FedPall: Prototype-based Adversarial and Collaborative Learning for Federated Learning with Feature Drift

Anonymous ICCV submission

Paper ID 13196

Abstract

Federated learning (FL) enables collaborative training of a global model in the centralized server with data from multiple parties while preserving privacy. However, data heterogeneity can significantly degrade the performance of the global model when each party uses datasets from different sources to train a local model. Among various cases of data heterogeneity, feature drift, feature space difference among parties, is prevalent in real-life data but remains largely unexplored. Feature drift can distract feature extraction learning in clients and thus lead to poor feature extraction and classification performance. To tackle the problem of feature drift in FL, we propose FedPall, an FL framework that utilizes prototype-based adversarial learning to unify feature spaces and collaborative learning to reinforce class information within the features. Moreover, FedPall leverages mixed features generated from global prototypes and local features to enhance the global classifier with classification-relevant information from a global perspective. Evaluation results on three representative feature-drifted datasets demonstrate FedPall's consistently superior performance in classification with feature-drifted data in the FL scenario. [1]

1. Introduction

Today, in computer vision, researchers often utilize large amounts of data from various parties to improve the accuracy of algorithms. However, this raises concerns, such as the potential for user privacy leakage caused by sharing private data [11]. Federated learning (FL) [20] is proposed as a privacy-preserving distributed learning paradigm to address these challenges. In the FL paradigm, each party maintains a local client model and collaborates with others to train a global model on the server without sharing the original data, effectively protecting user privacy.

Particularly, the FL paradigm faces challenges from data heterogeneity [13]. Due to the limited local view of each client, data distribution discrepancies arise across clients (commonly referred to as the non-independent and identically distributed data issue, or non-independent and identically distributed(non-IID) data issue), which can increase generalization error for local models and degrade the performance of the globally aggregated model [9]. Although recent work on non-IID data in FL primarily addresses issues such as stability, client drift, and heterogeneous label distributions among clients [10, 17, 30], feature drift(i.e., variations in feature distributions across clients) is a prevalent yet underexplored challenge in FL. As shown in Fig. 1a, feature drift refers to the phenomenon where samples of the same class exhibit different feature distributions across different clients due to variations in data collection methods, devices, and other factors. This leads to ambiguous decision boundaries, severely impacting the classification performance of federated learning models. However, tra-

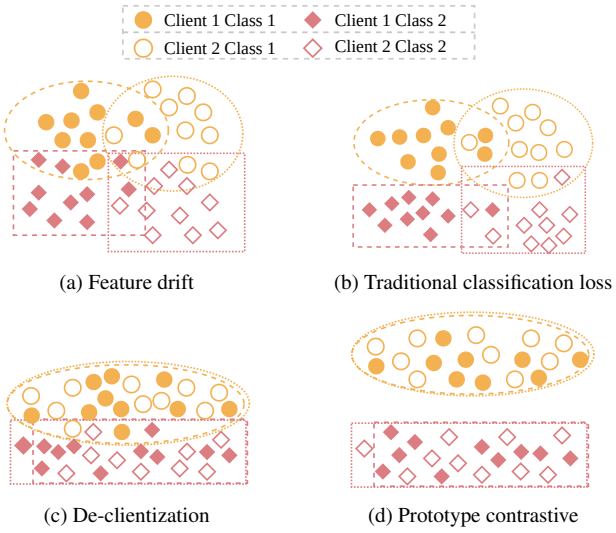


Figure 1. We show a schematic diagram of feature drift and use different techniques to drive feature distribution to update in different directions.

[1] The code is attached in the supplementary material for review.

ditional classification losses (e.g., cross-entropy loss (CE loss), as shown in Fig. 1b) do not account for feature drift. As a result, the collaborative effect among different clients causes the feature space of the same-class samples to influence each other, leading to only slight or even no clearer distinction between decision boundaries for various classes within the same client. As illustrated in Fig. 1c, decentralization is a common approach to addressing feature drift, which involves ignoring inter-client differences to make the feature space of same-class samples across different clients more clustered. In feature drift scenarios, some state-of-the-art algorithms do not explore the role of loss functions but instead optimize local update algorithms to achieve decentralization. For example, FedBN [18] compresses the feature space across clients by adding batch normalization layers to local models. Some methods [1, 33] align feature spaces by sharing partial data for synthetic data augmentation. Additionally, ADCOL sends raw features to the server to update a amplifier and uses the Kullback-Leibler (KL) loss function to achieve decentralization. However, these methods have drawbacks: on the one hand, they may lead to loss of class information, and on the other hand, they pose potential risks of privacy leakage.

To tackle the above challenge, we propose *FedPall*, a novel prototype-based adversarial and collaborative learning framework for FL with feature drift. FedPall applies adversarial learning between clients and the server to unify feature spaces, as well as collaborative learning across clients to enhance global decision boundaries. Specifically, FedPall uses adversarial learning to train a feature enhancer and uses KL divergence to align heterogeneous feature spaces between clients, while using prototype contrastive loss to reinforce class information (see Fig. 1d). Finally, adversarially aligned features are securely mixed with the global prototypes and uploaded to the server, where a global-view classifier is trained to enhance overall performance. Our contributions are summarized as follows:

- We propose a novel FL framework, FedPall, to address the feature drift problem. FedPall introduces adversarial learning between clients and the server, and collaborative learning among clients aiming to project feature representations into a unified feature space and reinforce the intrinsic class information. This approach effectively mitigates the feature drift problem in FL settings.
- We develop a technical strategy that hierarchically integrates the global prototypes with local features to orchestrate client-server collaboration. The mixed prototype features are then used to train a global classifier, which induces the classifier to distill discriminative patterns through cross-client knowledge consolidation.
- Empirical evaluation on three typical feature-drifted benchmarks demonstrates that our proposed method achieves state-of-the-art classification accuracy.

