Hierarchical Asynchronous Federated Learning Algorithm for Edge Computing Networks

Aijun Wen, Yunxi Fu, Zesan Liu, Zhenya Wang, Wenjuan Zhang*

State Grid Information and Communication Industry Group Co., Ltd., China
wenaijun@sgitg.sgcc.com.cn, yunxifu2018@163.com, liuzesan@sgitg.sgcc.com.cn,
wangzhenya@sgitg.sgcc.com.cn, zhangwenjuan@sgitg.sgcc.com.cn

Abstract

Federated learning aims to enable clients to jointly train a model with privacy without sharing the original data. Compared to centralized model training, federated learning introduces heterogeneous data among the distributed participants and communication bottlenecks problems. This article proposes a hierarchical Bayesian federated learning approach to achieve local model personalization and hierarchical model parameter aggregation, thereby addressing the heterogeneous data problem and reducing communication costs in federated learning. The variational inference method can effectively solve the heterogeneous data problem encountered by each participant in federated learning, demonstrating excellent robustness when handling different types of statistical heterogeneity problems, thereby effectively realizing the personalization of local models. Multilevel hierarchical model parameter aggregation and resource scheduling can also reduce communication costs in federated learning. Therefore, the hierarchical Bayesian federated learning framework proposed in this article controls the random variables of each participant's local model with global variables, and the model construction process is completed hierarchically and collaboratively, realizing robustness improvement and communication optimization.

Keywords: Hierarchical asynchronous federated learning, Multilevel collaborative, Personalized model, Variational inference, Separable optimization

1 Introduction

Federated learning [1-5] aims to enable clients to jointly train a model without sharing data while protecting privacy. Federated learning can efficiently support data fusion and sharing applications across organizations, institutions, departments, systems, etc., under the conditions of ensuring information security, data privacy security, and policy regulation fulfillment. Federated learning technology may be comparable to the traditional centralized model training strategy [6], but it faces statistical and systematic challenges regarding scattered participants and communication bottlenecks.

A significant statistical challenge is the non-IID data problem [7-8], where bias is present in the data between the clients and in the data aggregation and label allocation strategies. The most prevalent federated learning algorithms, FedAvg and FedProx, perform well in terms of training global models in highly heterogeneous scenarios [9] but often perform poorly with respect to addressing the local data distribution of each client. Recent research has attempted to alleviate this problem by allowing each client to train personalized and regional models that deviate from the shared global model. However, considering that the local data used by each client for customized training may be limited, building effective local models remains a challenging problem. Another core challenge in federated learning is the high cost of communication. In the real world, the huge number of terminal devices and the complex communication environment lead to the great communication pressure of Federated learning. In order to reduce the traffic in this complex environment, the Federated learning algorithm for communication cost optimization is mainly studied from these two aspects: reducing the communication times between client and server; Reduce the size of data transmitted in each communication.

This article introduces Bayesian theory [10-11], and by adopting multilevel collaborative hierarchical processing, the regional models are aggregated into edge models at the edge server, and the edge models are aggregated into global models at the cloud server, effectively addressing the user data privacy and security protection issues while reducing communication costs. This article elaborates on our research content step-by-step, starting with a background introduction to the current status and challenges of federated learning. The second part introduces layered federated learning, including the basic concept of hierarchical asynchronous federated learning, hierarchical model aggregation, and the hierarchical Bayesian federal learning framework. The third part presents the model training in the hierarchical Bayesian federal learning framework, including updating the model parameters, personalized model construction, and algorithm execution. The fourth part demonstrates the framework's performance analysis, including the convergence/generalization analysis and the transmission optimization. In the fifth part, we display the simulation results. Furthermore, finally, we summarize the entire thesis.

*Corresponding Author: Yunxi Fu; Email: yunxifu2018@163.com
DOI: https://doi.org/10.70003/160792642025092605005

2 Layered Federated Learning

Unlike traditional two-layer Federated learning, multilevel collaborative hierarchical asynchronous federated learning applies edge computing to federated learning and introduces edge servers as intermediaries between participating nodes and cloud servers. The edge server receives local model parameters from participating nodes and aggregates edge models-multiple rounds of model parameter distribution and aggregation on the edge and terminal sides. Then, the edge server uploads the edge aggregation model to the cloud server for global model aggregation, reducing the amount of uplink data transmission.

2.1 Hierarchical Asynchronous Federated Learning

In traditional two-layer federated learning, participating nodes use local data for local model training and upload it to the central aggregation server for global model aggregation. The objective function of the whole model training is to find the global model that minimizes the loss function. The objective function to minimize the loss function is usually defined as:

$$\min_{\omega} F(\omega) \tag{1}$$

$$F(\omega) = \frac{\sum_{m=1}^{M} |D_m| F_m(\omega)}{D} \tag{2}$$

$$F_m(\omega) = \frac{\sum_{j=1}^{|D_m|} f_j(\omega)}{|D_m|} \tag{3}$$

Among them, $D = \{x_j, y_j\}_{j=1}^{|D|}$ is the total training data set, $|D|$ is the total amount of data from all local participating nodes, x_j is the j-th input sample, y_j is the output label corresponding to the j-th input sample, and M is the total number of nodes participating in model training. $F_m(\omega)$ represents the local loss function of participating node m, and $f_j(\omega)$ represents the loss function generated by the model with parameter ω on the instance (x_j, y_j, ω) in data set D_m. In the federated learning system, the participating nodes usually use the random gradient descent (SGD) algorithm to update the local model; η represents the learning rate, then the update of the local model in iteration t is as follows:

$$\omega_m(t) = \omega_m(t-1) - \eta \nabla F_m(\omega_m(t-1)) \tag{4}$$

The aggregation of the global model of the central aggregation server in the t-round iteration is as follows:

$$\omega_t = \frac{\sum_{m=1}^{M} |D_m| \omega_m(t)}{|D|} \tag{5}$$

According to the above federated learning model training process, many model updates need to be exchanged between participating nodes and the central aggregation server, which requires a large number of network resources and increases uplink communication costs. Hierarchical asynchronous federated learning applies edge computing to federated learning, uses edge servers' computing and transmission capabilities to aggregate some models, reduces the amount of data transmitted on the uplink, and effectively solves the above problems.

This paper considers a typical cloud edge end three-layer hierarchical asynchronous federated learning scenario, and its workflow is shown in Figure 1. There are three main entities in hierarchical asynchronous federated learning: participating nodes on the terminal side, edge aggregation servers on the edge side, and the cloud aggregation server. Participating nodes generate or collect data, with each node having its private dataset; The edge server receives local model parameters uploaded by participating nodes and aggregates edge models; And, after a T-times updating for the model parameters between the edge aggregation server layer and terminal client layer, the edge model parameters are updated to the cloud server from the edge server.; The cloud server stores the original model and shares the global model with all participating nodes, receiving the edge model uploaded by the edge server for global model aggregation.

2.2 Hierarchical Asynchronous Model Aggregation

The aggregation process is outlined as follows. Multilevel collaborative hierarchical asynchronous federated learning is implemented in a cloud-edge-end environment. During this process, the local partial model aggregation strategy generates edge models on the edge server, and the edge model aggregation step forms the global model on the cloud server. This innovative approach effectively addresses the issues of user data privacy and security protection while reducing communication costs. The hierarchical model aggregation process is divided into the following steps.

Step 1: Model Retrieval. In the hierarchical Bayesian Federated learning framework, the edge aggregation server retrieves the global model from the cloud server in a hierarchical Bayesian federated learning framework, while the terminal-side client obtains the edge model from the edge server.

Step 2: The edge aggregation server layer and terminal client layer update the model parameters T times, and each update includes the following steps:

① The terminal side client uses the local data set to train the local model, generate a personalized local model, and update the model parameters;

② The terminal client uploads the updated personalized model parameters to the edge aggregation server connected to it;

③ The edge aggregation server performs Bayesian aggregation on the received local model parameters to generate an edge model;

④ The edge aggregation server sends the edge model parameters after Bayesian aggregation to the terminal client connected to it.

