International Journal of Machine Learning and Cybernetics (2023) 14:3783–3798
https://doi.org/10.1007/s13042-023-01864-z

ORIGINAL ARTICLE

Personalized federated learning based on multi-head attention algorithm

Shanshan Jiang[1] · Meixia Lu[2] · Kai Hu[2] · Jiasheng Wu[2] · Yaogen Li[2] · Liguo Weng[2] · Min Xia[2] · Haifeng Lin[3]

Received: 20 July 2022 / Accepted: 11 May 2023 / Published online: 27 May 2023

Abstract
Federated Learning (FL) is an algorithm for the encrypted exchange of model parameters while ensuring the independence of participants. Classic federated learning does not take into account the correlation between features, nor does it take into account the data differences caused by the reasonable personalization of each client. Therefore, this paper proposes a personalized federated learning algorithm based on a multi-head attention mechanism. First, in order to improve the personalization of local models, attention mechanism is used to capture the relevance of local features. Then, when aggregating local models, the weight λ is generated for local models based on the differences between models, and finally aggregate them into a new global model. Finally, the multi-head attention is proposed to calculate the importance score of the global model parameters on the current local model, and assign it to the local model as the attention coefficient, so as to realize personalized federated learning. Through experiments on MNIST, SVHN and STL10 datasets, the validity of Personalized Federated Learning is verified, and the rationality of hyperparameter setting is discussed through visualizing results.

Keywords Federated learning · Multi-head attention mechanism · Personalize

1 Introduction

Federated learning can ensure the security of data owned by each participant in distributed learning. Federated Learning (FL) has become a distributed collaborative AI method. By allowing AI training on distributed IoT devices without data sharing, it can make many intelligent IoT applications possible [1–3]. In 2016, Google first proposed a federated learning algorithm for mobile terminals [4]. Each client trains a local model and then aggregates it to obtain a global model. The process of exchanging model information between clients is carefully designed so no client can learn the private data content of other clients. The data sources seem to be integrated when the global model is obtained. This is the core idea of Federated Learning. After the concept of federated learning was proposed by Google, McMahan et al. proposed a practical method for the federated learning of deep networks based on iterative model averaging in 2017 [5]. This algorithm uses relatively small communication rounds to train a high-quality model, which is the classic federated averaging algorithm. In the later period, many federated learning algorithms were further developed based on McMahan's work, and many excellent algorithms were formed [6, 7].

In commercial applications, not only data confidentiality is very important, personalization is also an important research content. For example, the core of personalized recommendation is to accurately locate the content that users are interested in. Therefore, extracting the user's hidden preferences from limited user information becomes the key to measuring the quality of a recommendation algorithm, which is also the main issue of research. Existing recommendation systems usually directly perform unified processing on the user's historical browsing records. It ignores the difference between users' long-term preferences and short-term preferences, which will directly affect the accuracy of user

✉ Shanshan Jiang
jss@nuist.edu.cn

[1] School of management Science and Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China

[2] Jiangsu Provincial Collaborative Innovation Center for Atmospheric Environment and Equipment Technology, Nanjing University of Information Science and Technology, Nanjing 210044, China

[3] College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China

behavior feature extraction [8]. In the field of personalized recommendation, people have proposed a recommendation algorithm based on deep learning [9] and a recommendation model based on self-attention mechanism [10], and they have achieved certain results. However, users have a variety of personalization problems. These personalization problems are intertwined with each other and affect the final result in both time and space. Therefore, only a single attention cannot be used to accurately solve the problem. In the commercial field, people need an FL that can mine multiple personalities and keep data confidential.

From the perspective of learning model, federated learning mainly includes methods based on traditional machine learning (such as Gradient Boosting Decision Tree, Extreme Gradient Boosting, support vector machine) and methods based on deep learning (such as Convolution Neural Networks, Recurrent Neural Networks). Federated learning algorithm based on traditional machine learning refers to the implementation of classical machine learning algorithm under the framework of federated learning. Its advantages are less computation, but its disadvantages are the following problems in practical application scenarios: when the data volume is large, the algorithm has poor scalability and adaptability, and the algorithm is not easy to convert. Compared with machine learning, deep learning has the advantages of strong scalability, good adaptability and easy conversion when the data volume is large [11, 12]. It is a powerful method that can automatically learn feature representation from data. The existing research direction of federated learning is mainly to improve the performance of statistics, solve the problem of small data volume, and improve security to protect data privacy [13, 14].

At present, among the many directions of federated learning [15–19], there are few types of research on personalization. In 2020, Fallah et al. proposed the Per-FedAvg algorithm [20], which uses meta-learning [21] to achieve personalized differential training of federated learning, introduces a Model Agnostic Meta Learning (MAML) algorithm, and treats client [22] training as a task. The global model finds a suitable initialization parameter during training for each client. In 2021, Tan et al. proposed the concept of a personalized federated learning [23]. Federated learning applications often face the heterogeneity of data distribution and device functions among data owners, thus accelerating the development of the personalized federation. The obstacles to personalized federation under the setting of federated learning are discussed in the paper. In addition, a unique classification of personalized federation technology into data-based and model-based methods is proposed. The existing personalized federated learning has slow convergence and poor performance on heterogeneous (non-IID) data, and the global model lacks personalization for local tasks or local data sets. This paper improves the local data set or task personalization task according to the global model, and gives different personalized weights to the personalization between different clients. The global model can better adapt to the personalized tasks or local data sets of the local model.

To improve the reasonable personalized degree of the local model, and reduce the impact of the data difference caused by the reasonable personalization of the local model on the global model when the model is aggregated, a personalized federated learning algorithm based on multi-head attention mechanism is proposed in this work. While mining the relevance of features, it also increases the reasonable personalized proportion of local models. The contributions of this article are as follows:

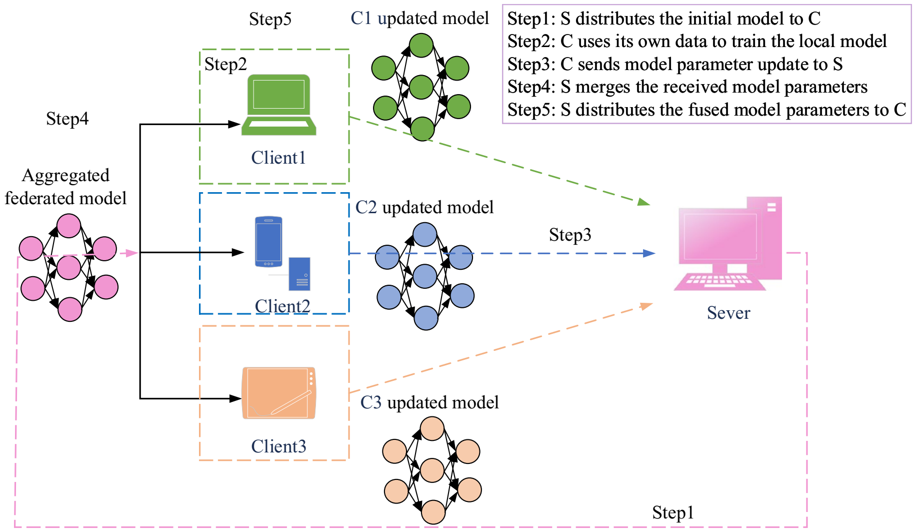
(1) In the local model, improving the reasonable personalized degree of the local model can increase the model's performance. Therefore, a multi-head attention mechanism is introduced to capture the correlation between local features and increase the personalized degree of local model parameters. It can improve the performance of local models.
(2) In the global model, we design a fusion framework of federated learning suitable for a multi-head attention mechanisms. The framework first uses random sampling of 30% of the data and uses the federated averaging model to train to obtain the global model on the server. Secondly, for the remaining 70% of the data, on the basis of the federated averaging model, we calculate the distance between current local model parameters and pre-processed global model parameters and obtain the differences between the models to formulate the personalization weight λ for the local models. Then the weighted average is used to obtain the global model.

This paper is organized as follows: The second part introduces the relevant background information of the research content; the third part details the specific content of a personalized federated learning algorithm; the fourth part shows the performance of this algorithm through a large number of experiments, and the results are discussed; the fifth part summarizes the work.

2 Background

2.1 The basic structure of federated learning

Figure 1 is a federated learning architecture that includes a coordinator. In this scenario, the coordinator is an aggregation server S, which can send the initial random model to each client C. The clients use their respective models for training and send the model weight updates to the aggregation server. Then, the aggregation server aggregates the



Step1: S distributes the initial model to C
Step2: C uses its own data to train the local model
Step3: C sends model parameter update to S
Step4: S merges the received model parameters
Step5: S distributes the fused model parameters to C

Fig. 1 Federated learning system example: client–server architecture

model received from the client and sends the aggregated model updates back to the clients. This process will be repeated until the model converges, the maximum number of iterations is reached, or the maximum training time is reached. Under this architecture, the original data of the client is always stored locally, which can protect user privacy and data security [24].