2. Related Work

In FL settings, the limited local view of each client directly induces the feature drift problem. Due to data-distribution differences, the same class label may have different feature representations, resulting in poor generalization of local models. Existing studies addressing this problem generally adopt two dominant paradigms: discriminative feature alignment and contrastive prototype learning.

Some studies have sought to address the problem of feature drift in FL by focusing on aligning feature representations. FRAug [1] employs data augmentation to generate synthetic embeddings encompassing global information and client-specific characteristics. FedSea [25] aims to mitigate feature drift by aligning feature distributions to transform raw features into an IID format. FedCiR [19] addresses feature drift by maximizing mutual information between representations and labels while minimizing mutual information between client-specific representations conditioned on labels. Similarly, MOON [15] encourages local models to align with the global feature distribution by constraining updates based on the similarity between local and global representations. Unlike traditional aggregation-based frameworks, ADCOL [16] employs adversarial learning to enforce a unified representation distribution across clients, thereby alleviating inter-client feature drift. However, it adopts a weak form of collaboration that does not address class boundaries from a global view. Moreover, its adversarial mechanism of directly transmitting features to the server introduces potential privacy risks. Our method adopts stronger collaborative learning to enhance global class boundaries and privacy-preserving adversarial learning to alleviate inter-client feature drift.

Some studies have focused on prototype-driven federated learning paradigms. Prototypes can compact feature embeddings through prototype abstraction, reducing communication bandwidth and preserving data privacy [29]. Tan et al. [26, 27] proposed a supervised contrastive loss function leveraging both global and local prototypes to minimize the distance between feature representations and class prototypes, thereby addressing data heterogeneity. However, using the average feature as a prototype for each class may overlook intra-class variability within the feature space. To address this, MP-FedCL [23] utilizes clustering on the client side to generate multiple prototypes per class, capturing intra-class variation and mitigating feature drift arising from these differences. Following the effort of MP-FedCL, FedPLVM [28] further enhances local training through a two-stage clustering process between clients and the server, incorporating an α-sparsity prototype loss function to optimize performance. By leveraging the privacy-preserving nature of prototypes, this approach effectively addresses privacy and security concerns while using global prototypes to strengthen class-specific information within

feature representations. In the FedPall framework, we enhance the collaboration between the client and the server through global prototypes. With the server's assistance, the client gains access to global category information, which helps to bring similar category data closer together and push data from different categories farther apart. We also use mixed features with global category information to enhance the global classifier.

3. Method

3.1. Problem Description

We define *feature drift* as follows: Given a dataset $\mathbb{D}$ with features x and labels y, while the conditional distribution $P_i(x|y)$ differs across clients, the marginal distribution $P(y)$ remains the same. This means that the same label may have significantly different features across clients. For example, due to variations in environment, geographic location, and cultural differences, the structural features of houses can vary widely.

With feature drift in FL, our goal is to optimize each client's personalized model loss while leveraging the potential performance gains from collaborative learning across clients [32]. In the FedPall framework, there are a total of N clients, each client n has a private dataset D_n. Based on this goal, we formulate the overall optimization objective of the FedPall framework as follows:

$$\min_{\theta_1,\theta_2,\ldots,\theta_N \in \mathbb{R}^{d_1}} F(\theta) := \frac{1}{N}\sum_{n=1}^{N} f_n(\theta_n), \tag{1}$$

where f_n represents the expected loss obtained from client n using the global model parameters under the dataset D_n, and θ_i represents the local model parameters of client i.

3.2. FedPall Framework

Existing approaches to addressing the feature drift problem in FL typically focus on either collaborative learning or adversarial learning in isolation. This can result in models that either fail to adequately capture class-related information in the feature representations or exhibit persistent discrepancies in feature distributions across clients. To address these limitations, we propose integrating both adversarial and collaborative learning to effectively mitigate feature drift in FL settings. In this section, we present the FedPall framework by elaborating on its adversarial and collaborative learning. The framework of the overall approach is shown in Fig. 2a. It is structured into four key procedures: generating global prototypes, training local models, training global model, and decentralizing global classifier.

We define certain model symbols here that will be used in this section. The local model $F(\cdot)$ consists of two components, a feature extractor $G(\cdot)$ (e.g., Resnet50 [6] for image data) and a classifier $H(\cdot)$. We use a multilayer perceptron (MLP) as our Amplifier, and except for the output layer, the number of nodes in other layers is consistent with that of the classifier.

3.2.1. Generating Global Prototypes

Several studies [21, 26] suggest that category-centered prototypes are a privacy-friendly form of global knowledge. We leverage collaboration between clients to aggregate and generate global prototypes. Typically, the class prototype for each category is represented by the mean of the features for that category. The local prototype for the k-th category on client n is defined as:

$$c_n^k = \frac{1}{N_n^k}\sum_{(x,y)\in D_n^k} G_n(x), \tag{2}$$

where D_n^k and N_n^k represent the data samples and the number of samples for the k-th category on client n, respectively.

Gathering all the local prototypes together forms a local prototype set, which can be defined as:

$$\mathcal{O}_n = \{c_n^1, c_n^2, ..., c_n^K\} \in \mathbb{R}^{K\times d}, \tag{3}$$

where K represents the number of categories owned by each client, and d denotes the output feature dimension.