Step 3: The edge aggregation server uploads the edge model after Bayesian aggregation to the cloud side server;

Step 4: The cloud-side server performs Bayesian aggregation on the edge model parameters. A global model can be generated through Bayesian aggregation;

Step 5: The cloud server sends the aggregated global model parameters to the corresponding edge servers;

Step 6: The edge aggregation server performs Bayesian aggregation on the received global model parameters from the cloud side and personalized local model parameters from the terminal to generate a personalized edge model;

Step 7: The edge aggregation server sends the edge model parameters after Bayesian aggregation to the terminal side client connected to it;

Step 8: The terminal client receives the edge model from the edge server and inputs personalized training data for local model training. Fine-tune the edge model to generate a personalized local model, which is used as its personalized model for subsequent personalized prediction.

The hierarchical aggregation process of the model is shown in Figure 1:

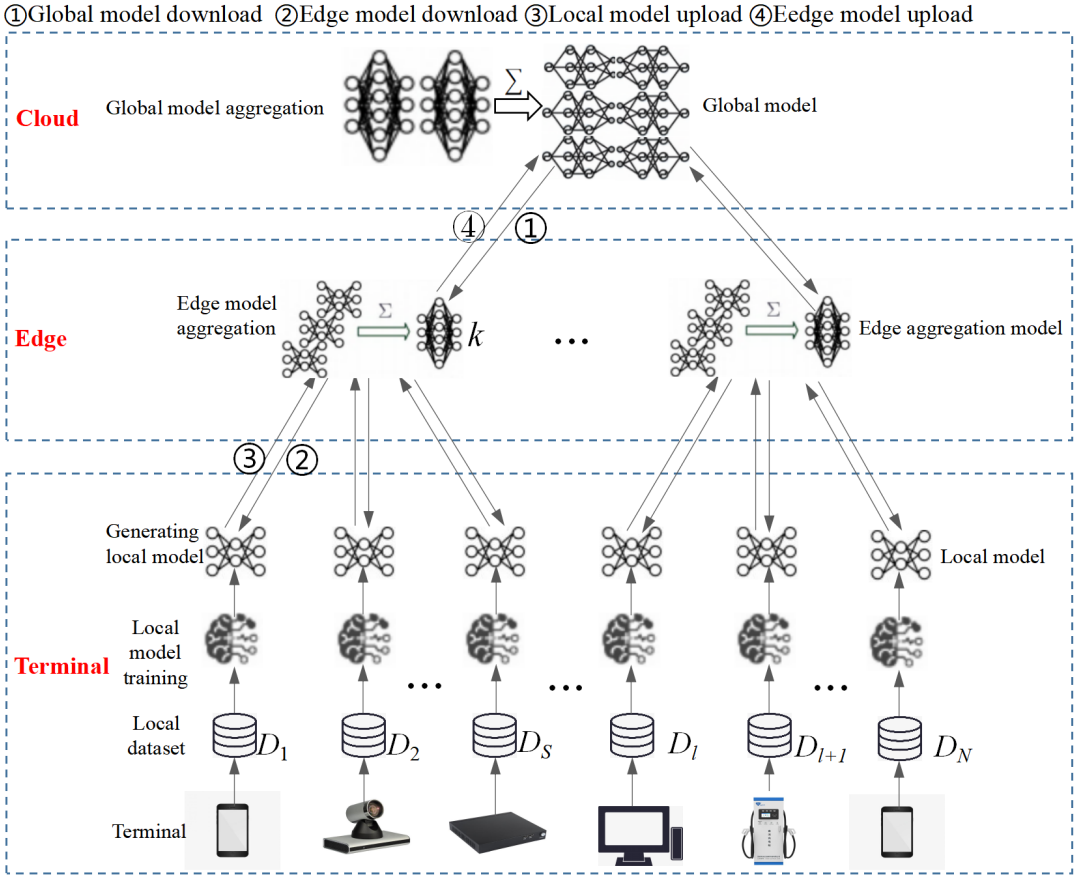


Figure 1. The process of hierarchical aggregation for the model

2.3 The Hierarchical Bayesian Federal Learning Framework

In the hierarchical Bayesian federal learning framework, three types of random variables Ψ, $\{\phi_k\}_{k=1}^{M}$ and $\{\theta_i\}_{i=1}^{S}$ are introduced. Among them, random model weight variables θ_i are assigned to the model weights of the terminal-side client (where i represents the i-th participating party on the terminal side), and ϕ_k represents the model weights of the edge aggregation server k on the edge side (where k represents the k-th participant on the upper level, i.e., the edge side), which is responsible for connecting the weights θ_i of various clients. Ψ can be regarded as the global shared model weights on the higher level (cloud side), which are responsible for securing the consequences ϕ_k of the different edge aggregation servers on the edge side. Furthermore, S represents the number of terminal-side clients involved in the edge-side federated learning process under the edge aggregation server k, M denotes the number of participating parties involved in federated learning on the edge side. The Bayesian prior probability and Bayesian likelihood (sample distribution) are hierarchically formed into Formulas (6), (7), (8), (9), (10), (11), (12), (13) and (14). Bayesian prior probability:

$$p(\phi_k,\theta_i)=p(\phi_k)\,p(\theta_i|\phi_k), \tag{6}$$

$$p(\phi_k,\theta_{1:S})=p(\phi_k)\prod_{i=1}^{S}p(\theta_i|\phi_k), \tag{7}$$

$$p(\psi,\phi_k)=p(\psi)\,p(\phi_k,\psi), \tag{8}$$

$$p(\psi,\phi_{1:M})=p(\psi)\prod_{k=1}^{M}p(\phi_k|\psi), \tag{9}$$

based on further deduction, we can derive the following formulas:

$$p(\psi,\phi_k,\theta_i)=p(\psi)\,p(\phi_k|\psi)\,p(\theta_i|\phi_k), \tag{10}$$

$$p(\psi,\phi_k,\theta_{1:S})=p(\psi)\,p(\phi_k|\psi)\prod_{i=1}^{S}p(\theta_i|\phi_k), \tag{11}$$

$$p(\psi,\phi_{1:M},\theta_{1:N})=p(\psi)\prod_{k=1}^{M}p(\phi_k|\psi)\prod_{i=1}^{N}p(\theta_i|\phi_k), \tag{12}$$

where, N represents the total number of terminal-side clients participating in federated learning, and there is $N = S \times M$. Sample distribution:

$$p(D_i|\theta_i)=\prod_{(x,y^k)\in D_i}p(y^k|x,\theta_i), \tag{13}$$

$$p(U_k|\phi_k)=\prod_{(x,y^k)\in U_k}p(y^k|x,\phi_k), \tag{14}$$

here, D_i represents the local dataset of terminal client i, and U_k represents the sum of all local datasets of the terminal clients covered by edge server k during federated training. If S terminal clients participate in the federated training process under edge server k, then $U_k = \sum_{i=1}^{S} D_i$. In the above formulas, $p(y^k|x, \theta_i)$ and $p(y^k|x, \phi_k)$ represent the traditional neural network model. Given the datasets D_1, D_2, D_3, …, D_N, the Bayesian posterior probabilities can be inferred:

$$p(\phi_k,\theta_{1:S}|D_{1:S})\propto p(\phi_k)\prod_{i=1}^{S}p(\theta_i|\phi_k)p(D_i|\theta_i), \tag{15}$$

$$p(\psi,\phi_{1:M}|U_{1:M})\propto p(\psi)\prod_{k=1}^{M}p(\psi|\phi_k)p(U_k|\phi_k), \tag{16}$$

$$p(\psi,\phi_{1:M}|U_{1:M},\theta_{1:N}|D_{1:N})\propto P(\psi)\prod_{k=1}^{M}p(\psi|\phi_k)\prod_{i=1}^{N}p(\theta_i|\phi_k)p(D_i|\theta_i). \tag{17}$$

