Revisiting Hypernetwork in Model Heterogeneous Personalized Federated Learning

Chen Zhang[1] , **Husheng Li**[1] , **Xiang Liu**[2*] , **Linshan Jiang**[3] and **Danxin Wang**[1]
[1]Shandong Key Laboratory of Intelligent Oil & Gas Industrial Software, Qingdao Institute of Software, College of Computer Science and Technology, China University of Petroleum (East China), China
[2]National University of Singapore
[3]Southern University of Science and Technology
{zhangchen, wangdx}@upc.edu.cn, z23070167@s.upc.edu.cn,
liu.xiang@u.nus.edu, linshan001@e.ntu.edu.sg

Abstract

Recent personalized federated learning research focuses on heterogeneous models across clients. However, existing methods often rely on external data, model decoupling, and partial learning, which makes them sensitive to settings. In contrast, we revisit hypernetworks and leverage their strong generalization ability to propose the first practical method for personalized federated learning. We first propose a **m**odel **h**eterogeneous **p**ersonalized **fed**erated learning framework based on **h**yper**n**etworks, **MH-pFedHN**, which quantifies clients with different architectures using customized embedding vectors and then generates client-specific model parameters through a server-side hypernetwork. Besides the shared feature extractor, our hypernetwork consists of multiple heads, where clients with similar numbers of parameters are assigned the same number of customized embedding vectors and consequently share the same head. This design enables knowledge sharing across different architectures and reduces the computation of parameter generation. To further enhance the hypernetwork's learning and generalization, we propose **MH-pFedHNGD**, which introduces a lightweight yet effective `plug-in` global model. Our framework requires no external data and does not disclose client model architectures, thereby effectively ensuring security and demonstrating great potential. Experiments across various models and tasks demonstrate that our approach outperforms standard baselines and exhibits strong generalization performance. Our code is available at https://github.com/DangDang1895/MH-pFL.

1 Introduction

Federated learning (FL) has been widely applied in various fields, such as intelligent transportation [Wang *et al.*, 2022; Zhou *et al.*, 2024], healthcare [Nguyen *et al.*, 2022; Murmu *et al.*, 2024], and recommendation systems [Yuan *et al.*, 2023; Feng *et al.*, 2024]. However, a single global model cannot meet all clients' needs due to non-IID data. To address this, personalized federated learning (pFL) [Dinh *et al.*, 2020; Deng *et al.*, 2020] emerges, aiming to craft personalized models for clients while enabling knowledge sharing under cross-device settings [Meng *et al.*, 2024; Qi *et al.*, 2025] , thus better matching their specific tasks and data distributions.

In practice, devices participating in pFL are often heterogeneous, as they have different computational resources [Chai *et al.*, 2020; Shin *et al.*, 2024], communication capabilities [He *et al.*, 2020; Shah and Lau, 2021], and model architectures [Li and Wang, 2019; Zhu *et al.*, 2021]. In particular, in resource-constrained environments, it is difficult to enforce a unified model architecture across all clients [Chan *et al.*, 2024]. Meanwhile, it is also necessary to prevent the privacy leakage of model architectures [Zhang and Liu, 2024]. These factors complicate the challenges that pFL faces in scenarios of model heterogeneity [Chen *et al.*, 2023]. To address the limitations of the model heterogeneous pFL (MH-pFL), several methods have been proposed by researchers, including partial training [Diao *et al.*, 2021; Shen *et al.*, 2024], federated distillation [Wang *et al.*, 2024; Liu *et al.*, 2025], and model decoupling [Liang *et al.*, 2020; Collins *et al.*, 2021; Xu *et al.*, 2023].

However, modeling decoupling [Jang *et al.*, 2023; Yi *et al.*, 2023a; Yi *et al.*, 2023b] separates local models into feature extractors and classifiers; this low-level knowledge sharing may hinder client collaboration and negatively impact performance. Partial training methods [Horváth and others, 2021; Lee *et al.*, 2024] allow clients to select sub-models for local training. However, differences in model architectures, data distributions, and resource conditions can lead to misalignment in parameter counts and feature spaces, causing suboptimal performance. Current federated distillation approaches [Gong *et al.*, 2021; Itahara *et al.*, 2021; Wang *et al.*, 2025] depend on a universal or synthetic dataset for knowledge integration. Such additional datasets may limit the applicability of the method and make the performance dependent on the quality of these datasets. Hence, an MH-pFL approach is needed to resolve these issues while ensuring pri-

*Corresponding authors.

vacy and computational efficiency effectively.

In this work, we revisit hypernetwork [Ha *et al.*, 2017], a model that generates personalized parameters for another neural network, and propose a novel MH-pFL framework, the ***M**odel-**H**eterogeneous **P**ersonalized **Fed**erated **H**yper**N**etwork* (`MH-pFedHN`), which uses a hypernetwork as a knowledge aggregator to enable knowledge fusion among heterogeneous clients. In our approach, MH-pFedHN customizes a set of embedding vectors for each client based on the number of parameters required by the client's model. Therefore, clients with a similar number of parameters will have an equal number of customized embedding vectors. In addition to the feature extractor from traditional hypernetwork shared across all clients to support generalizable feature learning, a shared head is created for clients that have the same number of embedding vectors, thereby increasing efficiency and promoting knowledge sharing among them. This fine-grained mapping mechanism improves the server's expressive and generalization abilities.

To further learn cross-client knowledge and enhance performance, we propose ***MH-pFedHN** with **G**lobal **D**istillation* (`MH-pFedHNGD`), with an additional lightweight but effective global model. This plug-in component on the server side is directly generated by our hypernetwork using the global customized embedding vectors. Compared to MH-pFedHN, MH-pFedHNGD leverages a global model to aggregate updates from clients, introducing one more round of lightweight update mechanisms, enabling the hypernetwork to learn a more comprehensive data distribution and more generalizable features. Meanwhile, the global model can serve as a teacher model [Hinton, 2015; Jeong *et al.*, 2018] to assist training via knowledge distillation on the client side, thus improving knowledge acquisition abilities of client-specific models and balancing personalization with generalization.

Both of our methods are data-free and preserve the structural privacy of personalized models. They are the first efficient frameworks designed to leverage hypernetworks for solving MH-pFL problems, and hold great promise for future applications. Our main contributions are as follows.

- We propose MH-pFedHN, a personalized FL method based on hypernetworks and specifically designed for heterogeneous models. This method allows the server to utilize customized embedding vectors and the shared head tailored to clients' needs to generate parameters without disclosing the model architecture to the server, thereby enhancing privacy protection.
- We propose MH-pFedHNGD based on MH-PFedHN, which further integrates a plug-in component lightweight global model. This approach enhances the hypernetwork's learning and generalization capabilities and allows personalized client models to learn from a global model with knowledge distillation efficiently, resulting in significant performance improvements.
- We evaluate our MH-pFedHN and MH-pFedHNGD for multiple tasks on four popular datasets. The experiments demonstrate that our method exceeds state-of-the-art performance against baselines across all tasks, highlighting the effectiveness and generalizability.

2 Related Work

2.1 Personalized Federated Learning with Hypernetworks

Hypernetworks are methods that use one network to generate weights for other neural networks [Ha *et al.*, 2017; Zeng *et al.*,], which are widely used in various applications [Suarez, 2017; Nirkin *et al.*, 2021; Beck *et al.*, 2024; Ruiz *et al.*, 2024]. In pFL, hypernetworks are used to generate personalized model parameters from clients' embedding vectors [Shamsian *et al.*, 2021; Zhu *et al.*, 2023; Scott *et al.*, 2024] and output the weight ratio during aggregation [Ma *et al.*, 2022]. This method has shown effectiveness in systems with diverse data and few-shot learning scenarios [Sendera *et al.*, 2023].

