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

AKPFL: A Personalized Federated Learning Architecture to Alleviate Statistical Heterogeneity

Shuoshuo Fang[1[0009-0003-0215-0803]] and Jialong Sun[2[0009-0001-1197-8624]]

Wenchao Zhang[3[0009-0009-6357-9666]] and Zongjian Yang[4[0009-0005-3200-0317]]

Kejia Zhang[1*[0000-0001-9715-6076]]

[1] School of Computer Science and Big Data, School of Cybersecurity,Heilongjiang University, Harbin, China
2232699@s.hlju.edu.cn, 20221543@s.hlju.edu.cn, zhangkejia@hlju.edu.cn
[2] Heilongjiang University, Harbin, China
20212644@s.hlju.edu.cn
[3] Nanjing Audit University, Nanjing, China
213080139@stu.nau.edu.cn
*Corresponding author: zhangkejia@hlju.edu.cn

Abstract. Federated Learning (FL) has gained considerable attention in machine learning for its ability to preserve data privacy while enabling collaborative modeling. However, statistical heterogeneity, such as non-independent and identically distributed (non-IID) data severely limits performance. To address this limitation, this study proposes an Adaptive Kernel Alignment-based Personalized Federated Learning framework (AKPFL). This approach achieves a balance between global model sharing and local adaptation by incorporating a dynamic kernel adjustment mechanism and a personalized model fusion strategy, thereby improving model generalization and robustness in heterogeneous data environments. Experimental results demonstrate that, compared to existing algorithms, AKPFL delivers substantial improvements in test accuracy on datasets such as Fashion MNIST, CIFAR-10, and CIFAR-100, particularly under high statistical heterogeneity. The code for the framework will be released publicly following the completion of the paper review process.

Keywords: Federated Learning, Personalized Federated Learning, Statistical Heterogeneity, Kernel Function, Meta-Learning.

1 Introduction

Modern machine learning has shown exceptional capability in handling high-dimensional and complex data. However, with the increasing demand for privacy protection and distributed data, traditional centralized learning has encountered challenges related to data security and storage costs in recent years. Federated Learning (FL), an emerging distributed learning paradigm, offers an effective solution to these issues [1]. It plays a unique role in the manufacturing [2], communication [3] and transportation industries [4].

Federated Learning strikes a balance between data privacy protection and collaborative optimization by training models on multiple local clients, uploading the trained model parameters to a central server for aggregation, and then distributing the aggregated model back to the clients [5]. However, the data on different clients often exhibit distinct local distributions in practice. This characteristic, known as local data statistical heterogeneity, presents significant challenges to the generalization performance of the global model [6]. Due to the severe statistical heterogeneity, a single global model struggles to generalize effectively across different local clients. As a result, balancing the global shared model with the regional, personalized models has become a critical issue in Federated Learning research.

Statistical heterogeneity refers to the non-IID (non-independent and identically distributed) problem in Federated Learning, caused by significant differences in data distributions among clients [7]. To address this challenge, several methods have been proposed. Ahmad et al. introduced a weight aggregation mechanism based on the Hessian matrix, which improves the robustness and convergence performance of the global model under statistical heterogeneity by more accurately capturing gradient information [8]. Azimi and Fodor proposed a hierarchical Federated Learning framework that integrates quantized communication optimization to coordinate gradient aggregation and model aggregation across multiple layers, thereby mitigating the impact of data distribution discrepancies on model performance [9]. Additionally, Li et al. developed a contrastive learning-based Federated Learning method, Model-Contrastive Federated Learning (MOON), which effectively alleviates the impact of statistical heterogeneity on model convergence by contrasting representations of the global and local models [10]. However, existing personalized aggregation schemes often face feature shift issues, where features from datasets with prominent characteristics disproportionately influence global clients. To systematically address these challenges, Tan et al. conducted a comprehensive review of personalized Federated Learning, analyzing current research progress and challenges while providing theoretical guidance for future studies [11]. Furthermore, Xu et al. proposed a balanced information and dynamic update prototype representation Federated Learning method (BD-FedProto), which dynamically adjusts the balance between local and global learning to enhance the adaptability and generalization of models under non-IID data [12].

This paper proposes a personalized Federated Learning scheme based on adaptive hierarchical kernel alignment to mitigate the feature shift phenomenon. By utilizing a central kernel alignment dynamic adjustment mechanism and a customized model layering method, the proposed approach alleviates the impact of statistical heterogeneity on the model's generalization ability. To provide a more intuitive understanding of the algorithm's overall framework and workflow, the core algorithm framework is illustrated in Figure1.