The Bayesian posterior probability can be approximated by using the method of variational inference as follows:

$$q(\phi_k,\theta_{1:S};L_k):=q(\phi_k;L_{00})\prod_{i=1}^{S}q_i(\theta_i;L_i), \tag{18}$$

$$q(\psi,\phi_{1:M};L):=q(\psi;L_0)\prod_{k=1}^{M}q_k(\phi_k;L_k), \tag{19}$$

based on further deduction, we can derive the following formula:

$$q(\psi,\theta_{1:N};L):\ q(\psi;L_0)\prod_{k=1}^{M}q_k(\phi_k;L_k)\left[q(\phi_k;L_{00})\prod_{i=1}^{S}q_i(\theta_i;L_i)\right], \tag{20}$$

here, the variational parameter L consists of L_0 (the parameters of $q(\Psi)$), $\{L_k\}_{k=1}^{M}$ (the parameters of edge server $q_k(\phi_k)$), L_{00} (the parameters of $q(\phi_k)$), and $\{L_i\}_{i=1}^{S}$ (the parameters of client $q_i(\theta_i)$).

Through hierarchical processing and multilevel collaborative Bayesian federated learning, the variational inference procedure is decomposed into separable optimization problems involving θ_i, ϕ_k and Ψ, and practical Bayesian learning algorithms that are compatible with federated learning constraints are derived. This method exhibits exceptional robustness when addressing various statistical heterogeneity problems. It can efficiently achieve personalized processing for the local partial models of each participant on the edge side, effectively resolving the issue of non-IID data.

3 Model Training in the Hierarchical Bayesian Federal Learning Framework

3.1 Updating the Model Parameters

Variational inference [12] is used to solve the difficult-to-compute true posterior distribution by using a convenient distribution to approximate the true posterior distribution. Thus, the log-likelihood function of parameter ϕ_k can be written as $\log p(\phi_k) = D_{KL}(q(\phi_k)||p(\phi_k) + \mathcal{L}(\phi_k)$. Here, $D_{KL}(q(\phi_k) \,||\, p(\phi_k)$ aims to make the learned distribution $q(\phi_k)$ approach the true posterior distribution $p(\phi_k)$, while $\mathcal{L}(\phi_k)$ is the variational bound, which is also known as the evidence lower bound (ELBO).

The ELBO objective function can be derived through standard variational inference techniques and hierarchical processing. The negative ELBO (to be minimized) can be represented in the form of a sum:

$$\begin{aligned}&\mathcal{L}_k(L_k):=\\&\sum_{i=1}^{S}\left(\mathbb{E}_{q(\phi_k)}\left[\mathrm{KL}\left(q_i(\theta_i)\middle\|p(\theta_i|\phi_k)\right)\right]\right)+\\&\mathbb{E}_{q_i(\theta_i)}\left[-\log p(D_i|\theta_i)\right]+\\&\mathrm{KL}\left(q(\phi_k)\middle\|p(\phi_k)\right)\end{aligned}\tag{21}$$

$$\begin{aligned}&\mathcal{L}(L):=\\&\sum_{k=1}^{M}\left(\mathbb{E}_{q(\psi)}\left[\mathrm{KL}\left(q_k(\phi_k)\middle\|p(\phi_k|\psi)\right)\right]\right)+\\&\mathbb{E}_{q_k(\phi_k)}\left[-\log p(U_k|\phi_k)\right]+\\&\mathrm{KL}\left(q(\psi)\middle\|p(\psi)\right)\end{aligned}\tag{22}$$

Here, D_i represents the local dataset of terminal-side client i, and U_k represents the sum of all local datasets covered by edge server k during the federated training process of the terminal-side clients. It is assumed that the number of terminal clients participating in federated activity under edge server k is S; then, $U_k = \Sigma_{i=1}^{S} D_i$. KL denotes the Kullback–Leibler divergence measure, and $\mathrm{E}_{q(\phi_k)}$ and $\mathrm{E}_{q(\Psi)}$ represent the log-likelihood estimates for ϕ_k and Ψ, respectively. In this paper, instead of jointly optimizing L via conventional methods, we consider a hierarchical and blockwise optimization strategy [13], which is divided into four alternating steps: (1) updating/optimizing all $L_i(i = 1, 2, \ldots, S)$ while fixing L_{00}; (2) updating L_{00} while fixing all $L_i(i = 1, 2, \ldots, S)$; (3) updating/optimizing all $L_k(k = 1, 2, \ldots, M)$ while fixing L_0; and (4) updating L_0 while fixing all $L_k(k = 1, 2, \ldots, M)$.

Regarding the optimization issues encountered during the parameter aggregation process between the edge and terminal sides, the following formula holds.

• Optimizing $L_1, L_2, \ldots, L_S$ (with L_{00} fixed)

$$\min_{\{L_i\}_{i=1}^{S}}\sum_{i=1}^{S}\begin{pmatrix}\mathbb{E}_{q(\phi_k)}\left[\mathrm{KL}\left(q_i(\theta_i)\middle\|p(\theta_i|\phi_k)\right)\right]\\+\mathbb{E}_{q_i(\theta_i)}\left[-\log p(D_i|\theta_i)\right]\end{pmatrix}\tag{23}$$

Formula (23) is completely separable in i, so we can independently optimize each addend. The following procedure can be obtained:

$$\begin{aligned}&\min_{L_i}\mathcal{L}_i(L_i):=\\&\mathbb{E}_{q(\phi_k;L_{00})}\left[\mathrm{KL}\left(q_i(\theta_i;L_i)\middle\|p(\theta_i|\phi_k)\right)\right]+\\&\mathbb{E}_{q_i(\theta_i;L_i)}\left[-\log p(D_i|\theta_i)\right]\end{aligned}\tag{24}$$

Hence, Formula (24) represents the local optimization process of the clients. Each client only needs to access its local private data D_i without relying on the data of other participants, which satisfies the requirements of federated learning.

• Optimizing L_{00} (with $L_1, L_2, \ldots, L_S$ fixed)

$$\begin{aligned}&\min_{L_{00}}\mathscr{J}_{00}(L_{00}):\\&=-\sum_{i=1}^{S}\mathbb{E}_{q(\phi_k;L_0)q_i(\theta_i;L_i)}\left[\log p(\theta_i|\phi_k)\right]\\&+\mathrm{KL}\left(q(\phi_k;L_{00})\middle\|p(\phi_k)\right)\end{aligned}\tag{25}$$

The above process establishes the update criterion for the edge-side edge aggregation server parameters, and the edge aggregation server does not need to access any local data on the terminal side, which is in line with federated learning.

Regarding the optimization issues encountered during the process of parameter aggregation between the cloud and edge sides, the following formula holds.

• Optimizing $L_1, L_2, \ldots, L_M$ (with L_0 fixed)

$$\min_{\{L_k\}_{k=1}^{M}}\sum_{k=1}^{M}\begin{pmatrix}\mathbb{E}_{q(\psi)}\left[\mathrm{KL}\left(q_k(\phi_k)\middle\|p(\phi_k|\psi)\right)\right]\\+\mathbb{E}_{q_k(\phi_k)}\left[-\log p(U_k|\phi_k)\right]\end{pmatrix}\tag{26}$$

Formula (26) is completely separable in k, so we can independently optimize each addend. The following formula can be obtained:

$$\begin{aligned}&\min_{L_k}\mathcal{L}_k(L_k):\\&=\mathbb{E}_{q(\psi;L_0)}\left[\mathrm{KL}\left(q_k(\phi_k;L_k)\middle\|p(\phi_k|\psi)\right)\right]\\&+\mathbb{E}_{q_k(\theta_k;L_k)}\left[-\log p(U_k|\phi_k)\right]\end{aligned}\tag{27}$$

Thus, edge aggregation server k is optimized, and the edge aggregation server does not need to access any local data. This rule is compatible with federated learning.