However, these methods are mainly designed to alleviate the problem of statistical heterogeneity, so that clients can obtain personalized models. The work [Shamsian *et al.*, 2021] proposes pFedHN and only explores generating limited heterogeneous models. FLHA-GHN [Litany *et al.*, 2022] uses graph hypernetworks to generate model parameters for different architectures. Training the hypernetwork on local clients adds computational overhead. They cannot be deployed in resource-constrained environments; in contrast, our efficient methods, MH-pFedHN and MH-pFedHNGD, demonstrate strong potential for deployment in practical scenarios.

2.2 Model-Heterogeneous Personalized Federated Learning Methods

Due to resource constraints [Chai *et al.*, 2020; Shin *et al.*, 2024], communication limitations [He *et al.*, 2020; Shah and Lau, 2021], and personalized requirements for models [Li and Wang, 2019; Zhu *et al.*, 2021], MH-pFL has become an important research direction. Even when clients share the same model architecture, two forms of model heterogeneity can still arise: 1) Due to resource differences, clients may need to employ different sub-models for training [Diao *et al.*, 2021]; 2) Clients with varying personalization requirements retain parameters of specific layers solely for local updates [Li *et al.*, 2021], resulting in model heterogeneity. Furthermore, the model architecture may impact model privacy [Zhang *et al.*, 2024], clients are often reluctant to further disclose the details of their model design information.

Partial Training is adopted in some methods, where each client selects a sub-model that originates with the global model. The parameters from the client sub-models are aggregated by averaging matched parameters. HeteroFL [Diao *et al.*, 2021] divides the global model into different sub-models, allowing clients to train appropriate models based on computational capabilities, enabling low-resource clients to contribute to the global model without being excluded from the aggregation. DepthFL [Kim *et al.*, 2023] utilizes different depths of local clients and prunes the deepest layers off the global model to help allocate the server model to all the clients based on computational resources. pFedGate [Chen *et al.*, 2023] explores to learn a sparse local model adaptively. HypeMeFed [Shin *et al.*, 2024] combines a multi-path network architecture with weight generation to support client

heterogeneity. However, due to differences in model architecture, parameters of different sub-models learning by partial training may not align, resulting in suboptimal performance.

Federated Distillation, similar to compression [Li and Ma, 2024; Zeng *et al.*, 2025], utilizes knowledge distillation [Hinton, 2015] to mitigate the impact of data heterogeneity and address model heterogeneity when facilitating collaboration between clients with different architectures by transferring knowledge from complex models to simpler ones in FL [Sattler *et al.*, 2020; Wu *et al.*, 2021].

FedMD [Li and Wang, 2019] and DS-FL [Itahara *et al.*, 2021] use a pre-existing public dataset for knowledge extraction, aggregating local soft predictions on the server. KT-pFL [Zhang *et al.*, 2021] trains personalized soft prediction weights on the server to improve heterogeneous models' performance further. Inspired by data-free knowledge distillation [Zhang *et al.*, 2022; Dai *et al.*, 2024], some methods [Zhu *et al.*, 2021; Zhang *et al.*, 2023] use GAN [Goodfellow *et al.*, 2020] to generate synthetic data on the server side. FedGD [Zhang *et al.*, 2023] and DFRD [Wang *et al.*, 2024] design a distributed GAN between the server and the clients to enable data-free knowledge transfer and effectively tackle model heterogeneity. However, training the key generator in GAN-based federated distillation poses significant challenges that can impact overall system performance. FedAKT [Liu *et al.*, 2025] combines knowledge distillation and model decoupling; however, homogeneous adapters increase computational and communication overhead for clients, and distillation and dual-head mechanisms can perform poorly in highly heterogeneous environments. Therefore, federated distillation still faces the challenges related to privacy risks and the quality of public and synthetic datasets.

Model Decoupling is another attempt, which divides client models into feature extractors and classifier heads. Further, scientists extend the previous model decoupling methods to share part of the global model while retaining the other part in a heterogeneous form locally. FedClassAvg [Jang *et al.*, 2023] aggregates classifier weights to establish a consensus on decision boundaries in the feature space, enabling clients with non-IID data to learn from scarce labels; FedGH [Yi *et al.*, 2023a] trains a shared, generalized global prediction head using representations extracted by clients' heterogeneous feature extractors on the FL server. In pFedES [Yi *et al.*, 2023b], clients are trained via their proposed iterative learning method to facilitate the exchange of global knowledge; small local homogeneous extractors are then uploaded for aggregation to help the server learn the knowledge across clients. However, these basic mechanisms might find it difficult to achieve effective knowledge fusion in complex heterogeneous scenarios, which results in limitations in final performance.

3 Our Framework

In this section, we first formalize the MH-pFL problem. Then, we introduce MH-pFedHN, which generates parameters for different models through customized embedding vectors and utilizes a hypernetwork for knowledge fusion across heterogeneous clients. Finally, we present MH-pFedHNGD, which utilizes a lightweight global model to enhance the learning and generalization capabilities of the hypernetwork. The lightweight plug-in component also assists personalized model training for clients via knowledge distillation, further improving model training across heterogeneous clients. The overviews of our two methods are shown in Figures 1 and 2, where one round of MH-pFedHN consists of 5 steps and one round of MH-pFedHNGD is made up of 10 steps.

3.1 Problem Formulation

Our goal is to develop an MH-pFL approach that, without requiring knowledge of model architectures, meets each client's requirements to address the challenges brought about by model and data heterogeneity. This can be turned into the following minimization problem, which aims to capture the personalized objectives of each client while accommodating heterogeneous data distributions and model architectures

$$\{\boldsymbol{\theta}_1^*, \ldots, \boldsymbol{\theta}_n^*\} = \underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n}{\arg\min} \sum_{i=1}^{n} \mathbb{E}_{x,y\sim P_i}\left[\ell_i(x, y; \boldsymbol{\theta}_i)\right], \quad (1)$$

where P_i represents the local data distribution of the i-th client, and $\ell_i(x_j, y_j; \boldsymbol{\theta}_i)$ denotes the loss function for the i-th client's model with parameters $\boldsymbol{\theta}_i$. In particular, $\boldsymbol{\theta}_i$ is client-specific and can correspond to models of varying architecture.

To further formalize the optimization problem, the training objective is expressed as

$$\underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n}{\arg\min} \sum_{i=1}^{n} L_i(\boldsymbol{\theta}_i) = \underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n}{\arg\min} \sum_{i=1}^{n} \frac{1}{m_i} \sum_{j=1}^{m_i} \ell_i(x_j, y_j; \boldsymbol{\theta}_i), \quad (2)$$

where m_i denotes the number of data samples in the local dataset of the i-th client, and $L_i(\boldsymbol{\theta}_i)$ represents the empirical loss over its dataset. By allowing each client to optimize its model, ensuring that the learned parameters $\boldsymbol{\theta}_i$ are suited to the client's unique data distribution and task.

3.2 MH-pFedHN

Here, we describe MH-pFedHN to the MH-pFL problem (Equation 2), and the complete algorithm for MH-pFedHN is provided in Algorithm 1. Let $h(\cdot; \boldsymbol{\varphi})$ denote the hypernetwork h with parameters $\boldsymbol{\varphi}$, where $\boldsymbol{\varphi}$ is composed by a feature extractor $\boldsymbol{\varphi}_f$ and multiple heads $\{\boldsymbol{\varphi}_{H_l}\}_{l=1}^{L}$, where l indexes the heads and L denotes the number of heads.

In the initialization stage, client i needs to upload the number of parameters K_i required for its personalized model. The server then dynamically determines the hypernetwork output dimension N (i.e., the length of each generated parameter block) based on $\{K_i\}_{i=1}^{n}$, and computes the number of customized embedding vectors for client i as $\tau_i = \lceil K_i/N \rceil$. In practice, we choose N such that τ_i is greater than the number of layers in the client model to enable finer-grained parameter generation. Therefore, clients with similar numbers of model parameters (i.e., $\lceil K_i/N \rceil = \lceil K_j/N \rceil$) will have the same number of customized embedding vectors (i.e., $\tau_i = \tau_j$) and thus share the same head parameters $\boldsymbol{\varphi}_{H_l}$, which enhance knowledge sharing and reduce computational overhead. In contrast, clients with different numbers of customized embedding vectors (i.e., $\tau_i \neq \tau_j$) use separate heads to mitigate the impact on personalized task accuracy.

