CSAHFL: Clustered Semi-Asynchronous Hierarchical Federated Learning for Dual-layer Non-IID in Heterogeneous Edge Computing Networks

Aijing Li[1,2,3], **Junping Du**[1,2*], **Dandan Liu**[1,2,4], **Yingxia Shao**[1,2], **Tong Zhao**[1,2] and **Guanhua Ye**[1,2]

[1]School of Computer Science (National Pilot School of Software Engineering), Beijing University of Posts and Telecommunications, China
[2]Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia, China
[3]Century College, Beijing University of Posts and Telecommunications, China
[4]Zaozhuang University, China
aijing_li@bupt.edu.cn, junpingdu@126.com, {luckydan6798, shaoyx, zhaotong, g.ye}@bupt.edu.cn

Abstract

Federated Learning (FL) enables collaborative model training across distributed devices without sharing raw data. Hierarchical Federated Learning (HFL) is a new paradigm of FL that leverages the Edge Servers (ESs) layer as an intermediary to perform partial local model aggregation in proximity, reducing core network transmission overhead. However, HFL faces new challenges: (1) The two-stage aggregation process between client-edge and edge-cloud results in a dual-layer non-IID issue, which may significantly compromise model training accuracy. (2) The heterogeneity and mobility of clients further impact model training efficiency. To address these challenges, we propose a novel **C**lustered **S**emi-**A**synchronous **H**ierarchical **F**ederated **L**earning (CSAHFL) framework that integrates adaptive semi-asynchronous intra-cluster aggregation at client-edge layer and dynamic distribution-aware inter-cluster aggregation at edge-cloud layer, collaboratively enhancing model performance and scalability in heterogeneous and mobile environments. We conducte experiments under varying degrees of dual-layer non-IID in both static and high-mobility scenarios. The results demonstrate significant advantages of CSAHFL over representative state-of-the-art methods.

1 Introduction

Federated learning (FL) [McMahan *et al.*, 2017] has emerged as a promising paradigm to enable collaborative training of machine learning models across distributed clients without sharing raw data. By ensuring data privacy, FL has found widespread applications in domains such as healthcare [Wen *et al.*, 2023], the Internet of Things (IoT) [Rani *et al.*, 2023], and smart cities [Pandya *et al.*, 2023]. However, the practical deployment of FL is hindered by the prohibitively high communication costs arising from the iterative exchange of model updates between clients and the cloud server. This issue is further exacerbated when clients' local training data are non-IID, as achieving the desired learning accuracy requires more aggregation rounds [Shahid *et al.*, 2021; Qin *et al.*, 2021]. Additionally, as model architectures grow increasingly complex, the volume of transmitted model updates grows significantly, further amplifying the communication burden in FL systems [Elbir *et al.*, 2021].

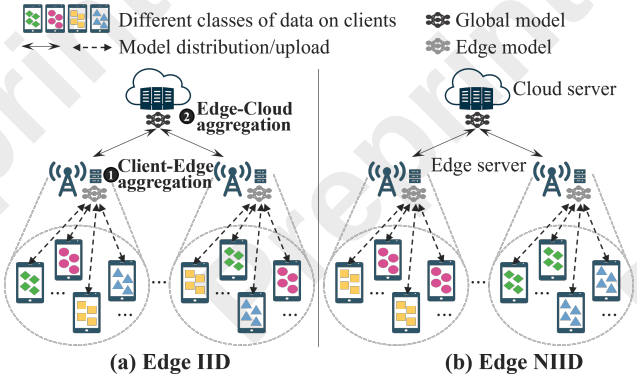


Figure 1: The illustration of Dual-layer non-IID in HFL. Each client only holds data samples from a single class, resulting in a non-IID distribution between clients. As shown in (a), both edge servers can aggregate updates from clients covering 4 classes. In (b), each edge server can only aggregate updates from clients covering 2 different classes, leading to a non-IID distribution across edge servers.