• Optimizing L_0 (with $L_1, L_2, \ldots, L_M$ fixed)

$$\begin{aligned}&\min_{L_0}\mathscr{J}_0(L_0):\\&=-\sum_{k=1}^{M}\mathbb{E}_{q(\psi;L_0)q_i(\phi_k;L_k)}\left[\log p(\phi_k|\psi)\right]\\&+\mathrm{KL}\left(q(\psi;L_0)\middle\|p(\psi)\right)\end{aligned}\tag{28}$$

The above process determines the specification for the server-side parameter updates, and the server does not need to access any local data, making this step consistent with federated learning.

Therefore, the specification for updating federated model parameters derived from variational inference forms the update specifications for the local model parameters at the terminal-side client, the edge aggregation model parameters at the edge-side edge aggregation server, and the global model parameters at the cloud server, guiding the construction of multilevel hierarchical asynchronous federated learning models.

3.2 Personalized Model Construction

The two core objectives of federated learning are global prediction and personalized adjustment. Global forecasting aims to evaluate the performance of the training model on new test data that may have a different distribution from that of the training data. On the other hand, personalized adjustment optimizes the training model on local, customized datasets. Due to the non-IID nature of the data distributed on the client side, it is difficult to train a single global model that applies to all clients. Therefore, constructing personalized models for each client is crucial. Personalized federated learning algorithm [14] is usually combined with Transfer learning [15-16], knowledge distillation, meta-learning, multi-task learning [17], and other machine learning technologies. Transfer learning enables the deep learning model to use the knowledge obtained when solving one problem to solve another related problem. The principles of global prediction and personalized prediction are illustrated in Figure 2.

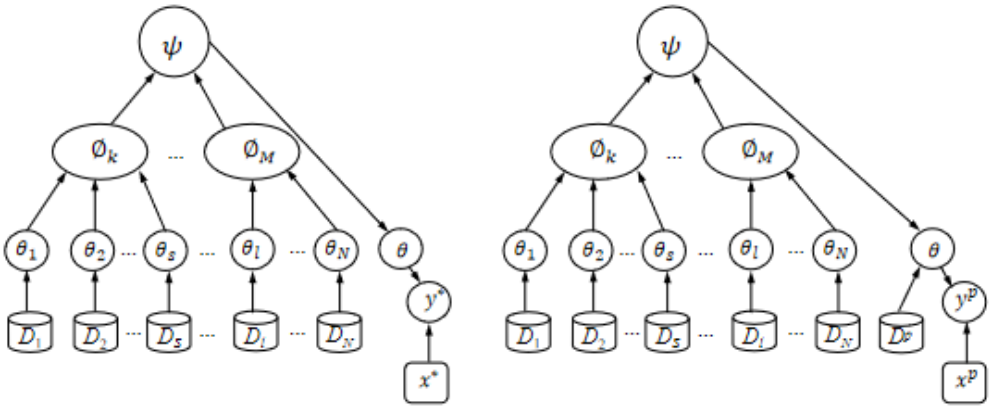


Figure 2. Regarding global and personalized prediction as probabilistic reasoning problems

(D_1, D_2, …, D_N represents the training set, y^* and y^p denote the target values to be predicted, x^* and x^p represent the test inputs for global prediction and personalized model adjustment, respectively, and D^p signifies the personalized training data.)

• Global Prediction. In the global prediction task, the model needs to generate a prediction result for a new test input x^*. The input value x^* comes from the dataset D^*, and the data distribution of D^* may be the same as or different from that of the training dataset D_1, D_2, …, D_N. In this case, N represents the total number of terminal-side client devices participating in federated learning model training. Under the framework of the hierarchical Bayesian model, this problem can be transformed into a probabilistic inference problem $P_k(y^*|x^*, D_{1:S})$ and $p(y^*|x^*, \varnothing_{1:M})$. In this context, the local model θ is responsible for receiving the input x^* and generating an output y^*. The probability can be described by the following formula:

$$\begin{aligned} & p_k\left(y^* \middle| x^*, D_{1:S}\right) \\ &= \iint p_k\left(y^* \middle| x^*, \theta_i\right) p\left(\theta_i \middle| \varnothing_k\right) p\left(\varnothing_k \middle| D_{1:S}\right) d\theta d\varnothing \\ &\approx \iint p_k\left(y^* \middle| x^*, \theta_i\right) p\left(\theta_i \middle| \varnothing_k\right) q\left(\varnothing_k\right) d\theta d\varnothing \\ &= \int p_k\left(y^* \middle| x^*, \theta_i\right)\left(\int p\left(\theta_i \middle| \varnothing_k\right) q\left(\varnothing_k\right) d\varnothing\right) d\theta \end{aligned} \tag{29}$$

In the above equation, $p(\varnothing_k | D_{1:S}) \approx q(\varnothing_k)$.

$$\begin{aligned} & p\left(y^* \middle| x^*, D_{1:S}, \varnothing_{1:M}\right) \\ &= \iint p\left(y^* \middle| x^*, \varnothing_k\right) p\left(\varnothing_k \middle| \psi\right) p\left(\psi \middle| \varnothing_{1:M}\right) d\varnothing d\psi \\ &\approx \iint p\left(y^* \middle| x^*, \varnothing_k\right) p\left(\varnothing_k \middle| \psi\right) q(\psi) d\varnothing d\psi \\ &= \int p\left(y^* \middle| x^*, \varnothing_k\right)\left(\int p\left(\varnothing_k \middle| \psi\right) q(\psi) d(\psi)\right) d\phi \end{aligned} \tag{30}$$

In the above equation, $p(\Psi | \varnothing_{1:M}) \approx q(\Psi)$.

• Personalized Model Adjustment. The task of training a prediction model $\hat{p}(y|x)$ is given a new and locally (privately) owned training dataset D^p with an unknown distribution $p^p(x, y)$. Existing methods typically rely only on fine-tuning to balance the fitting of the initial federated learning model and the private local data to avoid underfitting and overfitting issues. However, practical solutions are still lacking. In the hierarchical Bayesian federated learning framework proposed in this paper, personalized model adjustment can be regarded as another

posterior probability inference problem with locally owned training data D^p. Predicting the test data x^p is equivalent to conducting Bayesian posterior probability inference. Therefore, personalized model adjustment can be reduced to the task of inferring the Bayesian posterior probability $p_k(\theta| D^p, D_{1:S})$ and $p(\theta| D^p, D_{1:S}, \varnothing_{1:M})$ given a locally owned training dataset D^p and a federated learning model training dataset $D_{1:N}$. Under the hierarchical Bayesian model, the posterior probability can be linked to the federated learning training step $q(\Psi)$.

$$\begin{aligned} & p_k\left(\theta\middle|D^p, D_{1:S}\right) \\ & \approx \int p_k\left(\theta\middle|D^p, \varnothing_k\right) p\left(\varnothing_k\middle|D_{1:S}\right) d\varnothing \\ & \approx \int p\left(\theta\middle|D^p, \varnothing_k\right) q\left(\varnothing_k\right) d\varnothing \end{aligned} \tag{31}$$

$$\begin{aligned} & p\left(\theta\middle|D^p, D_{1:S}, \varnothing_{1:M}\right) \\ & \approx \int p\left(\theta\middle|D^p, \varnothing_k\right) q\left(\varnothing_k\right) p\left(\psi\middle|\varnothing_{1:M}\right) d\varnothing d\psi \\ & \approx \int p\left(\theta\middle|D^p, \varnothing_k\right) q\left(\varnothing_k\right) q\left(\psi\right) d\varnothing d\psi \\ & \approx \int p\left(\theta\middle|D^p, \psi\right) q\left(\psi\right) d\psi \\ & \approx p\left(\theta\middle|D^p, \psi^*\right) \end{aligned} \tag{32}$$