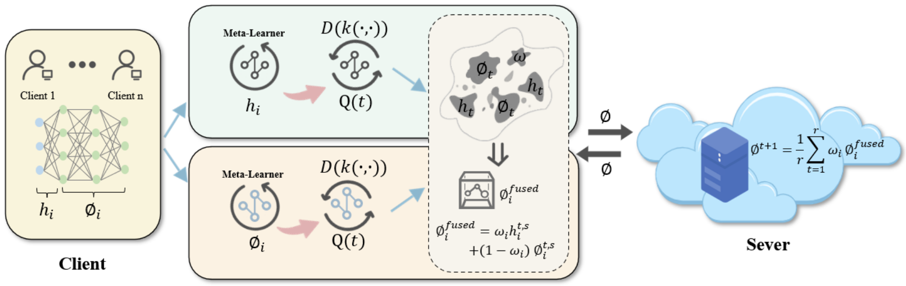

Fig. 1. This figure illustrates the training process of the personalized and global models in the Federated Learning algorithm. First, the clients split the local model into a personalized model h_i and global model $\emptyset_i$. The Meta-Learner performs preprocessing operations on both components, and the kernel distance function $D(k,\cdot)$ is used to calculate the distance between the model's previous and current iterations. A custom smoothing function $Q(t)$ is applied to dynamically adjust the model's optimization direction. Subsequently, the personalized and global models are fused using the weight ω_i, resulting in $\emptyset_i^{fused}$.The client uploads the fused model to the server, where the server aggregates the models uploaded by all clients to generate a new global model $\emptyset^{t+1}$ and sends it back to the clients, completing one iteration of the update.

Specifically, the main contributions of this paper are as follows:

1. We propose a dynamic adjustment mechanism based on kernel alignment to measure the feature deviation between local and global clients. This mechanism overcomes the limitations of traditional distance metrics, which tend to fail in high-dimensional spaces.
2. We empirically demonstrated the effectiveness of AKPFL,with its performance improving by several percentage points across different datasets. Notably, it exhibited outstanding robustness and generalization capabilities on the CIFAR-10 and CIFAR-100 datasets under pathological data distribution.
3. We establish a comprehensive and easily modifiable framework to ensure its effective application in real-world scenarios.

2 Related Work

In recent years, Federated Learning (FL) has garnered significant attention for its advantages in preserving data privacy and enabling distributed collaborative modeling. Li et al. [13] proposed the FedProx algorithm, which mitigates the global performance degradation caused by differences in client updates by incorporating a proximal regularization term. Reddi et al. [14] introduced the FedOpt framework, which employs adaptive optimization strategies to enhance model convergence and generalization. Ghosh et al. [15] proposed an aggregation mechanism based on kernel distance, which optimizes global model performance on heterogeneous data by measuring the similarity of client models.

Personalized Federated Learning (PFL) has emerged as a critical research direction by balancing global sharing with local adaptability. Fallah et al. [16] designed a

Personalized Federated Learning method based on MAML, enabling rapid client adaptation. T. Dinh et al. [17] proposed the pFedMe framework, which incorporates a Moreau regularization term to improve local model performance. Hanzely and Richtárik [18] introduced a hybrid optimization approach to effectively balance global and personalized needs.

The application of kernel functions has further advanced Federated Learning in heterogeneous data scenarios. Yurochkin et al. [19] applied Bayesian Nonparametric Federated Learning (BNFL) to integrate global and local distributions using kernel methods, significantly enhancing generalization. Tang et al. [20] proposed a Federated Learning method based on dynamically adjusted kernel functions, which improves robustness in nonlinear data scenarios through adaptive kernel selection. These studies highlight the potential of kernel functions in addressing the challenges posed by statistical heterogeneity.

3 Preliminary knowledge

Preliminary knowledge provides the theoretical foundation for model design and optimization. The proposed hierarchical personalized Federated Learning framework integrates the rapid adaptation capability of meta-learning, the dynamic adjustment mechanism of kernel functions, and a model fusion strategy. This combination enables the collaborative optimization of global sharing and local, personalized models, offering theoretical support for efficient modeling in heterogeneous data scenarios.

3.1 Federated Learning

Federated Learning (FL) is a distributed machine learning framework designed to train a global model through collaborative optimization while preserving data privacy. In this framework, the system consists of multiple clients, each with a local dataset $D_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{M_i}$, where M_i represents the size of the local dataset. Clients first train the model parameters $\emptyset_i$on their local data to minimize the local loss function $f_i(\emptyset)$:

$$\emptyset_i^t = argmin\, f_i(\emptyset) \tag{1}$$

where $\emptyset_i^t$ represents the locally optimized model parameters for client i. Subsequently, the central server receives the parameters $\emptyset_i$ uploaded by the clients and updates the global model parameters $\emptyset^{t+1}$ using a weighted average:

$$\emptyset^{t+1} = \frac{1}{n}\sum_{i=1}^{n} \emptyset_i^t \tag{2}$$

This process optimizes the global model through multiple rounds of iteration between the clients and the server. However, due to the non-IID nature of client data distributions, Federated Learning faces challenges in the generalization performance of the global model. This requires optimization strategies to balance the needs of the globally shared model and local adaptability.