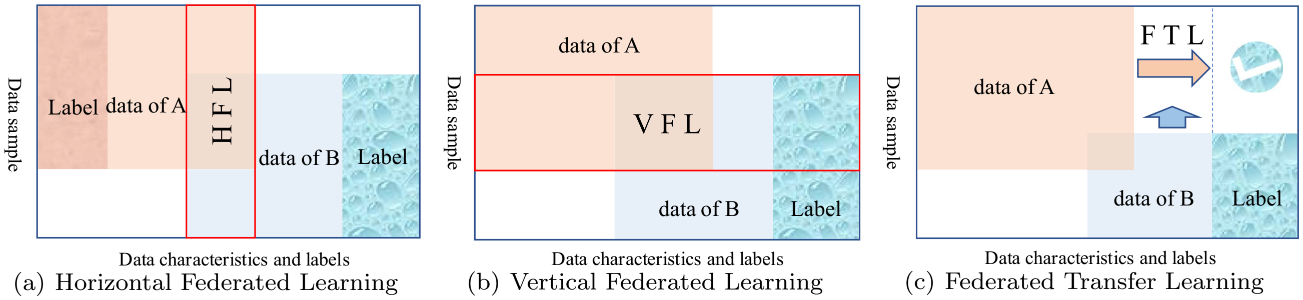
2.2 The way of federated learning

The feature space and sample ID space of the data owned by different clients may be different. According to the distribution of the data feature space and sample ID space of the training data among different participants, federated learnings are divided into Horizontal Federated Learning (HFL), Vertical Federated Learning (VHL), and Federated Transfer Learning (FLT). Taking a federated learning scenario with two participants as an example, Fig. 2 shows the definitions of three types of federated learning. HFL [25] is suitable for federated learning participants whose data has overlapping data features, VHL [4] is suitable for federated learning clients whose training data has overlapping data samples, and FLT [9] is suitable for situations where the client's data samples and data features overlap very slightly.

Horizontal federated learning is suitable for scenarios where the client's dataset has the same feature space but different sample spaces and can be easily used to build applications supported by a large number of mobile devices. In these scenarios, the federation's goals apply to different user groups. However, in many practical scenarios, the clients of federated learning are organizations or institutions that have the same user group. These institutions collect different data characteristics for the same group to achieve different business goals. Therefore, the vertical federated learning method is the most suitable for actual application scenarios, and we choose the vertical federated learning method in this work.

2.3 Federated averaging algorithm

In the classic federated learning algorithm, the federated averaging algorithm is generally used for model training. The federated averaging algorithm is mainly model averaging. In 2017, Mcmahan et al. proposed an equivalent federated model training method [5]. In this work, each client



(a) Horizontal Federated Learning (b) Vertical Federated Learning (c) Federated Transfer Learning

Fig. 2 The way of federated learning

locally performs stochastic gradient descent on the existing model parameters $\overline{\omega}_t$ using local data, the updated model parameter $\omega_{t+1}^{(p)}$ is sent to the server, the server aggregates the received model parameters, that is, uses a weighted average of the received model parameters, and the updated parameters $\overline{\omega}_{t+1}$ is sent to each client. This method is called model averaging [26]. Finally, the server checks the model parameters, and if it converges, the server sends a signal to each participant to stop the model training.

$$\omega_{t+1}^{(p)} = \overline{\omega}_t - \eta \nabla_p, \tag{1}$$

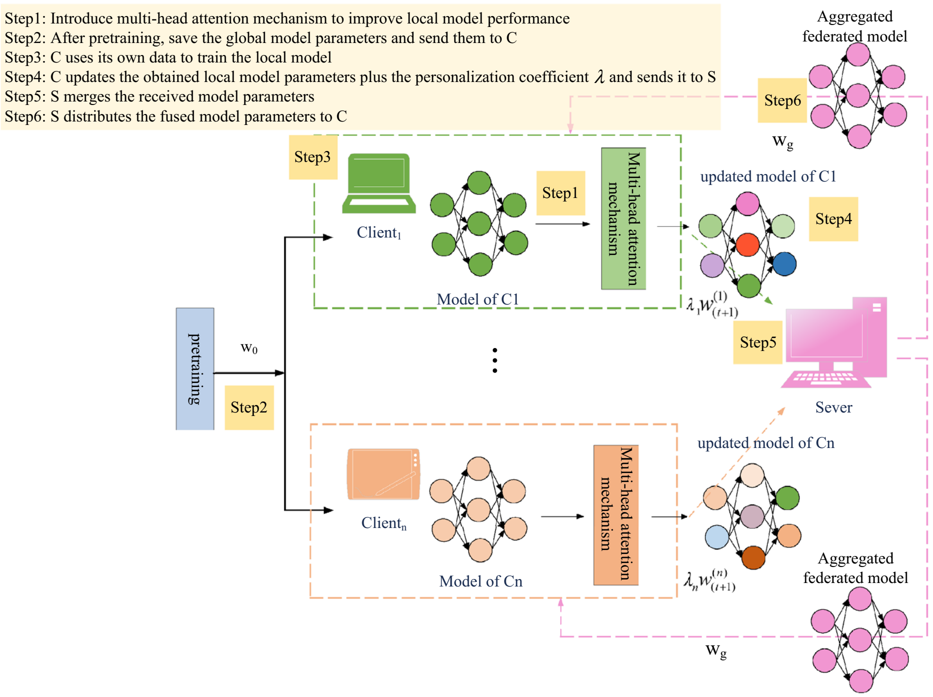
$$\overline{\omega}_{t+1} = \sum_{p=1}^{P} \frac{n_p}{n} \omega_{t+1}^{(p)}, \tag{2}$$

where η is the learning rate, ∇_p is the local gradient update of the p_{th} participant, n_p is the local data volume of the p^{th} participant, n is the local data volume of all participants, $\omega_{t+1}^{(p)}$ are the parameters of the local model of the p^{th} participant at this time, and $\overline{\omega}_{t+1}$ are the aggregated global model parameters. However, when applying the federated averaging algorithm to federated learning based on deep learning, it did not consider the impact of the reasonable personalization degree of the local model on the model performance, nor did it consider the impact of the data difference brought by the reasonable personalization of the local model on the global model during model aggregation. Therefore, this paper proposes a personalized federated learning algorithm.

3 Personalized federated learning algorithm

In the business field, people not only need a guarantee for data security but also need to be able to further mine personalized content to bring value. In the field of personalization, scholars have found that users have more than one personalization, such as long-term personalities, short-term personalities in time, and personalities under different conditions in space. Therefore, the single attention mechanism in the literature [10, 27] cannot meet this complex demand. The attention mechanism can increase attention to the correlation between data features. Scholars define this association as personalization, which can improve accuracy. Therefore, the multi-head attention mechanism can focus on the various associations between data features from many aspects. Based on the traditional federated average algorithm, this paper introduces a multi-headed attention mechanism, we hope to improve the model's ability to mine user personalization under the framework of federated learning, and achieve greater commercial value. The framework of this algorithm is shown in Fig. 3. The specific algorithm is shown in Algorithm 1.

Fig. 3 Overall framework of personalized federated learning


Step1: Introduce multi-head attention mechanism to improve local model performance
Step2: After pretraining, save the global model parameters and send them to C
Step3: C uses its own data to train the local model
Step4: C updates the obtained local model parameters plus the personalization coefficient λ and sends it to S
Step5: S merges the received model parameters
Step6: S distributes the fused model parameters to C

Aggregated federated model
Step6
W_g
Step3
Client$_1$
Step1
Multi-head attention mechanism
updated model of C1
Step4
Model of C1
$\lambda_1 W_{(t+1)}^{(1)}$
Step5
pretraining
W_0
Step2
Sever
updated model of Cn
Client$_n$
Multi-head attention mechanism
Model of Cn
$\lambda_n W_{(t+1)}^{(n)}$
Aggregated federated model
W_g

```
Algorithm 1: M-FedAvg
    Pretrain the local model:
1       Input:Datasets $D_p$
2       Output:pre-trained glob model parameters $\omega_0$
3       calculate the coefficient $\lambda_p$ (formula 9)
    Global model optimization:
    \\the number P of participants in FL,
    datasets $D_p$, the fraction of L of clients C is 0.1
4       Initialize the global model parameter
        $\overline{\omega}_{t+1} = \sum_{p=1}^{P} \frac{n_p}{n}(\lambda_p \omega_{t+1}^{(p)})$ (equation 11)
5       for each round $t = 0, 1, \ldots$ do
6           $m =$ (random set of max$(L \cdot P, 1)$ clients)
7           for each client $p \epsilon m$ in parallel do
8               $\omega_g^{(p)} = ParticipantUpdate(p, \omega_{t+1}^{(p)})$
9               $\omega_g = \sum_{p=1}^{P} \frac{n_p}{n} \omega_g^{(p)}$
10          end for
11          Information distribution:pass $\omega_g$ back to the local model
12      end for
    Local participants update:
    \\local model parameters $\omega_{t+1}^{(p)}$,datasets $D_p$
13      batches = (data $D_p$ split into batches of size B)
14      Download the newest global model optimization parameter $\omega_g$
15      to the current local model optimization $\omega_{t+1}^{(p)}$
16      for each local epoch i from 1 to E do
17          for batch b in batches do
18              Information distribution:
19              $\omega \leftarrow$ Concat$(head_1, head_2, ..., head_n)\omega$
20              $\omega = \omega - \eta \nabla \ell(\omega; b)$
21      Return $\omega_{t+1}^{(p)}$ to step 5
22      end for
```