In the above equations, $p(\varnothing_k \mid D_{1:S}) \approx q(\varnothing_k)$ and $p(\Psi \mid \varnothing_{1:M}) \approx q(\Psi)$, furthermore we introduce a tractable variational distribution $f(\theta) \approx p(\theta| D^p, \Psi^*)$ by utilizing variational inference. After going through the typical variational inference process, we obtain the negative ELBO objective for personalized model calibration.

$$\mathrm{KL}\left(f(\theta)\middle\|p\left(\theta\middle|\psi^*\right)\right) + \min_f \mathbb{E}_{f(\theta)}\left[-\log p\left(D^p\middle|\theta\right)\right] \tag{33}$$

The predictive distribution becomes:

$$\begin{aligned} & p\left(y^p\middle|x^p, D^p, D_{1:S}, \varnothing_{1:M}\right) \\ & \approx \frac{1}{MC}\sum_{MC=1}^{MC} p\left(y^p\middle|x^p, \theta^{(MC)}\right), where\ \theta^{(MC)} \sim f(\theta) \end{aligned} \tag{34}$$

Here, the MC (Monte Carlo) symbol is the Monte Carlo sample. The Monte Carlo approach is often used in various optimization scenarios, including optimization applications involving neural networks. The hierarchical Bayesian federated learning framework discussed above demonstrates how the variational inference of hierarchical Bayesian models applies to federated learning problems.

3.3 Hierarchical Bayesian Federated Learning Algorithm Execution

To more comprehensively elaborate the implementation steps of the proposed hierarchical asynchronous update strategy, we demonstrate its operation mode in the form of pseudocode for a cloud server, edge aggregation servers, and terminal-side clients. The cloud server initializes the parameters of the federated model and broadcasts parameter ω_0 to all edge aggregation servers. Then, the edge servers broadcast parameter ω_1 to all terminal clients participating in the federated learning process. The clients locally perform model training in parallel and send the updated parameters ω_{t+1}^k obtained from training to the edge aggregation servers. After receiving the model parameters from all participants, the edge aggregation servers aggregate the model parameters and upload the aggregated edge model parameters to the central cloud server. Finally, the current training round t updates the parameters of each edge server and each client. The federal learning algorithm developed in this article is shown as Algorithm 1 below.

Algorithm 1. Multi-level cooperative hierarchical Bayesian FL algorithm

We define the following hyperparameters:

- N_f ($\leq S$) = The number of terminal-side clients participating in each round.
- T_S = the number of iterations per round for updating the (participating) clients.
- Let Q_S = the number of iterations per round for updating the edge server.
- T_M = the number of iterations per round for updating the (participating) edge serves.
- Let Q_M = the number of iterations per round for updating the cloud server.
- Let η_t = the reciprocal of the learning rate at iteration t.

Initialize the global iteration counter $t = 0$.

for each round **do**

- Select N_f clients uniformly at random from u_t {1, ..., S} without replacement. Let blocks $u_t \in$ {1, ..., S} be the set of the participants ($|u_t| = N_f$).

Terminal clients update:

- **for** each of T iterations,

1. Perform an update for the block u_t. Thus,

$$x^{t+1} := \left[x_{u_t}^{t+1}; x_{-u_t}^t\right], where\ x_{u_t}^{t+1} = x_{u_t}^t - \frac{1}{\eta_t}\nabla_{u_t} f_t\left(x^t\right)$$

where f_t is the mini batch version of f defined on the mini batch data. Note that this update is actually done independently over the participating clients $i \in u_t$ due to the separable objective.

2. $t \leftarrow t + 1$.

end

Edge servers update:

- **for** each of Q_S iterations,

1. Perform update for the index 0. Thus,

$$x^{t+1} := \left[x_0^{t+1}; x_{-0}^t\right], where\ x_0^{t+1} = x_0^t - \frac{1}{\eta_t}\nabla_0 f_t\left(x^t\right)$$

2. $t \leftarrow t + 1$.
end
Cloud server update:
• **for** each of Q_M iterations,
1. Perform update for the index 0. Thus,

$$x^{t+1} := \left[x_0^{t+1}; x_{-0}^{t} \right], where\ x_0^{t+1} = x_0^t - \frac{1}{\eta_t} \nabla_0 f_t\left(x^t\right)$$

2. $t \leftarrow t + 1$.
end

• Training algorithm

Algorithm 2. Training algorithm

Input: Initial parameters L_0 in the variational posterior $q(\Psi, L_0)$.
Output: Training parameters L_0.
For each epoch $r = 1, 2, 3, \ldots, R$ **do:**
1. Sample a Γ subset of participating clients ($|\Gamma| = N_f \leq N$).
2. Amount of edge servers is M.
3. Amount of clients at the lower level of edge server k is S.
4. Cloud sever sends L_0 to all edge servers $k \in M$.
5. Each edge sever sends L_{00} to all the clients at its lower level $i \in S$.
6. **For** each edge server $k \in M$ **in parallel do:**
Solve with L_0 fixed:

$$\min_{\{L_k\}_{k=1}^{M}} \mathbb{E}_{q(\Psi;L_0)}\left[\mathrm{KL}\left(q_k\left(\phi_k; L_k\right) \middle\| p\left(\phi_k \middle| \psi\right)\right)\right]$$

$$+\mathbb{E}_{q_k(\phi_k;L_k)}\left[-\log p\left(U_k \middle| \phi_k\right)\right]$$

The initial data L_k can be sourced either from a previous iteration L_0.
7. **For** each terminal clients $i \in k$ **in parallel do:**
Solve with L_{00} fixed:

$$\min_{\{L_i\}_{i=1}^{S}} \mathbb{E}_{q(\phi_k;L_i)}\left[\mathrm{KL}\left(q_i\left(\theta_i; L_i\right) \middle\| p\left(\theta_i \middle| \phi_k\right)\right)\right]$$

$$+\mathbb{E}_{q_i(\theta_i;L_i)}\left[-\log p\left(D_i \middle| \theta_i\right)\right]$$

The initial data L_i can be sourced either from a previous iteration L_{00} or from local storage L_i if the client has that capability.
8. Each client $i \in S$ sends the updated L_k back to the edge server.
9. Each client $k \in M$ sends the updated L_k back to the cloud server.
10. Upon receiving $\{L_i\} \in S$, the server updates L_{00} by solving (with $\{L_i\} \in S$ fixed):

$$\min_{L_0}\left[\mathrm{KL}\left(q\left(\phi; L_{00}\right) \middle\| p\left(\phi\right)\right)\right]$$

$$-\frac{N}{N_f}\sum\nolimits_{i\in} \mathbb{E}_{q(\theta_i;L_{00})q_i(\theta_i;L_i)}\left[\log p\left(\theta_i \middle| \phi\right)\right]$$

11. Upon receiving $\{L_k\} \in M$, the server updates L_0 by solving (with $\{L_k\} \in M$ fixed):

$$\min_{L_0} \mathrm{KL}\left(q\left(\Psi; L_0\right) \middle\| p\left(\Psi\right)\right)$$

$$-\frac{N}{N_f}\sum\nolimits_{k\in M} \mathbb{E}_{q(\phi_k;L_0)q_i(\phi_k;L_k)}\left[\log p\left(\phi_k \middle| \Psi\right)\right]$$

• Prediction algorithm

Algorithm 3. Prediction algorithm

Global prediction:
Sample

$$\theta^{(s)} \sim \iint p\left(\theta_i \middle| \phi_k\right) q\left(\phi_k; L_{00}\right) p\left(\phi_k \middle| \Psi\right) q\left(\Psi; L_0\right) d\Psi,$$

$$for\ s = 1, 2, 3, \ldots, S.$$

Input: Test input x^*. Trained model L_0 in the variational posterior $q(\Psi, L_0)$.
Output: Predictive distribution $p(y^*|x^*, D_{1:S}, \emptyset_{1:M})$.