Specifically, for example, if two client models have the same number of parameters and are both determined to have

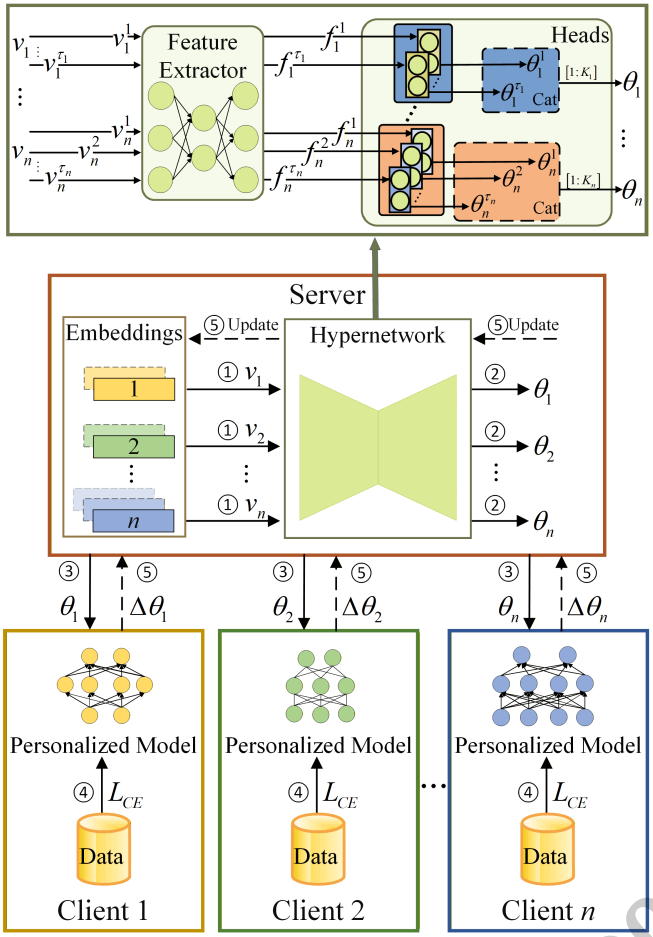

Figure 1: Framework of MH-pFedHN. The workflow contains 5 steps: ① input the embedding matrix $\boldsymbol{v}_i$ of client i into the hypernetwork; ② generates the parameters $\boldsymbol{\theta}_i$ of client i; ③ client i receives the parameters $\boldsymbol{\theta}_i$ from the server; ④ client i trains the personalized model $\boldsymbol{\theta}_i$ using the private data; ⑤ client i uploads the update of parameters $\Delta\boldsymbol{\theta}_i$ to the server; server updates customized embedding vectors and hypernetworks.

three customized embedding vectors, they will share the same head $\boldsymbol{\varphi}_{H_l}$. This head has three output channels (the channel number equals the number of customized embedding vectors), each tailored for a specific customized embedding vector input. The shared feature extractor $\boldsymbol{\varphi}_f$ for all clients and this head $\boldsymbol{\varphi}_{H_l}$ will generate three subsets of parameters with length N for each model. These three subsets of parameters, when combined and truncated according to the specific parameter size of each client, constitute the personalized parameters for each client. *Generally*, for the j-th customized embedding vector of client i (associated with the l-th head), the hypernetwork outputs $\boldsymbol{\theta}_i^j = h(\boldsymbol{v}_i^j; \boldsymbol{\varphi}_f, \boldsymbol{\varphi}_{H_l})$. We then obtain the personalized model parameters for client i by concatenating all blocks and truncating to match K_i:

$$\boldsymbol{\theta}_i := \text{concat}(\boldsymbol{\theta}_i^1, \boldsymbol{\theta}_i^2, \cdots, \boldsymbol{\theta}_i^{\tau_i})_{[1:K_i]}. \tag{3}$$

For notational convenience, let $\boldsymbol{v}_i = [\boldsymbol{v}_i^1, \ldots, \boldsymbol{v}_i^{\tau_i}]$ denote the customized embedding vectors of client i, and we write $\boldsymbol{\theta}_i = \boldsymbol{\theta}_i(\boldsymbol{\varphi}) := h(\boldsymbol{v}_i; \boldsymbol{\varphi})_{[1:K_i]}$.

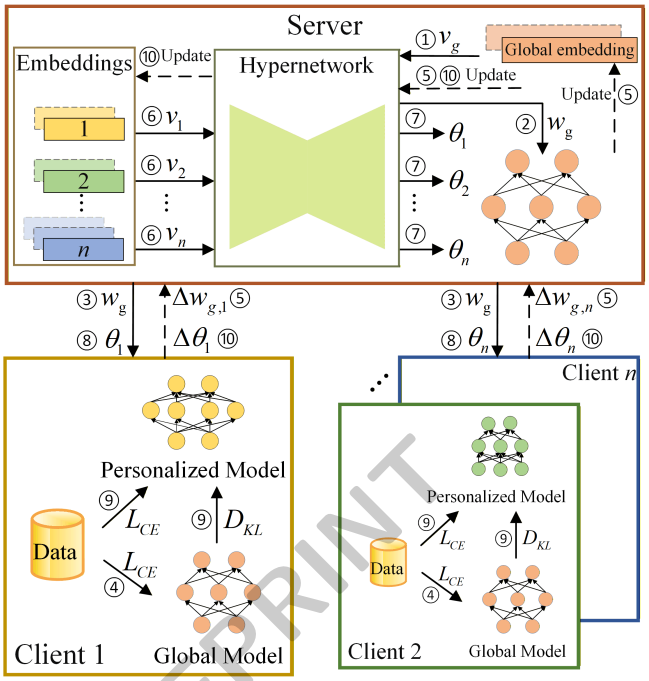

Figure 2: Framework of MH-pFedHNGD. The workflow contains 10 steps: ① input the embedding matrix $\boldsymbol{v}_g$ of the global model into the hypernetwork; ② generates parameters $\boldsymbol{w}_g$ of the global model; ③ each client receives the global model parameters $\boldsymbol{w}_g$ from the server; ④ each client trains the global model using private data; ⑤ The server receives $\Delta\boldsymbol{w}_{g,i}$ from each client and updates the hypernetwork and global embedding vector parameters; ⑥ input the embedding matrix $\boldsymbol{v}_i$ of client i into the hypernetwork; ⑦ generates parameters $\boldsymbol{\theta}_i$ of client i; ⑧ client i receives parameters $\boldsymbol{\theta}_i$ from the server; ⑨ global model $\boldsymbol{w}_g$ assists in training personalized model $\boldsymbol{\theta}_i$ of client i on private data; ⑩ client i uploads update of parameters $\Delta\boldsymbol{\theta}_i$ to the server; server updates customized embedding vectors and hypernetworks.

Based on the aforementioned setup, we adopt the MH-pFL objective (Equation 2) and get our MH-pFedHN optimization function as[1]:

$$\begin{aligned}
&\min_{\boldsymbol{\varphi}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n} \sum_{i=1}^{n} L_i\big(h(\boldsymbol{v}_i; \boldsymbol{\varphi})_{[1:K_i]}\big) \\
&= \min_{\boldsymbol{\varphi}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n} \sum_{i=1}^{n} \frac{1}{m_i} \sum_{j=1}^{m_i} \ell_i\big(x_j, y_j; h(\boldsymbol{v}_i; \boldsymbol{\varphi})_{[1:K_i]}\big) \\
&= \min_{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n} \sum_{i=1}^{n} \frac{1}{m_i} \sum_{j=1}^{m_i} \ell_i(x_j, y_j; \boldsymbol{\theta}_i).
\end{aligned} \tag{4}$$

3.3 MH-pFedHNGD

To enhance the hypernetwork's learning capacity and generalization ability while enabling clients to efficiently extract knowledge representations from a global model, we propose MH-pFedHNGD, which introduces a lightweight global model based on MH-pFedHN. This design aims to improve overall performance at the cost of minimal system and

[1]The theoretical analysis is in Appendix B.

communication overhead. The complete algorithm for MH-pFedHNGD can be found in Algorithm 2.