3.2 Meta-Learning

To address the personalized modeling challenges posed by non-IID data in Federated Learning, we introduce the Model-Agnostic Meta-Learning (MAML) method. MAML optimizes the initial parameters $\emptyset$ of the global model, enabling it to rapidly adapt to client-specific local data, thereby enhancing personalized adaptability with minimal computational overhead. During training, MAML balances global sharing and local personalization through two optimization loops: inner and outer.

For the global model, MAML employs stochastic gradient descent (SGD) to quickly adjust the global model parameters, with the update formula given by:

$$\emptyset' = \emptyset - \alpha \nabla_{\emptyset} L(h(x), y) \tag{3}$$

where $\emptyset$ represents the global model parameters, L is the local loss function, and α is the learning rate.

MAML optimizes the personalized model parameters on local data to adjust the model for improved client adaptability. The updated formula for the personalized model is:

$$h' = h - \alpha \nabla_h L(h(x), y) \tag{4}$$

By jointly optimizing both the global and personalized model parameters, MAML enables the model to rapidly adapt to client-specific data, significantly improving modeling performance in non-IID environments.

3.3 Kernel Functions

Kernel Functions are used in Federated Learning to measure the similarity between personalized models and global models by calculating the distance between the model parameters from previous and current rounds, dynamically adjusting the model update process. In heterogeneous data scenarios, Kernel Functions ensure the generalization of the global model while enabling the personalized fitting process. For two sets of model parameters θ_1 and θ_2, the distance based on the Kernel Functions is defined as:

$$D(\theta_1, \theta_2) = \sqrt{k(\theta_1, \theta_1) + k(\theta_2, \theta_2) - 2k(\theta_1, \theta_2)} \tag{5}$$

where $k(\cdot,\cdot)$ denotes the kernel function used to calculate the similarity between vectors.

In this study, we utilized various kernel functions, including linear, RBF, polynomial, Laplacian, and Matern kernels. The linear kernel computes the inner product of parameters directly and is suitable for scenarios with linear correlations. The formula is:

$$k_{linear}(x, y) = x^T y \tag{6}$$

For complex nonlinear relationships, the RBF kernel captures the nonlinear features of data distribution through a Gaussian function:

$$k_{rbf}(x, y) = \exp\left(-\frac{|x - y|^2}{2\sigma^2}\right) \tag{7}$$

The polynomial kernel extends the feature space to capture nonlinear relationships, and its formula is:

$$k_{poly}(x,y) = (\gamma x^T y + c)^d \quad (8)$$

The Laplacian kernel, based on linear distance, emphasizes local nonlinear features with the following form:

$$k_{laplacian}(x,y) = \exp\left(-\frac{|x-y|}{\sigma}\right) \quad (9)$$

Additionally, the Matern kernel provides flexibility by adjusting the smoothness of the kernel function via the smoothness parameter ν. For instance, when $\nu = 1.5$, the formula is:

$$k_{matern}(x,y) = \left(1+\sqrt{3}\frac{|x-y|}{l}\right)\exp\left(-\sqrt{3}\frac{|x-y|}{l}\right) \quad (10)$$

The above Kernel Functions provide diverse tools for calculating data similarity. By selecting the appropriate kernel function based on task requirements, it is possible to construct a framework that effectively integrates heterogeneous data and personalized models, thereby improving modeling performance and optimization capabilities under heterogeneous data distributions.

3.4 PFL algorithm

We propose a hierarchical optimization framework that combines a global shared model and local, personalized models to address modeling challenges in non-IID data environments across clients. Through collaborative optimization, the global model captures shared features across clients. In contrast, the local, personalized models refine adjustments based on the distribution characteristics of client data, satisfying both global and local requirements.

Assume the system includes n clients, where each client i has a dataset $D_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{M_i}$.The global model parameters are denoted as ϕ, and the personalized model parameters as h_i.The joint optimization objective for the global and local models is

$$h_i(x) = E_{x\sim D_i}\left[L\left(h_i\left(q_\emptyset(x)\right),y\right)\right] \quad (11)$$

Where $f_i(\emptyset, h_i)$ is the loss function for client i,which accounts for the collaborative optimization of the global and personalized models. The personalized model is refined based on local data, and its predictive output is expressed as:

$$\min_{\Phi \in H} \frac{1}{n}\sum_{i=1}^{n} f_i(\emptyset, h_i) \quad (12)$$

where L measures the error between the predicted output $h_i(q_\emptyset(x))$ and the true label y.