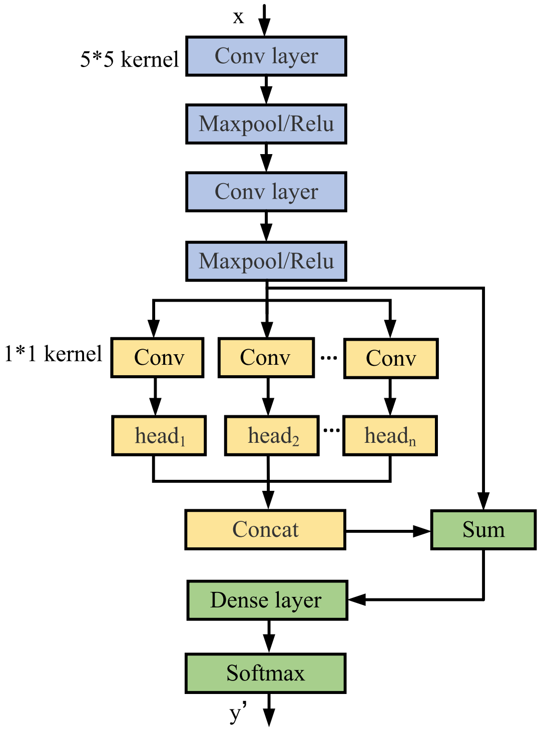


Fig. 5 Local model multi-head attention mechanism module flowchart

3.1 Local model multi-head attention mechanism

In the algorithm of this paper, there can be multiple models of local models, such as SVM, CNN and other algorithms combined with multi-head attention mechanism. The architecture of CNN combined with a multi-head attention mechanism is shown in Fig. 4. The specific process is shown in Fig. 5.

As shown in Fig. 4, for a given feature map input $I\epsilon R^{(C\times H\times W)}$, this article uses a CNN to extract image features [28, 29]. It first passes through the convolutional layer and convolves the image with a 5×5 convolution kernel. The convolutional layer can preserve the input shape so that the correlation of the pixels of the image in both the height and width directions may be effectively identified [30–33]. At the same time, the same convolution kernel and the input of different positions are repeatedly calculated through the sliding window to avoid the parameter size being too large [34–36]. After that, it passes through the excitation layer, performs a nonlinear mapping on the output of the convolutional layer, and then passes through the pooling layer to down-sample the previous layer, which can reduce the size of the previous

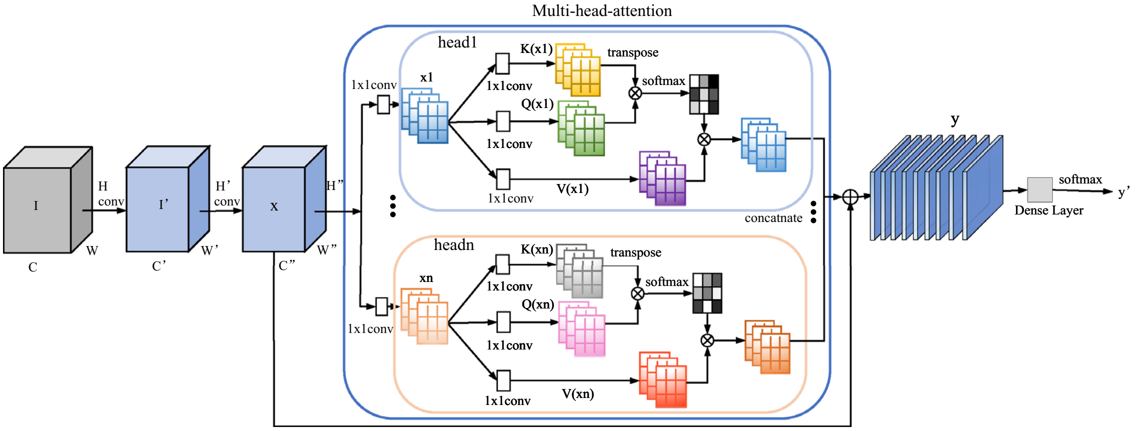


Fig. 4 Local model multi-head attention mechanism architecture diagram

layer, thereby reducing the amount of calculation [37]. We get the feature map $I^{'}\epsilon R^{C^{'}\times H^{'}\times W^{'}}$ and repeat the operation. The number of output channels of the first convolutional layer is 6, and the number of output channels of the second convolutional layer is increased to 16. Since the second convolutional layer has a smaller input height and width than the first, increasing the output channel makes the parameter sizes of the two convolutional layers similar, we can get the feature map $x^{'}\epsilon R^{C^{\varepsilon}\times H^{\varepsilon}\times W^{\varepsilon}}$. Since 3 parallel attention heads are used in this article, a 1×1 convolution kernel with a number of 3 is added to obtain image features x_1, x_2, x_3. Then the image features of $x_1\epsilon R^{C^{\varepsilon}\times N}$ from the previous convolutional layer are transformed into two feature spaces K, Q to calculate attention, where $K(x_1) = W_k x_1, Q(x_1) = W_q x_1$. Where C^{ε} is the number of channels, and N is the number of feature positions of the feature x_1 from the previous convolutional layer.

$$\beta_{j,i}^{(1)} = \frac{exp(s_{ij}^{(1)})}{\sum_{i=1}^{N} exp(s_{ij}^{(1)})}, \tag{3}$$

$$s_{ij}^{(1)} = K(x_i^{(1)})^T Q(x_j^{(1)}), \tag{4}$$

where $\beta_{j,i}^{(1)}$ indicates the extent to which the model attends to the i^{th} location when synthesizing the j^{th} region of x_1. The output of the attention layer $o^{(1)} = (o_1^{(1)}, o_2^{(1)}, \cdots, o_j^{(1)}, \cdots, o_N^{(1)})\epsilon R^{C^{\varepsilon}\times N}$.

$$o_j^{(1)} = \sum_{i=1}^{N} \beta_{j,i}^{(1)} V(x_i^{(1)}), \tag{5}$$

$$V(x_i^{(1)}) = W_v x_i^{(1)}, \tag{6}$$

where $W_k\epsilon R^{\overline{C^{\varepsilon}}\times C^{\varepsilon}}$, $W_q\epsilon R^{\overline{C^{\varepsilon}}\times C^{\varepsilon}}$, $W_v\epsilon R^{\overline{C^{\varepsilon}}\times C^{\varepsilon}}$, are the learned weight matrices, which are implemented as 1×1 convolutions. Since We did not notice any significant performance decrease when reducing the channel number of $\overline{C^{\varepsilon}}$ to be C^{ε}/k, where $k = 1, 2, 4, 8$ after a few training epochs on three datasets. For memory efficiency, we choose $k = 8$ in all our experiments. After that, we further perform Concat operations on the output of the three attention layers to obtain the output of the multi-head attention layer.

$$o = (o^{(1)} + o^{(2)} + o^{(3)})/3, \tag{7}$$

$$y_i = \gamma o_i + x_i, \tag{8}$$

where γ is a learnable scalar, and it is initialized as 0. Introducing the learnable γ allows the network to rely on the local neighbourhood's cues and then gradually learn to assign more weight to the non-local evidence. After that, we multiply the output of the multi-head attention layer by the scale parameter γ and add it back to the input feature map to obtain the feature map $y\epsilon R^{C^{\varepsilon}\times H^{\varepsilon}\times W^{\varepsilon}}$, and then pass y to a dense layer (fully connected layer). The fully connected layer will flatten each sample in the mini-batch. That is, the input shape of the fully connected layer will become two-dimensional, where the first dimension is the sample in the mini-batch, the second dimension is the vector representation after each sample is flattened, and the vector length is the product of the channel, height, and width. Finally, the output $y^{'}$ is transformed into a legal category prediction distribution through the softmax operation, and the category with the largest prediction probability is used as the output category. The proposed method can strengthen the useful information of the image and suppress the useless information, improve the performance of the local model, and help to increase the reasonable degree of personalization of the local model.

3.2 The federated learning personalized aggregation

In federated learning based on deep learning, the global model parameters of the traditional federated learning algorithm are randomly generated and have no reference values. If we first use the federated averaging algorithm to aggregate the local model parameters to the global model and send the aggregated model updates back to the client. This process would be repeated until the model converges or when the number of iterations or the maximum training time is reached. Then, the global model parameters are saved, and this step is used as pre-training. Afterwards, the pre-trained parameters are saved to the global model parameters for retraining, which can improve the personalization ability of model processing and reduce the impact of data noise. The tranditional aggregation method of federated learning is directly uses the federated averaging algorithm to average the local model parameters and does not take into account the influence of the data difference caused by the reasonable personalization of the local model on the global model when the model is aggregated. The method of aggregating local model parameters into the global model will harm its model performance.

