Dynamic Sub-Model Aggregation and Clustering for Intelligence Data via Hierarchical Federated Learning with Pre-Training

Jianfeng Wang*, Lin Ma, Xinyan Pei, Ruonan Shi, Qi Jing, Chen Yang
State Grid Shanxi Electric Power Company Yuncheng Power Supply Company, Yuncheng 044000, China
E-mail: wjf056391@126.com
*Corresponding author

Keywords: federated learning algorithm, intelligence data integration, pre-training mechanism, hierarchical clustering, decomposition and combination technology

Received: August 21, 2025

To address the challenges of slow convergence speed, poor dynamic adaptability, and low communication efficiency in intelligence data processing, a dynamic integration and clustering method for intelligence data based on an improved federated learning algorithm is proposed. First, an improved federated learning algorithm combining decomposition and combination is designed, where the global model is decomposed into multiple sub-models for local training, and a dynamic combination strategy is applied to integrate these sub-models, thereby improving the adaptability and accuracy of the global model. Then, a pre-training mechanism is introduced to initialize the global model using feature information from historical data, enhancing the model's initialization performance in dynamic data environments and accelerating convergence. Experiments are conducted on the MNIST and CIFAR-10 datasets, with comparisons made against baseline methods including FedAvg, FedProx, and ScaFFL. The results show that the proposed algorithm achieves accuracies of 98.69% and 90.26% on the pathological heterogeneity client, and 98.14% and 89.87% on the actual scenario heterogeneity client, respectively, on the two datasets. The normalized mutual information values of the proposed intelligence dynamic data integration and clustering method are 0.91 and 0.79, respectively. In a practical medical Internet of Things scenario test, the running time and memory usage of the proposed method are 18.23s and 1681MB, respectively. Our research denotes that the designed method can effectively improve the quality of dynamic integration of intelligence data and reduce resource consumption, providing a feasible solution for efficient processing of multi-source heterogeneous intelligence data.

Povzetek: Predlagana izboljšana metoda federativnega učenja z dinamično integracijo in gručenjem inteligentnih podatkov izboljša prilagodljivost, natančnost in učinkovitost obdelave večizvornih heterogenih podatkov ter hkrati zmanjša porabo virov.

1 Introduction

With the rapid development of information technology, the scale and complexity of intelligence data have increased sharply, and the demand for efficient processing and real-time analysis of large-scale intelligence data is becoming increasingly urgent [1]. In this study, intelligence data is defined as multi-source, heterogeneous data streams that are dynamically generated from distributed sensors, Internet of Things (IoT) devices, and edge computing nodes in real-world scenarios such as smart healthcare, industrial monitoring, and security systems. These data are characterized by their diverse modalities, high dimensionality, non-independent and identically distributed (non-IID) nature, and temporal dynamics. However, intelligence data has characteristics such as multi-source heterogeneity, high dimensionality, and dynamic evolution. Traditional data analysis methods face issues such as data silos, insufficient privacy protection, and poor real-time performance, making it difficult to meet the dynamic, collaborative, and intelligent requirements of modern intelligence processing [2]. Therefore, exploring an efficient method for integrating and clustering intelligence data has become a key focus of current research. In recent years, machine learning algorithms have been broadly employed in the area of intelligence data analysis [3]. Among them, Federated Learning (FL), as a distributed machine learning method, not only addresses data privacy and security issues, but also enables multi-party collaborative learning, demonstrating great potential [4]. Researchers have conducted extensive research on the FL algorithm, aiming to address issues such as non independent and identically distributed data, high communication overhead, and poor model convergence [5].

Guo et al. proposed a real-time medical data processing method based on FL, which integrates old and new models and selects representative samples to mitigate catastrophic forgetting, effectively learning diagnostic models from continuous medical data streams [6]. Gafni et al. introduced a signal processing-driven FL framework, combining signal processing and communication techniques to design optimized solutions that enhance FL

efficiency [7]. Yazdinejad et al. proposed an auditable privacy-preserving FL framework for medical electronic devices, using trusted execution environments to ensure secure training and aggregation, thereby preventing privacy leaks [8]. Bao and Guo presented a systematic research approach for FL under a cloud-edge collaborative architecture, filling the theoretical gap in cloud-edge FL [9]. Wang et al. designed an FL scheme for edge computing environments, integrating secret sharing and digital signatures to improve training efficiency by 40% while maintaining privacy [10]. Chatterjee et al. developed a recommendation model based on FL and blockchain, enhancing system security and transparency [11]. Akter et al. proposed an FL-based privacy protection framework for edge-based smart healthcare, balancing privacy and performance with an accuracy of 90% [12]. Gao et al. designed an FL framework based on cross-technology communication, improving model performance and communication efficiency in heterogeneous IoT environments [13]. Qu et al. introduced a quantum fuzzy FL algorithm, increasing training efficiency by 23% and accuracy by 15% while maintaining over 90% fidelity in quantum noise environments [14].

The summary of federal learning related work is shown in Table 1.

As illustrated in Table 1, prior research has made significant strides in applying FL to various domains, enhancing privacy, and improving efficiency through different strategies. However, several technical gaps remain. Many existing methods exhibit slow convergence speeds and poor adaptability under highly non-IID and dynamic data environments. Furthermore, considerations for the complex relationships between data sources are often insufficient, and communication efficiency remains a challenge. To systematically address these challenges and clearly define the scope of our contribution, this study is guided by the following research questions:

(1) Can the proposed dynamic sub-model aggregation and combination mechanism significantly improve model accuracy and convergence speed under non-independent and identically distributed data distributions, compared to standard FL baselines?

(2) To what extent does the integration of a pre-training mechanism and hierarchical similarity clustering enhance the quality of data integration and reduce communication overhead in dynamic environments?

In view of this, this study proposes a dynamic integration and clustering method for intelligence data based on an improved FL algorithm, aiming to enhance the real-time, adaptability, and accuracy of intelligence data processing. The novelty of this study lies in using a dynamic sub-model aggregation mechanism to solve the problem of insufficient adaptability of traditional methods to changes in data distribution. Moreover, a pre-training mechanism that utilizes historical data feature information to enhance the initialization performance and convergence speed of the model is introduced. Besides, by combining hierarchical similarity clustering techniques, efficient grouping and personalized modeling of data can be achieved, reducing communication overhead and improving the efficiency of dynamic integration and clustering of intelligence data.

Table 1: Summary of related works in federated learning.

References	Datasets Used	Methodological Innovations	Accuracy / Performance Metrics	Identified Limitations
Guo et al. [6]	Continuous medical data streams	Model fusion with sample selection	Supports continuous data stream learning	Limited adaptability to dynamic non-IID data
Gafni et al. [7]	Not specified (Theoretical)	Signal processing-inspired optimization	Improves FL efficiency	Lacks validation on real-world data
Yazdinejad et al. [8]	Medical data from electronic devices	Privacy protection using trusted execution environments	Effectively prevents privacy leaks	High hardware dependency and overhead
Bao and Guo [9]	Not specified (Survey)	Cloud-edge collaborative architecture analysis	Provides theoretical framework	No algorithmic innovation or validation
Wang et al. [10]	Medical IoT data	Secret sharing with digital signatures	40% training efficiency improvement	High communication and computation costs
Chatterjee et al. [11]	Financial consumer service data	FL combined with blockchain	Enhanced security and transparency	Latency and scalability issues
Akter et al. [12]	Smart healthcare data	Artificial noise injection	90% accuracy with high privacy rate.	Difficult to balance privacy and accuracy
Gao et al. [13]	Heterogeneous IoT data	Cross-technology communication coordination	Improved performance in heterogeneous environments	Requires dedicated coordination devices
Qu et al. [14]	Not specified (Simulation)	Quantum fuzzy FL	23% efficiency gain, 15% accuracy improvement	Requires quantum resources, low practicality

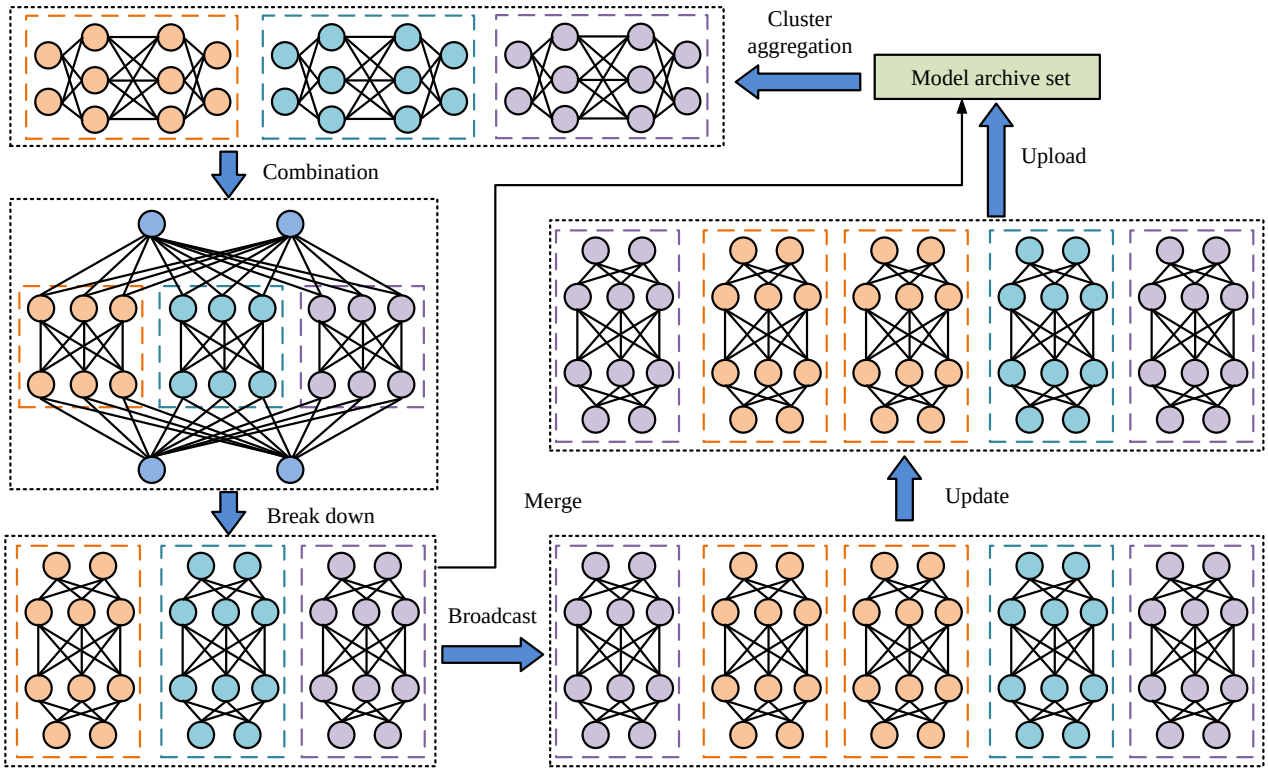


Figure 1: Flow diagram of improved FL algorithm combining decomposition and combination.