Building on this, we introduce a model fusion strategy that dynamically adjusts the weights ω to combine the global and personalized models efficiently. The dynamic weights are adjusted based on the changes in the client's loss, represented as:

$$\omega = f\left(loss_{prev}, loss_{current}\right) \quad (13)$$

Where ω is the weighting coefficient used to balance the contribution of the global and personalized models in the current round. The fusion and update formulas for the personalized model and global model parameters are as follows:

$$h_i = \omega h_i^{prev} + (1-\omega) h_i^{current} \tag{14}$$
$$\emptyset_i = \omega \emptyset_i^{prev} + (1-\omega) \emptyset_i^{current} \tag{15}$$

The server aggregates the fused model parameters uploaded by the clients to update the global model:

$$\emptyset^{t+1} = \frac{1}{n} \sum_{i=1}^{n} \emptyset_i^{fused} \tag{16}$$

Through the dynamic fusion mechanism, clients can adaptively adjust between the global shared and personalized models based on task requirements. This reduces the generalization error of the global model and enhances the personalized model's adaptability to local data. Finally, the server optimizes the global model's performance by aggregating the fused parameters from clients, providing theoretical support and practical value for federated learning in federated data scenarios.

4 AKPFL Algorithm

This section elaborates on the core design and implementation of our algorithm, focusing on two key components: adaptive kernel function adjustment and personalized model fusion. By introducing a dynamic kernel adjustment mechanism, we accurately capture the feature shift between local and global models, enabling adaptive adjustments to the optimization direction and enhancing the model's robustness and generalization capabilities in heterogeneous data scenarios.

The personalized model fusion strategy also integrates kernel distance with a dynamic weight smoothing mechanism, achieving an efficient balance between global sharing and local personalization requirements. The server performs weighted aggregation of the fused model parameters uploaded by clients, further improving the stability and performance of the global model. These designs effectively address the challenge of statistical heterogeneity, significantly enhancing the applicability of federated learning in complex real-world scenarios.

4.1 Client update

Client-Side Personalized Model: In local, personalized model updates on the client side, the initial parameters of the personalized model are first adjusted through meta-learning to enable rapid adaptation to local data. Based on these initial parameters, further personalized optimization for local tasks is performed, generating the personalized head parameters for the current round $h_i^{t,s}$.

To enhance the robustness and generalization capability of the personalized model, a structured regularization constraint is introduced during the optimization process. An L2 regularization term is applied to the personalized head parameters to effectively suppress overfitting. The regularization formula is as follows:

$$R\left(h_i^{t,s}\right) = \tau \sum_{\rho \in head} |p|_2^2 \tag{17}$$

where τ is the regularization weight, and $|p|_2^2$ represents the L2 norm of the personalized head parameters. This regularization improves the stability and adaptability of the model. The updated formula for the personalized head is as follows:

$$h_i^{t,s} = SGD\left(f_i\left(h_i^{t,s-1}, \emptyset^t\right), h_i^{t,s-1}, \delta\right) + R\left(h_i^{t,s}\right) \tag{18}$$

where $SGD(\cdot)$ denotes the stochastic gradient descent algorithm used to update the personalized head parameters.

After completing the gradient updates, the difference between the personalized head in the current and previous rounds is calculated using a kernel function. The kernel function adaptively computes the kernel distance between the personalized parameters from the current and previous rounds to measure the extent of the parameter change:

$$D\left(h_i^{t,s}, h_i^{t,s-1}\right) = \sqrt{k\left(h_i^{t,s}, h_i^{t,s}\right) + k\left(h_i^{t,s-1}, h_i^{t,s-1}\right) - 2k\left(h_i^{t,s}, h_i^{t,s-1}\right)} \tag{19}$$

Combining the kernel distance D with an auxiliary adjustment function $Q_{head}(t)$, the smoothing coefficient μ is dynamically adjusted. The function $Q_{head}(t)$ is adapted based on the current training round it and the total number of rounds T_{head}:

$$Q_{head}(t) = 1 - \frac{t}{T_{head}} \tag{20}$$

The dynamic smoothing coefficient μ_i is then calculated as:

$$\mu_i = D\left(h_i^{t,s}, h_i^{t,s-1}\right) \cdot Q_{head}(t) \tag{21}$$

Using the dynamic smoothing coefficient μ, the update strategy for the personalized head parameters in the current round is as follows:

$$h_i^{t,s} = h_i^{t,s-1} + (1 - \mu_i)h_i^{t,s} \tag{22}$$

This design enables the model to adaptively smooth the current optimization using historical information, ensuring that the training process for the personalized head parameters converges gradually. It also effectively enhances the robustness and generalization capability of the personalized model. By incorporating regularization terms and a dynamic smoothing mechanism based on kernel functions, the client achieves efficient optimization and stable updates for the personalized model.

Global Model: Building upon the optimization of personalized models, global model training aims to integrate local information from all clients, achieving efficient generalization of shared knowledge across clients. Meta-learning is used to pre-train the global model parameters, enabling the model to quickly adapt to the diverse data distributions of different clients, thereby reducing local optimization time and providing a robust initialization for subsequent global model updates.