Since we are only addressing the problem of feature drift, all clients have the same number of categories. Upon receiving the local prototype set and local label proportions $\mathcal{N} = \{\{N_n^k\}_{k=1}^K | n \in \mathcal{A}\}$ sent by client set $\mathcal{A}$, the server integrates the local prototypes from all clients to form the global prototypes,

$$\mathcal{G}^k = \sum_n \frac{N_n^k}{\sum_n N_n^k} c_n^k \tag{4}$$

The global prototypes set can be represented as

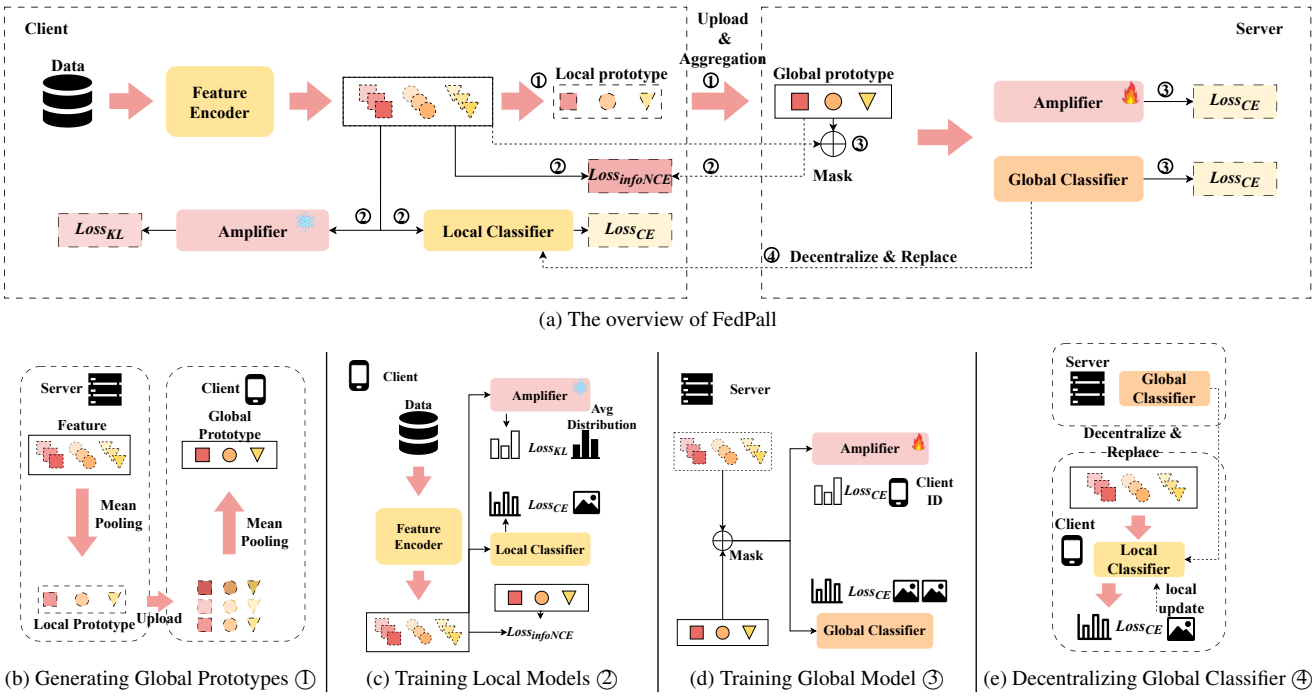
$$\mathcal{G} = [\mathcal{G}^1, \ldots, \mathcal{G}^k, \ldots, \mathcal{G}^K] \tag{5}$$

Next, the server sends the global prototype set $\mathcal{G}$ to each client to guide local model training. You can refer to Fig. 2b for a better understanding of the generation of the global prototype.

3.2.2. Training Local Models

The goal of this module is to train an effective feature encoder that maps the raw data from different clients into a unified feature space, where the feature distributions are aligned and the class-related information is enhanced.

As mentioned earlier, due to feature drift, training a local classifier alone is not sufficient for accurately classifying data with feature drift. To address this, we apply adversarial learning to train a feature encoder. Specifically, we use a global amplifier, trained on the server, which amplifies



(a) The overview of FedPall

(b) Generating Global Prototypes ① (c) Training Local Models ② (d) Training Global Model ③ (e) Decentralizing Global Classifier ④

Figure 2. The FedPall framework and detailed phases

the heterogeneous information in the features from different clients. At the client's side, we apply the amplifier and use Kullback-Leibler (KL) divergence to reduce the heterogeneous information in the features, thereby creating an adversarial learning setup between the client and server. The objective is to improve the generalization ability of the feature encoders across clients while minimizing the client-specific heterogeneity in the feature representations. Let $x^{(i)}$ denote the i-th dimension of vector x, the KL divergence is calculated as:

$$\begin{aligned}\mathcal{L}_{KL} &= \sum_{(\mathbf{x},\mathbf{y})\in D_n} D_{KL}(A_n(G(\mathbf{x}))||[\frac{1}{N}]^N) \\ &= \sum_{(\mathbf{x},y)\in D_n} \sum_{i}^{N} A_n(G(\mathbf{x}))^{(i)} log N A_n(G(\mathbf{x}))^{(i)}.\end{aligned} \tag{6}$$

where, $x^{(i)}$ denotes the i-th dimenson of vector x, $A_n(\cdot)$ represents the amplifier of the n-th client, $D_{KL}(P||Q)$ represents the KL divergence of P and Q.