To address these challenges, Hierarchical federated learning (HFL) has been proposed as an advanced extension of traditional FL [Wang *et al.*, 2022]. HFL adopts a multi-layer architecture, comprising clients, edge servers (ESs), and a central cloud server. By leveraging the intermediate ESs layer for local model aggregation, HFL significantly reduces the frequency of direct communication between clients and the cloud, thereby lowering communication overhead. Recent advancements in HFL have addressed key challenges in communication efficiency, computational scalability, and

*Corresponding author

model convergence by introducing multi-layer design [Cui *et al.*, 2022; Pervej *et al.*, 2024; Ma *et al.*, 2024a]. HFL aligns naturally with the layered structure of many real-world systems. For instance, in IoT applications, sensors first transmit updates to local gateways (i.e., ESs), which then coordinate with a cloud server for global model training, reducing latency and enabling distributed learning efficiency [Singh *et al.*, 2022a].

Despite its potential, the practical implementation of HFL is hindered by several challenges:

Dual-layer Non-IID in HFL. Unlike traditional FL, where heterogeneity is primarily observed at the client level [Li *et al.*, 2022], HFL introduces an intermediate aggregation layer at ESs, resulting in intra-edge heterogeneity (variation among clients under the same ES) and inter-edge heterogeneity (variation across different ESs). This dual-layer non-IID characteristic further complicates the convergence of the global model. For example, clients associated with the same edge server might exhibit highly diverse data distributions, while pronounced disparities across ESs can further hinder global aggregation [Huang *et al.*, 2022]. Figure 1 illustrates the dual-layer non-IID data distribution.

Heterogeneous client capabilities. Clients in HFL often exhibit varying computational resources, including differences in CPU/GPU performance, memory capacity, and network bandwidth, which give rise to variable update latencies and potential bottlenecks during the aggregation process. To address these, adaptive scheduling is required to accommodate such variations without stalling the overall training [Singh *et al.*, 2022b].

Client mobility and dynamic participation. In dynamic environments such as vehicular systems, clients frequently join or drop out of the training process due to mobility or resource constraints. In addition, clients may move between ESs regions during training. Such mobility disrupts local update consistency and complicates hierarchical aggregation, thereby impeding convergence of the global model [Prigent *et al.*, 2024; Morell Martínez *et al.*, 2022].

While existing studies have explored solutions to address individual aspects of these challenges in HFL, a comprehensive framework capable of simultaneously mitigating multiple issues remains lacking. In this paper, we propose a novel **C**lustered **S**emi-**A**synchronous **H**ierarchical **F**ederated **L**earning (**CSAHFL**) framework, which integrates adaptive semi-asynchronous intra-cluster aggregation at the client-edge layer and dynamic distribution-aware inter-cluster aggregation at the edge-cloud layer, collaboratively enhancing model performance and scalability in heterogeneous and dynamic environments. Specifically, we introduce a privacy-preserving one-shot clustering method that groups clients by data distribution similarity. This method incurs minimal computational overhead and supports efficient cross-edge cluster identification. At the client-edge layer, our adaptive semi-asynchronous aggregation mechanism dynamically adjusts each client's workload based on real-time availability, thereby increasing participation rates for low-capacity clients and reducing idle time. At the edge-cloud layer, we employ a dynamic distribution-aware inter-cluster aggregation strategy that assigns weights to aggregated edge models according to their similarity to the global objective, effectively mitigating inter-cluster disparities. Together, these features enable CSAHFL to comprehensively address the dual-layer non-IID heterogeneity inherent in HFL. Our contributions in this paper are as follows.

- We propose the CSAHFL framework, which effectively tackles the dual-layer non-IID heterogeneity in HFL by incorporating a privacy-preserving one-shot clustering, intra-cluster aggregation at the client-edge layer and inter-cluster aggregation at the edge-cloud layer.
- We design an adaptive semi-asynchronous update mechanism that reduces the impact of client heterogeneity by dynamically balancing workload allocation while ensuring efficient model training.
- We introduce three different Edge non-IID (ENIID) configurations to gain insights into the impact of dual-layer non-IID on model performance. We conducted experiments in both static and high-mobility scenarios, and the experimental results demonstrate that our proposed method CSAHFL exceeds the performance of state-of-the-art algorithms.

2 Related Work

HFL extends traditional FL by introducing an intermediate layer of ESs, thereby reducing communication overhead and enabling more efficient model aggregation. However, communication efficiency and heterogeneity in both client capabilities and data distributions remain significant barriers to its practical deployment.