2 Methods and materials

Firstly, an improved FL algorithm combining decomposition and combination is designed to enhance the adaptability and accuracy of the global model. Then, a dynamic intelligence data integration and clustering method based on improved joint pre-training and hierarchical similarity is proposed to achieve efficient grouping and personalized modeling of data.

2.1 Improved FL algorithm combining decomposition and combination

In the big data era, intelligence data presents characteristics such as multi-source heterogeneity, high dimensionality, and dynamic evolution. Although FL is a distributed machine learning paradigm that can effectively protect data privacy, it still faces problems such as slow convergence speed, poor adaptability to dynamic data and an inability to consider complex relationships between data sources when processing intelligence data [15]. Therefore, a study proposes an improved FL algorithm that combines decomposition and combination. This algorithm breaks down the global model into local sub-models, optimizes local training efficiency and combines the sub-models dynamically to enhance the global model's adaptability and accuracy. The flowchart of the improved FL algorithm combining decomposition and combination is shown in Figure 1.

To adapt complex global models to different data feature spaces, the study first decomposes the global model, with each sub-model corresponding to a data feature subspace, as shown in equation (1) [16].

$$m_k = M \cdot \mathbf{W}_k, \quad \mathbf{W}_k \in \mathbf{R}^{d \times d_k} \tag{1}$$

In equation (1), m_k represents the k th sub-model obtained after decomposition; M represents the global model; $\mathbf{W}_k$ represents the decomposition matrix; d and d_k respectively represent the dimensions of the global model and sub-models; $\mathbf{R}$ represents the set of real numbers. The decomposition matrix $\mathbf{W}_k$ is predefined based on the structural characteristics of the global model. Specifically, the decomposition is performed by partitioning the global model into multiple sub-models, each corresponding to a distinct feature subspace. This partitioning is conducted according to the layer-wise or block-wise architecture of the neural network, ensuring that each sub-model captures a specific subset of features. The decomposition matrix is not learned during training, nor is it a random projection. Instead, it is constructed as a fixed, structured matrix that maps the global model parameters to the respective sub-models. This approach allows for efficient local training and dynamic recombination while maintaining the interpretability and structural consistency of the global model. The number of sub-models used in the experiments was determined through empirical validation and sensitivity analysis. With fewer sub-models, the feature subspaces were too coarse, limiting adaptability to heterogeneous data distributions. With more sub-models, the communication and computation costs increased without significant gains in accuracy. After the model decomposition is completed, each node needs to train the sub-model based on local data. The node trains a sub-model based on local data, and its Loss Function (LF) is designed as shown in equation (2).

$$L_i(m_k) = \frac{1}{|D_i|} \sum_{(x,y)\in D_i} \left\| y - m_k(x) \right\|^2 \tag{2}$$

In equation (2), $L_i(m_k)$ represents the LF when training the sub-model on the local dataset of node i; D_i denotes the local dataset of node i; $|D_i|$ represents the amount of samples in dataset D_i; x and y respectively represent input features and real labels; $m_k(x)$ denotes the predicted output of the sub model m_k on the input x. To optimize the performance of sub models, each node needs to calculate the gradient update of model parameters [17]. After receiving local updates from all nodes, the central server needs to preliminarily integrate these updates, as shown in equation (3).

$$\begin{cases} \Delta m_k^i = \eta \cdot \nabla L_i(m_k) \\ \Delta m_k^{merge} = \dfrac{1}{N} \sum_{i=1}^{N} \Delta m_k^i \end{cases} \tag{3}$$

In equation (3), Δm_k^i refers to the parameter update amount of node i to submodel m_k; η stands for learning rate; $\nabla L_i(m_k)$ refers to the parameter gradient of the LF $L_i(m_k)$ for the submodel m_k; Δm_k^{merge} represents the merging and updating amount of sub-models by the central server; N means the total amount of nodes. To strengthen the robustness of the model, it needs to cluster local updates to eliminate the influence of noisy data. The study uses the K-means clustering (K-means) algorithm for local update clustering, and the calculation of cluster centers is shown in equation (4) [18].

$$\Delta m_k^c = \frac{1}{|S_c|} \sum_{\Delta m_k^i \in S_c} \Delta m_k^i \tag{4}$$

In equation (4), Δm_k^c represents the center of c clusters; S_c represents the collection of the c th cluster. Based on the clustering results, the algorithm needs to assign different weights to updates of different clusters to achieve dynamic combination. The new sub-model obtains equation (5) by weighting and combining the updates of each cluster center.

$$m_k^{new} = m_k + \sum_{c=1}^{C} \alpha_c \cdot \Delta m_k^c \tag{5}$$

In equation (5), m_k^{new} represents the updated sub-model; m_k represents the current sub-model parameters; α_c means the weight of the c th cluster. The design of weights takes into account both cluster size and data quality, as shown in equation (6).

$$\alpha_c = \frac{|S_c|}{\sum_{c'=1}^{C} |S_{c'}|} \cdot \exp(-\beta \cdot Var(S_c)) \tag{6}$$

In equation (6), β represents the adjustment parameter; $Var(S_c)$ represents the variance of local updates within cluster S_c, measuring the level of data noise. To further enhance the robustness of the improved FL algorithm, eliminate the influence of noisy data on model updates, and achieve dynamic adaptation of different data feature spaces, clustering and aggregation operations are studied for local updates. The schematic diagram of sub-model clustering and aggregation for improving the FL algorithm is shown in Figure 2.

In Figure 2, sub-model clustering and aggregation involve three key steps, and in the merging stage, the local sub-model updates uploaded by each node are preliminarily integrated. The clustering stage uses clustering algorithms to group the merged updates. During the aggregation stage, weights are dynamically allocated based on clustering results to form an optimized global sub model. After all sub-models are updated, they need to be recombined into a complete global model. The new global model obtains equation (7) through a linear combination of sub-models and their decomposition matrices [19].

$$M^{new} = \sum_{k=1}^{K} m_k^{new} \cdot W^T \tag{7}$$

In equation (7), M^{new} represents the updated global model; W^T represents the transpose matrix of the decomposition matrix, used to map sub-models to the global model space; K represents the total number of sub-models. The iterative process of the algorithm requires monitoring the global LF to determine whether it converges, as shown in equation (8).

$$L_{global} = \frac{1}{K} \sum_{k=1}^{K} L(m_k^{new}) \tag{8}$$

In equation (8), L_{global} means the global LF; $L(m_k^{new})$ means the local LF of the k th sub-model. Finally, the optimized model needs to be properly saved for future use. The model archiving operation is achieved by adding new models to the archive set.

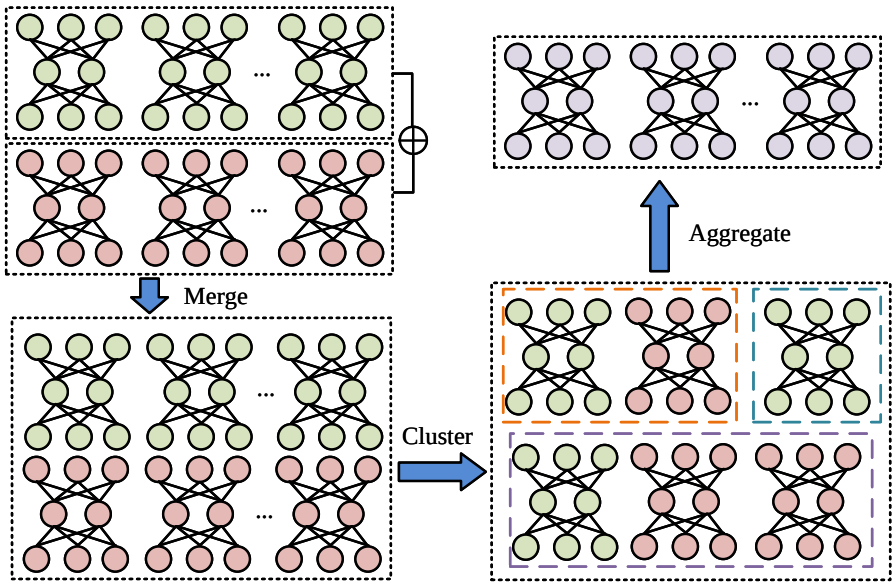


Figure 2: Schematic diagram of sub-model clustering and aggregation for improving FL algorithm.