After adversarial learning, although the feature representations of different clients are mapped to similar feature distributions, the class-related information may be blurred. To address this problem, we propose using contrastive learning to reinforce the class-related information within the feature encoder. To prevent information leakage, we enable collaboration between the global prototypes (server-side) and local features (client-side). Specifically, we employ the InfoNCE loss to minimize the distance between local features and their corresponding global prototypes, while maximizing the distance between local features and global prototypes of other classes. This approach strengthens the class-related representation within the feature encoder. The formula for the global-prototype contrastive loss is as follows:

$$\mathcal{L}_{infoNCE} = \sum_{(\mathrm{x},y)\in D_n} -log\frac{exp(sim(G(\mathrm{x}),\mathcal{G}^y)/\tau)}{\sum_{y_\alpha\in A(y)} exp(sim(\mathrm{x},\mathcal{G}^{(y_\alpha)})/\tau)}, \tag{7}$$

where $A(y) := \{y_\alpha \in [1,|\mathcal{G}|] : y_\alpha \neq y\}$ is the set of labels distinct from y, τ is the temperature to adjust the tolerance for feature difference, and $sim(x,y)$ represents the cosine similarity of x and y.

We combine adversarial learning and collaborative learning to address the feature drift problem in FL. By leveraging the two loss functions defined above, along with the local cross-entropy loss, we progressively train the local feature encoder at each client. The overall loss function for each client is as follows:

$$\mathcal{L} = \mathcal{L}_{CE(\mathrm{x},y)\sim D_n}(F(\mathrm{x}),y) + \mu\mathcal{L}_{KL} + \delta\mathcal{L}_{infoNCE}, \tag{8}$$

where μ denotes the weight of the $\mathcal{L}_{KL}$ divergence and $\mathcal{L}_{CE}$ denotes the cross-entropy loss.

The local model $F(\cdot)$ is updated using Eq. (8). For a comprehensive visualization of the global prototype generation process, consult Fig. 2c where the workflow is systematically delineated.

3.2.3. Training Global Model

Due to the limited local view of clients, it is difficult for them to train an accurate classifier. Therefore, we upload the encrypted mixed features to the server to train a classifier with a global perspective. Additionally, we leverage the global view from the server to train an amplifier used for adversarial learning with the clients.

For each feature z_n^k of class k on client n, we obtain a prototype mixed feature by performing a weighted fusion with the corresponding global prototype:

$$r_n^k = \alpha \times z_n^k + (1-\alpha) \times \mathcal{G}^k, \tag{9}$$

where $z_n^k \in Z_n^k = \{z_n^{i,k}\}_{i=1}^S$. $\alpha \sim U(u_f, u_r)$ are the mix parameters, with u_f and u_d representing two hyperparameters of a uniform distribution. S represents the number of samples of the k-th category in client n.

Building on this, we employ a Bernoulli mask to further reduce the risk of privacy leakage,

$$\begin{aligned} Mask &= \{X_1, X_2, \ldots, X_d\}, \\ X_i &\sim \text{Bernoulli}\,(\beta)\,, \quad \forall i \in [1, d], \end{aligned} \tag{10}$$

The final output of the prototype mixing mechanism is derived by selecting the mixed feature elements based on the noise mask:

$$\tilde{r}_n^k = Mask \odot r_n^k, \tag{11}$$

where $\odot$ is an element-wise *and* operator.

After generating the prototype mixed features, the client will form the set $\mathcal{D}_{R_L}(R, Y)$ using the prototype mixed feature set $R_n = \{\tilde{r}_n^1, ..., \tilde{r}_n^k, ..., \tilde{r}_n^K\} \in \mathbb{R}^{K \times d}$ and the corresponding label set Y, which will be sent to the server.

The server updates the global amplifier A using the mixed feature sets from each client along with the corresponding client IDs. Specifically, we first construct the dataset for training the amplifier, denoted as $\mathcal{D}_{R_I}(R, I)$, where I represents the client IDs. The amplifier is then updated by minimizing the empirical risk:

$$\mathbb{E}_{(R,I)\sim\mathcal{D}_{R_I}} \ell_{CE}(R, I). \tag{12}$$

At the same time, the server updates the global classifier C_g using the mixed prototype features and the class labels from the clients Y. This is done by minimizing the empirical risk:

$$\mathbb{E}_{(R,Y)\sim\mathcal{D}_{R_L}} \ell_{CE}(R, Y). \tag{13}$$

We show the training process of the global classifier in Fig. 1c.

3.2.4. Decentralizing Global Classifier

Finally, we deploy the global classifier C_g to each client to replace the original local classifiers C_c. The purpose of this is to obtain a more generalized classifier that can alleviate the feature drift problem. To allow the global classifier to adapt to the personalized characteristics of local data, we retrain it on the client's local data, thereby enhancing the classifier's accuracy and improving its performance on individual client data distributions. And you can understand the deployment method of the server-side global classifier through Fig. 2e.

4. Discussion

Computational Cost Compared to the standard federated learning model, our adversarial collaborative learning approach introduces an amplifier and a global classifier. However, the number of parameters for these two components is much smaller than those of the other components. In our design, both the amplifier and the classifier are designed as three-layer MLPs. Compared to the feature extractor (with a total of 23.5M parameters), the classifier (with a total of 1.32M parameters) and the amplifier (with a total of 1.33M parameters) account for approximately 5.61% and 5.59%, respectively. Moreover, the client-side amplifier remains frozen, functioning exclusively in the forward pass for loss calculation while being disabled during backpropagation cycles.