Return $p(y^*|x^*, D_{1:S}, \emptyset_{1:M} \approx \frac{1}{S}\sum_{s=1}^{S} p\left(y^* \middle| x^*, \theta^{(s)}\right)$.

Personalization:
• **Estimate the variationa**l density $f(\theta) \approx p(\theta|D^p, \phi^*)$.
• Sample $\theta^{(s)} \sim f(\theta)$, for $s = 1, 2, 3, ..., S$.
Input: Personal training data D^p, test input x^p. Trained model L_0 in the variational posterior $q(\Psi, L_0)$.
Output: Predictive distribution $p(y^p|x^p, D^p, D_{1:N}, \emptyset_{1:M})$.

Return $p(y^p|x^p, D^p, D_{1:N}, \emptyset_{1:M}) \approx \frac{1}{S}\sum_{s=1}^{S} p\left(y^p \middle| x^p, \theta^{(s)}\right)$.

• Cloud server-side algorithm execution

Algorithm 4. Cloud server-side algorithm execution

Input: K, ρ.
Initialize ω_0 and broadcast ω_0 to all edge servers.
m $\leftarrow$ max (k * ρ, 1).
$C_i \leftarrow$ {random set of m edge servers}.
for each edge server $k \in C_i$ **in parallel do:**

$$\omega_{t+1}^{k} \leftarrow \text{Edge server - update}\left(k, \omega_p\right)$$

end

• Edge server-side algorithm execution

Algorithm 5. Edge server-side algorithm execution

Input: K, ρ.
Initialize ω_0 and broadcast ω_0 to all clients.
$m \leftarrow$ max (k * ρ, 1).
$C_i \leftarrow$ {random set of m Client}.
for each edge server $k \in C_i$ **in parallel do:**

$$\omega_{t+1}^{k} \leftarrow \text{Client - update}\left(k, \omega_p\right)$$

end

• Client-side algorithm execution

Algorithm 6. Client-side algorithm execution

Input: K, ω.
Output: ω.
Get newest ω from the server.
Initialize I, M.
for each iteration i=1, 2, 3, ..., S **do:**
$B \leftarrow$ split the dataset into batches of size M.
for *batch b* B **do:**
calculate gradient g_k;
$\omega \leftarrow \omega \leftarrow \eta * g_k$;
end
end
Return ω to the edge server.

4 Performance Analysis for the Framework

4.1 Analysis of the Convergence and Generalization

The proposed hierarchical Bayesian federated learning algorithm is evaluated through two theoretical analyses, which provide insights into the algorithm's convergence and generalization error. (1) Convergence analysis: The developed algorithm, as a special hierarchical block-coordinate optimization algorithm, is theoretically proven to converge to a local optimum. (2) Generalization error analysis: Theoretically, the optimal model trained on test data exhibits predictive capabilities.

• Convergence analysis. We denote the objective function as $f(x)$, where $x = [x_0, x_1, \ldots, x_S]$ corresponds to the variational parameters $x_0 := L_0$, $x_1 := L_1$, $x_2 := L_2$, …, $x_S := L_S$. *Let* $\eta_t = \bar{L} + \sqrt{t}$ for some constant $\bar{L}$, and let $\bar{x}^T = \frac{1}{T}\Sigma_{t=1}^{T} x^t$, where t is the batch iteration counter, x is the iteration at t following the hierarchical Bayesian federated learning algorithm, and $N_f(\leq S)$ is the number of clients participating in each round. For any T, the following formula holds:

$$\mathbb{E}\left[f\left(\bar{x}^T\right) - f\left(x^*\right)\right] \leq \frac{N+N_f}{N_f} \cdot \frac{\frac{\sqrt{T}+\bar{L}}{2} D^2 + R_f^2\sqrt{T}}{T} = O\left(\frac{1}{\sqrt{T}}\right) \tag{35}$$

Here, x^* denotes the local optima, and D and R_f are some constants. $\bar{x}^t$ converges to the expected optimal point x^* at a rate of $O(1/\sqrt{t})$, and this rate approaches the rate of the standard stochastic gradient descent (SGD) algorithm.

• Generalization error analysis. Given $d^2(P_{\theta_i}, P^i)$ as the square of the expected Hellinger distance between the true class distribution $P^i(y|x)$ and the model distribution $P_{\theta_i}(y|x)$, the optimal solution $(\{q_i^*(\theta_i)\}_{i=1}^N, q^*(\phi_k))$ of the optimization problem satisfies the following criteria:

$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{q_i^*(\theta_i)}\left[d^2\left(P_{\theta_i}, P^i\right)\right] \leq O\left(\frac{1}{n}\right) + C\cdot\varepsilon_n^2 + C'\left(r_n + \frac{1}{N}\sum_{i=1}^{N}\lambda_i^*\right) \tag{36}$$

Here, C and C' are constants such that ≥ 0, $\lambda_i^* = \min_{\theta\in\Theta}||f_\theta - f^i||_\infty^2$, and r_n, $\epsilon_n \to 0$ when the amount of training data $n \to \infty$, which means that the optimal solution is asymptotically optimal.

4.2 Analysis of Transmission Optimization in the Framework

The mobility of nodes and the instability of communication links also challenge the training of the hierarchical asynchronous federated learning model. Combining federated learning, homomorphic encryption, and secure multi-party computation technologies can enhance privacy protection. However, at the same time, it will also cause large communication transmission overhead for model training [18-21]. In edge networks, there are thousands or hundreds of users participating in model training, and a large number of nodes and servers require continuous model update transmission, making network transmission efficiency a key bottleneck for model training convergence [22-25]. In the whole model training process, the local model update and model update upload of a large number of participating nodes will cause excessive network transmission overhead, and the privacy technologies such as homomorphic encryption and secure multi-party computation applied in the privacy security research field mentioned above will cause large network transmission overhead for model training.

Therefore, it is necessary to study network transmission optimization technology to reduce network transmission overhead in model training in federated learning research. Researchers have studied the typical problems in federated learning, such as massive participation nodes, limited network bandwidth, data heterogeneity, heterogeneous computing power, node collaboration selfishness, improving efficiency, reducing energy consumption, and improving model accuracy. The main solutions include node selection, enhanced local computing, reduced number of model updates uploaded, model compression, decentralized training, parameter aggregation transmission oriented, and each method focuses on a particular problem. We present the relevant research information in Table 1.

The overall goal of local model updates is to use local datasets for machine learning model training on the distributed node side of the terminal. However, performing local model calculations in a single communication round and sharing local model updates on edge servers can result in low communication efficiency. A transmission optimization method to address this issue is to enhance local computing to reduce communication frequency, thereby reducing the number of communication rounds required for model training.

The Bayesian hierarchical asynchronous federated learning framework described in this paper achieves transmission optimization based on enhanced local computing. In the framework, participating nodes achieve local model accuracy through multiple iterations of local model calculation and then upload it to the edge server; The edge server undergoes multiple iterations of edge model aggregation to achieve edge model accuracy and then uploads it to the cloud server for a global model aggregation. Before the participating nodes upload the local model updates to the edge server, they must carry out multiple rounds of local iterations to reduce the number of communications with the server. The edge aggregation model is subject to multiple edge iterations

before uploading to the cloud server, thus reducing communication overhead, model training time, and energy consumption of the Edge device. Furthermore, the hierarchical Bayesian algorithm in this paper can correct the 'drift' introduced in the local model update of equipment when the local data is not Independent and identically distributed random variables and form a personalized local model.