After analysis, similar data aggregation can improve the performance of the global model. For example, the Tik Tok app, which everyone loves, has become popular all over the world in recent years because it can recommend corresponding videos according to users' preferences, realize preferences of local models and consider the similarity indicators of users' favorite types. Data with completely different similarity indicators cannot improve the global model and may also have a performance degradation effect. At the same time, using the federated averaging algorithm to average the local model parameters directly cannot deal with the impact of noise in the data either. Joint training is challenging to obtain a high-performance model with such data. Suppose it is possible to reduce the proportion of local model parameters that are significantly different from the global model parameters during model aggregation. In that case, it will initially reduce the impact of local model personalization on the performance of the global model. Based on this, we propose to add the variable coefficient λ in front of the local model parameters to reduce the impact of the personalized model on the global model. The local model personalized aggregation process is shown in Fig. 6.

As shown in Fig. 6, we perform pre-training and calculate the distance between current local model parameters and initial model parameters, and obtain the difference between models to formulate the personalization coefficient λ for local models. The variable coefficient λ is calculated as shown in the following formula. Then we assign the variable coefficient λ to the local model according to the difference between the local model parameters and the initial model parameters, and aggregate them into the global model, it can initially solve the problem of poor joint training model performance and increase data noise that model personalization problems may cause.

$$\lambda_p = \frac{e^{f(|\omega_0-\omega_{t+1}^{p}|)}}{\sum_P e^{f(|\omega_0-\omega_{t+1}^{p}|)}}, \tag{9}$$

$$f(x) = \begin{cases} \left|\underbrace{\frac{1}{\sum\sum x(i,j)}}_{a}\right| \ (a \geq 1\mathrm{or}a \leq -1) \\ \left|\underbrace{\sum\sum x(i,j)}_{a}\right| \quad (-1 < a < 1), \end{cases} \tag{10}$$

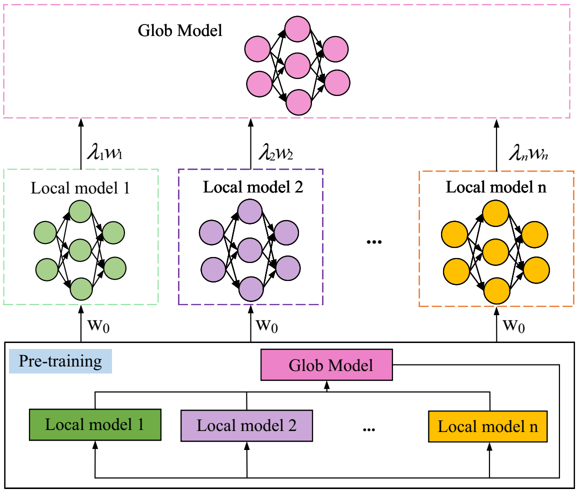


Fig. 6 Procedure chart of local model personalized distribution

where λ_p is the local model coefficient factor initially aggregated into the global model, w_0 is the weight parameter of the global model after pre-processing, and $w_{t+1}^{(p)}$ is the weight parameter of the p^{th} client after training the local model on the local data, it is to calculate the difference between the pre-processed weight parameters and the weight parameters of the current local model. The f function ensures that the data is between 0 and 1, which can reduce the proportion of data with large differences to solve the impact of model personalization problems on aggregation and reduce the risk of increasing data noise. After that, the data is normalized to ensure that the sum of the probabilities of multiple classifications is 1. $\sum\sum x(i,j)$ is the sum of all elements in x, and the local updated model parameter $w_{t+1}^{(p)}$ is assigned to the personalization coefficient λ_p and then sent to the server. The server aggregates the received model parameters and finally uses the weighted average of the received model parameters:

$$\overline{\omega}_{t+1} = \sum_{p=1}^{P} \frac{n_p}{n}(\lambda_p \omega_{t+1}^{(p)}), \tag{11}$$

where n_p is the local data volume of the p^{th} participant, $w_{(t+1)}^{(p)}$ is the parameter of the local model at this time, λ_p is the local model coefficient factor initially aggregated into the global model, and then the server will aggregate the model parameter $\overline{\omega}_{t+1}$ send it to all participants. The local model parameter pre-processing can solve the problems of poor joint training and of increasing data noise caused by model personalization.

4 Experimental analysis

In order to prove that the personalized federated learning algorithm proposed in this paper based on the multi-head attention can not only improve the accuracy in the standard dataset, but also improve the degree of data personalization, this paper implements comparative experiments on three standard datasets. By comparing with FedAvg algorithm, pFedMe algorithm [19], per-FedAvg algorithm [21], FedAdagrad algorithm [38] and FedDC algorithm [39], it is proved that the personalized part has a positive impact on the overall accuracy. In particular, in order to simulate the federated learning settings, the experimental part of this paper adopts the serial training method for each client, and uploads the model parameters after the training to the terminal server in turn to aggregate the global model.

The experimental environment configuration is as follows: CPU (AMD R5-3600), memory (16 G) DDR4, GPU (NVIDIA Geforce RTX2070S), operating system (64-bit Windows 10). The experimental framework is the Pytorch open-source framework. Stochastic gradient descent is used as the learning rate. The initial learning rate is 0.001, the weight attenuation coefficient is set to 0.01, the momentum is set to 0.9, The training batch size is 10, the number of local clients is 10, and the training is 50 communication times (equivalent to 500 cycles of CNN training). The experiments use the classic CNN as the network model in the architecture of this article and compares it with the Federated Averaging model, which uses the same structure of CNN. The result of the experiment is an average of 10 times.

The datasets used in this experiment are MNIST, SVHN, and STL10. The public dataset of MNIST consists of 60,000 training samples and 10,000 test samples, and each sample is a 28×28-pixel grey handwritten number Picture, SVHN is similar in style to MNIST (for example, the image is a small cropped number), but contains much more labelled data, 73257 training samples and 26,032 test samples. The STL-10 dataset is an image recognition data set used to develop unsupervised feature learning, deep learning, and self-learning algorithms. It contains 10 types of objects, each of which consists of 500 training samples and 800 test samples. Each sampled pixel is 96×96. In addition to pictures with category tags, there are 100,000 pictures without category information. In which the data set is divided, taking the MNIST data set as an example, firstly, the data is sorted according to the digital label, and it is divided into 200 data blocks with a sample of 300. A total of 100 clients are set up, and each client is assigned 10 data blocks. This work uses precision index to evaluate the model, sets TP to represent the true class, that is, the positive samples that are correctly predicted by the model, and FP to represent the true negative class, that is, the positive samples that are predicted to be negative by the model. The precision formula is as follows:

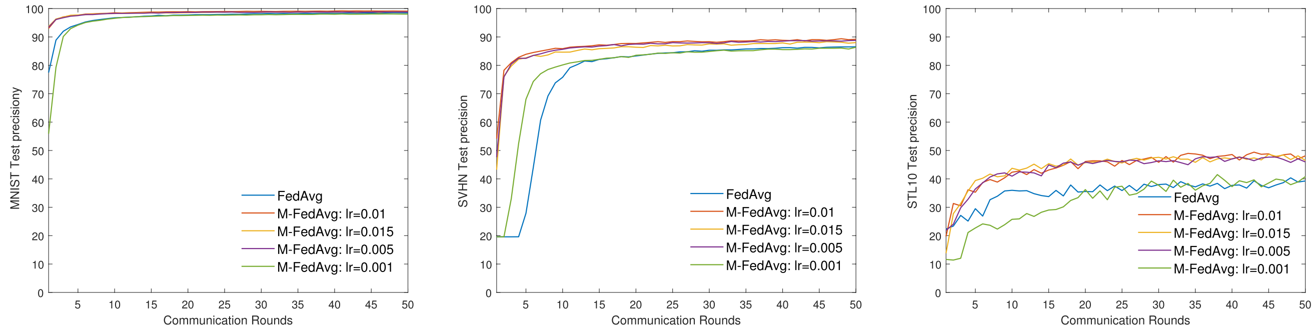
$$precision = TP/(TP + FP). \tag{12}$$

In the following, the experiment is mainly composed of three parts. Firstly, the classification accuracy of this method and the existing federated learning methods is compared and analyzed. Then, this paper conducted two ablation experiments, namely, the ablation experiment of multi-head attention and the ablation experiment of federation personalized aggregation. These two experiments verified the effectiveness of multi-head attention and federation personalized aggregation. Finally, an experiment is set to verify that the personalized parameters of the algorithm in this paper are better than those of the traditional federal average algorithm, and the method in this paper is more consistent with the actual scenario application.

4.1 Analysis of comparison results

Figure 7 shows the test results of the algorithm in this paper and the traditional FedAvg algorithm in 3 test sets, and Fig. 8 compares the loss function of the algorithm in this paper with the traditional FedAvg algorithm. M-FedAvg is evaluated in the setting of $lr \in (0.001, 0.005, 0.01, 0.015)$. When the learning rate is too large or too small, it will have a considerable negative impact on the performance of M-FedAvg, so the learning rate is set to 0.01. It can be seen from the results that the average precision of the algorithm in this paper is significantly better than the FedAvg algorithm, and the training loss convergence rate is significantly faster than the classic FedAvg algorithm.