In each round of global optimization, clients perform local training on their datasets using stochastic gradient descent (SGD) to update the global model parameters $\emptyset_i$, The optimization formula is as follows:

$$\emptyset_i^{t,s} = SGD\left(f_i\left(\emptyset_i^{t,s-1}, \emptyset_i^{t,s}\right), \emptyset_i^{t,s-1}, \delta\right) \tag{23}$$

To measure the difference between the global model parameters of the current and previous rounds, the kernel function module is used to compute the similarity between the parameters. By dynamically selecting the optimal kernel function, the kernel distance between the global model parameters of the current and previous rounds is calculated

$$D\left(\emptyset_i^{t,s}, \emptyset_i^{t,s-1}\right) = \sqrt{k\left(\emptyset_i^{t,s}, \emptyset_i^{t,s}\right) + k\left(\emptyset_i^{t,s-1}, \emptyset_i^{t,s-1}\right) - 2k\left(\emptyset_i^{t,s}, \emptyset_i^{t,s-1}\right)} \tag{24}$$

Here, $k(\cdot)$ is the kernel function, which can adaptively select the kernel type based on the data characteristics. Combining the kernel distance D and an auxiliary adjustment function $Q_{base}(t)$, the smoothing coefficient α_i is dynamically calculated as:

$$\alpha_i = D\left(\emptyset_i^{t,s}, \emptyset_i^{t,s-1}\right) \cdot Q_{base}(t) \tag{25}$$

where the auxiliary adjustment function is defined as:

$$Q_{base}(t) = 1 - \frac{t}{T_{base}} \tag{26}$$

Here, $Q_{base}(t)$ represents the total number of training rounds, and \$s\$ is the current round. As training progresses, the auxiliary adjustment function gradually decreases, allowing the smoothing coefficient to converge and ensuring stability during synchronized optimization.

After completing local optimization, the server performs a smoothing fusion of the personalized model parameters uploaded by the clients with the global model parameters from the current round to achieve dynamic updates of the global model parameters:

$$\emptyset_i^{t,s} = \emptyset_i^{t,s-1} + (1 - \alpha_i)\emptyset_i^{t,s} \tag{27}$$

Through this mechanism, global model training not only preserves the personalized updates uploaded by the clients but also dynamically adjusts and smooths the contributions using kernel functions. This balances global sharing with local adaptation, achieving cross-client model generalization and efficient optimization. It establishes a solid foundation for further optimization and stable convergence of personalized models.

Model Fusion Strategy: After optimizing the personalized and global models, we introduce a dynamic model fusion mechanism to balance their contributions. During the fusion process, the kernel distance d between the personalized head parameters $h_i^{t,s}$ and the global model parameters $\emptyset_i^{t,s}$ is computed using a kernel function. The fusion weight ω_i is dynamically adjusted based on the changes in the loss, following the weight adjustment strategy:

$$\omega_i = \begin{cases} 0.3, if\ the\ loss\ decreases\ significantly \\ 0.7, if\ the\ loss\ increases\ significantly \\ 0.5, if\ the\ loss\ remains\ stable \end{cases} \tag{28}$$

Incorporating the kernel distance and an exponential decay mechanism, the final weight ω_i is further dynamically adjusted as:

$$\omega_i = \omega_i \cdot \exp(-d)\,, \omega_i \in [0.05, 0.95] \tag{29}$$

Based on the dynamic weight ω_i, the fused model parameters are updated using a weighted average:

$$\emptyset_i^{fused} = \omega_i h_i^{t,s} + (1 - \omega_i)\emptyset_i^{t,s} \tag{30}$$

This mechanism achieves a dynamic balance between globally shared information and personalized adaptability. It effectively combines the generalization performance of the global model with the adaptability of the personalized head model, thereby enhancing model performance in non-IID data scenarios.

4.2 Server update

In each training round, the server receives the fused model parameters $\emptyset_i^{fused}$ uploaded by the clients and updates the global model parameters $\emptyset$ through a weighted average based on the local sample sizes of the clients. The specific update formula is:

$$\emptyset^{t+1} = \frac{1}{r}\sum_{i=1}^{r} \omega_i \, \emptyset_i^{fused} \tag{31}$$

Where $\emptyset_i^{fused}$ represents the fused parameters uploaded by client i, it is the number of participating clients in the training ground, and ω_i is the weight for client i, typically determined by the proportion of local sample size.

Using this weighted averaging strategy, the server effectively aggregates the local updates from clients to generate new global model parameters $\emptyset^{t+1}$. The server then distributes the updated global model parameters to all clients as the initialization for the next training round, continuing until the global model converges.

This process ensures stable updates of the global model under clients' diverse data distributions while maintaining the generalization ability of the globally shared knowledge. The detailed implementation steps are presented in Algorithm 1.

Algorithm 1 The training process of AKPFL.