5 Simulation Results

We evaluate the sensitivity of our models to critical hyperparameter K (Figure 3). Table 2 shows the parameter settings as follows.

In summary, this paper proposes a hierarchical Bayesian federated learning method, thus realizing local model personalization, solving the problem concerning non-IID data in federated learning, and effectively reducing communication costs through multilevel hierarchical model parameter aggregation and resource scheduling.

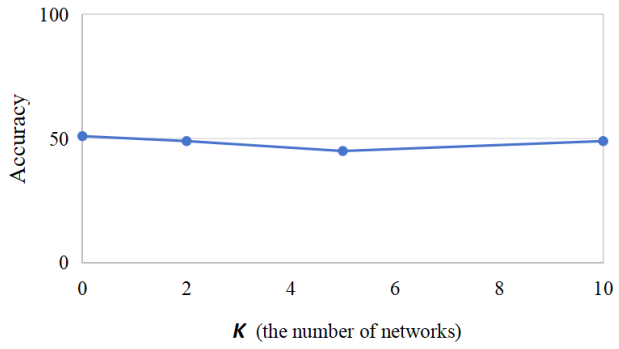
In this paper, the developed model obtained by local model personalization and hierarchical model parameter aggregation processing exhibits outstanding robustness, effectively addressing different types of statistical heterogeneity and communication bottleneck problems. The progress of building a hierarchical asynchronous federated learning model enables the participating parties to achieve efficient personalized local model processing at the edge side while solving the issue of non-IID data. Therefore, this article addresses the non-IID data and communication bottlenecks issues encountered in federated learning.

Table 1. Different optimization method

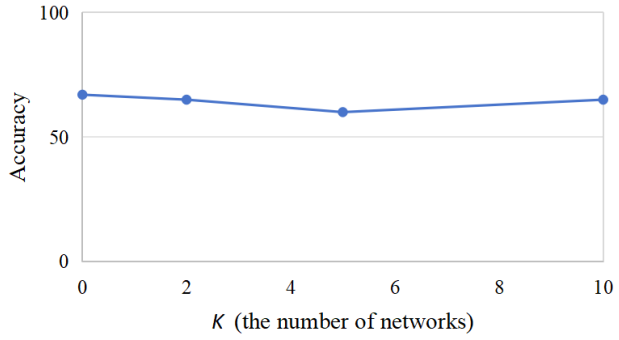
Optimization method	Faced challenge					Optimization objective		
	Massive participation nodes	Limited network bandwidth	Data heterogeneity	Heterogeneous computing power	Node collaboration selfishness	Improving efficiency	Reduce energy consumption	Improve model accuracy
Node selection	★★★ ★★★ ★	★★★ ★★	★	★★★	★★★ ★★	★★★★★★★	★	★★★
Enhance local computing		★★★	★			★★★		★★
Reduce the number of model updates uploaded		★★		★		★★		★★
Model compression	★★★ ★★★ ★	★				★★★★★★★		★★★
Decentralized training	★	★				★	★	★
Parameter aggregation transmission oriented	★★★	★				★★★		★

Table 2. Parametric configuration for simulation

Parameter	Numeric	Describe
K	2/5/10	The number of networks
B	1MHZ	Bandwidth
δ_b	5MB	Block size
n_0	-125dBM/HZ	Noise power
D	2MB	Training sample size
c_k	15cycles/bits	Frequency required for training completion
p_k	30W	Transmission power
$f_k^{\max}$	0.5GHz-1.0GHz	Maximum computing power
s	100	Sharding parameter
f	0.1	Participating client fraction
τ	1	Number of local epochs



(a) Global prediction (s = 100)



(b) Personalisation (s = 100)

Figure 3. Hyperparameter sensitivity analysis

Other researchers are also endeavoring to incorporate Bayesian theory into federated learning modeling [26-27] to address the formidable issue of non-IID data. By incorporating the distribution of the model weight parameters into the global model's posterior probability calculation, we can derive the global model's parameters from the posterior probability calculations of the local models. However, these methods regard the model weight parameters as random variables shared by all clients, and they still face challenges in terms of handling local models when each participant receives non-IID distributed data from each client [28].

For instance, FedPA [9] aims to establish the product-of-experts decomposition to allow client-wise inference. However, this decomposition does not hold in general unless a strong assumption of uninformative prior is made. FedBE (Bayesian Ensemble) [26] aims to build the global posterior distribution from the individual posteriors. pFedBayes [27] can be seen as an implicit regularisation-based method to approximate from individual posteriors. To this end, they introduce the so-called global distribution $\omega(\theta)$, which essentially serves as a regulariser to prevent local posteriors from deviating from it. The introduction of $\omega(\theta)$ and its update strategy appears to be a hybrid treatment rather than solely Bayesian perspective. FedPop [29] has a similar hierarchical Bayesian model structure as ours, but their model is limited to a linear deterministic model for the shared variate.

6 Conclusion

In this article, we propose a novel research method, a hierarchical Bayesian federated learning framework, with the aim of solving the communication and non-IID data problems that are commonly encountered in federated learning. Through this framework, we can achieve collaborative machine learning model training for distributed nodes, use global variables to control the random local model variables of the participants and achieve hierarchical collaborative model construction to optimize communication. In addition, we adopt the variational inference method to effectively solve the non-IID data problem of federated learning for each participant. This framework exhibits strong robustness in terms of coping with different types of statistical heterogeneity and realizes personalized local models. Although some aspects of this article still need to be improved, we plan to continue our in-depth research in future work and release relevant results in subsequent papers. Such as, in the process of model aggregation, when the central server receives model information uploaded by all participants, how to evaluate the participants' model information, give higher weights to parameter information that is more valuable and conducive to joint model training, improve model convergence rate, reduce communication between local models with smaller contributions and global models to enhance effective communication efficiency between different levels. That is one of the subsequent research focuses of this article's related research content.

Acknowledgments

The work is funded by the National Key R&D Program of China (2022YFB2403900).

References

[1] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, M. Guizani, A Survey on Federated Learning: The Journey From Centralized to Distributed On-Site Learning and Beyond, *IEEE Internet of Things Journal*, Vol. 8, No. 7, pp. 5476-5497, April1, 2021. https://doi.org/10.1109/JIOT.2020.3030072

[2] A. Sheth, J. A. Larson, Federated database systems for managing distributed, heterogeneous, and autonomous databases, *ACM Computing Surveys*, Vol. 22, No. 3, pp. 183-236, September, 1990. https://doi.org/10.1145/96602.96604

[3] R. Caruana, Multitask Learning, *Machine Learning*, Vol. 28, No. 1, pp. 41-75, July, 1997. https://doi.org/10.1023/A:1007379606734

[4] Y. Zhang, Q. Yang, An Overview of Multi-Task Learning, *National Science Review* (*NSR*), Vol. 5, No. 1, pp. 30-43, January, 2018. https://doi.org/10.1093/nsr/nwx105

[5] Q. Yang, Y. Liu, T. J. Chen, Y. X. Tong, Federated Machine Learning: Concept and Applications, *ACM Transactions on*

Intelligent Systems and Technology (TIST), Vol. 10, No. 2, pp. 1-19, March, 2019.
https://doi.org/10.1145/3298981

[6] X. Q. Zhang, Y. W. Liu, J. X. Liu, Y. N. Han, An Overview of Federated Learning in Edge Intelligence, *Jisuanji Yanjiu yu Fazhan/Journal of Computer Research and Development*, Vol. 60, No. 6, pp. 1276-1295, June, 2023.
https://dx.doi.org/10.7544/issn1000-1239.202111100

[7] X. Li, K. X. Huang, W. H. Yang, S. S. Wang, Z. H. Zhang, On the Convergence of FedAvg on Non-IID Data, 2020 *8th International Conference on Learning Representations (ICLR*), Addis Ababa, Ethiopia, 2020, pp. 1-26.