Thus, the global model's number of parameters is set as $K_g = \min\{K_1, \cdots, K_n\}$, thus, the global model shares the same head with the client having the fewest number of parameters to make sure the global model is lightweight. Then the number of global customized embedding vectors is set as $\tau_g = \lceil K_g/N \rceil$, the global embedding vector $\boldsymbol{v}_g = [\boldsymbol{v}_g^1, \ldots, \boldsymbol{v}_g^{\tau_g}]$. Assuming the client with the fewest number of parameters corresponds to the l-th head, the global model parameters are generated by:

$$\begin{aligned} \boldsymbol{w}_g := h(\boldsymbol{v}_g; \boldsymbol{\varphi})_{[1:K_g]} = \text{concat}\Big(& h(\boldsymbol{v}_g^1; \boldsymbol{\varphi}_f, \boldsymbol{\varphi}_{H_l}), \\ & h(\boldsymbol{v}_g^2; \boldsymbol{\varphi}_f, \boldsymbol{\varphi}_{H_l}), \cdots, h(\boldsymbol{v}_g^{\tau_g}; \boldsymbol{\varphi}_f, \boldsymbol{\varphi}_{H_l}) \Big)_{[1:K_g]}. \end{aligned} \tag{5}$$

Then, each client receives the parameters $\boldsymbol{w}_{g,i} = \boldsymbol{w}_g$ and trains on private data, which is to optimize:

$$\arg\min_{\boldsymbol{\varphi}, \boldsymbol{v}_g} \sum_{i=1}^{n} \frac{m_i}{M} L_i \left(h(\boldsymbol{v}_g; \boldsymbol{\varphi})_{[1:K_g]} \right) = \arg\min_{w_i} \sum_{i=1}^{n} \frac{m_i}{M} L_i(\boldsymbol{w}_{g,i}), \tag{6}$$

$$L_i(\boldsymbol{w}_{g,i}) = \frac{1}{m_i} \sum_{j=1}^{m_i} \ell_i(x_j, y_j; \boldsymbol{w}_{g,i}), \tag{7}$$

where $M = \sum_i m_i$. After training its own global model is completed in Step ④ in Figure 2, clients upload $\Delta \boldsymbol{w}_{g,i} = \boldsymbol{w}_{g,i} - \boldsymbol{w}_g$ to the server to update the global customized embedding vectors and the hypernetwork, enhancing the learning and generalization capabilities of the hypernetwork.

Finally, in the distillation training phase, client i reloads the global model in Step ③ and receives the parameters $\boldsymbol{\theta}_i$ of the personalized model, utilizing the global model as a teacher model, directing the training of the personalized model for client i. We adopt the optimization problem (Equation 4) and get our MH-pFedHNGD optimization function as follows:

$$\begin{aligned} \arg\min_{\boldsymbol{\varphi}, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n} \sum_{i=1}^{n} \Bigg[& \lambda L_{KL} \left(h(\boldsymbol{v}_i; \boldsymbol{\varphi})_{[1:K_i]}, h(\boldsymbol{v}_g; \boldsymbol{\varphi})_{[1:K_g]} \right) \\ & + (1 - \lambda) L_i \left(h(\boldsymbol{v}_i; \boldsymbol{\varphi})_{[1:K_i]} \right) \Bigg]. \end{aligned} \tag{8}$$

where L_{KL} is the Kullback–Leibler divergence [Csiszár, 1975], and λ balances the distillation and cross-entropy losses. Therefore, the lightweight plug-in component further improves training across heterogeneous clients.

4 Experiments

4.1 Experiment Setup

Datasets. We evaluate the MH-pFedHN and MH-pFedHNGD framework over four datasets, EMNIST, CIFAR-10, CIFAR-100, and Tiny-ImageNet. We adopt two non-IID settings [Dinh *et al.*, 2020; Liu *et al.*, 2024].1) Quantity-based label imbalance (non-IID_1). For the EMNIST, CIFAR-100, and Tiny-ImageNet datasets, we randomly allocate 6, 10, and 20 classes to each client, respectively. We draw $\alpha_{i,c} \sim U(0.4, 0.6)$, and allocate $\frac{\alpha_{i,c}}{\sum_j \alpha_{j,c}}$ of the samples for the class c selected on client i. 2) Distribution-based label imbalance (non-IID_2). We employ the Dirichlet distribution $Dir(0.01)$ to partition the dataset among the clients. For each client, 75% of the data is used for training, and 25% is used for testing. As EMNIST dataset is simpler than others, we only report results for 200 clients.

Models. For fair comparison, we use the LeNet model [LeCun *et al.*, 1998] in the homogeneous setting. In the heterogeneous setting, we additionally use VGGNet [Simonyan and Zisserman, 2015] and three residual networks(ResNet-10,12,18) [He *et al.*, 2016].[2] Unless otherwise specified, all experiments under heterogeneous settings use these five models, which are evenly distributed among clients. The feature extractor of the hypernetwork is comprised of a three-layer fully connected network. Details are in Appendix G.

Baselines. We choose various state-of-the-art. **pFedHN** [Shamsian *et al.*, 2021] uses hypernetwork to produce personalized model directly; **pFedLA** [Ma *et al.*, 2022] computes aggregation weights for local models of each client; **FedGH** [Yi *et al.*, 2023a] uses a generalized global prediction header for diverse model structures; **pFedLHN** [Zhu *et al.*, 2023] leverages a layer-wise hypernetwork to achieve fine-grained personalization across different layers; **PeFLL** [Scott *et al.*, 2024] uses a learning-to-learn approach to generate models; **FedAKT** [Liu *et al.*, 2025] employs adapters and distillation to enhance knowledge transfer and model adaptation in heterogeneous environments. We also include **Local** Training without aggregation and **FedAvg** [McMahan *et al.*, 2017].

Training Strategies. We run all experiments for 500 communication rounds and report the mean and standard deviation over 3 independent runs based on random seeds. For MH-pFedHN, we set local epochs to 2, SGD optimizer with the learning rate $1e{-3}$ and weight decay $1e{-4}$ and momentum 0.9, batch size 64. The hypernetwork uses the Adam optimizer with a learning rate of $2e{-4}$, embedding dimension 64, and the output dimension 3072. For MH-pFedHNGD, we configure the global model LeNet. On CIFAR-100, Tiny-ImageNet, and EMNIST datasets, we set distillation temperatures as 15, 24, and 10, and distillation loss coefficients as 0.01, 0.2, and 0.1, respectively. More settings and design choices are in Appendix C and F.

4.2 An Overall Comparison

Table 1 summarizes results under two data partitions for both homogeneous and heterogeneous settings.

In the homogeneous setting, MH-pFedHN and MH-pFedHNGD consistently outperform all baseline approaches across datasets and client scales under both non-IID scenarios, which shows the advantages of enhancing the knowledge sharing among clients. Although the performance of most methods degrades as the number of clients increases, our methods maintain strong accuracy, demonstrating robustness to data heterogeneity and scalability to large client populations. Moreover, MH-pFedHNGD achieves consistently higher accuracy than MH-pFedHN, supporting the effectiveness of introducing a global model for hypernetwork updates

[2]Experiments with Vision Transformer (ViT) are in Appendix D.4, which shows that our method supports large models. Experiments with CIFAR-10 are in Appendix D.2.