Communication Efficiency Compared to traditional federated learning methods like FedAvg, which sends local model parameters to the server, we use prototype-mixed features as the communication medium. Assume consistent tensor representations between model parameters and prototype-fused features, with the local model's feature extractor specifically employing a ResNet-50 architecture that produces 2048-dimensional embeddings. During the uploading process, our approach requires all clients to upload an average of 12,122 prototype-mixed features. In contrast, with FedAvg, each client needs to upload approximately 24.8 million parameters of model parameters. During the downloading process, our method only uses the parameters of the amplifier and global classifier as the communication content, which accounts for about 10.6% of FedAvg's communication cost (refer to the above paragraph).

Privacy Preserving Research has shown that the mutual information between input data and extracted latent features remains statistically insignificant [24]. Our framework implements a two-stage protection mechanism: The communication medium (mixed prototype features) first applies dual obfuscation: (1) prototype-based feature blending followed by (2) stochastic Bernoulli masking. This layered approach establishes information-theoretic security guarantees against model inversion attacks.furthermore, the class prototypes retain only a minimal amount of class information, exhibiting strong privacy-preserving properties. In

contrast to ADCOL [16], which simply sends raw features to the server, our prototype-mixed feature structure incorporates this dual encryption process, further enhancing privacy protection. This integration means that through the shared amplifier mechanism, feature spaces from different clients become intrinsically unified, ultimately diminishing privacy leakage risks.

Limitation While our framework currently specializes in image recognition tasks, its extension to NLP or time-series analysis remains unexplored. Successful cross-domain adaptation requires two key developments: (1) establishing domain-specific feature representations and prototype definitions, and (2) redesigning loss functions according to task semantics. For NLP applications, this implies reconfiguring the standard classification paradigm into autoregressive prediction frameworks. Architectural adaptations are equally crucial - particularly the incorporation of RNN-based structures with inherent temporal modeling capabilities for sequential data processing.

5. Experiments

5.1. Experimental Setup

Datasets We conduct experiments on three publicly available feature drift datasets: Digits [31], Office-10 [5], and PACS [14]. Specifically, (1) the Digits dataset consists of five different domain sources: MNIST [12], SCHN [22], USPS [8], SynthDigits [4], and MNIST-M [4]; (2) the Office-10 dataset includes four distinct sources: Amazon, Caltech, DSLR, and WebCam; (3) the PACS dataset consists of four sources: Art Painting, Cartoon, Photo, and Sketch. Datasets Office-10 and PACS are real-world images from natural scenes, which inherently exhibit feature drift due to the diversity of their sources. Digits is a digit recognition dataset. In line with [18, 27], we do not use the entire Digits dataset for feature transformation experiments but rather a subset of 10% of the data. For datasets Office-10 and PACS, we used all of the datasets for the experiment. Additionally, we split each dataset into training and testing sets with an 8:2 ratio.

Baselines We compare FedPall with ten baselines, including SingSet (where each client independently trains a model). FedAvg [20] is the most classic federated learning algorithm, while FedProx [17], PerFedAvg [3], and FedRep [2] are personalized federated learning methods. FedBN [18], ADCOL [16], MOON [15] and FedProto [26] are personalized federated learning algorithms for cross-domain learning, all of which address the issue of non-IID features to some extent. In addition, we explored the ability of RUCR [7] to solve the feature drift problem.

Model and Hyper-parameter Setup All algorithms use the same local model architecture to ensure a fair comparison. The local model consists of three components: (1) a feature extractor based on ResNet50 (without the classifier layer); (2) a classifier, which is a three-layer MLP with a hidden layer size of 512. The amplifier is also a three-layer MLP, with an input dimension of 2048 and a hidden layer size of 512. The output dimensions of both the predictor and the amplifier are adjusted according to the number of categories and data sources in the dataset. By default, the number of clients is the same as the number of data sources in the dataset, with each client owning a data source. The output dimension of the predictor is aligned with the number of categories in the dataset. For the Digits dataset, we set the number of local training epochs to 5. For the Office-10 and PACS datasets, we set the number of local training epochs to 10. The global training epochs are set to 10 for all datasets. We use the SGD optimizer with a learning rate of 0.01 for model training. Due to varying degrees of feature drift in the dataset itself, our algorithm makes slight adjustments to the hyperparameter values in Eq. (8). However, to ensure fair comparison, the loss value common to all algorithms uses the same hyperparameter value. Except for the digits dataset, for which the values of mu and delta are set to 0.7 and 0.3 respectively, the values of mu and delta are set to 0.1 for all other datasets. All experiments are conducted on an NVIDIA GeForce RTX 4090 GPU. By default, for each result, we run the three experiments with different random seeds for data processing and initialization of model parameters.

5.2. Experimental Analysis

We conduct the evaluation on three publicly available feature-drifted datasets (Digits, Office-10, and PACS) and compare the performance of the FedPall framework with classical and state-of-the-art baselines. As shown in Table 1, our proposed framework achieves state-of-the-art accuracy on all three datasets.