Table 1 shows the precision of multiple algorithms on three datasets. Its average test accuracy is higher than other modules. The single addition is lower than the average test precision of CNN training with the multi-head attention mechanism module after the data sets of all clients are concentrated in one place. We not only compared M-FedAvg with FedAvg but also compared with the most advanced personalized FL algorithms (such as pFedMe [19], Per-FedAvg [21], FedAdagrad [38] and FedDC [39]). The client is set to 10, the client data is independent and identically distributed, and comparisons are made on all three real datasets. In general, M-FedAvg maintains the highest performance in almost all scenarios. In the current federated learning based

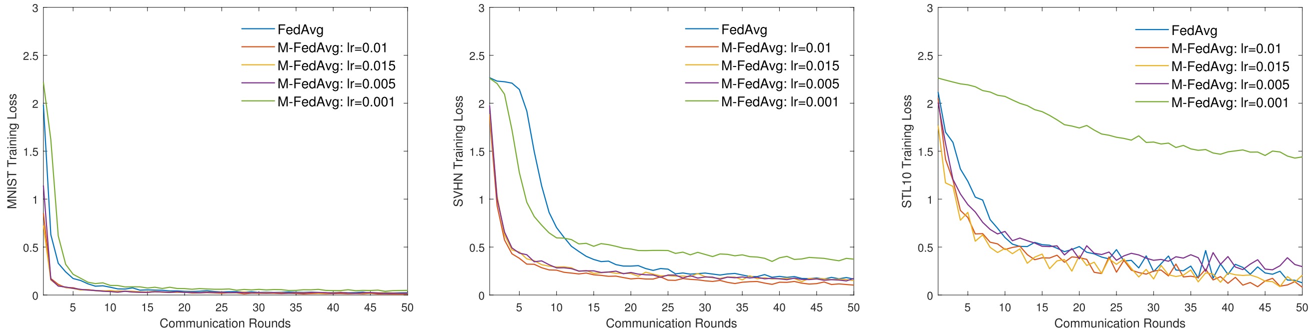


(a) MNIST precision curve graph

(b) SVHN precision curve graph

(c) STL10 precision curve graph

Fig. 7 Federated learning precision curve graph



(a) MNIST loss function comparison (b) SVHN loss function comparison (c) STL10 loss function comparison

Fig. 8 Federated learning loss function graph

Table 1 The influence of personalized federated learning algorithm based on multi-head attention mechanism

Dataset	Model	Average (%)	Best (%)
MNIST	FedAvg	96.95	98.56
	Per-FedAvg	97.51	98.70
	pFedMe	97.86	98.77
	FedAdagrad	97.98	98.75
	FedDC	98.52	99.21
	M-FedAvg	**98.63**	99.20
SVHN	FedAvg	76.30	86.54
	Per-FedAvg	79.69	86.86
	pFedMe	80.20	87.45
	FedAdagrad	80.71	86.32
	FedDC	84.78	89.32
	M-FedAvg	**86.75**	89.37
STL10	FedAvg	35.52	40.36
	Per-FedAvg	39.44	46.00
	pFedMe	40.97	47.66
	FedAdagrad	40.21	47.73
	FedDC	41.65	48.12
	M-FedAvg	**44.28**	49.41

Bold value represents the optimal value

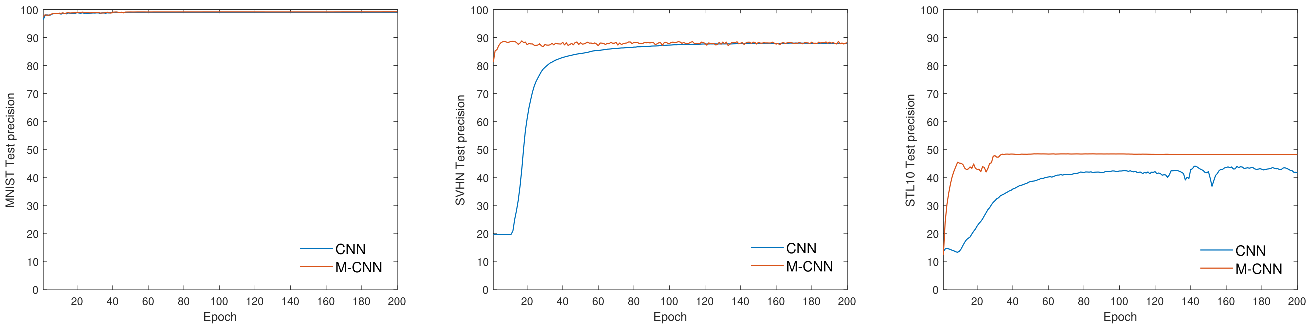
on deep learning, it is not considered to improve the personalized performance of the local model. Selecting information that is more critical to the current task goal from a large number of information can help improve the personalized performance of the local model. To solve this problem, this paper introduces a multi-head attention mechanism, which can help capture the relevance of local features, and may improve the performance of local models, thus increasing the degree of model parameters' personalization. In addition, through the multi-head attention, key information can be retained for better feature extraction and selection, and the accuracy of recognition can be improved. Compared with the traditional FL and personalized FL algorithms, the performance improvement of M-FedAvg mainly comes from three aspects: improving the reasonable degree of personalization of the local model, reducing the impact of data differences caused by the reasonable personalization of the local model on the global model when the model is aggregated, and reducing the impact of the global model on the local model when it is assigned to the local model.

4.1.1 Ablation experiment results of multi-head attention

This experiment uses the CNN model combined with the multi-head attention mechanism. We do not use the model framework of the second point of innovation in this algorithm but use the traditional FedAvg aggregation method to verify the precision of the CNN model combined with the multi-head attention mechanism (M-CNN).

Figure 9 shows the results of the traditional method and the improved method on each dataset. The traditional method here is to gather the datasets of all clients in one place and then use CNN to train. The improved traditional method is to gather the datasets of all clients in one place and then use the CNN training with the multi-head attention mechanism module, where the x-axis represents the training period (Epoch), and the y-axis represents the test precision. The set period is 200, which is equivalent to 20 communication times. Among them, the performance of the M-CNN algorithm using the CNN model combined with the multi-head attention mechanism has large fluctuations and the fastest convergence. In 60 training cycles, the performance of the two algorithms is basically stable. The accuracy of the improved traditional method is better than that of the traditional method. Finally, in MNIST, SVHN, and STL10 dataset average precision rate reached 99.04%, 87.88%, and 47.18%. The experimental results show that the multi-head attention is effective, which can improve the effectiveness of feature extraction and the precision of the results.

Figure 10 shows the test results of the federated averaging algorithm of the multi-head attention module and



(a) MNIST precision curve graph (b) SVHN precision curve graph (c) STL10 precision curve graph

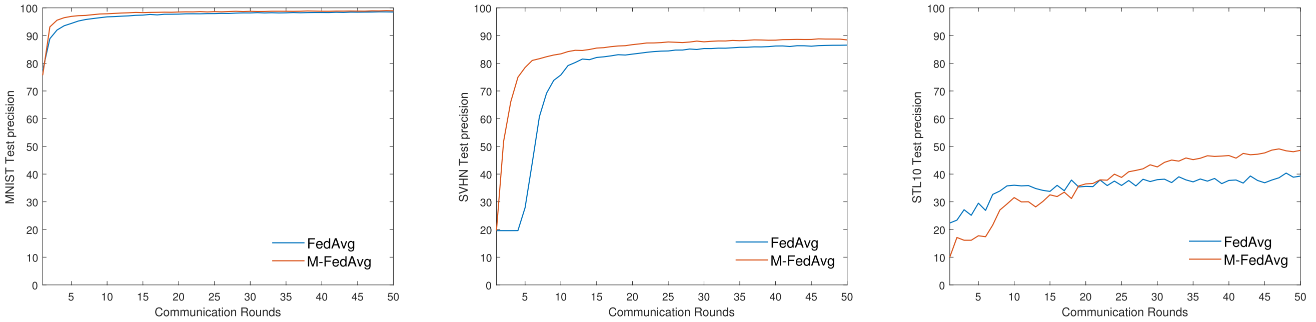
Fig. 9 Traditional method precision curve graph

the traditional federated averaging algorithm in each test set. The x-axis represents the number of communication rounds, and the y-axis represents the test precision, in which the performance of the multi-head attention module fluctuates greatly, and its convergence is the fastest. At 10 communication times, the performance of these two algorithms is basically stable. The average accuracy of the algorithm in this paper is significantly better than that of the FedAvg algorithm. Finally, the average accuracy rates on the MNIST, SVHN, and STL10 data sets reach 98.51%, 85.66%, and 42.92%. The experimental results show that the average accuracy of M-FedAvg is lower than that of M-CNN because M-CNN directly uses neural network CNN to train data sets. In federated learning, the participants will not expose the data to the server or other participants, so the performance of the federated learning model is slightly worse than that of the centralized training model. Compared with the loss of accuracy, the additional security and privacy protection in many scenarios is undoubtedly more valuable. Fig. 11 compares the improved federated average algorithm in this paper with the traditional federated average algorithm in loss functions. The x-axis represents the number of communication rounds, and the y-axis represents the training loss. At the beginning of training, two curves converge very quickly. When the training reaches 40 times of communication, the loss function of the traditional federated averaging algorithm remains stable and no longer converges and maintains a high loss value. The federated averaging algorithm with the multi-head attention module has a faster convergence rate and has a tendency to be lower than the traditional federated averaging algorithm.