[8] B. J. Wei, J. Li, Y. Liu, W. P. Wang, Non-IID Federated Learning With Sharper Risk Bound, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 35, No. 5, pp. 6906-6917, May, 2024.
https://doi.org/10.1109/TNNLS.2022.3213187

[9] M. Al-Shedivat, J. Gillenwater, E. Xing, A. Rostamizadeh, Federated Learning via Posterior Averaging: A New Perspective and Practical Algorithms, *2021 9th International Conference on Learning Representations (ICLR)*, Virtual Event, 2021, pp. 1-23.

[10] J. Bernardo, A. Smith, *Bayesian theory*, John Wiley & Sons Ltd., Chichester, 1994.

[11] G. F. Cooper, E. Herskovits, A Bayesian Method for the Induction of Probabilistic Networks from Data, *Machine Learning*, Vol. 9, No. 4, pp. 309-347, October, 1992.
https://doi.org/10.1007/BF00994110

[12] D. M. Blei, A. Kucukelbir, J. D. McAuliffe, Variational inference: A review for statisticians, *Journal of the American Statistical Association*, Vol. 112, No. 518, pp. 859-877, July, 2017.
https://doi.org/10.1080/01621459.2017.1285773

[13] J. M. Hofman, C. H. Wiggins, Bayesian approach to network modularity, *Physical Review Letters*, Vol. 100, No. 25, Article No. 258701, June, 2008.
https://doi.org/10.1103/PhysRevLett.100.258701

[14] V. Kulkarni, M. Kulkarni, A. Pant, Survey of Personalization Techniques for Federated Learning, *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, London, UK, 2020, pp. 794-797.
https://doi.org/10.1109/WorldS450073.2020.9210355

[15] S. J. Pan, Q. Yang, A survey on transfer learning, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 10, pp. 1345-1359, October, 2010.
https://doi.org/10.1109/TKDE.2009.191

[16] F. L. Da Silva, A. H. R. Costa, A survey on transfer learning for multiagent reinforcement learning systems, *Journal of Artificial Intelligence Research*, Vol. 64, No. 1, pp. 645-703, January, 2019.
https://doi.org/10.1613/jair.1.11396

[17] C. Zhang, H. Bai, Y. G. Zhang, X. Y. Niu, B. Yu, Y. Gao, Y. Xie, Federated Multitask Learning for HyperFace, *IEEE Transactions on Artificial Intelligence*, Vol. 3, No. 5, pp. 788-797, October, 2022.
https://doi.org/10.1109/TAI.2021.3133816

[18] N. Hudson, M. J. Hossain, M. Hosseinzadeh, H. Khamfroush, M. Rahnamay-Naeini, N. Ghani, A Framework for Edge Intelligent Smart Distribution Grids via Federated Learning, *2021 International Conference on Computer Communications and Networks (ICCCN*), Athens, Greece, 2021, pp. 1-9.
https://doi.org/10.1109/ICCCN52240.2021.9522360

[19] N. Kotsehub, M. Baughman, R. Chard, N. Hudson, P. Patros, O. Rana, I. Foster, K. Chard, FLoX: Federated Learning with FaaS at the Edge, *2022 IEEE 18th International Conference on e-Science (e-Science)*, Salt Lake City, UT, USA, 2022, pp. 11-20.
https://doi.org/10.1109/eScience55777.2022.00016

[20] M. Baughman, N. Hudson, I. Foster, K. Chard, Balancing Federated Learning Trade-Offs for Heterogeneous Environments, *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, Atlanta, GA, USA, 2023, pp. 404-407.
https://doi.org/10.1109/PerComWorksho
ps56833.2023.10150228

[21] M. Baughman, I. Foster, K. Chard, Exploring Tradeoffs in Federated Learning on Serverless Computing Architectures, *2022 IEEE 18th International Conference on e-Science (e-Science)*, Salt Lake City, UT, USA, 2022, pp. 433-434.
https://doi.org/10.1109/eScience55777.2022.00074

[22] M. Aledhari, R. Razzak, R. M. Parizi, F. Saeed, Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications, *IEEE Access*, Vol. 8, pp. 140699-140725, July, 2020.
https://doi.org/10.1109/ACCESS.2020.3013541

[23] Z. K. Taha, C. T. Yaw, S. P. Koh, S. K. Tiong, K. Kadirgama, F. Benedict, J. D. Tan, Y. A/L Balasubramaniam, A Survey of Federated Learning From Data Perspective in the Healthcare Domain: Challenges, Methods, and Future Directions, *IEEE Access*, Vol. 11, pp. 45711-45735, 2023.
https://doi.org/10.1109/ACCESS.2023.3267964

[24] M. Seol, T. Kim, Performance Enhancement in Federated Learning by Reducing Class Imbalance of Non-IID Data, *Sensors*, Vol. 23, No. 3, Article No. 1152, February, 2023.
https://doi.org/10.3390/s23031152

[25] A. A. Al-Saedi, V. Boeva, E. Casalicchio, FedCO: Communication-Efficient Federated Learning via Clustering Optimization, *Future Internet*, Vol. 14, No. 12, Article No. 377, December, 2022.
https://doi.org/10.3390/fi14120377

[26] H. Y. Chen, W. L. Chao, FedBE: Making Bayesian Model Ensemble Applicable to Federated Learning, *2021 9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria, 2021, pp. 1-21.

[27] J. Zhu, X. Ma, M. B. Blaschko, Confidence-Aware Personalized Federated Learning via Variational Expectation Maximization, *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 2023, pp. 24542-24551.
https://doi.org/10.1109/CVPR52729.2023.02351

[28] C. Briggs, Z. Fan, P. Andras, Federated learning with hierarchical clustering of local updates to improve training on non-IID data, *2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, 2020, pp. 1-9.
https://doi.org/10.1109/IJCNN48605.2020.9207469

[29] P. Wu, T. Imbiriba, J. Park, S. Kim, P. Closas, Personalized Federated Learning over non-IID Data for Indoor Localization, *2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Lucca, Italy, 2021, pp. 421-425.
https://doi.org/10.1109/SPAWC51858.2021.9593115

Biographies

Aijun Wen is the Vice President of the Information and Communication Research Institute of State Grid Information and Communication Industry Group Co., Ltd. Having a profound knowledge of software engineering, proficient in information technology-related knowledge, leading multiple large-scale software development and engineering implementation work, accumulating 11 scientific and technological progress awards in related technical fields, and numerous honorary titles such as the Top 10 Model and Excellent Union Cadre of China Electric Power Research Institute.

Yunxi Fu received a Ph.D. from the University of Science and Technology of China in 2018. She is a senior engineer at the Information and Communication Research Institute of State Grid Information and Communication Industry Group Co., Ltd. She has been engaged in various fields such as network information security, data security, privacy computing, federated learning, and information system technology research for a long time.

Zesan Liu is deputy director of the Information Technology Research and Development Center of the Information and Communication Research Institute of State Grid Information and Communication Industry Group Co., Ltd. Engaged in core technology research and development in comprehensive energy digitization, essential common platforms, power grid informatization, information security, etc. for a long time, participated in writing one monograph, published more than six core journal/EI/SCI search papers, authorized five invention patents, and accepted more than ten patents.

Zhenya Wang received a Ph.D. degree in Computer Science from the Beijing University of Posts and Telecommunications, China in 2023. He is a Middle Engineer at State Grid Information and Telecommunication Group CO., LTD. His research interests include privacy computing and a new generation of power systems.

Wenjuan Zhang received the master's degree from the School of Computer Science and Technology, Beijing Institute of Technology, in 2009. I am a system architect at the Information Industry Research Institute of State Grid Information and Communication Industry Group. Engaged in long-term work in information system development and Java microservice architecture design.