We first discuss the experimental results based on each individual dataset. On the Office-10 dataset, the overall accuracy of the FedPall framework surpasses that of the second-best method, ADCOL, by approximately 3 percentage points. By examining the accuracy across the four sub-datasets (as for the four clients), FedPall improves the overall performance by significantly elevating the performance of some sub-datasets while not decreasing that of others. On the Digits dataset, it is evident that FedPall outperforms all other models, achieving an accuracy that is approximately 1.1 percentage points higher than the second-best model, FedBN. Moreover, FedPall surpasses the popular baseline with a comparative idea, ADCOL, by about 2.2 percentage points. The Digits dataset contains images that are relatively easy to classify, and the de-

		SingleSet	FedAvg	FedProx	PerfedAvg	FedRep	FedBN	MOON	FedProto	ADCOL	RUCR	ours(FedPall)
Office-10	amazon	73.96(2.71)	56.94(2.46)	56.60(2.57)	57.12(2.17)	45.31(1.88)	40.80(15.75)	51.74(16.11)	69.44(2.10)	73.26(4.37)	52.08(8.53)	72.92(1.38)
	caltech	44.74(3.15)	46.52(4.63)	50.96(5.19)	50.81(1.56)	38.37(4.92)	33.93(6.48)	41.33(13.62)	39.41(6.32)	37.19(1.68)	44.30(1.03)	44.74(8.74)
	dslr	60.22(6.72)	30.11(4.93)	33.33(10.37)	31.18(4.93)	34.41(4.93)	38.71(3.23)	24.73(1.86)	65.59(4.93)	76.34(4.93)	30.11(6.72)	77.42(3.23)
	webcam	71.26(2.63)	37.93(6.22)	43.68(7.18)	47.13(7.77)	55.75(2.63)	30.46(6.05)	33.33(12.71)	71.26(4.34)	71.26(2.63)	37.36(4.98)	74.71(1.00)
	avg	62.54(0.38)	42.88(1.18)	46.14(2.64)	46.56(2.89)	43.46(1.34)	35.97(6.54)	37.78(10.89)	61.43(1.74)	64.51(1.79)	40.96(0.61)	**67.45(2.69)**
Digits	MNIST	95.51(0.22)	92.86(2.24)	91.79(2.95)	90.10(4.79)	86.54(6.08)	96.69(0.11)	93.41(1.14)	96.37(0.50)	96.30(0.41)	92.59(1.96)	97.24(0.42)
	SVHN	71.09(0.91)	77.39(0.21)	76.92(0.28)	75.64(0.42)	67.17(1.73)	79.44(0.25)	79.63(0.75)	72.50(0.29)	75.12(2.08)	77.94(0.25)	78.00(0.36)
	USPS	86.40(0.27)	89.25(0.89)	89.23(1.41)	88.69(0.69)	89.95(2.95)	90.07(0.54)	81.76(0.72)	87.01(0.83)	86.72(1.25)	88.85(2.39)	87.28(1.29)
	SynthDigits	95.15(0.13)	95.49(0.07)	95.39(0.12)	95.00(0.16)	94.21(0.78)	95.61(0.06)	96.63(0.23)	95.29(0.61)	96.43(0.29)	95.98(0.17)	95.26(0.43)
	MNIST-M	76.56(0.40)	73.81(1.45)	74.02(1.49)	73.21(0.78)	69.11(0.94)	76.25(0.39)	72.16(0.92)	78.27(1.20)	78.28(4.39)	72.65(0.37)	85.90(1.39)
	avg	84.94(0.06)	85.76(0.86)	85.47(1.14)	84.53(1.31)	81.40(2.47)	87.61(0.11)	84.72(0.60)	85.89(0.23)	86.57(1.32)	85.60(0.87)	**88.74(0.15)**
PACS	art_painting	33.58(0.84)	25.79(1.93)	24.33(4.14)	26.52(2.19)	26.93(3.32)	36.66(1.76)	30.58(1.97)	32.68(0.70)	34.87(1.15)	24.66(1.10)	35.60(0.56)
	cartoon	58.53(2.48)	45.36(2.29)	51.38(0.56)	48.27(1.24)	44.37(2.09)	55.63(1.95)	51.52(1.78)	57.25(1.51)	57.18(0.80)	47.49(3.32)	59.73(2.34)
	photo	63.01(1.93)	48.66(3.08)	49.55(1.95)	46.88(2.64)	41.94(2.76)	66.07(1.04)	53.02(3.34)	64.00(1.34)	62.12(1.98)	47.48(6.22)	64.69(1.29)
	sketch	79.70(0.13)	49.03(1.98)	40.74(1.54)	44.42(3.77)	40.48(1.25)	79.57(1.65)	55.12(1.37)	79.61(0.81)	80.12(1.03)	42.17(2.40)	82.23(0.71)
	avg	58.70(1.23)	42.21(1.59)	41.50(1.82)	41.52(1.90)	38.43(1.11)	59.48(1.44)	47.56(0.88)	58.39(0.25)	58.57(0.58)	40.45(1.98)	**60.56(0.36)**

Table 1. The top-1 accuracy (%) of each algorithm on each sub-dataset of the Office-10, Digits, and PACS datasets is compared, along with the average top-1 accuracy across all sub-datasets. The mean and standard deviation (std) from three random trials (using different random seeds, with other experimental settings remaining the same) are reported. The highest accuracy for each dataset is highlighted in bold, and the second-highest accuracy is underlined.

gree of feature drift is smaller compared to the Office-10 dataset. All baseline models achieve reasonably good accuracy on this dataset. Specifically, adversarial learning helps mitigate the heterogeneous information in the MNIST-M client. In addition, collaborative learning allows the local model on the MNIST-M client to benefit from the knowledge shared by other clients and the server. Similarly, our algorithm demonstrates strong performance on the PACS dataset, achieving an overall accuracy that is approximately 1.1 percentage points higher than the second-highest result produced by FedBN. FedPall achieves the highest or second-highest accuracy across all sub-datasets.