Clients	CIFAR-100						Tiny-ImageNet						EMNIST	
	50		100		200		50		100		200		200	
Local	50.32 ± 1.81	72.79 ± 1.73	40.41 ± 1.68	73.52 ± 1.67	34.65 ± 1.05	73.56 ± 1.17	29.17 ± 2.41	50.65 ± 1.12	20.73 ± 0.91	55.14 ± 1.03	14.90 ± 0.63	55.81 ± 0.34	96.26 ± 0.13	98.70 ± 0.28
FedAvg	22.94 ± 0.82	22.59 ± 0.97	24.80 ± 1.48	24.27 ± 1.33	25.55 ± 0.83	23.07 ± 0.77	8.13 ± 0.43	6.64 ± 0.31	8.80 ± 0.85	7.54 ± 0.88	9.12 ± 0.92	8.22 ± 0.23	81.49 ± 0.24	80.18 ± 0.31
pFedHN	63.66 ± 0.52	76.76 ± 0.95	58.90 ± 0.62	79.74 ± 0.53	32.77 ± 0.45	74.14 ± 0.84	40.10 ± 0.61	56.76 ± 0.25	35.39 ± 0.45	58.93 ± 0.64	32.10 ± 0.84	61.12 ± 0.62	97.50 ± 0.17	99.18 ± 0.09
pFedLA	63.33 ± 0.33	72.86 ± 1.28	55.83 ± 0.57	75.22 ± 0.91	55.36 ± 0.46	75.88 ± 1.14	39.69 ± 0.88	48.30 ± 0.39	30.00 ± 1.71	53.59 ± 0.62	23.42 ± 0.97	54.84 ± 1.52	95.41 ± 0.21	98.40 ± 0.19
FedGH	61.01 ± 0.36	75.54 ± 0.84	53.61 ± 1.22	76.65 ± 0.59	38.70 ± 0.97	77.30 ± 0.43	31.67 ± 0.76	54.30 ± 1.67	23.64 ± 0.52	57.97 ± 0.90	17.80 ± 0.64	57.44 ± 0.31	96.17 ± 0.32	96.42 ± 0.22
pFedLHN	61.00 ± 0.47	77.03 ± 0.92	58.39 ± 0.63	79.06 ± 0.38	53.43 ± 0.79	79.49 ± 0.56	39.49 ± 0.34	55.71 ± 0.88	35.00 ± 0.51	60.62 ± 0.97	31.31 ± 0.69	58.30 ± 0.41	97.52 ± 0.05	**99.25 ± 0.09**
PeFLL	50.64 ± 0.62	66.79 ± 1.34	49.20 ± 0.47	71.95 ± 0.91	40.97 ± 0.78	73.07 ± 1.57	32.41 ± 0.39	51.44 ± 0.86	28.07 ± 1.12	55.86 ± 0.54	23.65 ± 0.96	56.26 ± 0.43	94.68 ± 0.37	98.88 ± 0.29
FedAKT	60.85 ± 0.49	74.28 ± 1.26	56.09 ± 0.78	77.92 ± 0.36	51.48 ± 0.92	78.56 ± 0.57	38.57 ± 0.64	55.23 ± 0.41	31.86 ± 0.88	60.23 ± 0.53	31.87 ± 1.68	61.58 ± 0.72	97.21 ± 0.19	98.96 ± 0.17
MH-pFedHN (ours)	**64.69 ± 0.73**	**77.93 ± 0.46**	**63.32 ± 1.09**	**80.93 ± 0.70**	**60.11 ± 0.84**	**81.60 ± 0.68**	**42.35 ± 1.08**	**58.30 ± 0.40**	**37.62 ± 0.81**	**62.68 ± 1.20**	**37.30 ± 0.68**	**63.61 ± 0.29**	**97.56 ± 0.06**	99.16 ± 0.03
MH-pFedHNGD (ours)	**68.30 ± 0.44**	**80.07 ± 0.40**	**63.97 ± 0.84**	**82.51 ± 0.85**	**61.59 ± 0.77**	**82.54 ± 0.59**	**44.44 ± 0.56**	**58.35 ± 0.58**	**40.99 ± 0.70**	**63.39 ± 0.95**	**39.96 ± 0.70**	**66.67 ± 0.43**	**97.76 ± 0.04**	98.83 ± 0.02
Local	50.74 ± 2.62	71.69 ± 1.84	40.41 ± 2.57	72.01 ± 1.93	31.95 ± 3.41	73.05 ± 1.62	29.64 ± 1.83	51.64 ± 1.36	19.63 ± 1.47	54.05 ± 1.28	14.44 ± 1.78	53.92 ± 1.51	96.77 ± 0.32	98.83 ± 0.21
FedAvg	22.80 ± 1.91	17.27 ± 1.74	24.72 ± 1.63	22.18 ± 1.52	24.28 ± 1.49	21.29 ± 1.38	8.17 ± 1.02	8.22 ± 0.94	8.59 ± 1.14	11.86 ± 1.21	9.11 ± 1.26	16.40 ± 1.43	85.09 ± 0.44	84.78 ± 0.39
pFedHN	50.93 ± 1.08	73.22 ± 0.94	42.82 ± 1.15	74.73 ± 1.02	12.49 ± 1.37	72.62 ± 1.21	32.15 ± 1.04	54.29 ± 0.88	24.33 ± 1.12	58.31 ± 1.03	15.64 ± 1.29	57.82 ± 1.18	97.39 ± 0.24	99.07 ± 0.16
pFedLA	51.97 ± 1.21	71.91 ± 1.37	52.06 ± 1.09	74.68 ± 1.12	40.11 ± 1.26	75.24 ± 1.19	26.97 ± 1.42	46.86 ± 1.31	19.70 ± 1.38	51.73 ± 1.17	17.28 ± 1.44	54.44 ± 1.26	93.82 ± 0.35	98.36 ± 0.27
FedGH	52.61 ± 1.17	71.38 ± 1.28	42.45 ± 1.23	73.25 ± 1.11	31.57 ± 1.36	71.57 ± 1.24	25.69 ± 1.51	50.73 ± 1.42	16.90 ± 1.44	53.57 ± 1.21	11.74 ± 1.32	55.04 ± 1.16	96.01 ± 0.28	96.37 ± 0.25
pFedLHN	55.91 ± 0.96	75.28 ± 0.88	48.46 ± 1.07	76.16 ± 0.94	40.96 ± 1.18	74.60 ± 1.06	38.13 ± 0.91	55.13 ± 0.84	29.20 ± 1.03	59.36 ± 0.96	20.85 ± 1.14	57.53 ± 1.02	97.47 ± 0.17	99.15 ± 0.11
PeFLL	55.87 ± 1.10	72.28 ± 0.98	52.09 ± 1.08	74.51 ± 0.95	42.56 ± 1.22	72.79 ± 1.07	35.70 ± 0.98	55.21 ± 0.86	30.08 ± 1.14	55.60 ± 0.93	25.24 ± 1.21	55.01 ± 1.02	97.46 ± 0.19	99.10 ± 0.12
FedAKT	53.28 ± 1.13	73.15 ± 1.02	43.80 ± 1.20	75.25 ± 1.08	35.37 ± 1.32	74.98 ± 1.19	35.57 ± 1.05	53.74 ± 0.91	28.25 ± 1.17	56.95 ± 0.98	20.17 ± 1.29	55.36 ± 1.06	97.43 ± 0.21	99.11 ± 0.13
MH-pFedHN (ours)	**57.09 ± 0.71**	**75.40 ± 0.58**	50.35 ± 1.35	**77.24 ± 1.14**	**42.98 ± 0.87**	**75.38 ± 1.24**	38.00 ± 1.04	**55.51 ± 0.85**	**30.32 ± 0.47**	58.93 ± 0.80	23.31 ± 0.36	**58.79 ± 1.03**	**97.58 ± 0.18**	**99.18 ± 0.10**
MH-pFedHNGD (ours)	**60.11 ± 0.32**	**76.76 ± 0.64**	**52.14 ± 1.11**	77.03 ± 1.23	**43.41 ± 1.61**	**76.46 ± 0.90**	**42.18 ± 0.69**	**58.02 ± 0.61**	**34.98 ± 1.28**	**61.11 ± 0.61**	**26.12 ± 1.29**	**60.38 ± 1.49**	**97.65 ± 0.16**	98.73 ± 0.15

Table 1: Homogeneous (green) and heterogeneous (red) settings. Each cell reports results for non-IID_1 (left) and non-IID_2 (right).