2.2 Intelligence dynamic ensemble clustering method based on improved joint pre-training

Although the improved FL algorithm improves the adaptability and convergence speed of the model through decomposition and combination techniques, the multi-source heterogeneity and high-dimensional characteristics of intelligence data still pose higher requirements for data integration and clustering [20]. To further optimize feature extraction and pattern discovery, a dynamic intelligence data integration and clustering method based on improved joint pre-training and hierarchical similarity is proposed. This method enhances the initialization ability of the model through pre-training mechanisms and utilizes hierarchical similarity clustering techniques to achieve efficient grouping and personalized modeling of data. In the decomposition and construction of the intelligence dynamic data integration model, the study first introduces a pre-training mechanism, whose objective function is shown in equation (9).

$$L_{pre} = \lambda \cdot \left\| M - M_{base} \right\|^2 + \frac{1}{N} \sum_{i=1}^{N} L_i(m_k) \tag{9}$$

In equation (9), L_{pre} represents the pre-trained additional LF utilized to constrain the differences between the global model and the baseline model; M_{base} represents a predefined benchmark model; λ represents the adjustment coefficient. The benchmark model M_{base} is defined as a model pre-trained on a publicly available dataset that shares similar feature characteristics with the target intelligence data, but contains no overlapping samples or private information. This model is used to provide a robust initialization, leveraging transfer learning to enhance convergence and stability, especially in environments with non-IID data distributions. The use of a publicly pre-trained model as the benchmark was motivated by its ability to offer a generalized feature representation, thereby improving the initial performance of the global model without introducing bias from any specific client or prior federated training round. This approach ensures fairness and supports faster adaptation to heterogeneous local data. In the local training phase, each node dynamically extracts features based on local data and projects the data onto a shared feature space through a feature mapping matrix [21]. The feature mapping process is shown in equation (10).

$$z_i = W_i \cdot x_i + b_i \tag{10}$$

In equation (10), z_i represents the feature vector extracted by node i; x_i and W_i represent the local input data and feature mapping matrix of node i, respectively; b_i represents the bias term. The feature mapping matrix W_i is precisely defined as a learned parameter matrix. It is not a random projection. For each client, the matrix is optimized during the local training phase to project the local input data into a shared feature space. This learning process is performed collaboratively across clients within the FL framework, with the goal of aligning the feature representations from different clients to facilitate effective model aggregation and improve overall performance. The decomposition and construction diagram of the intelligence dynamic data integration model is shown in Figure 3.

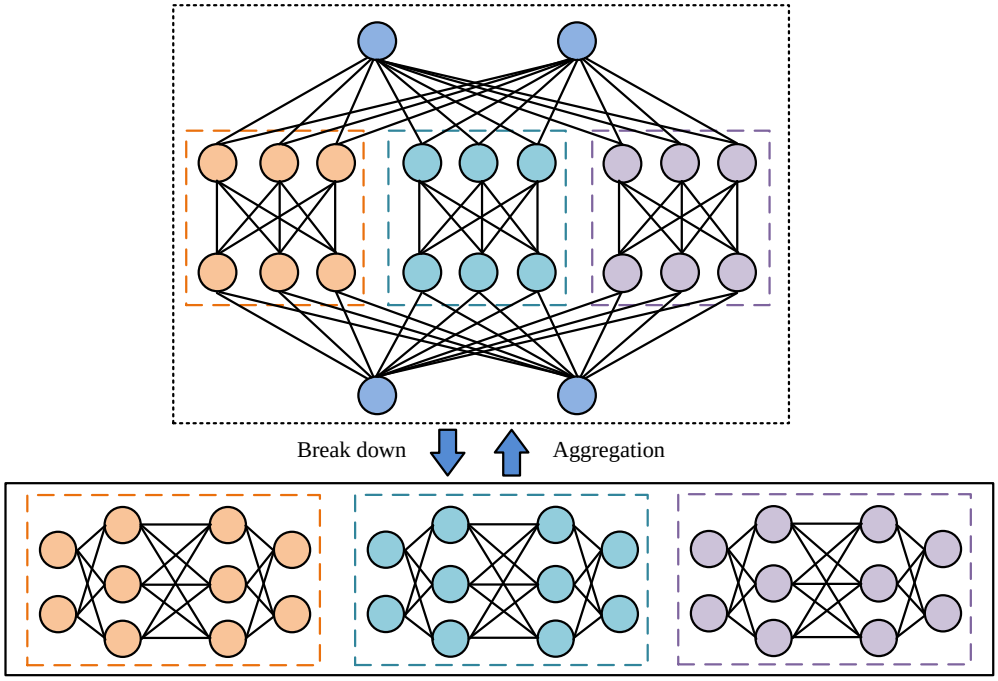


Figure 3: Decomposition and construction diagram of intelligence dynamic data integration model.

Aiming at the heterogeneity of client data, a hierarchical clustering algorithm based on cosine similarity is proposed to calculate the similarity matrix of client data distribution, as shown in equation (11) [22].

$$S_{uv} = \frac{D_u \cdot D_v}{\|D_u\| \cdot \|D_v\|} \tag{11}$$

In equation (11), S_{uv} represents the cosine similarity between client u and client v; D_u represents the data feature vector of client u; D_v represents the corresponding feature vector of client v; $\|D_u\|$ represents the Euclidean norm of vector D_u. Hierarchical clustering is primarily used in the initial phase to form a dendrogram, providing insights into the potential number of clusters and the multi-level data structure. However, the final client grouping is determined by the Spherical K-means algorithm, which operates directly on the normalized feature vectors. Spherical K-means is chosen as the final clustering driver due to its efficiency and compatibility with cosine similarity on normalized data, ensuring clients are partitioned into hyperspherical clusters. Further research is conducted using the spherical K-means algorithm for grouping, with the objective function shown in equation (12) [23].

$$C_o = \arg\min_h \sum_{u \in h} (1 - S_{uh}) \tag{12}$$

In equation (12), C_o represents the central client of the o th cluster; h stands for candidate center client; S_{uh} refers to the cosine similarity between client u and the current center h. After obtaining the client group, it is necessary to design a hierarchical model aggregation strategy [24]. To achieve more refined personalization, the intra group client model is fine tuned and its calculation is shown in equation (13).

$$\begin{cases} M_g = \sum_{u \in G} w_u \cdot m_u \\ m_u^{per} = M_g + \varepsilon \cdot \nabla L_u(M_g) \end{cases} \tag{13}$$

In equation (13), M_g represents the global model generated by aggregation; G means the set of clients in the current group; m_u and w_u respectively represent the local model and aggregation weights of client u; m_u^{per} represents the personalized model of the client; ε represents fine-tuning step size; $\nabla L_u(M_g)$ represents the gradient of the local LF L_u of client u on the global model M_g. The schematic diagram of intelligence dynamic data clustering is shown in Figure 4.

In Figure 4, the sub-model is selected through client selection and used for local training. During the local training phase, each client trains based on the selected model and local data to optimize model performance. Considering the timeliness of intelligence data, it is necessary to dynamically adjust the model weights. The temporal decay strategy is implemented to address the concept drift and potential data quality degradation that may occur in dynamic intelligence data environments. The primary rationale is to gradually reduce the influence of clients that have not provided recent updates, as their local models might become less representative of the current global data distribution over time. The weight update strategy for time decay is shown in equation (14) [25].

$$w_u(t) = w_u \cdot e^{-\gamma t} \tag{14}$$

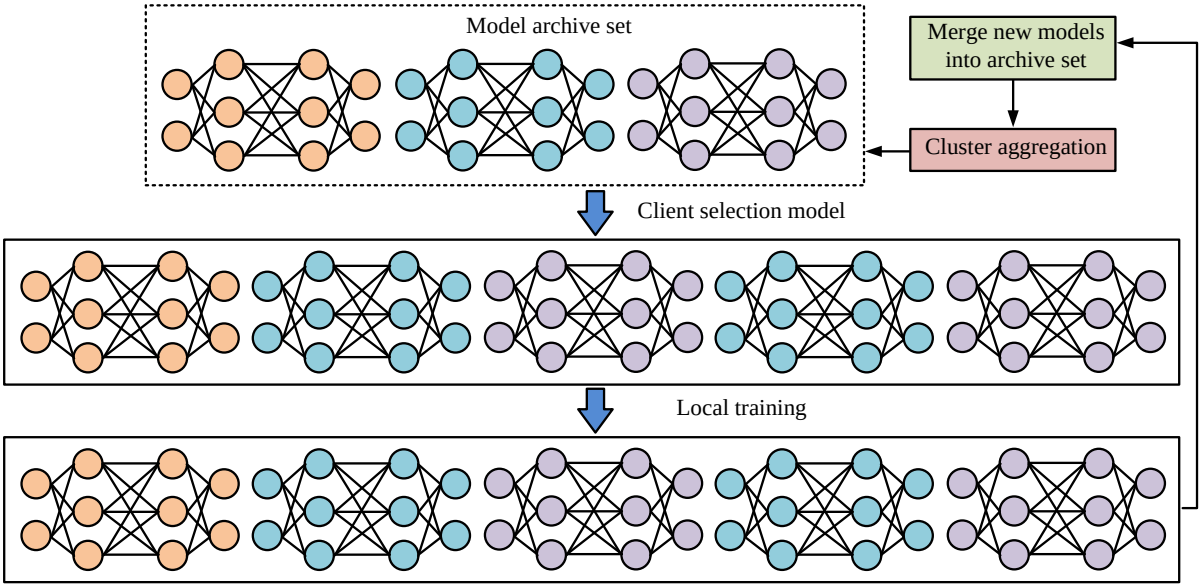


Figure 4: Schematic diagram of intelligence dynamic data clustering.