Require: Number of rounds T, number of clients N, local epochs τ_g, head epochs τ_h (personalized), learning rates δ_g and δ_h, initial global model $\emptyset^0$, initial personalized model h^0, smoothing coefficient α and μ, fusion weight ω

Ensure: Final global model $\emptyset^{t+1}$ and personalized head model h^t

For each round $\boldsymbol{t} = \mathbf{0}, \dots, -\mathbf{1}$ **do**

Server sends global model $\emptyset^t$ and personalized model h^t to selected clients S_t

For each client $\boldsymbol{i} \in S_t$ **in parallel do**

Initialize global model $\emptyset_i^{t,0} \leftarrow \emptyset^t$, personalized model $h_i^{t,0} \leftarrow h^t$

Local Update: Personalized Head

For each head epoch $\boldsymbol{s} = \mathbf{1}, \dots, \tau_h$ do

$h_i^{t,s} \leftarrow h_i^{t,s-1} = \delta_h \nabla_h L_i(h_i^{t,s-1})$

$\mu_i = D\left(h_i^{t,s}, h_i^{t,s-1}\right) \cdot Q_{head}(t)$

$h_i^{t,s} = h_i^{t,s-1} + (1-\mu_i) h_i^{t,s}$

End for

Local Update: Global Model

For each global epoch $\boldsymbol{s} = \mathbf{1}, \dots, \tau_g$ do

$\emptyset_i^{t,s} \leftarrow \emptyset_i^{t,s-1} = \delta_g \nabla_g L_i(\emptyset_i^{t,s-1})$

$\alpha_i = D\left(\emptyset_i^{t,s}, \emptyset_i^{t,s-1}\right) \cdot Q_{base}(t)$

$\emptyset_i^{t,s} = \emptyset_i^{t,s-1} + (1-\alpha_i) \emptyset_i^{t,s}$

End for

Fusion Step: Combine Global and Personalized Models

$\emptyset_i^{fused} = \omega_i h_i^{t,\tau_h} + (1-\omega_i) \emptyset_i^{t,\tau_g}$

Send fused parameters $\emptyset_i^{fused}$ to the server

End for

Server Aggregation:

$\emptyset^{t+1} = \frac{1}{|S_t|} \sum_{i=1}^{r} \omega_i \, \emptyset_i^{fused}$

End for

Output: Global model $\emptyset^{t+1}$

5 Experiment

To demonstrate the efficacy of AKPFL, two aspects are empirically investigated:

1. **Does AKPFL work?** We compare our algorithm with classic baselines on public datasets under varying levels of data heterogeneity.
2. **How does AKPFL work?** We conduct ablation experiments to compare how different kernel functions contribute to the FL process.

5.1 Set up

Dataset: The datasets used in this study include FashionMNIST [21], CIFAR-10, and CIFAR-100 [22]. These datasets are commonly used as benchmarks commonly used to to evaluate the performance of federated learning in image classification tasks.

Baselines: Our baselines include two classic federated models: classical federated learning and personalized federated learning. 1) Classical federated learning includes FedAvg [23] and FedProx [13]; 2) Personalized federated learning includes FedBABU [24], MOON [10], FedBN [25], FedRod [26], and FedRep [27].

Implementation: Our baseline reproduction is based on the PFLIB framework [28]. All experiments are conducted on a high-performance computational setup with the following specifications: NVIDIA RTX 4090 GPU (25.2 GB memory), AMD EPYC 9354 CPU (16 cores), 60.1 GB RAM, and 751.6 GB storage.

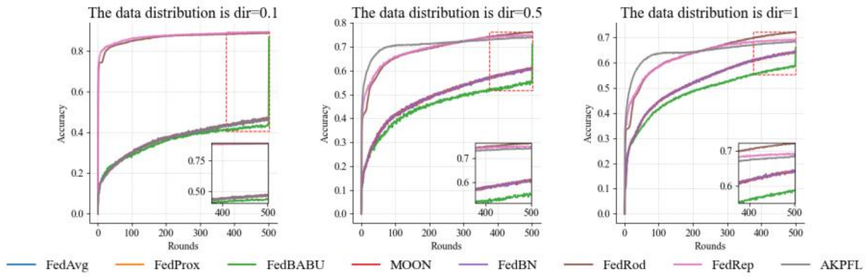

Fig. 2. Experimental comparison chart of different α heterogeneities on the Cifar100 dataset.

Data Heterogeneity Settings: We employ two dataset partitioning methods: Dirichlet partitioning and pathological label partitioning. In Dirichlet partitioning method, denoted as $Dir(\alpha)$, the parameter α is set to 0.1, 0.5, and 1, while in pathological label partitioning method, denoted as $Pat(\beta)$, the parameter β is assigned values of 1, 2, and 3.