Several studies have targeted communication optimization in HFL. HierFedAVG [Liu *et al.*, 2020] was the first to propose the client-edge-cloud hierarchical Federated Learning system, enabling faster model training and better communication-computation trade-offs by leveraging partial model aggregation at edge servers. RAF [Yang *et al.*, 2022] proposed a dynamic aggregation strategy to minimize the number of communication rounds, while HED-FL [Pervej *et al.*, 2024] introduced a compression technique to reduce transmitted data size. HFEL [Luo *et al.*, 2020] formulates a joint resource-allocation and edge-association problem, developing an efficient scheduling algorithm that minimizes global cost and enhances training performance. Although these approaches have markedly improved communication efficiency, they do not consider how data heterogeneity affects model accuracy.

To address devices and data heterogeneity, HELCHFL [Cui *et al.*, 2022] developed a client selection mechanism that prioritizes devices with greater computational power. SARE [Deng *et al.*, 2024] maximized diversity in edge-aggregated data by shaping local distributions to alleviate non-IID effects. HARMONY [Tian *et al.*, 2022] designed a hierarchical coordination scheme that balances local data distribution characteristics and global training round requirements to improve model accuracy and convergence speed. HiFlash [Wu *et al.*, 2023] integrated deep reinforcement learning for adaptive staleness control alongside a heterogeneity-aware client-edge association strategy.

Nonetheless, these approaches focuses primarily on the client level, lacking investigation into the non-IID discrepancies among ESs.

Although prior HFL research has enhanced communication efficiency and mitigated client-level heterogeneity, existing approaches still fall short of addressing the combined challenges posed by dual-layer non-IID data distributions and heterogeneous device capabilities. In this paper, we bridge these gaps.

3 The Proposed CSAHFL Method

3.1 Problem Definition

We provide the definition of the problem based on the Cloud-based FL. HFL consists of N clients, M Edge Servers and one cloud server. Each client n owns a local data set $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}_n|}$ where x_i denotes the i-the input sample and y_i is the corresponding labeled output for client n's federated learning task. Clients are divided into M disjoint groups based on their geographical locations, each of which is associated with one ES. The overall objective of the HFL is to minimize the global loss function, which is defined as:

$$F(w) = \sum_{j=1}^{M} \frac{n_j}{N} F_j(w) \tag{1}$$

where $F_j(w)$ is the loss of function for ES j, n_j is the total data size of clients assigned to ES j, N is the total data size across all clients.

At each ES j, the local loss function $F_j(w)$ is defined as:

$$F_j(w) = \sum_{i \in S_j} \frac{n_i}{n_j} F_i(w) \tag{2}$$

where S_j is the set of clients assigned to ES k, n_i is the data size of client i, $F_i(w) = \mathbb{E}_{(x,y)\sim P_i}[\ell(f_w(x), y)]$ is the local loss function for client i, with ℓ being the loss function.

The final optimization objective is to minimize:

$$\min_w \sum_{j=1}^{M} \frac{n_j}{N} \sum_{i \in S_j} \frac{n_i}{n_j} \mathbb{E}_{(x,y)\sim P_i}[\ell(f_w(x), y)] \tag{3}$$

This problem encapsulates the dual-layer aggregation process, aiming to produce a globally optimized model while efficiently handling hierarchical data distributions.

3.2 Overview of CSAHFL

We propose Clustered Semi-Asynchronous Hierarchical Federated Learning (CSAHFL), a framework that integrates three interdependent components to work in harmony to handle data and system heterogeneity. As shown in Figure 2. The CSAHFL comprises: (1) *Privacy-preserving one-shot clustering module*: Clients are grouped into clusters based on data-distribution similarity without revealing raw data. It ensures efficient clustering with minimal computational and communication overhead. These clusters form the basis for all subsequent aggregations. (2) *Adaptive semi-asynchronous intra-cluster aggregation* at the client-edge layer: Each client