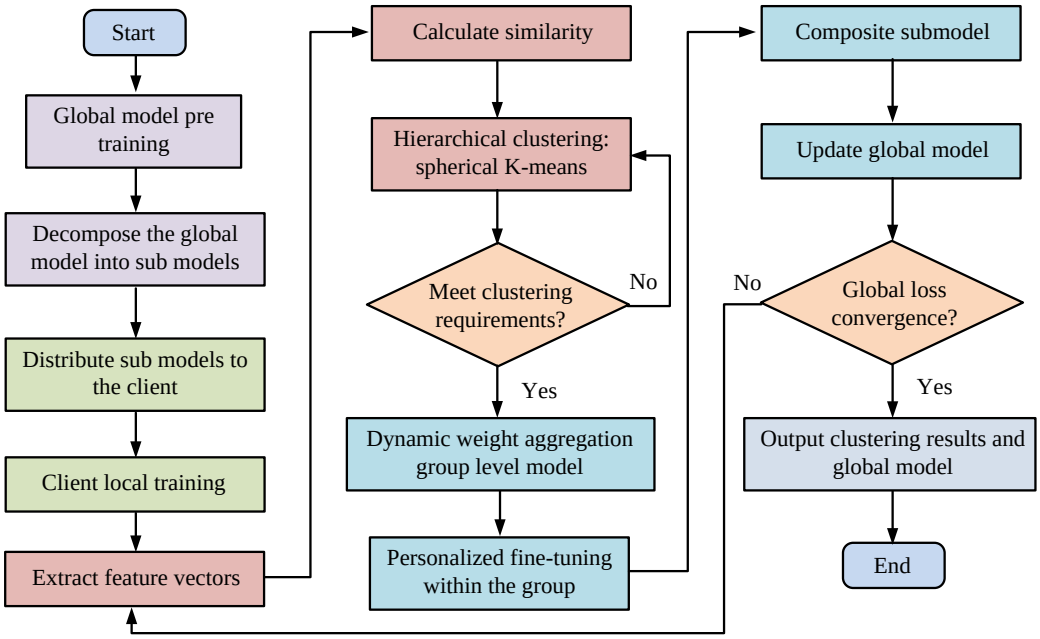


Figure 5: Flow chart of intelligence dynamic data integration and clustering based on improved joint pre-training and hierarchical similarity.

In equation (14), $w_u(t)$ represents the dynamic weight of client u at time t; w_u represents the initial weight of client u; γ represents attenuation coefficient; t represents the time variable. Among them, the calculation of sub-model weights needs to reflect data quality, and its expression is shown in equation (15).

$$\beta_k = (\sum_{u \in S_k} \|D_u\|) / (\sum_{v=1}^{K} \sum_{u \in S_v} \|D_u\|) \quad (15)$$

In equation (15), β_k represents the weight of sub model k; S_k represents the set of clients belonging to the sub model k. The intelligent dynamic data integration and clustering process based on improved joint pre-training and hierarchical similarity is shown in Figure 5.

The pseudocode of the proposed method is as follows.

Algorithm 1: Dynamic Sub-Model Aggregation and Clustering via Hierarchical Federated Learning with Pre-Training

Input: Number of clients, total rounds, number of sub-models, clustering epochs, historical dataset
Output: Final global model
/ **Step 1: Pre-training Phase**
1: **Initialize** global model by pre-training on historical dataset
// **Step 2: Federated Learning Rounds**

```
2: for round t=1to total rounds do
3:     // Server executes:
4:     Decompose global model into sub-models
5:     Send relevant sub-models to a subset of active clients
6:     // Client execution (in parallel):
7:     for each client in selected clients do
8:         Train received sub-model on local data
9:         Compute feature vector and upload model update
10:    end for
11:    // Server executes:
12:    Perform preliminary integration of updates
13:    if t mod Ec=0 then
14:        Calculate client similarity matrix
15:        Perform hierarchical client clustering via Spherical K-means
16:    end if
17:    Cluster client updates via K-means
18:    Dynamically combine sub-models
19:    Apply temporal weight decay
20:    Recombine sub-models into global model
21:    Evaluate global loss
22: end for
23: return final global model
```

3 Results

Firstly, the performance of the improved FL algorithm was analyzed to verify the advantages of decomposition and combination mechanisms in improving model convergence speed and adapting to dynamic data. Then, the performance of intelligence dynamic data integration and clustering methods was evaluated.

3.1 Performance validation of improved FL algorithm

To prove the effect of the improved FL algorithm, experiments were carried out on two common datasets, MNIST and CIFAR-10. The MNIST dataset contains 70000 handwritten digit images, split into 10 categories, with 60000 for training and 10000 for testing. The CIFAR-10 dataset contains 60000 color images, split into 10 categories, with 50000 for training and 10000 for testing. The number of algorithm iterations was 400. For the selection of attenuation coefficient, sensitivity analysis was conducted by changing its value, and the results are shown in Table 2. In Table 2, the attenuation coefficient has a significant impact on model performance and participation fairness. When the attenuation coefficient was 0.05, the model achieved the highest accuracy of 98.69% and 90.26% on the MNIST and CIFAR-10 datasets, respectively. At this time, the client dropout rate was 5.1% and the fairness index was 0.89. When the attenuation coefficient increased to 0.20, the accuracy decreased to 97.33% and 86.41% respectively, and the fairness index dropped to 0.61. The results indicate that when the attenuation coefficient is 0.05, the model achieves the best balance between accuracy, dropout rate, and fairness, and is the recommended optimal parameter.

To substantiate the use of K-means clustering for update grouping, a comparative experiment was conducted under varying levels of simulated data noise. The proposed method was compared against the standard FedAvg aggregation. The clustering quality was quantitatively assessed using the Silhouette Score, and the model's robustness was evaluated by its performance on a clean test set. The results are shown in Table 3. The results confirm that as noise levels increase, the K-means-based aggregation mechanism effectively identifies and isolates anomalous updates into separate clusters. This is evidenced by the maintenance of a high Silhouette Score and a lower Intra-cluster Distance for the dominant cluster, indicating coherent grouping of reliable updates. The clustering metrics provide clear empirical evidence that the K-means grouping enhances robustness by prioritizing the aggregation of updates from clients with consistent and trustworthy data distributions.

Table 2: Sensitivity analysis results of decay coefficient.

Decay coefficient	MNIST Accuracy (%)	CIFAR-10 Accuracy (%)	Client Dropout Rate (%)	Participation Fairness Index
0.01	98.45	89.12	3.2	0.92
0.03	98.61	89.87	4.2	0.90
0.05	98.69	90.26	5.1	0.89
0.10	98.12	88.95	12.7	0.78
0.15	97.68	87.23	19.3	0.70
0.20	97.33	86.41	28.4	0.61

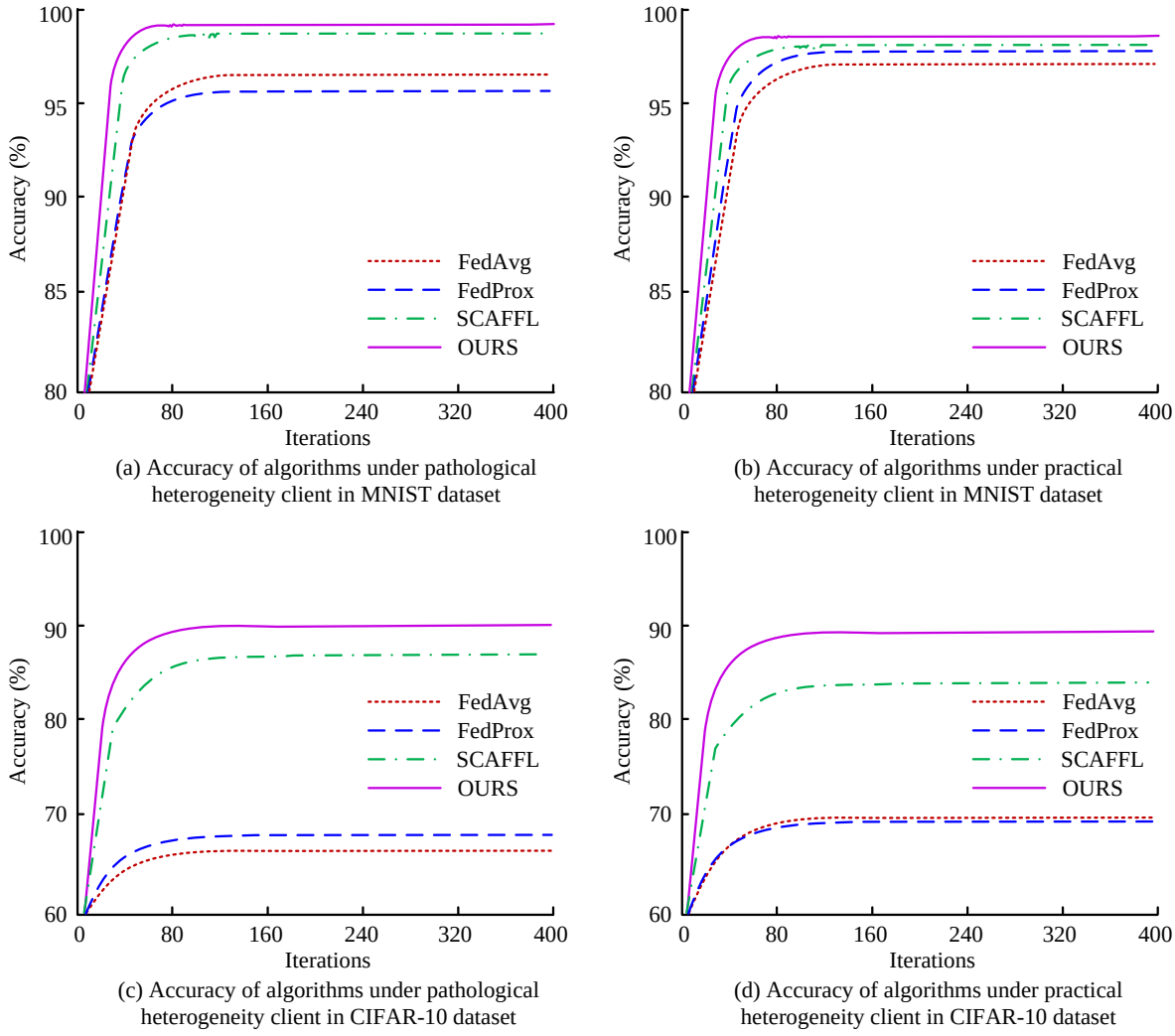
Table 3: Performance and clustering quality comparison under noisy conditions.