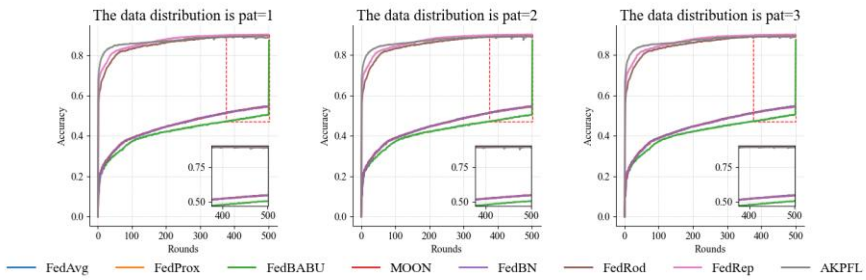

Fig. 3. E Experimental comparison chart of different β heterogeneities on the Cifar10 dataset.

5.2 Comparative experiment

The experimental results of AKPFL under Dirichlet data heterogeneity are shown in Table 1, while the results under pathological data heterogeneity are presented in Table 2. Our method demonstrates superior performance compared to classic baseline models across most settings. The comparative experiments conducted on the CIFAR-10 dataset are summarized in Fig. 2 and Fig. 3.

Table 2. Test Accuracy of Different Algorithms on Fashion MNIST, CIFAR-10, and CIFAR-100 Datasets under Dirichlet Distribution Parameters $\alpha = 0.1,0.5,1$.

Algorithm	Fashion MNIST			Cifar-10			Cifar-100		
	Dir 0.1	Dir 0.5	Dir 1	Dir 0,1	Dir 0.5	Dir 1	Dir 0.1	Dir 0.5	Dir 1
FedAvg	74.91	83.00	82.87	36.92	48.58	52.19	18.83	21.06	21.24
FedProx	74.89	82.98	82.86	36.91	48.62	52.18	18.83	21.05	21.24
FedBabu	72.54	81.07	80.94	35.69	44.90	48.20	16.31	17.96	17.90
MOON	74.85	83.00	82.86	36.95	48.16	52.18	18.82	21.06	21.26
FedBN	74.87	83.00	82.86	36.91	48.60	52.18	18.87	21.07	21.24
FedRod	95.85	88.72	87.68	86.39	68.58	63.21	41.76	**28.26**	23.88
FedRep	**96.01**	88.53	87.32	**87.16**	68.92	62.14	**43.08**	24.32	18.58
AKPFL	95.26	**90.02**	**89.48**	84.77	**70.71**	**64.14**	38.59	27.47	**25.86**

Table 2. Test Accuracy of Different Algorithms on FashionMNIST, CIFAR-10, and CIFAR-100 Datasets under Pathological Distribution Parameters $\beta = 1,2,3$.

Algorithm	Fashion MNIST			Cifar-10			Cifar-100		
	Pat1	Pat 2	Pat 3	Pat 1	Pat 2	Pat 3	Pat 1	Pat 2	Pat 3
FedAvg	73.19	73.21	73.22	44.77	44.78	44.78	18.30	18.30	18.35
FedProx	73.21	72.22	72.22	44.75	44.76	44.76	18.32	18.32	18.32
FedBabu	71.14	71.80	70.95	41.83	41.85	41.89	16.70	16.70	16.64
MOON	73.26	73.22	73.19	44.79	44.80	44.77	18.29	18.29	18.32
FedBN	73.20	73.21	73.21	44.75	44.77	44.76	18.32	18.32	18.31
FedRod	98.70	98.72	98.63	85.24	85.24	85.23	52.72	52.70	52.68
FedRep	98.71	98.70	98.61	86.72	86.70	86.68	**57.65**	**57.64**	**57.72**
AKPFL	**98.75**	**98.72**	**98.64**	**86.86**	**86.85**	**64.89**	49.20	49.91	49.69

5.3 Ablation experiment

Table 3. Performance of Different Kernels and Meta-Learning Methods in Ablation experiment.

Algorithm	Fashion MNIST			Cifar-10			Cifar-100		
	Dir0.5	Dir1	Pat3	Dir0.5	Dir1	Pat3	Dir0.5	Dir1	Pat3
Linear	98.94	89.58	98.64	70.52	63.94	86.86	27.39	25.83	49.84
RBF	90.03	89.57	98.66	70.34	64.12	86.86	27.33	25.88	49.04
Poly	89.68	89.05	98.36	69.82	63.57	86.48	26.86	25.52	49.39
Lap	89.56	88.94	97.66	70.45	63.43	86.34	26.84	25.21	48.48
Matern	90.07	89.75	98.72	70.62	64.25	86.92	27.30	25.83	49.80
MAML	89.06	89.31	75.70	60.84	63.73	53.81	27.72	27.47	21.54

The ablation experiments validate the significant contributions of the adaptive kernel distance formula and the meta-learning module to the framework's performance. The adaptive kernel selection enhances the model's generalization capability for heterogeneous data, while the meta-learning module accelerates model training and improves local data adaptability. The complete framework leverages the synergy of these two components, demonstrating superior performance across various tasks and distribution settings. The ablation experiment results are shown in Table 3.