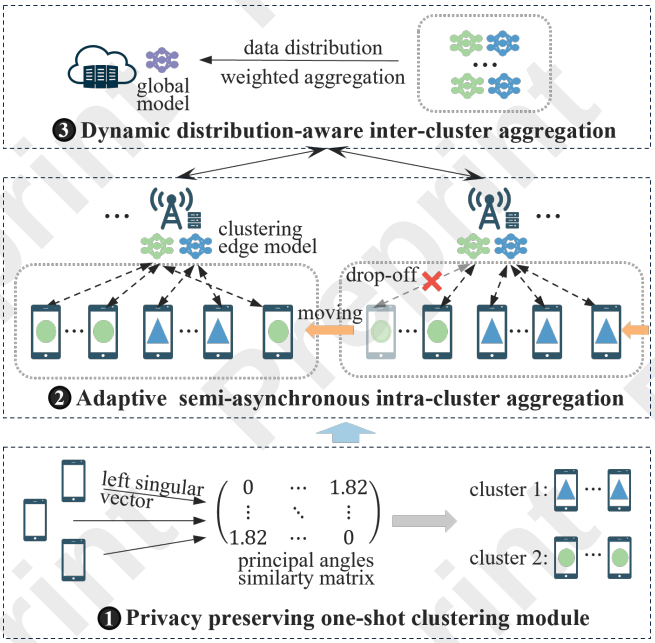


Figure 2: The overview of CSAHFL

dynamically adjusts its aggregation interval according to real-time availability and resource constraints, minimizing idle time and ensuring that low-capacity clients continuously contribute to local model updates. (3) *Dynamic distribution-aware inter-cluster aggregation* at the edge-cloud layer: Updated cluster models are aggregated at the cloud by assigning weights proportional to inter-cluster distribution similarity, thereby mitigating the impact of heterogeneity across ESs and yielding a robust, globally consistent model. We provide one global aggregation iteration of CSAHFL in Algorithm 1.

3.3 Privacy Preserving One-Shot Clustering Module

Clients in HFL often exhibit heterogeneous (non-IID) data distributions, which impede both local and global model convergence. To alleviate these effects, clients with similar distributions can be grouped into clusters, thereby reducing inter-client variability and enabling localized training within each cluster. This strategy naturally exploits the HFL architecture, where ESs mediate between clients and the central cloud. Here, we employ principal angles to quantify distributional similarity among clients. Unlike iterative clustering methods, this approach performs clustering in a single step, incurring no additional computational overhead during training[Vahidian *et al.*, 2023].

The cosine similarity metric is extended to measure the similarity between subspaces using principal angles. For two subspaces $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^n$, the smallest principal angle Θ_1 is defined as:

$$\Theta_1(X, Y) = \min_{\mathbf{x}\in X, \mathbf{y}\in Y} \arccos\left(\frac{|\mathbf{x}^\top \mathbf{y}|}{\|\mathbf{x}\|\|\mathbf{y}\|}\right) \tag{4}$$

Principal angles quantify subspace distances, providing a robust similarity metric for clustering[Qian *et al.*, 2004].

To ensure privacy, each client computes a truncated singular value decomposition (SVD) of its dataset $\mathcal{D}_i$, retaining only the top p singular vectors:

$$\mathbf{U}_p^n = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_p], \quad p \ll \text{rank}(\mathcal{D}_i) \tag{5}$$

These vectors, serving as low-dimensional data signatures, are transmitted to the server while preserving privacy by avoiding raw data sharing.

We construct a proximity matrix $\mathbf{A}$, where each element $\mathbf{A}_{i,j}$ represents the smallest principal angle between the subspaces $\mathbf{U}_p^i$ and $\mathbf{U}_p^j$:

$$\mathbf{A}_{i,j} = \Theta_1\left(\mathbf{U}_p^i, \mathbf{U}_p^j\right), \quad i,j = 1,\ldots,N \tag{6}$$

Smaller values of $\mathbf{A}_{i,j}$ indicate higher similarity. Hierarchical clustering (HC) is applied to $\mathbf{A}$ to group clients into K clusters, with the clustering threshold β determining the number of clusters $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, ...\mathcal{C}_K\}$.

3.4 Adaptive Semi-Asynchronous Intra-Cluster Aggregation at the Client-Edge Layer

To address resource heterogeneity among clients and improve their participation rates in HFL, we propose an adaptive semi-asynchronous intra-cluster aggregation mechanism at the client-edge layer, which tries to unify each client's round time to the limited aggregation interval T_θ by adaptively adjusting the workload concerning its real-time availability, effectively mitigate the impact of weak or stale clients while enhancing training efficiency.