The algorithm proposed in this paper has made a series of improvements. The influence of the multi-head convolutional attention module on the federated average algorithm is shown in Table 2. We add the multi-head attention module, which can increase the reasonable degree of personalization of the local model and improve the precision of the image classification task.

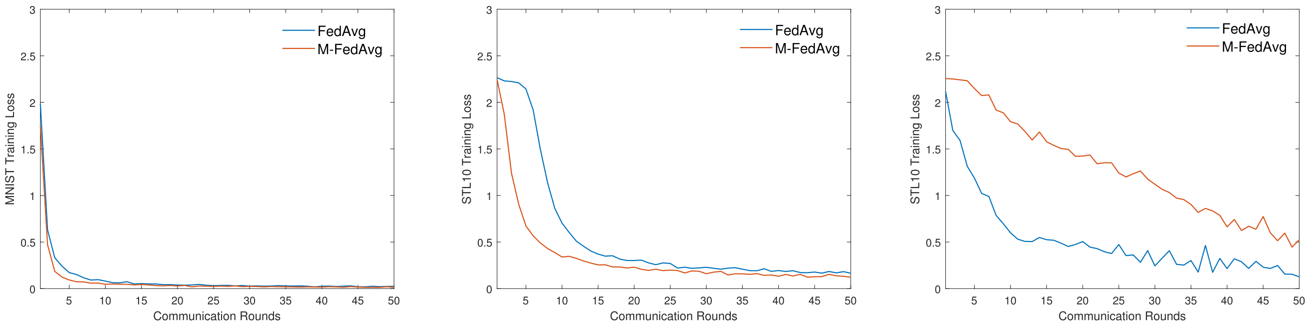
4.1.2 Ablation experiment results of personalized aggregation

In federated learning based on deep learning, the aggregation method directly use the federated averaging algorithm to average local model parameters. Neither the relevance of features nor the data differences brought about by the reasonable personalization of each client are considered.



(a) MNIST precision curve graph (b) SVHN precision curve graph (c) STL10 precision curve graph

Fig. 10 Multi-head attention module precision curve graph



(a) MNIST loss function comparison (b) SVHN loss function comparison (c) STL10 loss function comparison

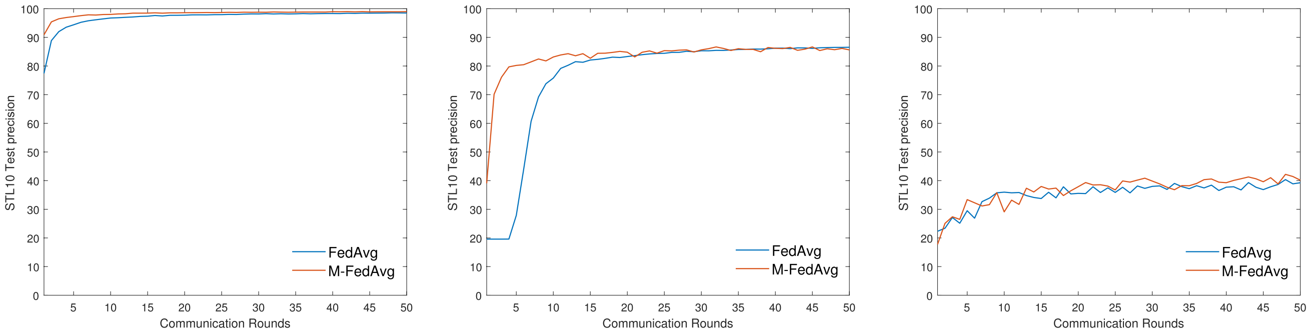
Fig. 11 Multi-head attention module loss function graph

Table 2 The influence of multi-head attention module on the federated averaging algorithm

DATASET	MODEL	Average (%)	Best (%)
MNIST	CNN	98.92	99.03
	M-CNN	**99.04**	99.16
	FedAvg	96.95	98.56
	M-FedAvg	**97.82**	99.01
SVHN	CNN	80.24	87.99
	M-CNN	**87.88**	88.78
	FedAvg	76.30	86.54
	M-FedAvg	**84.02**	88.81
STL10	CNN	38.00	43.98
	M-CNN	**47.18**	48.39
	FedAvg	35.52	40.36
	M-FedAvg	**37.18**	49.08

Bold value represents the optimal value

Aggregating the local model parameters at this time into the global model will harm its model performance. Considering the influence of the personalization of each local model on the global model, this paper makes corresponding changes according to the personalization characteristics of the model when sending the global model parameters to the local model. By calculating the difference between the model parameters obtained from real-time training of the local model with local data and the parameters of the real-time global model, the weight coefficients are obtained and returned to the client. This experiment is an ablation experiment, only uses the aggregation method, without using the combined multi-head attention mechanism, and compare it with the traditional FedAvg aggregation method to verify the new aggregation algorithm. Figure 12 shows the test results of the M-FedAvg and FedAvg algorithms added to the local model personalized aggregation module in each test set. Among them, the performance of M-FedAvg fluctuates greatly, and the convergence is the fastest. At 20 communication times, the performance of the two algorithms is basically stable. The average accuracy of the algorithm in this paper is significantly better than that of the FedAvg algorithm, and the average accuracy of the MNIST, SVHN, and STL10 data sets is 98.31%, 83.32%, and 36.72%. Figure 13 compares the loss functions of M-FedAvg and FedAvg algorithms with the addition of the local model personalized aggregation module. In the early stage of training, the two curves converge very quickly, and the M-FedAvg training loss converges significantly faster than FedAvg.



(a) MNIST precision curve graph (b) SVHN precision curve graph (c) STL10 precision curve graph

Fig. 12 Local model personalized aggregation module accuracy curve graph

Table 3 shows the impact of the local model personalized aggregation module on the federated average algorithm. The improved federated averaging algorithm in this paper is pretrained first, then the local model personalized aggregation module is added, which can reduce the impact of the data differences when aggregating local models on the global model, and improve the precision of the image classification task.

Table 3 The influence of local model personalized aggregation module on the federated average algorithm

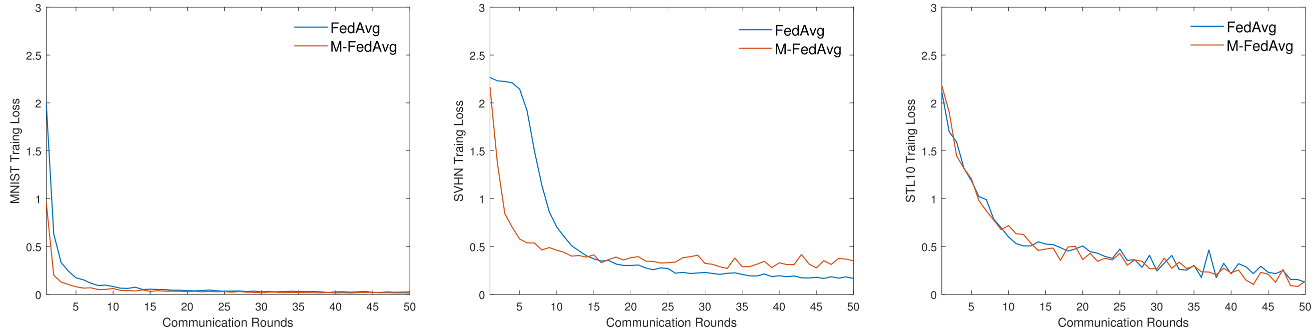
Dataset	Model	Average (%)	Best (%)
MNIST	FedAvg	96.95	98.56
	M-FedAvg	**98.31**	99.00
SVHN	FedAvg	76.30	86.54
	M-FedAvg	**83.32**	86.66
STL10	FedAvg	35.52	40.36
	M-FedAvg	**36.72**	42.19

Bold value represents the optimal value

4.2 Personalized analysis

The above experiment proves that the personalized federated learning algorithm can improve overall precision, but it cannot accurately determine the effective proportion of the main part to the personalized part. Currently, there is no dataset specifically for verifying personalization problems, so this work constructs a dataset that can be used for personalization problems to verify the effect of the personalization part on the size of the fitting data deviation. The training set is also used as the test set for linear fitting problems.

