communication round, lowering the communication cost and enhancing th e transmission efficiency.

Performance of PFPL compared to the single-domain FL method: As shown in Fig.5, PFPL achieves higher accuracy than the FL method in single heterogeneous sc enarios with skewed label distributions and skewed feature distributions. We suspect t hat this is due to the fact that previous FL methods focused on solving the heterogeneo us federation problem in single scenarios and ignored other heterogeneous scenarios, r esulting in lower performance in mixed heterogeneous scenarios.

6 Conclusion

In this paper, we explore the personalized federated learning PFPL for handling mixed heterogeneous scenarios. Our work introduces prototype as a communication standar d. We use prototype (prototype-like representation) to learn knowledge of different do mains and stable convergence goals by generating specific personalized prototypes for different client sides and introducing personalized prototype consistency during the lo cal update phase. Ultimately, our method achieves higher accuracy than single-scenari o heterogeneous federated learning methods.

References

1. Wang, Songping, Hanqing Liu, and Haochen Zhao. "Public-Domain Locator for Boosting Attack Transferability on Videos." *2024 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2024.

2. Hu, Xiantao, et al. "Towards modalities correlation for RGB-T tracking." *IEEE Transactions on Circuits and Systems for Video Technology* (2024).
3. Yang, Qiang, et al. "Federated machine learning: Concept and applications." *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019): 1-19.
4. McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial intelligence and statistics*. PMLR, 2017.
5. Tan, Yue, et al. "Federated learning on non-iid graphs via structural knowledge sharing." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. No. 8. 2023.
6. Xing, Wolong, et al. "Personalized federated learning based on feature fusion." *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE, 2024.
7. Zhang, Jie, et al. "Model Decomposition and Reassembly for Purified Knowledge Transfer in Personalized Federated Learning." *IEEE Transactions on Mobile Computing* (2024).
8. Hanzely, Filip, and Peter Richtárik. "Federated learning of a mixture of global and local models." *arXiv preprint arXiv:2002.05516* (2020).
9. Li, Junnan, et al. "Prototypical contrastive learning of unsupervised representations." *arXiv preprint arXiv:2005.04966* (2020).
10. Yan, Haorui, et al. "Global or Local Adaptation? Client-Sampled Federated Meta-Learning for Personalized IoT Intrusion Detection." *IEEE Transactions on Information Forensics and Security* (2024).
11. Zhu, Fei, et al. "Prototype augmentation and self-supervision for incremental learning." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021.
12. Li, Xiaoxiao, et al. "Fedbn: Federated learning on non-iid features via local batch normalization." *arXiv preprint arXiv:2102.07623* (2021).
13. Huang, Wenke, et al. "Rethinking federated learning with domain shift: A prototype view." *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2023.
14. Yi, Liping, et al. "pFedKT: Personalized federated learning with dual knowledge transfer." *Knowledge-Based Systems* 292 (2024): 111633.
15. Tan, Yue, et al. "Fedproto: Federated prototype learning across heterogeneous clients." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 36. No. 8. 2022.
16. Mu, Xutong, et al. "Fedproc: Prototypical contrastive federated learning on non-iid data." *Future Generation Computer Systems* 143 (2023): 93-104.
17. Chen, Haokun, et al. "FRAug: Tackling federated learning with Non-IID features via representation augmentation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2023.
18. Li, Qinbin, Bingsheng He, and Dawn Song. "Adversarial collaborative learning on non-IID features." *International Conference on Machine Learning*. PMLR, 2023.
19. Huang, Wenke, Mang Ye, and Bo Du. "Learn from others and be yourself in heterogeneous federated learning." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022.
20. Oh, Jaehoon, Sangmook Kim, and Se-Young Yun. "Fedbabu: Towards enhanced representation for federated image classification." *arXiv preprint arXiv:2106.06042* (2021).
21. Chen, Hong-You, and Wei-Lun Chao. "On bridging generic and personalized federated learning for image classification." *arXiv preprint arXiv:2107.00778* (2021).
22. Dai, Yutong, et al. "Tackling data heterogeneity in federated learning with class prototypes." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. No. 6. 2023.
23. Roy, Prasun, et al. "Effects of degradations on deep neural network architectures." *arXiv preprint arXiv:1807.10108* (2018).

24. Li, Da, et al. "Deeper, broader and artier domain generalization." *Proceedings of the IEEE international conference on computer vision*. 2017.
25. He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
26. Li, Tian, et al. "Ditto: Fair and robust federated learning through personalization." *International conference on machine learning*. PMLR, 2021.
27. Deng, Yuyang, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. "Adaptive personalized federated learning." *arXiv preprint arXiv:2003.13461* (2020).
28. Wu, Chuhan, et al. "Communication-efficient federated learning via knowledge distillation." *Nature communications* 13.1 (2022): 2032.
29. Saenko, Kate, et al. "Adapting visual category models to new domains." *Computer vision–ECCV 2010: 11th European conference on computer vision, Heraklion, Crete, Greece, September 5-11, 2010, proceedings, part iV 11*. Springer Berlin Heidelberg, 2010.