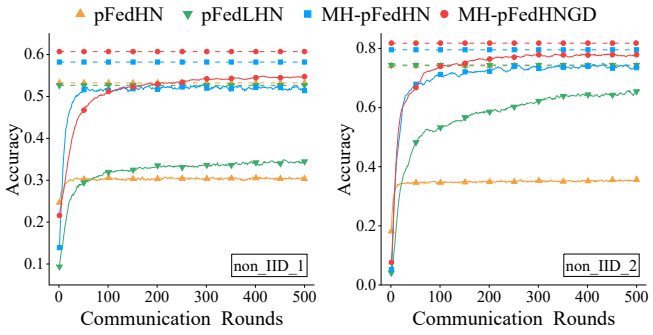

Figure 3: Generalization experiments (homogeneous), where solid lines denote test clients and dashed lines denote training clients.

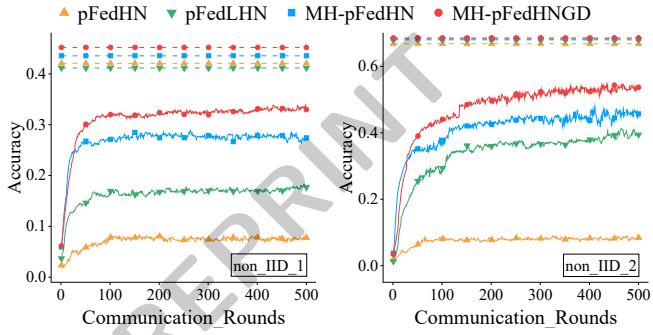

Figure 4: Generalization experiments (heterogeneous), where solid lines denote test clients and dashed lines denote training clients.

and leveraging it for knowledge distillation. Similar trends are also observed in the heterogeneous setting.

In the heterogeneous setting, we adapt pFedHN and pFedLHN by adding multiple heads so that they can generate parameters for different client models. Since PeFLL, pFedLA, FedAvg, and Local Training do not natively support model heterogeneity, we repeat experiments with different models and report the average test accuracy as the final results. Overall, allowing model heterogeneity generally leads to a performance drop compared to the homogeneous setting under the same configuration, suggesting that effective collaboration among heterogeneous clients remains challenging and deserves further investigation. Note that for methods that do not support heterogeneity, averaging results over multiple models (e.g., LeNet, VGGNet, and ResNet variants) can yield higher numbers than the homogeneous results where only a single model (e.g., LeNet) is used. Despite these challenges, MH-pFedHN and MH-pFedHNGD still achieve the best performance in most settings, and the gain of MH-pFedHNGD over MH-pFedHN is particularly evident under heterogeneous models, highlighting the necessity of combining a global model with distillation for MH-pFL.

Finally, most methods attain high accuracy on EMNIST (often above 96%), likely due to the dataset's relative simplicity, which makes meaningful comparisons more challenging.

4.3 Experiment with Generalization

Generalization in FL refers to the model's ability to perform well on unseen clients. This section evaluates the generalization capabilities of MH-pFedHN and MH-pFedHNGD. These experiments utilize the CIFAR-100 dataset with 100 clients, of which 80 clients are used to train the hypernetwork, while the remaining 20 clients are designated for generalization testing. During the testing phase, the parameters of the hypernetwork remain fixed, and only the customized embedding vectors corresponding to each client are optimized.

Generalize to homogeneous clients. Figure 3 shows that MH-pFedHNGD and MH-pFedHN exhibit outstanding generalization capabilities, significantly outperforming other baseline methods up to 20%. In the early stages of generalization, both algorithms converge rapidly and achieve accuracy on the test set comparable to that of the training clients. Moreover, the generalization ability of MH-pFedHNGD is superior to that of MH-pFedHN, indicating that introducing a global model further improves the hypernetwork's generalization.

Generalize to heterogeneous clients. We use LeNet, VGGNet, and ResNet10 from the heterogeneous setting, and further add MLP and SqueezeNet1_0 [Iandola, 2016]. These five models were evenly distributed among clients. Figure 4 shows that our method still exhibits the better generalization performance (up to 50%) of all the baselines for heterogeneous models, significantly outperforming the other two baselines. Again, the generalization ability of MH-pFedHNGD is notably superior to MH-pFedHN.

Generalize to new architectures. All the baselines can not handle this challenge. We use ResNet18 and SqueezeNet1_1 for the added clients, while training uses the same five models as above. In this case, only the feature extractor of the hypernetwork is frozen; the new head will be added and trained. Figure 5 shows no significant difference

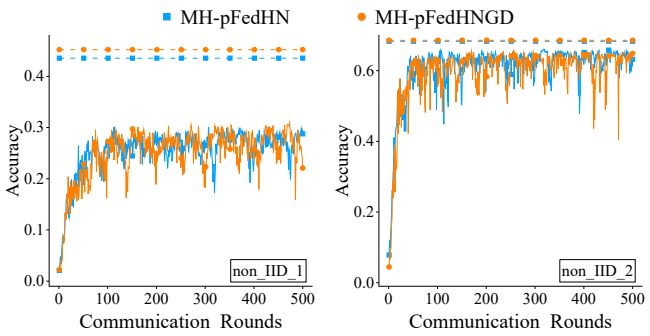

Figure 5: Generalization experiments (new architectures), where lines denote test clients and dashed lines denote training clients.

Clients	CIFAR-100					
	50		100		200	
homogeneous	58.32 ± 0.94	76.01 ± 0.53	47.86 ± 1.54	77.26 ± 0.84	40.63 ± 0.13	76.76 ± 0.64
heterogeneous	39.72 ± 1.51	57.74 ± 0.98	35.40 ± 1.88	64.27 ± 1.21	29.28 ± 0.85	60.68 ± 1.01
homogeneous	53.51 ± 0.43	73.70 ± 0.56	–	–	–	–
heterogeneous	–	–	–	–	–	–

Table 2: Ablation study of MH-pFedHN. Green indicates the setting without a head, and Red indicates the setting where each client has one head. "–" indicates out of memory. Each cell reports results for non-IID_1 (left) and non-IID_2(right).

Clients	CIFAR-100					
	50		100		200	
MH-pFedHN	64.69 ± 0.73	77.93 ± 0.46	63.32 ± 1.09	80.93 ± 0.70	60.11 ± 0.84	81.60 ± 0.68
MH-pFedHNG	66.19 ± 0.56	79.18 ± 0.42	63.36 ± 0.93	81.80 ± 0.78	59.71 ± 0.68	81.85 ± 0.64
MH-pFedHNGD	68.30 ± 0.44	80.07 ± 0.40	63.97 ± 0.84	82.51 ± 0.85	61.59 ± 0.77	82.54 ± 0.59
MH-pFedHN	57.09 ± 0.71	75.40 ± 0.58	50.35 ± 1.35	77.24 ± 1.14	42.98 ± 0.87	75.38 ± 1.24
MH-pFedHNG	57.01 ± 0.65	75.44 ± 0.60	49.81 ± 1.03	76.32 ± 1.21	41.45 ± 0.99	76.04 ± 1.06
MH-pFedHNGD	60.11 ± 0.32	76.76 ± 0.64	52.14 ± 1.11	77.03 ± 1.23	43.41 ± 1.61	76.46 ± 0.90

Table 3: Ablation study of MH-pFedHNGD under homogeneous (green) and heterogeneous (red) settings. Each cell reports results for non-IID_1 (left) and non-IID_2(right).

between MH-pFedHN and MH-pFedHNGD. The assistance provided by the global model is limited, and the learning during the generalization process primarily relies on the data distribution itself. Still, our methods outperform baselines.