In this experiment, the idea of personalized understanding in the recommendation system is used for reference. The core task of the recommendation system is to model and understand the user's preference information for personalized recommendations. In 2019, Zhao et al. [40] proposed the LSIC model, which uses adversarial training to use long-term and short-term information for content-aware movie recommendations. The long-term model indicates that the interaction between the user and the movie should change slowly over time (long-term preference information), while the conversation-based model dynamically encodes the user's interest information and short-term changes in movie attributes (short-term preference information). Since users' long-term preferences are the same [40–42], but short-term preferences are different, the experiment was designed this time. Long-term preference information is the main part and has a high frequency of occurrence. Short-term preference information is personalized, and the frequency should be lower than the main frequency. At the same time, there will be some noise in user data. Since there is no data set specifically aimed at verifying personalization issues, this article is to visually observe the changes in the personalization part of the data set. We construct datasets V1 and V2, and datasets M1 and M2 with artificially added Gaussian noise. Then we set the long-term preference information (main part) to occur at high frequency and the short-term preference information (personalized part) to occur at low frequency. In the model, we use stochastic gradient descent as the learning rate, the initial learning rate is 0.0005, and the momentum is set to 0.9, the weight attenuation coefficient is set to 0.0005, the training batch size is 100. The V1, V2, M1, and M2 datasets are randomly divided into 2 local clients, and 1000 communication times are trained.



(a) MNIST loss function comparison (b) SVHN loss function comparison (c) STL10 loss function comparison

Fig. 13 Local model personalized aggregation module loss function graph

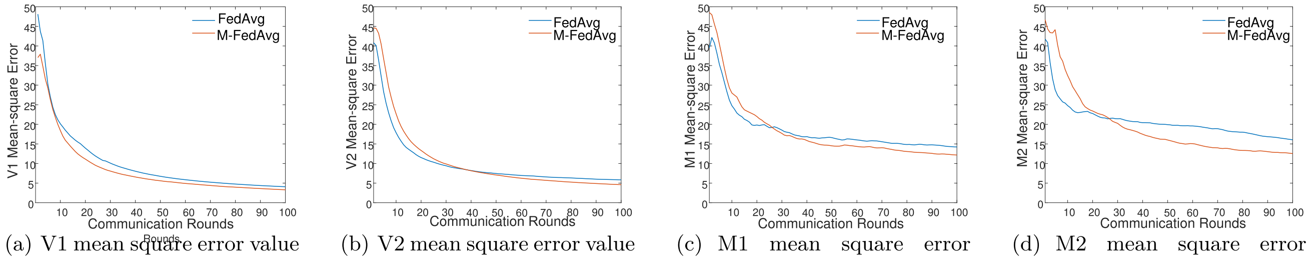
$$
\begin{aligned}
&V1\epsilon\{x,\underbrace{a\sin(w_1x)}_{MainPart}+\underbrace{b\cos(w_2x)}_{PersonalizedPart}\}\\
&V2\epsilon\{x,\underbrace{a\sin(w_1x)}_{MainPart}+\underbrace{c\cos(w_3x)}_{PersonalizedPart}\}\\
&M1\epsilon\{(x+\underbrace{noise}_{Gussian}),\underbrace{a\sin(w_1x)}_{MainPart}+\underbrace{b\cos(w_2x)}_{PersonalizedPart}\}\\
&M2\epsilon\{(x+\underbrace{noise}_{Gussian}),\underbrace{a\sin(w_1x)}_{MainPart}+\underbrace{c\cos(w_3x)}_{PersonalizedPart}\},
\end{aligned}
\tag{13}
$$

where x randomly selects 400 points on $(-2\pi, 2\pi)$, and y is the corresponding label. In any dataset, the main part of the data accounts for a relatively large proportion, and the personalized part of the data accounts for a relatively small proportion. A and b represent the amplitudes of the two parts, and w is the frequency of occurrence. Therefore, the main part parameters $a = 10$, $w_1 = 2$, the personalized part parameters $b = 2$, $w_2 = 1$, $c = 1$, $w_2 = 0.5$, and the noise is Gaussian noise. Due to the small amount of data, Multilayer Perceptron (MLP) is used to train the data, and the Mean Square Error (MSE) is used to evaluate the degree of change of the model. MSE is the mean value of the sum of squared errors of the corresponding points between the predicted data and the original data. The standard for evaluating estimators under the sample is defined as follows for the constructed dataset:

$$
MSE = \sum_{i=1}^{n} \frac{1}{n}(f(x_i) - y_i)^2, \tag{14}
$$

where f and y are the predicted value and the true value, respectively. The smaller the mean square error value, the better the model's ability to fit the experimental data. Figure 14 is the comparison of the mean square error values of M-FedAvg and FedAvg. M-FedAvg and FedAvg are used to train directly the dataset V1, V2, M1, M2, where the x-axis represents the number of communications (Communication Rounds), and the y-axis represents the mean square error value (Mean-square Error). The mean square error of the algorithm in this paper is significantly lower than that of the FedAvg algorithm. Finally, the mean square error values on the V1, V2, M1, and M2 data sets reach 3.293 – 4.076, 4.604 – 5.834, 12.167 – 14.229, 12.532 – 16.059 (– left is the mean square error value of the algorithm in this paper, and the right is the mean square error value of the FedAvg algorithm), the algorithm in this paper has a small mean square error. After adding noise, the advantage of M-FedAvg is more obvious, indicating that the model with the multi-head attention mechanism module has a good performance in fitting experimental data.

The ARMA algorithm is used to calculate the specific parameters for the prediction results. The specific parameters of the FedAvg algorithm and the training data under the algorithm in this paper are shown in Table 4 below. It can be seen from the data in the table that when no noise is added to



(a) V1 mean square error value comparison

(b) V2 mean square error value comparison

(c) M1 mean square error value comparison

(d) M2 mean square error value comparison

Fig. 14 Federated learning mean square error graph

Table 4 FedAvg and M-FedAvg training model parameters algorithm

	Predict a	Basic	Predict w_1	Basic	Predict b	Basic	Predict w_2	Basic
V1(FedAvg)	9.753	–0.247	1.986	–0.014	1.796	–0.204	1.049	+0.049
V1(M-FedAvg)	9.968	**–0.032**	1.998	**–0.002**	2.059	**+0.059**	0.986	**–0.014**
V2(FedAvg)	9.197	–0.803	1.972	–0.028	1.104	**+0.104**	0.585	+0.085
V2(M-FedAvg)	9.651	**–0.349**	1.990	**–0.010**	1.130	+0.130	0.545	**+0.045**
M1(FedAvg)	7.499	–2.501	1.982	**–0.018**	2.697	+0.697	1.110	**+0.110**
M1(M-FedAvg)	7.781	**–2.219**	1.972	–0.028	2.686	**+0.686**	1.112	+0.112
M2(FedAvg)	7.230	–2.770	1.975	–0.025	0.893	–0.107	0.315	–0.185
M2(M-FedAvg)	7.810	**–2.190**	1.974	–0.026	1.064	**+0.064**	0.946	+0.446

Bold value represents the optimal value

Table 5 Introduction of main parameters

Symbol	Meaning	Situation
B	Batch size for client updates	
C	Client	
D_p	Dataset of each local client	
E	local epoch	
FP	The positive sample predicted by the model to be negative.	Eq. 12
I	The feature map	
I'	The feature map is obtained after 5×5 convolution of I	
L	The fraction of clients C	
m	Random set of $\max(L \cdot P, 1)$ clients	
n	Number of local samples for all clients	Eq. 2
n_p	Number of local samples on the client	Eq. 2
$o_j^{(1)}$	The output of the self-attention layer at the j^{th} position of the feature map x_1	Eq. 5
o	The output of multi-headed attention layer	Eq. 7
P	The number of participants in FL	Eq. 9
S	Server	
t	Number of communication cycles	
TP	The positive sample predicted correctly by the model	Eq. 12
ω_0	Pre-trained glob model parameters	Eq. 9
ω_g	The global model parameter	
$\omega_{t+1}^{(p)}$	Updated local model parameters	Eq. 1
$\overline{\omega}_t$	Average model parameter	Eq. 1
$\overline{\omega}_{t+1}$	Average model parameters after aggregation	Eq. 2
W_k	The learned weight matrix	
W_q	The learned weight matrix	
W_v	The learned weight matrix	Eq. 6
x	The feature map is obtained after 5×5 convolution of I'	
$x_i^{(1)}$	The i^{th} position of feature map x_1	Eq. 4
$x_j^{(1)}$	The j^{th} position of feature map x_1	Eq. 4
∇_p	The average gradient of the local client under the current model.	Eq. 1
λ_p	Local model coefficient factors that are initially aggregated to the global model	Eq. 9
γ	A learnable scalar	Eq. 8
η	Learning rate	Eq. 1

the dataset, the parameters of the personalized algorithm are closer to the true value than FedAvg. After noise is added, the algorithm in this paper and the most advanced personalized FL algorithm per-FedAvg and pFedMe's main part of the parameters a and personalized parameters b are closer to the true value than FedAvg. Some random, discrete, and isolated noises may appear due to the channel transmission error of the picture during the transmission process, so the dataset with added noise will be more consistent with the requirements of this article. The most advanced personalized FL algorithm and the personalized parameters of the algorithm in this paper are closer to the true value than the personalized parameters of the traditional federated average algorithm, indicating that the personalized part has a positive impact and improves the overall accuracy). In order to better enable readers to understand the parameters in this article, we provide explanations of the parameters in Table 5.