Noise Level	Aggregation Method	Final Test Accuracy (%)	Avg. Silhouette Score	Intra-cluster Distance (Majority Cluster)
10%	FedAvg (Baseline)	97.85	N/A	N/A
	Proposed (K-means)	98.41	0.72	0.15

20%	FedAvg (Baseline)	95.12	N/A	N/A
	Proposed (K-means)	97.56	0.68	0.18
30%	FedAvg (Baseline)	90.33	N/A	N/A
	Proposed (K-means)	95.88	0.61	0.23
40%	FedAvg (Baseline)	83.47	N/A	N/A
	Proposed (K-means)	92.15	0.55	0.29

Table 4: Experimental environments and parameters.

Experimental environments		Parameters	
Names	Configuration	Names	Values
Graphics processing unit	NVIDIA Tesla V100	Learning rate	0.01
Central processing unit	Intel Xeon Gold 6248R	Number of clusters	3
Memory	64GB DDR4	Number of clients	20
Operating System	Windows 10	Batch size	32
Deep learning framework	PyTorch 1.10	Number of sub-models	5
Programming language	Python 3.8	Decay coefficient	0.05



(a) Accuracy of algorithms under pathological heterogeneity client in MNIST dataset

(b) Accuracy of algorithms under practical heterogeneity client in MNIST dataset

(c) Accuracy of algorithms under pathological heterogeneity client in CIFAR-10 dataset

(d) Accuracy of algorithms under practical heterogeneity client in CIFAR-10 dataset

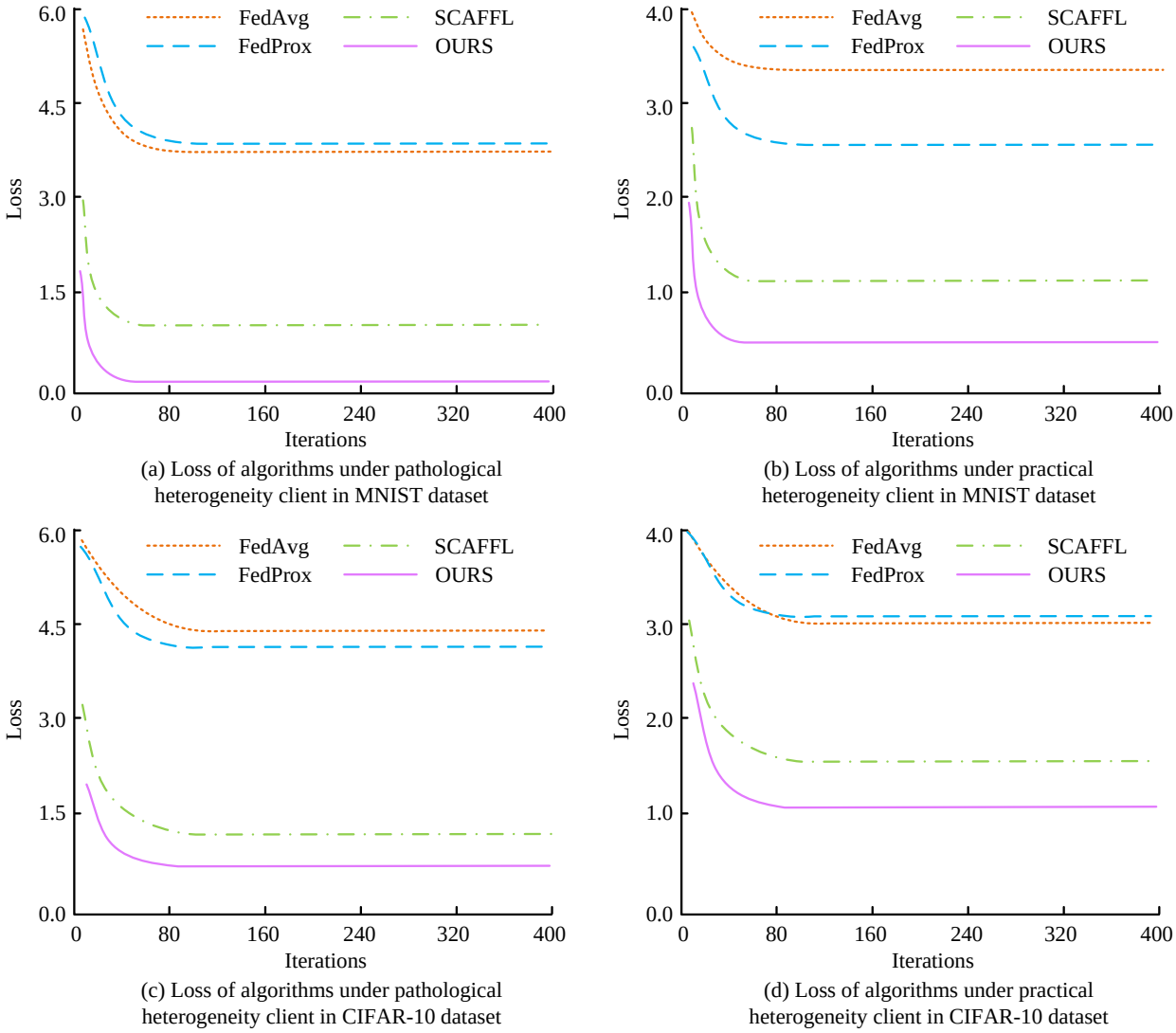
Figure 6: Accuracy of algorithms under two heterogeneous clients in MNIST and CIFAR-10 datasets.

The experimental environment and parameters are denoted in Table 4.

To prove the robustness of the designed algorithm, the accuracy of the algorithm was analyzed on two heterogeneous clients, pathology and real-world scenarios, in the MNIST and CIFAR-10 datasets. Compared with current mainstream algorithms, including FedAvg, FedProx, and Stochastic Controlled Averaging for Federated Learning (ScaFFL), the findings are denoted in Figure 6. In Figure 6 (a), in the MNIST dataset, with an iteration of 160, the accuracy of FedAvg, FedProx, SCAFFL, and the proposed algorithm on the pathological heterogeneity client were 96.24%, 95.16%, 97.68%, and 98.69%, respectively. In Figure 6 (b), when the iteration number was 160, the accuracy of the four algorithms in the MNIST dataset was 96.58%, 97.10%, 97.35%, and

98.14%, respectively, under the actual scenario of heterogeneous clients. In Figure 6 (c), in the CIFAR-10 dataset, with an iteration of 160, the accuracy of FedAvg, FedProx, and SCAFFL under pathological heterogeneity client was 65.37%, 67.69%, and 86.74%, respectively. The accuracy of the proposed algorithm was 90.26%. In Figure 6 (d), under the actual scenario of heterogeneous clients and with 160 iterations, the accuracy of the four algorithms in the CIFAR-10 dataset was 69.38%, 69.12%, 83.56%, and 89.87%, respectively. The findings show that the designed algorithm exhibits higher accuracy and robustness in different data distributions and scenarios.

The loss of different comparison algorithms was analyzed in the MNIST and CIFAR-10 datasets to validate the convergence of the designed algorithm. The findings are denoted in Figure 7. In Figure 7 (a), under the pathological heterogeneity client, when the iteration number was 160, the losses of FedAvg, FedProx, SCAFFL, and the proposed algorithm in the MNIST dataset were 0.348, 0.352, 1.025, and 0.113, respectively. In Figure 7 (b), under the actual scenario of heterogeneous clients, when the iteration number was 160, the losses of the four algorithms were 0.341, 0.326, 1.104, and 0.505, respectively. In Figure 7 (c), in the CIFAR-10 dataset, at an iteration of 160, the losses of the four algorithms under pathological heterogeneity clients were 4.472, 4.210, 1.389, and 0.857, respectively. In Figure 7 (d), under the actual scenario of heterogeneous clients, when the iteration number was 160, the losses of FedAvg, FedProx, and SCAFFL were 3.021, 3.098, and 1.610, respectively, and the loss of the proposed algorithm was 1.024. The findings demonstrate that the designed algorithm can improve the convergence speed of the model and achieve lower losses under different data distributions.