4.4 Ablation Study

First, we investigate the head components of MH-pFedHN. We have two configurations: (1) MH-pFedHN does not have any heads; the hypernetwork consists of only a feature extractor; (2) MH-pFedHN where each client is equipped with an individual head with an output dimension of N. Table 2 indicates that without the head, the performance degrades significantly, demonstrating the importance of our head architecture in generating effective parameters. When each client is equipped with an individual head, most settings encounter out-of-memory issues, and the remaining two cases show limited performance, which highlights that our carefully designed shared head can greatly reduce computation overhead and hold strong practical potential.

Next, we examine the global model. We use MH-pFedHNG to denote MH-pFedHNGD without knowledge distillation. Table 3 shows that in most cases, MH-pFedHNGD outperforms MH-pFedHNG, which in turn outperforms MH-pFedHN. This indicates that the global model alone can enhance the hypernetwork's performance, while

Clients	CIFAR-100					
	50		100		200	
VGGNet	60.04 ± 0.88	77.72 ± 0.57	50.67 ± 0.63	77.71 ± 0.79	39.35 ± 0.74	76.94 ± 0.60
ResNet10	62.03 ± 0.61	75.57 ± 0.82	55.38 ± 0.86	72.50 ± 0.55	42.09 ± 0.72	65.50 ± 0.77
MLP	52.14 ± 0.93	69.97 ± 0.52	49.09 ± 0.58	73.13 ± 0.69	43.48 ± 0.80	74.32 ± 0.63
VGGNet/ResNet10/MLP	56.16 ± 0.67	73.73 ± 0.74	48.94 ± 0.91	75.32 ± 0.56	42.11 ± 0.60	75.01 ± 0.70
VGGNet	64.75 ± 0.46	79.82 ± 0.63	61.03 ± 0.54	79.93 ± 0.38	44.65 ± 0.41	78.07 ± 0.49
ResNet10	64.47 ± 0.58	75.66 ± 0.36	55.40 ± 0.43	74.59 ± 0.61	49.43 ± 0.62	73.67 ± 0.44
MLP	52.20 ± 0.37	69.98 ± 0.59	49.17 ± 0.64	73.58 ± 0.33	45.37 ± 0.49	74.75 ± 0.40
VGGNet/ResNet10/MLP	59.11 ± 0.63	74.66 ± 0.35	50.05 ± 0.39	76.68 ± 0.58	43.72 ± 0.52	76.25 ± 0.37

Table 4: Experiments with different architectures for global and personalized models. Green and red indicate MH-pFedHN and MH-pFedHNGD, respectively. Each cell reports results for non-IID_1 (left) and non-IID_2 (right).

knowledge distillation with the global model further improves model training across heterogeneous clients. The observation that results in homogeneous settings are better than those in heterogeneous ones is also consistent with other experiments.

4.5 Experiments with Different Architectures

Under the heterogeneous setting of MH-pFedHNGD, some clients and the global model also used the same LeNet architecture. Therefore, we further investigate the scenario where the architectures of the global model and the personalized client models are completely different. Thus, we set up experiments where the personalized client models are all VGGNet, MLP, and ResNet10, while the global model is LeNet. We also consider the case where these three types of models are evenly distributed among the clients, with the global model still being the LeNet model. Table 4 shows that for MH-pFedHNGD, even when the architectures of the global and client models are completely different, its performance still surpasses that of MH-pFedHN, which is consistent with our previous findings. Under such a challenging scenario, both models achieve promising results, demonstrating the strong robustness and generalization ability of our method.

4.6 Supplementary Experiments

Experiments with CIFAR-10, ViT, scalability, client participation ratios, different architectures of global model, redundant parameter, shared heads for heterogeneous models with similar parameter sizes, more baselines, overhead, architecture-specific clustering, progressive client enrollment, communication efficiency, extremely personalized setting, and resource constraint setting are in Appendix D. Experiments with **privacy concerns** (iDLG Attacks and differential privacy) are in Appendix E. Hyperparameter choice experiments are in Appendix F.

5 Conclusion

We propose a novel data-free approach for model-heterogeneous personalized federated learning, termed MH-pFedHN. This method employs a hypernetwork to generate the model parameters for each client, allowing clients to customize their network architectures without exposing them to the server. Furthermore, we present an improved MH-pFedHN with minimal effort, denoted as MH-pFedHNGD, which significantly enhances the learning and generalization abilities of the hypernetwork, improves the learning effectiveness of personalized models for clients, and reduces the risk of overfitting. Overall, the effectiveness of our approaches stands out through extensive experiments over baselines.

Acknowledgements

The work is supported by the National Natural Science Foundation of China (Grant No. 62402527), Natural Science Foundation of Shandong Province, China (Grant No. ZR2023QF153, ZR2023QF101), the Fundamental Research Funds for the Central Universities (Grant No. 26CX02004A), and Self-developed research project of Engineering Research Center of Intelligent Oilfields of the Ministry of Education.

References

[Beck *et al.*, 2024] Jacob Beck, Risto Vuorio, et al. Recurrent hypernetworks are surprisingly strong in meta-rl. *NeurIPS*, 2024.

[Chai *et al.*, 2020] Zheng Chai, Ahsan Ali, et al. Tifl: A tier-based federated learning system. In *Proc. of HPDC*, 2020.

[Chan *et al.*, 2024] Yun-Hin Chan, Zhihan Jiang, et al. Fedin: Federated intermediate layers learning for model heterogeneity, 2024.

[Chen *et al.*, 2023] Daoyuan Chen, Liuyi Yao, et al. Efficient personalized federated learning via sparse model-adaptation. *ArXiv*, 2023.

[Collins *et al.*, 2021] Liam Collins, Hamed Hassani, et al. Exploiting shared representations for personalized federated learning. In *ICML*, 2021.

[Csiszár, 1975] Imre Csiszár. i-divergence geometry of probability distributions and minimization problems. *Annals of Probability*, 1975.

[Dai *et al.*, 2024] Rong Dai, Yonggang Zhang, et al. Enhancing one-shot federated learning through data and ensemble co-boosting. *arXiv preprint arXiv:2402.15070*, 2024.

[Deng *et al.*, 2020] Yuyang Deng, Mohammad Mahdi Kamani, et al. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

[Diao *et al.*, 2021] Enmao Diao, Jie Ding, et al. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. In *ICLR*, 2021.

[Dinh *et al.*, 2020] Canh T. Dinh, Nguyen Tran, et al. Personalized federated learning with moreau envelopes. *NeurIPS*, 2020.

[Feng *et al.*, 2024] Chenyuan Feng, Daquan Feng, et al. Robust privacy-preserving recommendation systems driven by multimodal federated learning. *IEEE TNNLS*, 2024.

[Gong *et al.*, 2021] Xuan Gong, Abhishek Sharma, et al. Ensemble attention distillation for privacy-preserving federated learning. In *ICCV*, 2021.

[Goodfellow *et al.*, 2020] Ian Goodfellow, Jean Pouget-Abadie, et al. Generative adversarial networks. *Communications of the ACM*, 2020.

[Ha *et al.*, 2017] David Ha, Andrew M. Dai, et al. Hypernetworks. In *ICLR*, 2017.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, et al. Deep residual learning for image recognition. In *CVPR*, 2016.

[He *et al.*, 2020] Chaoyang He, Murali Annavaram, et al. Group knowledge transfer: federated learning of large cnns at the edge. In *NeurIPS*, 2020.

[Hinton, 2015] Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Horváth and others, 2021] Samuel Horváth et al. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. In *NeurIPS*, 2021.

[Iandola, 2016] Forrest N Iandola. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[Itahara *et al.*, 2021] Sohei Itahara, Takayuki Nishio, et al. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE TMC*, 2021.