6 Conclusion

This study investigates methods to mitigate statistical heterogeneity in federated learning. Our primary focus is on an adaptive kernel alignment-based federated learning approach to address model discrepancies caused by heterogeneity across clients. The proposed AKPFL algorithm demonstrates that selecting different kernel functions for feature alignment during model aggregation significantly reduces. Future work will explore the application of the client kernel alignment approach to more complex model heterogeneity and large-scale federated learning models, further examining whether kernel alignment can effectively mitigate heterogeneity across various dimensions.

Acknowledgment. This work was supported by the National Natural Science Foundation of China under Grant 62271234, the Fundamental Research Funds for Heilongjiang Universities under Grant 2022-KYYWF-1042, Open Foundation of State Key Laboratory of Public Big Data (Guizhou University) under Grant No. PBD2022-16, Double First-Class Project for Collaborative Innovation Achievements in Disciplines Construction in Heilongjiang Province under Grant No. LJGXCG2022-054 and LJGXCG2023-028, and Advanced Programs of Heilongjiang Province for the overseas scholars.

References

1. C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," Knowledge-Based Systems, vol. 216, p. 106775, 2021
2. Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach," IEEE Internet of Things Journal, vol. 8, no. 8, pp. 6348-6358, 2021
3. K. Zhang, J. Sun, Q. Feng, X. Huang, C. Jiang, and M. S. Hossain, "Joint optimization of fairness and energy efficiency in zero-trust federated learning for consumer internet of things: A lossy communication perspective," IEEE Transactions on Consumer Electronics, pp. 1-1, 2024
4. Y. Zhu, Y. Ye, Y. Liu, and J. J. Q. Yu, "Cross-area travel time uncertainty estimation from trajectory data: A federated learning approach," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 12, pp. 24966-24978, 2022
5. P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings et al., "Advances and open problems in federated learning," Foundations and Trends® in Machine Learning, vol. 14, no. 1-2, pp. 1-210, 2021
6. V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," Advances in Neural Information Processing Systems, vol. 30, 2017
7. Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," arXiv preprint arXiv:1806.00582, 2018
8. Ahmad, W. Luo, and A. Robles-Kelly, "Robust federated learning under statistical heterogeneity via hessian-weighted aggregation," Machine Learning, vol. 112, no. 2, pp. 633-654, 2023
9. S. M. Azimi-Abarghouyi and V. Fodor, "Quantized hierarchical federated learning: A robust approach to statistical heterogeneity," arXiv preprint arXiv:2403.01540, 2024
10. Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 10713-10722
11. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," IEEE Transactions on Neural Networks and Learning Systems, vol. 34, no. 12, pp. 9587-9603, 2022
12. X. Weiqin, X. Ting, and W. Zhe, "Balance information and dynamic update in prototype-based federated learning," Journal of East China University of Science and Technology, vol. 50, no. 6, pp. 905-912, 2024
13. T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50-60, 2020
14. S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečny, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," arXiv preprint arXiv:2003.00295, 2020
15. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," Advances in Neural Information Processing Systems, vol. 33, pp. 19,586-19,597, 2020
16. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," arXiv preprint arXiv:2002.07948, 2020
17. T. Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with Moreau envelopes," Advances in Neural Information Processing Systems, vol. 33, pp. 21,394-21,405, 2020
18. F. Hanzely and P. Richtérik, "Federated learning of a mixture of global and local models," arXiv preprint arXiv:2002.05516, 2020

19. M. Yarochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in International Conference on Machine Learning, PMLR, 2019, pp. 7,252-7,261
20. M. Tang, X. Ning, Y. Wang, J. Sun, Y. Wang, H. Li, and Y. Chen, "Fedcor: Correlation-based active client selection strategy for heterogeneous federated learning," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10,102-10,111
21. H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," arXiv preprint arXiv:1708.07747, 2017
22. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Handbook of Systemic Autoimmune Diseases, vol. 1, no. 4, 2009
23. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in Artificial Intelligence and Statistics, PMLR, 2017, pp. 1273-1282
24. J. Oh, S. Kim, and S.-Y. Yun, "Fedbabu: Towards enhanced representation for federated image classification," arXiv preprint arXiv:2106.06042, 2021
25. X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," arXiv preprint arXiv:2102.07623, 2021
26. H.-Y. Chen and W.-L. Chao, "On bridging generic and personalized federated learning for image classification," arXiv preprint arXiv:2107.00778, 2021
27. L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in International Conference on Machine Learning, PMLR, 2021, pp. 2089-2099
28. J. Zhang, Y. Liu, Y. Hua, H. Wang, T. Song, Z. Xue, R. Ma, and J. Cao, "Pfllib: Personalized federated learning algorithm library," arXiv preprint arXiv:2312.04992, 2023
29. Krizhevsky, G. Hinton et al., "Learning multiple layers of features from tiny images," 2009