(a) Loss of algorithms under pathological heterogeneity client in MNIST dataset

(b) Loss of algorithms under practical heterogeneity client in MNIST dataset

(c) Loss of algorithms under pathological heterogeneity client in CIFAR-10 dataset

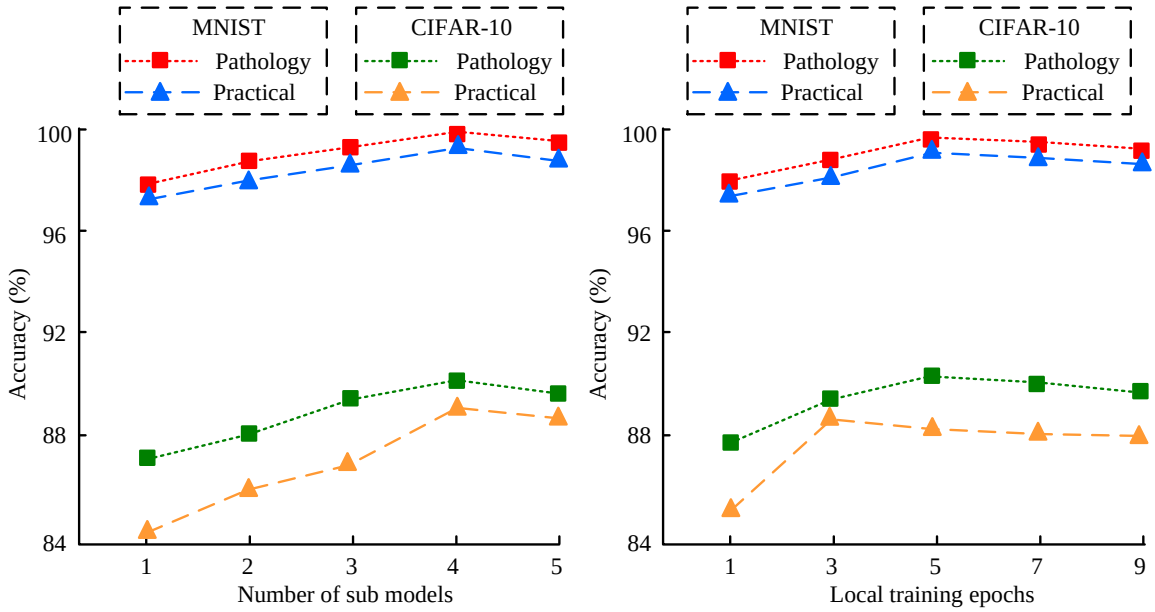
(d) Loss of algorithms under practical heterogeneity client in CIFAR-10 dataset

Figure 7: Loss of algorithms under two heterogeneous clients in MNIST and CIFAR-10 datasets.

Table 5: Comprehensive comparison of model performance and statistical significance.

Dataset	Methods	Pathological Heterogeneity			Real-world Heterogeneity		
		95% CI (Accuracy)	95% CI (Loss)	p-value	95% CI (Accuracy)	95% CI (Loss)	p-value
MNIST	FedAvg	[95.98, 96.44]	[0.335, 0.361]	<0.01	[96.34, 96.76]	[0.328, 0.354]	<0.01
	FedProx	[94.83, 95.45]	[0.339, 0.365]	<0.01	[96.85, 97.31]	[0.313, 0.339]	<0.01
	SCAFFL	[97.47, 97.83]	[1.001, 1.049]	<0.01	[97.13, 97.51]	[1.080, 1.128]	<0.01
	OURS	[98.52, 98.82]	[0.105, 0.121]	/	[98.00, 98.24]	[0.490, 0.520]	/
CIFAR-10	FedAvg	[64.68, 66.00]	[4.350, 4.594]	<0.01	[68.84, 69.86]	[2.909, 3.133]	<0.01
	FedProx	[67.07, 68.23]	[4.098, 4.322]	<0.01	[68.54, 69.64]	[2.986, 3.210]	<0.01
	SCAFFL	[86.31, 87.11]	[1.367, 1.411]	<0.01	[83.09, 83.97]	[1.588, 1.632]	<0.01
	OURS	[89.97, 90.51]	[0.841, 0.873]	/	[89.57, 90.13]	[1.010, 1.038]	/

Note: p<0.01 indicates reaching a highly significant level.



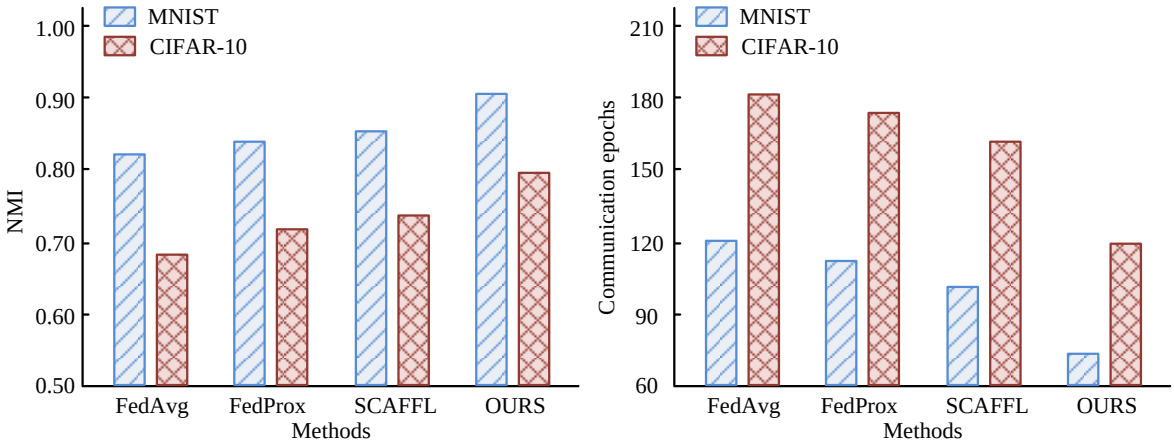
(a) The impact of different numbers of sub models on the accuracy of the algorithm

(b) The impact of different local training epochs on the accuracy of the algorithm

Figure 8: The impact of different numbers of sub models and local training epochs on the accuracy of improved algorithms.

This study conducted a statistical analysis of the accuracy loss of the proposed algorithm under two heterogeneous clients, pathology and real-world scenarios, in the MNIST and CIFAR-10 datasets. The results are shown in Table 5. The statistical results show that the method proposed in this study significantly outperforms the baseline algorithm in both datasets and heterogeneous scenarios. On the MNIST dataset, the accuracy and 95% confidence interval of the proposed method were [98.52, 98.82] and [98.00, 98.24], respectively, demonstrating applicability to heterogeneous pathology and real-world scenarios, and the 95% confidence intervals for loss values were [0.105, 0.121] and [0.490, 0.520], respectively. On the more complex CIFAR-10 dataset, the proposed method also outperformed all baselines in terms of confidence intervals. All p-values compared were less than 0.001, indicating that the performance improvement is highly statistically significant.

A comparative analysis was conducted on the accuracy of the MNIST and CIFAR-10 datasets under different numbers of sub-models and local training epochs, to investigate the impact of these factors on the performance improvement of the proposed algorithm. The findings are denoted in Figure 8. In Figure 8 (a), under the pathological heterogeneity client, when the number of sub-models was 4, the accuracy of the designed algorithm in the two datasets was 99.82% and 90.33%, respectively. In actual heterogeneous client scenarios, their accuracy rates were 99.47% and 89.11%, respectively. In Figure 8 (b), when the local training epochs were 5, the accuracy of the proposed algorithm in two datasets was 99.86% and 90.62% respectively under pathological heterogeneity client. In actual scenarios with heterogeneous clients, the accuracy rates were 99.43% and 88.94%, respectively. Research has found that increasing the number of sub-models and local training rounds can improve algorithm performance.



(a) Comparison of NMI using different methods on two datasets

(b) Comparison of communication epochs using different methods on two datasets

Figure 9: Integration quality and clustering efficiency of different methods on two datasets.

3.2 Performance evaluation of intelligence dynamic data integration and clustering methods

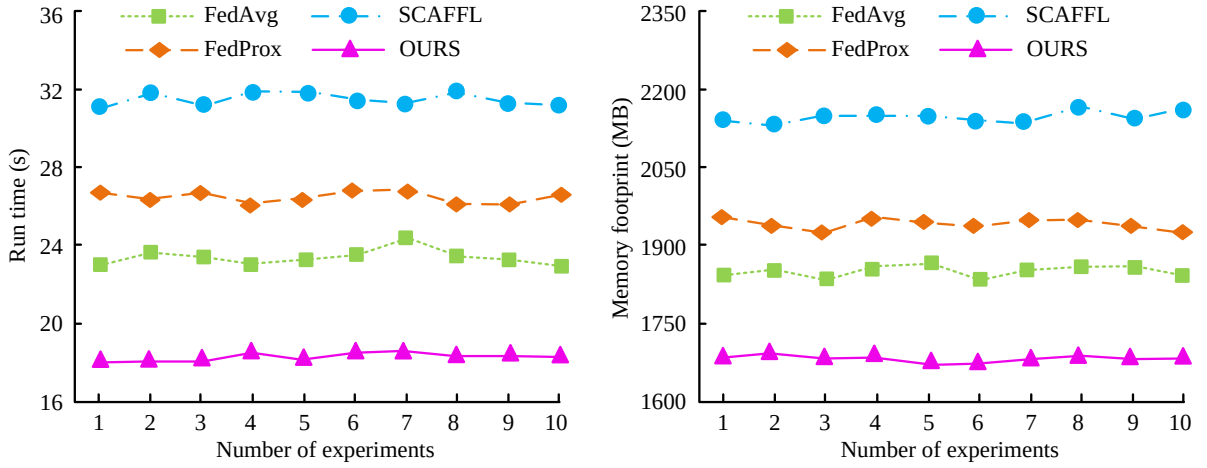
The ensemble quality and clustering efficiency of the proposed method were analyzed in the MNIST and CIFAR-10 datasets, and compared with FedAvg, FedProx, and SCAFFL. The indicator for evaluating integration quality was Normalized Mutual Information (NMI), and the clustering efficiency indicator was communication epochs. The findings are denoted in Figure 9. In Figure 9 (a), in the MNIST dataset, the NMIs of FedAvg, FedProx, SCAFFL, and the designed algorithm were 0.82, 0.84, 0.86, and 0.91, respectively. The NMIs in the CIFAR-10 dataset were 0.68, 0.71, 0.73, and 0.79, respectively. In Figure 9 (b), the communication epochs of FedAvg, FedProx, and SCAFFL in the MNIST dataset were 121, 112, and 101, respectively, and in the CIFAR-10 dataset were 179, 171, and 162, respectively. Compared with it, the proposed algorithm had 75 and 119 communication epochs in the two datasets, respectively. The findings show that the designed algorithm can ensure high integration quality while reducing communication overhead, verifying its efficiency in dynamic data integration and clustering tasks.