[Jang *et al.*, 2023] Jaehee Jang, Heoneok Ha, et al. Fedclassavg: Local representation learning for personalized federated learning on heterogeneous neural networks. In *ICPP*, 2023.

[Jeong *et al.*, 2018] Eunjeong Jeong, Seungeun Oh, et al. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

[Kim *et al.*, 2023] Minjae Kim, Sangyoon Yu, et al. Depthfl: Depthwise federated learning for heterogeneous clients. In *ICLR*, 2023.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, et al. Gradient-based learning applied to document recognition. *Proc. IEEE*, 1998.

[Lee *et al.*, 2024] Royson Lee, Javier Fernandez-Marques, et al. Recurrent early exits for federated learning with heterogeneous clients, 2024.

[Li and Ma, 2024] Xinjin Li and Yu et.al. Ma. Synergized data efficiency and compression (sec) optimization for large language models. In *2024 EIECS*, 2024.

[Li and Wang, 2019] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[Li *et al.*, 2021] Xiaoxiao Li, Meirui Jiang, et al. FedBN: Federated learning on non-IID features via local batch normalization. In *ICLR*, 2021.

[Liang *et al.*, 2020] Paul Pu Liang, Terrance Liu, et al. Think locally, act globally: Federated learning with local and global representations. *arXiv:2001.01523*, 2020.

[Litany *et al.*, 2022] Or Litany, Haggai Maron, et al. Federated learning with heterogeneous architectures using graph hypernetworks. *arXiv*, 2022.

[Liu *et al.*, 2024] Xiang Liu, Liangxi Liu, et al. Fedlpa: One-shot federated learning with layer-wise posterior aggregation. *NeurIPS*, 2024.

[Liu *et al.*, 2025] Shichong Liu, Haozhe Jin, et al. Adapter-guided knowledge transfer for heterogeneous federated learning. *J. Syst. Archit.*, 2025.

[Ma *et al.*, 2022] Xiaosong Ma, Jie Zhang, et al. Layer-wised model aggregation for personalized federated learning. In *CVPR*, 2022.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, et al. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[Meng *et al.*, 2024] Lei Meng, Zhuang Qi, et al. Improving global generalization and local personalization for federated learning. *IEEE TNNLS*, 2024.

[Murmu *et al.*, 2024] Anita Murmu, Piyush Kumar, et al. Reliable federated learning with gan model for robust and resilient future healthcare system. *IEEE TNSM*, 2024.

[Nguyen *et al.*, 2022] Dinh C Nguyen, Quoc-Viet Pham, et al. Federated learning for smart healthcare: A survey. *ACM Comput. Surv.*, 2022.

[Nirkin *et al.*, 2021] Yuval Nirkin, Lior Wolf, et al. Hyperseg: Patch-wise hypernetwork for real-time semantic segmentation. In *CVPR*, 2021.

[Qi *et al.*, 2025] Zhuang Qi, Lei Meng, et al. Cross-silo feature space alignment for federated learning on clients with imbalanced data. In *AAAI*, 2025.

[Ruiz *et al.*, 2024] Nataniel Ruiz, Yuanzhen Li, et al. Hyperdreambooth: Hypernetworks for fast personalization of text-to-image models. In *CVPR*, 2024.

[Sattler *et al.*, 2020] Felix Sattler, Arturo Marban, et al. Communication-efficient federated distillation. *arXiv:2012.00632*, 2020.

[Scott *et al.*, 2024] Jonathan A Scott, Hossein Zakerinia, et al. Pefll: Personalized federated learning by learning to learn. In *ICLR*, 2024.

[Sendera *et al.*, 2023] Marcin Sendera, Marcin Przewiezlikowski, et al. Hypershot: Few-shot learning by kernel hypernetworks. In *WACV*, 2023.

[Shah and Lau, 2021] Suhail Mohmad Shah and Vincent K. N. Lau. Model compression for communication efficient federated learning. *IEEE TNNLS*, 34, 2021.

[Shamsian *et al.*, 2021] Aviv Shamsian, Aviv Navon, et al. Personalized federated learning using hypernetworks. In *ICML*, 2021.

[Shen *et al.*, 2024] Leming Shen, Qiang Yang, et al. Fedconv: A learning-on-model paradigm for heterogeneous federated clients. In *MobiSys*, 2024.

[Shin *et al.*, 2024] Yujin Shin, Kichang Lee, et al. Effective heterogeneous federated learning via efficient hypernetwork-based weight generation. In *SenSys*, 2024.

[Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[Suarez, 2017] Joseph Suarez. Language modeling with recurrent highway hypernetworks. *NeurIPS*, 2017.

[Wang *et al.*, 2022] Xiaoding Wang, Wenxin Liu, et al. Ai-empowered trajectory anomaly detection for intelligent transportation systems: A hierarchical federated learning approach. *IEEE TITS*, 2022.

[Wang *et al.*, 2024] Shuai Wang, Yexuan Fu, et al. Dfrd: Data-free robustness distillation for heterogeneous federated learning. *NeurIPS*, 2024.

[Wang *et al.*, 2025] Zichen Wang, Feng Yan, et al. Feddfa: Federated distillation for heterogeneous model fusion through the adversarial lens. *AAAI*, 2025.

[Wu *et al.*, 2021] Chuhan Wu, Fangzhao Wu, et al. Fedkd: Communication efficient federated learning via knowledge distillation. *arXiv:2108.13323*, 2021.

[Xu *et al.*, 2023] Jian Xu, Xinyi Tong, et al. Personalized federated learning with feature alignment and classifier collaboration. *arXiv:2306.11867*, 2023.

[Yi *et al.*, 2023a] Liping Yi, Gang Wang, et al. Fedgh: Heterogeneous federated learning with generalized global header. In *ACM MM*, 2023.

[Yi *et al.*, 2023b] Liping Yi, Han Yu, et al. pfedes: Model heterogeneous personalized federated learning with feature extractor sharing. *arXiv:2311.06879*, 2023.

[Yuan *et al.*, 2023] Wei Yuan, Hongzhi Yin, et al. Federated unlearning for on-device recommendation. In *WSDM*, 2023.

[Zeng *et al.*,] Yiming Zeng, Jinghan Cao, and et.al. Hyperedit: Unlocking instruction-based text editing in llms via hypernetworks.

[Zeng *et al.*, 2025] Yiming Zeng, Wanhao Yu, and et.al. Bridging the editing gap in llms: Fineedit for precise and targeted text modifications. *EMNLP*, 2025.

[Zhang and Liu, 2024] Guangsheng Zhang and Bo et.al. Liu. How does a deep learning model architecture impact its privacy? a comprehensive study of privacy attacks on {CNNs} and transformers. In *USENIX Security*, 2024.

[Zhang *et al.*, 2021] Jie Zhang, Song Guo, et al. Parameterized knowledge transfer for personalized federated learning. *NeurIPS*, 2021.

[Zhang *et al.*, 2022] Jie Zhang, Chen Chen, et al. Dense: Data-free one-shot federated learning. *NeurIPS*, 2022.

[Zhang *et al.*, 2023] J. Zhang, Song Guo, et al. Towards data-independent knowledge transfer in model-heterogeneous federated learning. *IEEE TC*, 2023.

[Zhang *et al.*, 2024] Guangsheng Zhang, Bo Liu, et al. How does a deep learning model architecture impact its privacy? a comprehensive study of privacy attacks on cnns and transformers. In *USENIX Security*, 2024.

[Zhou *et al.*, 2024] Xiaokang Zhou, Wei Liang, et al. Adaptive segmentation enhanced asynchronous federated learning for sustainable intelligent transportation systems. *IEEE TITS*, 2024.

[Zhu *et al.*, 2021] Zhuangdi Zhu, Junyuan Hong, et al. Data-free knowledge distillation for heterogeneous federated learning. In *ICML*, 2021.

[Zhu *et al.*, 2023] Suxia Zhu, Tianyu Liu, et al. Layer-wise personalized federated learning with hypernetwork. *Neural Process. Lett.*, 2023.