We also discuss the performance of FedPall as compared to other state-of-the-art baselines across all three datasets. The average accuracy of FedPall consistently outperforms that of ADCOL in all three datasets, achieving an increase ranging from about 1.1 to 2.9 percentage points. In addition, even though FedBN can achieve accuracy comparable to our method on datasets Digits and PACS, our method outperforms it significantly by 31.5 percentage points on datasets Office-10. As mentioned earlier, the Office-10 dataset comes from real-world data, where feature drift is particularly prominent, and there is also a significant distribution difference between the training and testing sets, leading to the suboptimal performance of the FedBN method on this dataset. In contrast, the special design incorporating both adversarial and collaborative learning in FedPall enables it to adapt well to the Office-10 dataset.

5.3. Ablation Study

Effect of loss combination In this section, we analyze the impact of KL loss and InfoNCE loss on the final model performance when training the local feature encoder. We conducted several ablation experiments, specifically setting up three comparative groups: removing both KL loss and InfoNCE loss simultaneously, removing only KL loss, and removing only InfoNCE loss. As shown in Figure 3, the algorithm performs best when all losses are retained, which validates the reliability of the loss combination we designed.

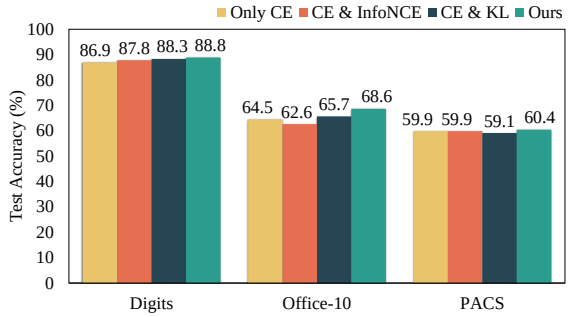


Figure 3. We evaluate the top-1 accuracy averaged over all clients using different loss function combinations on different datasets.

Specifically, on the Office-10 dataset, our method outperforms the model using only CE loss by nearly 4 percentage points. Interestingly, the combination of CE loss and InfoNCE loss performs even worse than using only CE loss. This suggests that, when the feature drift problem is more pronounced, reinforcing the category information solely through InfoNCE loss may exacerbate the feature drift. Although the combination of CE loss and KL loss performs better than using CE loss alone on the Office-10 dataset, its performance on the PACS dataset falls below expectations (even lower than CE loss), indicating that the model's robustness is weaker when using CE and KL losses together. This can lead to some loss of category information, and while the model can adapt to some heterogeneous datasets, it remains highly unstable.

To analyze the effects of KL and InfoNCE losses on feature distributions, we visualize categorical feature distributions across clients in the Office-10 dataset. For clarity, Figure 4 displays randomly sampled data points through

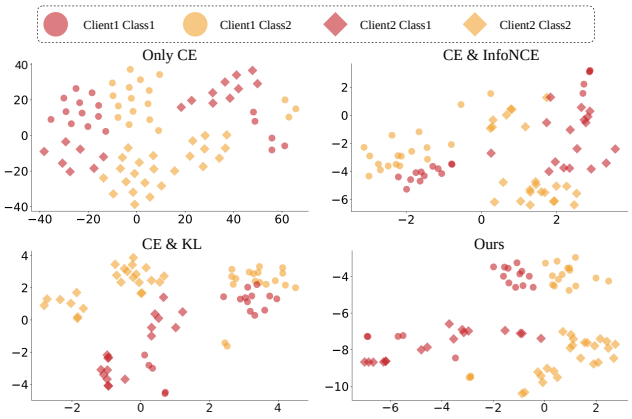


Figure 4. We plotted the feature distribution of different categories under different clients, corresponding to the four loss combination strategies of Fig. 3

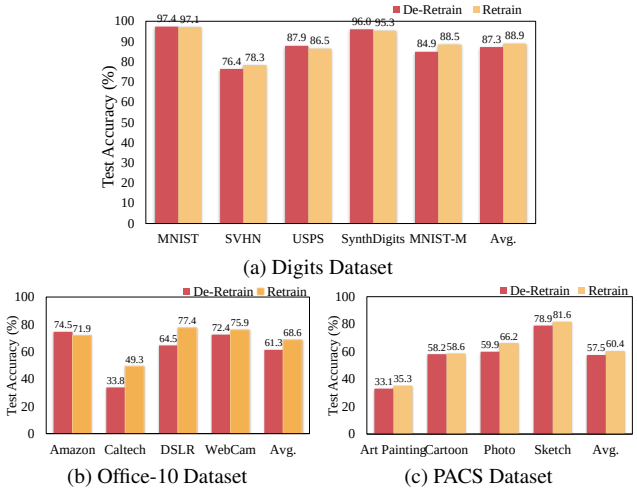


Figure 5. Comparison of accuracy with and without training the global classifier on the three datasets.

t-SNE projections. The CE-loss-only model exhibits poor feature drift handling, particularly in Client 1 where decision boundaries remain ill-defined. Introducing InfoNCE loss improves intra-client class separation compared to CE loss alone, though inter-client feature drift persists, causing ambiguous global boundaries. The CE+KL combination demonstrates dual effects: KL loss effectively reduces inter-client distance for identical classes, achieving clearer global boundaries. However, this comes at the cost of compressed intra-class spacing in Client 1, manifested as overlapping clusters and outlier formation that impair local classification. Our unified loss function synergistically coordinates these effects: KL loss aligns cross-client same-class features while CE and InfoNCE losses jointly enforce intra-client class separation. This balanced interaction produces well-separated yet compact clusters, ultimately enhancing classification performance and validating our algorithm's efficacy.