Ablation experiments were conducted to verify the contribution of each core module to the overall performance of the dynamic data integration and clustering methods. The findings are presented in Table 6. In ablation analysis, the roles of each module are as follows: the decomposition and combination mechanism adapts to heterogeneous data distributions by dividing sub models; The pre-training mechanism utilizes historical features to optimize model initialization and accelerate convergence; Hierarchical clustering identifies intrinsic relationships between clients through a multi-level structure, improving grouping stability and aggregation quality, which is superior to methods that rely solely on planar partitioning; The dynamic weighting mechanism adjusts client contributions based on data timeliness to alleviate concept drift. In Table 6, on the MNIST dataset, the F1 score of the complete method was 0.924 and the precision was 0.931, significantly higher than other configurations. On the CIFAR-10 dataset, the F1 score and precision of the complete method were 0.802 and 0.810, respectively. The F1 score and precision of the baseline method were the lowest, with values of 0.852 and 0.806 in the MNIST dataset and 0.723 and 0.730 in the CIFAR-10 dataset, respectively. The findings denote that each core module contributes significantly to the improvement of model performance.

The study analyzed the running time and memory usage of the designed method in a practical scenario of a medical IoT, and compared it with other methods. The findings are denoted in Figure 10. In Figure 10 (a), the average running times of FedAvg, FedProx, SCAFFL, and the proposed algorithm were 23.42s, 26.75s, 31.52s, and 18.23s, respectively. Compared with the comparative algorithm, the running time of the proposed algorithm was reduced by 22.16%, 31.85%, and 42.16%, respectively. In Figure 10 (b), the average memory usage of FedAvg, FedProx, and SCAFFL was 1853MB, 1926MB, and 2157MB, respectively. Compared with them, the proposed algorithm had an average memory usage of 1681MB, which was reduced by 9.28%, 12.72%, and 22.07%, respectively. The outcomes demonstrat that the designed algorithm can substantially decrease the consumption of computational resources while maintaining performance, thereby verifying its practicality and deployment benefits in IoT scenarios with limited resources.

Table 6: Results of ablation experiment.

Method configuration	MNIST dataset				CIFAR-10 dataset			
	Precision	Recall	F1 score	(*p*-value vs. FedAvg)	Precision	Recall	F1 score	(p-value vs. FedAvg)
Complete method	0.931	0.918	0.924	<0.001	0.810	0.795	0.802	<0.001
No decomposition combination	0.879	0.863	0.871	<0.001	0.742	0.728	0.735	<0.001
No pre training	0.900	0.886	0.893	<0.001	0.775	0.761	0.768	<0.001
Non hierarchical clustering	0.893	0.879	0.886	<0.001	0.760	0.745	0.752	<0.001
No dynamic weight	0.908	0.894	0.901	<0.001	0.786	0.772	0.779	<0.001
Baseline method (FedAvg)	0.806	0.845	0.852	/	0.730	0.716	0.723	/

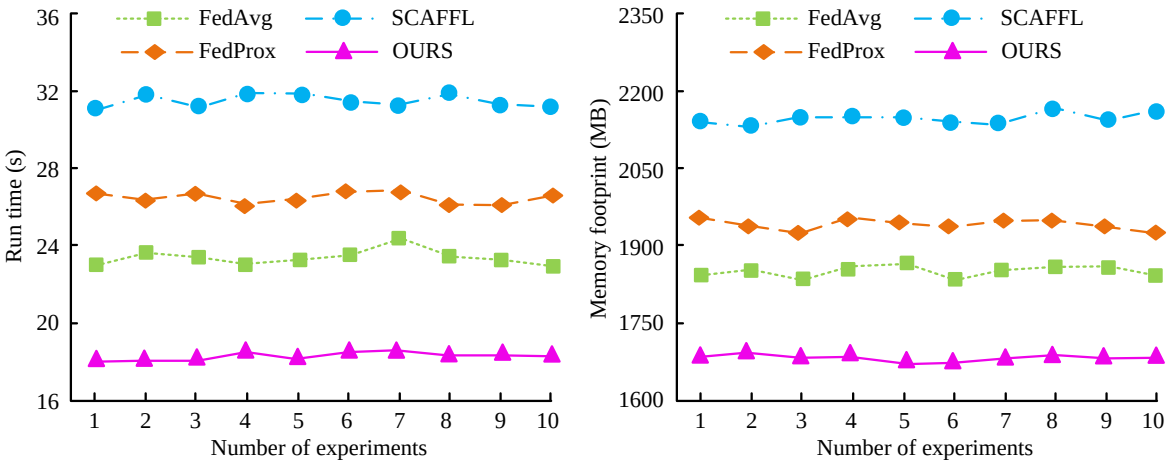


(a) Comparison of run time of different methods in practical scenarios

(b) Comparison of memory usage of different methods in practical scenarios

Figure 10: Comparison of runtime and memory usage of different methods in practical scenarios.

To verify the effectiveness of the proposed method in practical datasets, an analysis was conducted on the running time and average memory of different methods under different numbers of clients in the PhysioNet dataset. The PhysioNet dataset is a widely recognized real-world clinical time series dataset containing records of 12000 ICU patients. The results are shown in Figure 11. In Figure 11 (a), when the number of clients was 20, the running times of FedAvg, FedProx, and SCAFFL were 28.45s, 32.11s, and 38.94s, respectively, and the running time of the proposed method was 21.08s. When the number of clients increased to 100, the running times of the four methods were 62.34s, 71.89s, 88.56s, and 45.12s, respectively. In Figure 11 (b), when the number of clients was 20, the average memory of FedAvg, FedProx, SCAFFL, and the proposed method was 2105MB, 2189MB, 2455MB, and 1950MB, respectively. When the number of clients reached 100, the average memory was 3521MB, 3744MB, and 1950MB, respectively. B, 4455MB, and 2850MB. The results show that the proposed method effectively reduces computational and storage costs on real medical datasets, demonstrating superior scalability and practicality.



(a) Comparison of run time of different methods in practical scenarios

(b) Comparison of memory usage of different methods in practical scenarios

Figure 11: Run time and average memory of different methods under different numbers of clients.

Table 7: Model complexity comparison across different datasets.

Methods	MNIST	CIFAR-10	PhysioNet	MNIST	CIFAR-10	PhysioNet
	Number of Parameters (Millions)			FLOPs (MegaFLOPs)		
FedAvg	4.21	23.45	1.85	8.45	125.67	3.72
FedProx	4.21	23.47	1.85	8.47	125.72	3.73
SCAFFL	4.26	23.51	1.87	8.52	126.05	3.76
OURS	5.18	25.83	2.21	10.31	135.42	4.45

Further analysis was conducted on the parameter count and Floating-Point Operations (FLOPs) of different methods on the MNIST, CIFAR-10, and PhysioNet datasets. The results are shown in Table 7. In Table 7, on the MNIST dataset, the proposed method had a parameter size of 5.18M and FLOPs of 10.31M, which were approximately 23% and 22% higher than FedAvg, respectively. The complexity of the proposed method on the real clinical dataset PhysioNet also maintained a similar increase. This controllable increase in complexity, compared to the specific performance improvements obtained in the previous experiments, demonstrates that the proposed method achieves a good balance between efficiency and performance.

4 Discussion

In this study, a dynamic integration and clustering method for intelligence data based on an improved FL algorithm was proposed. The experimental results demonstrated significant improvements in accuracy, convergence speed, and communication efficiency compared to several existing approaches.

In terms of accuracy, this method achieved accuracies of 98.69% and 90.26% respectively on the MNIST and CIFAR-10 datasets for pathological heterogeneous clients, and 98.14% and 89.87% respectively in real-world heterogeneous clients. Compared with the real-time medical data processing method proposed in reference [6], the accuracy of this method was improved by about 6-8 percentage points. This improvement is mainly due to the dynamic sub model aggregation mechanism, which decomposes the global model into multiple specialized sub models, enabling the model to better adapt to the data distribution characteristics of different clients.

In terms of convergence performance, this method only required 75 communication rounds to converge on the MNIST dataset and 119 communication rounds on the CIFAR-10 dataset. In contrast, the privacy preserving FL method in reference [10] required 121 and 179 communication epochs respectively in similar tasks. The improvement in convergence speed is mainly due to the introduction of pre-training mechanisms, which utilize historical data feature information for model initialization, enabling the model to have a good parameter foundation in the early stages of training, thereby accelerating the training process.

In terms of communication efficiency, this method had a running time of 18.23 seconds and a memory usage of 1681MB in actual medical IoT scenario testing. Compared with the method in reference [12], the running time was reduced by 27% and the memory usage was reduced by 12%. This improvement is due to the application of hierarchical similarity clustering technology, which significantly reduces unnecessary communication overhead by intelligently grouping clients.