5 Conclusion

In business, people need one to ensure data security and to be able to further mine personalized content to bring value. In personalization, scholars have found that users have more than one personalization, such as long-term personalities, short-term personalities in time, and personalities under different conditions in space. Based on the traditional federated averaging algorithm, this paper proposes a personalized federated learning algorithm based on the multi-head attention mechanism. The algorithm uses a multi-headed attention mechanism on the local side to improve the learning of personalized data. It uses weights on the server side to ensure that the personalized model is not quickly melted away. The experiment shows that the algorithm can not only improve the accuracy of the standard dataset but also improve the personalization degree of the data.

Acknowledgements This work is supported in part by the National Natural Science Foundation of PR China (72101121) and Ministry of Education, Humanities and social science projects (21YJC790054).

Author contributions Conceptualization, SJ, KH, ML, and JW; methodology, SJ, KH and ML; software, SJ; validation, YL and ML; formal analysis, KH, ML, HL and JW; investigation, KH and ML; resources, KH; data curation, HL; writing-original draft preparation, SJ; writing-review and editing, LW; visualization, MX; supervision, MX; project administration, KH; funding acquisition, KH. All authors have read and agreed to the published version of the manuscript.

Data availibility statement The data and code used to support the findings of this study are available from the corresponding author upon request (*jss@nuist.edu.cn*).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Honghao Gao, Wanqiu Huang, Tong Liu, Yuyu Yin, Youhuizi Li (2022) Ppo2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems. IEEE Trans Transp Syst. https://doi.org/10.1109/TITS.2022.3169421
2. Xiao Junsheng, Huahu Xu, Gao Honghao, Bian Minjie, Li Yang (2021) A weakly supervised semantic segmentation network by aggregating seed cues: the multi-object proposal generation perspective. ACM Trans Multimed Comput Commun Appl 17:15
3. Honghao Gao, Binyang Qiu, Barroso Ramon J, Duran Hussain Walayat, Yueshen Xu, Xinheng Wang (2022) Tsmae: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder. IEEE Trans Netw Sci Eng. https://doi.org/10.1109/TNSE.2022.3163144
4. Brendan Mcmahan H, Eider Moore, Daniel Ramage, Seth Hampson, Blaise Arcas (2016) Communication-efficient learning of deep networks from decentralized data. PMLR 54:1273–1282
5. Mcmahan H Brendan, Moore Eider, Ramage Daniel, Arcas Blaise (2017) Federated learning of deep networks using model averaging. arXiv preprint arXiv:1602.05629
6. Ke Guolin, Meng Qi, Thomas Finley (2017) A highly efficient gradient boosting decision tree. NIPS 30:3149–3157
7. Chen Lu, Xia Min, Lin Haifeng (2022) Multi-scale strip pooling feature aggregation network for cloud and cloud shadow segmentation. Neural Comput Appl 34:6149–6162
8. T Xu M (2021) Research on long and short-term neural network recommendation model based on self-attention mechanism
9. Zhu Qiannan, Zhou Xiaofei, Song Zeliang, Tan Jianlong, Guo Li (2019) Dan: deep attention neural network for news recommendation. AAAI 33:5973–5980
10. An Mingxiao Wu, Chuhan Fangzhao, Wu, Kun Zhang, Zheng Liu, Xing Xie (2019) Neural news recommendation with long-and short-term user representations. ACL 57:336–345
11. Smith Virginia, Chiang Chaokai, Sanjabi Maziar, Talwalkar Ameet (2017) Federated multi-task learning. NIPS:4427–4437
12. Liu Yang, Chen Tianjian, Yang Qiang (2018) Secure federated transfer learning. arXiv preprint arXiv:1812.03337
13. Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Qiang Yang (2019) SecureBoost: A lossless federated learning framework. IEEE Intell Sys 36:7–98
14. Deng Yuyang, Kamani, Mohammad Mahdi, Mahdavi Mehrdad (2020) Adaptive personalized federated learning. arXiv preprint arXiv:2003.13461
15. Yang Qiang, Liu Yang, Cheng Yong (2019) Federated machine learning: concept and applications. ACM 10:1–19
16. Zhuo Hankz Hankui, Feng Wenfeng, Lin Yufeng, Xu Qian, Yang Qiang (2019) Federated deep reinforcement learning. arXiv preprint arXiv:1812.03337
17. Arivazhagan Manoj Ghuhan, Aggarwal Vinay, Singh Aaditya Kumar, Choudhary Sunav (2019) Federated learning with personalization layers. arXiv preprint arXiv:1901.08277v1
18. Jiang Yihan, Konečný Jakub, Rush Keith, Kannan Sreeram (2019) Improving federated learning personalization via model agnostic meta learning. arXiv preprint arXiv:1909.12488
19. Dinh Canh T, TranNguyen H, Nguyen Tuan Dung (2020) Personalized federated learning with moreau envelopes. arXiv preprint arXiv:2006.08848
20. Fallah Alireza, Mokhtari Aryan, Ozdaglar Asuman (2020) Personalized federated learning: a meta-learning approach. arXiv preprint arXiv:2002.07948
21. Nichol Alex, Achiam Joshua, Schulman John (2018) On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999
22. Wang Jialei, Kolar Mladen, Srerbo Nathan (2016) Distributed multi-task learning. Artif Intell Stat 51:751–760
23. Tan Alysa Ziying, Yu Han, Cui Lizhen, Yang Qiang (2021) Towards personalized federated learning. arXiv preprint arXiv:2103.00710
24. Liu Yang, Yang Qiang, Chen Tianjian (2019) Tutorial on federated learning and transfer learning for privacy, security and confidentiality.*AAAI'19*
25. Kairouz Peter, McMahan H. Brendan, Avent Brendan, Bellet Aurélien, Bennis Mehdi, Bhagoji Arjun Nitin, Bonawitz Kallista, et al Charles Zachary (2019) Advances and open problems in federated learning
26. Lecun Yann, Bottou Leon (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324
27. Song Lei, Xia Min, Weng Liguo, Lin Haifeng, Qian Min, Chen Bingyu (2023) Axial cross attention meets cnn: bibranch fusion network for change detection. IEEE J Sel Top Appl Earth Observ Remote Sens 16:32–43
28. Hao Yu, Yang Sen (2018) and Shenghuo Zhu. Demystifying why model averaging works for deep learning, parallel restarted sgd with faster convergence and less communication
29. Cho Kyunghyun, van Merrienboer Bart, Bahdanau Dzmitry, Yoshua Bengio (2014) Encoder-decoder approaches on the properties of neural machine translation. arXiv preprint arXiv:1409.1259
30. Krizhevsky Alex (2009) Learning multiple layers of features from tiny images. Tech Report. 1–60
31. Miao Shoukuan, Xia Min, Qian Ming, Zhang Yonghong, Liu Jia, Lin Haifeng (2022) Cloud/shadow segmentation based on multi-level feature enhanced network for remote sensing imagery. Int J Remote Sens 43(15–16):5940–5960
32. Yi Qu, Xia Min, Zhang Yonghong (2021) Strip pooling channel spatial attention network for the segmentation of cloud and cloud shadow. Comput Geosci 157:104940

33. Chen Bingyu, Xia Min, Qian Ming, Huang Junqing (2022) Manet: a multi-level aggregation network for semantic segmentation of high-resolution remote sensing images. Int J Remote Sens 43(15–16):5874–5894
34. Min Xia Xu, Zhang Wan’an Liu, Weng Liguo, Yiqing Xu (2020) Multi-stage feature constraints learning for age estimation. IEEE Trans Inf Forensics Secur 15:2417–2428
35. Wang Zhiwei, Xia Min, Min Lu, Pan Lingling, Liu Jun (2022) Parameter identification in power transmission systems based on graph convolution network. IEEE Trans Power Deliv 37(4):3155–3163
36. Liu Jingjing, Liu Yefeng, Zhang Qichun (2022) A weight initialization method based on neural network with asymmetric activation function. Neurocomputing 483:171–182
37. Gao Jiahong, Weng Liguo, Xia Min, Lin Haifeng (2022) MLNet: multichannel feature fusion lozenge network for land segmentation. J Appl Remote Sens 16(1):1–19
38. Gao Liang, Fu Huazhu, Li Li, Chen Yingwen, Xu Ming, Xu Cheng-Zhong (2022) Feddc: Federated learning with non-iid data via local drift decoupling and correction. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 22094283
39. Reddi Sashank, Charles Zachary, Zaheer Manzil, Garrett Zachary, Rush Keith, Konecny Jakub, Kumar Sanjiv, McMahan H Brendan (2021) Adaptive federated optimization. International Conference on Representation Learning, page 13
40. Wei Zhao, Benyou Wang, Min Yang, Jianbo Ye, Zhou Zhao, Xiaojun Chen (2019) Leveraging long and short-term information in content-aware movie recommendation via adversarial training. IEEE Trans Cybern 50:4680–4693
41. Robin C (2017) Geyer, Tassilo Klein, and Moin Nabi. A client level perspective. NIPS, differentially private federated learning
42. Mukund Deshpande, George Karypis (2004) Item-based top-n recommendation algorithms. ACM Trans Inf Syst 22:143–147

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.