In summary, our method not only outperforms other combinations in the simple handwritten digit recognition scenario but also demonstrates superior performance on other real-world datasets. This highlights the robustness and generalization capability of the loss combination we designed.

Comparison of different classifier replacement methods. We validate the global classifier's effectiveness via an ablation study by removing it. Results in Figure 5 reveal that while the Digits dataset's simplicity causes slightly lower accuracy for the global classifier than local classifiers on some sub-datasets (panel 5a), it outperforms the non-global-classifier baseline across multiple sub-datasets. Notably, our method achieves 3.61% higher accuracy on MNIST-M, yielding superior overall performance. Panels 5b and 5c highlight the global classifier's advantages for datasets with significant feature drift: It surpasses the baseline on nearly all Office-10 sub-datasets (panel 5b), achieving 49.33% accuracy (15.55% improvement) on Caltech. For PACS (panel 5c), it consistently outperforms the baseline across all sub-datasets with up to 3% gains.

These results confirm the necessity of FedPall's global classifier, which captures cross-client category information to enhance client-server collaboration and improve the framework's generalization against feature drift.

6. Conclusion

In this study, we focus on the feature drift problem in FL. The feature drift problem causes the same class samples on different clients to have distinct feature distributions, making it difficult for traditional model aggregation methods to handle such data heterogeneity. To tackle this problem, we design a prototype-based adversarial collaborative framework to unify feature spaces and enhance classification boundaries. The global classifier is retrained with mixed features to further grasp classification-relevant information from a global perspective. Our method has empirically achieved state-of-the-art performance in popular feature-drifted datasets with multiple data sources.

Currently, our work is limited to image classification tasks and has not been extended to other domains. In the future, we aim to systematically validate the framework's generalizability across multimodal learning scenarios.

References

[1] Haokun Chen, Ahmed Frikha, Denis Krompass, Jindong Gu, and Volker Tresp. Fraug: Tackling federated learning with non-iid features via representation augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4849–4859, 2023. 2

[2] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pages 2089–2099. PMLR, 2021. 6

[3] 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. 6

[4] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015. 6

[5] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012. 6

[6] 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. 3

[7] Wenke Huang, Yuxia Liu, Mang Ye, Jun Chen, and Bo Du. Federated learning with long-tailed data via representation unification and classifier rectification. *IEEE Transactions on Information Forensics and Security*, 2024. 6

[8] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994. 6

[9] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019. 1

[10] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2(6), 2019. 1

[11] Jakub Konečnỳ, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015. 1

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[13] Boyuan Li, Shengbo Chen, and Zihao Peng. New generation federated learning. *Sensors*, 22(21):8475, 2022. 1

[14] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, . 6

[15] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722, 2021. 2, 6

[16] Qinbin Li, Bingsheng He, and Dawn Song. Adversarial collaborative learning on non-iid features. In *International Conference on Machine Learning*, pages 19504–19526. PMLR, 2023. 2, 6

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020. 1, 6

[18] Xiaoxiao Li, Meirui JIANG, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. In *International Conference on Learning Representations*, . 2, 6

[19] Zijian Li, Zehong Lin, Jiawei Shao, Yuyi Mao, and Jun Zhang. Fedcir: Client-invariant representation learning for federated non-iid features. *IEEE Transactions on Mobile Computing*, 2024. 2

[20] 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. 1, 6

[21] Xutong Mu, Yulong Shen, Ke Cheng, Xueli Geng, Jiaxuan Fu, Tao Zhang, and Zhiwei Zhang. Fedproc: Prototypical contrastive federated learning on non-iid data. *Future Generation Computer Systems*, 143:93–104, 2023. 3

[22] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011. 6

[23] Yu Qiao, Md Shirajum Munir, Apurba Adhikary, Huy Q Le, Avi Deb Raha, Chaoning Zhang, and Choong Seon Hong. Mp-fedcl: Multi-prototype federated contrastive learning for edge intelligence. *IEEE Internet of Things journal*, 2023. 2

[24] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017. 5

[25] Min Tan, Yinfu Feng, Lingqiang Chu, Jingcheng Shi, Rong Xiao, Haihong Tang, and Jun Yu. Fedsea: Federated learning via selective feature alignment for non-iid multimodal data. *IEEE Transactions on Multimedia*, 2023. 2

[26] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8432–8440, 2022. 2, 3, 6

[27] Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. *Advances in neural information processing systems*, 35:19332–19344, 2022. 2, 6

[28] Lei Wang, Jieming Bian, Letian Zhang, Chen Chen, and Jie Xu. Taming cross-domain representation variance in federated prototype learning with heterogeneous data domains. *arXiv preprint arXiv:2403.09048*, 2024. 2

[29] Li Wang, Quangui Zhang, Lei Sang, Qiang Wu, and Min Xu. Federated prototype-based contrastive learning for privacy-preserving cross-domain recommendation. *arXiv preprint arXiv:2409.03294*, 2024. 2

[30] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018. 1

[31] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 561–578. Springer, 2020. 6

[32] Tailin Zhou, Jun Zhang, and Danny HK Tsang. Fedfa: Federated learning with feature anchors to align features and classifiers for heterogeneous data. *IEEE Transactions on Mobile Computing*, 2023. 3

[33] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pages 12878–12889. PMLR, 2021. 2