However, this method also has some limitations. Firstly, due to the adoption of a multi submodel architecture and clustering process, its computational complexity is relatively high, which may limit its application in resource constrained environments. Secondly, the method is sensitive to hyperparameter settings, especially the selection of the number of clusters K and learning rate, which can significantly affect performance. Compared with the heterogeneous IoT FL framework in reference [13], this method improved

accuracy by about 4% on the CIFAR-10 dataset, but increased computational load by about 15%. This indicates that while pursuing performance improvement, a balance needs to be struck between accuracy and computational efficiency. Compared with the TEE based method in reference [8], although this method avoided the dependence on dedicated hardware, it may be slightly inadequate in combating advanced security threats.

In summary, the proposed method has achieved significant improvements in accuracy, convergence speed, and communication efficiency through the organic combination of dynamic submodel aggregation, pre training mechanism, and hierarchical clustering. Future research will focus on developing adaptive hyperparameter optimization strategies, reducing computational complexity, and exploring decentralized aggregation mechanisms to enhance system robustness.

5 Conclusion

The study proposes a dynamic integration and clustering method for intelligence data based on an improved FL algorithm to strengthen the effectiveness of the FL algorithm in intelligence data processing. The findings denote that the designed method can effectively solve the challenges of slow convergence speed, poor adaptability to dynamic data, and low clustering efficiency in FL. It not only improved precision and convergence speed, but also reduced computational resource consumption, making it suitable for practical application scenarios such as medical IoT.

6 Funding

The research is supported by: the Science and Technology Project of State Grid Shanxi Electric Power Company “Research and Application of Key Technologies for Information Extraction and Association Based on Federated Learning Large Models” (Grant number: 5205M024000K).

References

[1] Iqbal H. Sarker. Machine learning for intelligent data analysis and automation in cybersecurity: Current and future prospects. Annals of Data Science, 10(6):1473-1498, 2023. https://doi.org/10.1007/s40745-022-00444-2

[2] Zhijuan Zong, and Yu Guan. AI-driven intelligent data analytics and predictive analysis in Industry 4.0: Transforming knowledge, innovation, and efficiency. Journal of the Knowledge Economy, 16(1):864-903, 2025. https://doi.org/10.1007/s13132-024-02001-z

[3] Lei Ren, Yingjie Li, Xiaokang Wang, Jin Cui, and Lin Zhang. An ABGE-aided manufacturing knowledge graph construction approach for heterogeneous IIoT data integration. International Journal of Production Research, 61(12):4102-4116, 2023. https://doi.org/10.1080/00207543.2022.2042416

[4] Jie Wen, Zhixia Zhang, Yang Lan, Zhihua Cui, Jianghui Cai, and Wensheng Zhang. A survey on federated learning: challenges and applications. International Journal of Machine Learning and Cybernetics, 14(2):513-535, 2023. https://doi.org/10.1007/s13042-022-01647-y

[5] Qiang Yang, Anbu Huang, Lixin Fan, Chee Seng Chan, Jian Han Lim, Kam Woh Ng, Ding Sheng Ong, and Bowen Li. Federated learning with privacy-preserving and model IP-right-protection. Machine Intelligence Research, 20(1):19-37, 2023. https://doi.org/10.1007/s11633-022-1343-2

[6] Kehua Guo, Tianyu Chen, Sheng Ren, Nan Li, Min Hu, and Jian Kang. Federated learning empowered real-time medical data processing method for smart healthcare. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 21(4):869-879, 2022. https://doi.org/10.1109/TCBB.2022.3185395

[7] Tomer Gafni, Nir Shlezinger, Kobi Cohen, Yonina C. Eldar, and H. Vincent Poor. Federated learning: A signal processing perspective. IEEE Signal Processing Magazine, 39(3):14-41, 2022. https://doi.org/10.1109/MSP.2021.3125282

[8] Abbas Yazdinejad, Ali Dehghantanha, and Gautam Srivastava. AP2FL: Auditable privacy-preserving federated learning framework for electronics in healthcare. IEEE Transactions on Consumer Electronics, 70(1):2527-2535, 2023. https://doi.org/10.1109/TCE.2023.3318509

[9] Guanming Bao, and Ping Guo. Federated learning in cloud-edge collaborative architecture: Key technologies, applications and challenges. Journal of Cloud Computing, 11(1):94-115, 2022. https://doi.org/10.1186/s13677-022-00377-4

[10] Ruijin Wang, Jinshan Lai, Zhiyang Zhang, Xiong Li, Pandi Vijayakumar, and Marimuthu Karuppiah. Privacy-preserving federated learning for internet of medical things under edge computing. IEEE Journal of Biomedical and Health Informatics, 27(2):854-865, 2022. https://doi.org/10.1109/JBHI.2022.3157725

[11] Pushpita Chatterjee, Debashis Das, and Danda B. Rawat. Federated learning empowered recommendation model for financial consumer services. IEEE Transactions on Consumer Electronics, 70(1):2508-2516, 2023. https://doi.org/10.1109/TCE.2023.3339702

[12] Mahmuda Akter, Nour Moustafa, Timothy Lynar, and Imran Razzak. Edge intelligence: Federated learning-based privacy protection framework for smart healthcare systems. IEEE Journal of Biomedical and Health Informatics, 26(12):5805-5816, 2022. https://doi.org/10.1109/JBHI.2022.3192648

[13] Demin Gao, Haoyu Wang, Xiuzhen Guo, Lei Wang, Guan Gui, and Weizheng Wang. Federated learning based on CTC for heterogeneous internet of things. IEEE Internet of Things Journal, 10(24):22673-22685, 2023. https://doi.org/10.1109/JIOT.2023.3305189

[14] Zhiguo Qu, Lailei Zhang, and Prayag Tiwari. Quantum fuzzy federated learning for privacy protection in intelligent information processing. IEEE Transactions on Fuzzy Systems, 33(1):278-289, 2024. https://doi.org/10.1109/TFUZZ.2024.3419559

[15] Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. Federated learning for healthcare: Systematic review and architecture proposal. ACM Transactions on Intelligent Systems and Technology (TIST), 13(4):1-23, 2022. https://doi.org/10.1145/3501813

[16] Shaoxiong Ji, Yue Tan, Teemu Saravirta, Zhiqin Yang, Yixin Liu, Lauri Vasankari, Shirui Pan, Guodong Long, Anwar Walid, Vasankari L, and Walid A. Emerging trends in federated learning: From model fusion to federated x learning. International Journal of Machine Learning and Cybernetics, 15(9):3769-3790, 2024. https://doi.org/10.1007/s13042-024-02119-1

[17] Ali Hatamizadeh, Hongxu Yin, Pavlo Molchanov, Andriy Myronenko, Wenqi Li, Prerna Dogra, Andrew Feng, Mona G Flores, Jan Kautz, Daguang Xu, and Holger R. Roth. Do gradient inversion attacks make federated learning unsafe. IEEE Transactions on Medical Imaging, 42(7):2044-2056, 2023. https://doi.org/10.1109/TMI.2023.3239391

[18] Weimin He, and Lei Zhao. Application of federated learning algorithm based on K-means in electric power data. Journal of New Media, 4(4):191-203, 2022. https://doi.org/10.32604/jnm.2022.032994

[19] Chaoli Sun, Xiaojun Wang, Junwei Ma, and Gang Xie. A composition-decomposition based federated learning. Complex & Intelligent Systems, 10(1):1027-1042, 2024. https://doi.org/10.1007/s40747-023-01198-x

[20] Mansoor Ali, Faisal Naeem, Muhammad Tariq, and Georges Kaddoum. Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey. IEEE Journal of Biomedical and Health Informatics, 27(2):778-789, 2022. https://doi.org/10.1109/JBHI.2022.3181823

[21] Pradyumna Kumar Tripathy, Anurag Shrivastava, Varsha Agarwal, Devangkumar Umakant Shah, Chandra Sekhar Reddy L., and S.V. Akilandeeswari. Federated learning algorithm based on matrix mapping for data privacy over edge computing. International Journal of Pervasive Computing and Communications, 20(5):633-647, 2024. https://doi.org/10.1108/IJPCC-03-2022-0113

[22] Zhiyuan Wang, Hongli Xu, Jianchun Liu, Yang Xu, He Huang, and Yangming Zhao. Accelerating federated learning with cluster construction and hierarchical aggregation. IEEE Transactions on Mobile Computing, 22(7):3805-3822, 2022. https://doi.org/10.1109/TMC.2022.3147792

[23] Jose A. Carrillo, Nicolas Garcia Trillos, Sixu Li, and Yuhua Zhu. FedCBO: Reaching group consensus in clustered federated learning through consensus-based optimization. Journal of Machine Learning Research, 25(214):1-51, 2024. DOI: 10.48550/arXiv.2305.02894

[24] Yongheng Deng, Feng Lyu, Tengxi Xia, Yuezhi Zhou, Yaoxue Zhang, Ju Ren, and Yuanyuan Yang. A communication-efficient hierarchical federated learning framework via shaping data distribution at edge. IEEE/ACM Transactions on Networking, 32(3):2600-2615, 2024. https://doi.org/10.1109/TNET.2024.3363916

[25] Chengtian Ouyang, Yehong Li, Jihong Mao, Donglin Zhu, Changjun Zhou, and Zhenyu Xu. Enhancing federated learning with dynamic weight adjustment based on particle swarm optimization. Discover Computing, 27(1):35-52, 2024. https://doi.org/10.1007/s10791-024-09478-x