Adaptive Local Trainning

At each ES communication round, the ES randomly samples n clients to construct the collection $\mathcal{S}$ and distributes the edge model to clients, each selected client would perform one data batch full model training to estimate its pre-epoch computation time t_{unit} and report it to the ES. Sequentially, the ES calculates each client's total time, the total time t_{total} consists of computation time t_{cmp} and communication time t_{com}, which is defined as:

$$\begin{aligned} t_{total} &= t_{cmp} + t_{com} \\ &= E_{max} \times N_{BS} \times t_{unit} + M/B_w \end{aligned} \tag{7}$$

where N_{BS} is the number of batches, determined by dividing the dataset size by the batch size B, E_{max} is a hyperparameter representing the upper bounds for training epochs, M is model's file size and B_w is real-time network bandwidth.

The ES dynamically adjusts the aggregation interval T_θ, is defined as:

$$T_\theta = \text{median}(T_{\text{total}}) + \alpha \cdot \text{IQR}(T_{\text{total}}) \tag{8}$$

where $\text{median}(T_{\text{total}})$ represents the typical time consumption among clients, $\text{IQR}(T_{\text{total}})$ captures the variability in client performance, and α is a tunable parameter to control the degree of tolerance for stragglers. By dynamically adjusting T_θ, the ES ensures that the majority of clients can complete their training tasks and report updates within the designated interval.

Each client i then adjusts its workload E_i to ensure timely completion within T_θ:

$$E_i = \max\left(\min\left(\frac{T_\theta - t_{\text{com}}}{N_{BS} \times t_{unit}}, E_{\max}\right), 1\right) \tag{9}$$

Semi-Asynchronous Intra-Cluster Aggregation

Clients that successfully report updates within T_θ are included in the aggregation set $\mathcal{S}_k$ for cluster k. For clients that fail to submit updates, their contributions are asynchronously included in the aggregation process during subsequent iterations to ensure their updates are not discarded. The aggregated edge model for cluster k is computed as:

$$w_{\text{edge}}^{k,t} = \frac{1}{N_{\text{edge}}^k}\left(\sum_{i\in S_k} n_i w_i^t + \sum_{j\in A_k} \tau_j n_j w_j^t\right) \tag{10}$$

where n_i and n_j are the local dataset sizes of synchronous client i and asynchronous client j, $N_{\text{edge}}^k = \sum_{i\in S_k} n_i + \sum_{j\in A_k} n_j$ is the total dataset size for all reporting clients in cluster k, including both synchronous S_k and asynchronous A_k sets. τ_j is a discount factor applied to asynchronous updates from client j to account for potential staleness in their contributions, where $\tau_j = 1/(1+\gamma_j)$, γ_j represents the number of delayed rounds for the update from client j. w_i^t and w_j^t represent the local model updates from synchronous and asynchronous clients at round t.

After intra-cluster aggregation, the ES maintains K updated cluster models $\{w_{\text{edge}}^1, w_{\text{edge}}^2, ..., w_{\text{edge}}^K\}$.

Algorithm 1 CSAHFL Under One Global Aggregation

Input: Cluster set $\mathcal{C}$, initial global model w_{g}^0, aggregation interval T_θ, maximum epoch $E_{\max}$
Output: Final global model w_{g}.

```
/* Intra-Cluster Aggregation                                   */
```
for *each edge communication round* $t = 1, 2, \ldots, \gamma_e$ **do**
  for *each cluster* $\mathcal{C}_k$, $k = 1, 2, \ldots, K$ ***in parallel*** **do**
    ES sends the current cluster model $w_{\text{edge}}^{k,t}$ to all clients in $\mathcal{C}_k$
    for *each client* $i \in \mathcal{C}_k$ ***in parallel*** **do**
      Client i estimates unit computation time t_{unit} and reports it to ES.
      ES computes total time T_{total} and adjusts the aggregation interval T_θ dynamically,
      $T_\theta = \text{median}(T_{\text{total}}) + \alpha \cdot \text{IQR}(T_{\text{total}})$
      Client i updates workload,
      $E_i = \max\left(\min\left(\frac{T_\theta - t_{\text{com}}}{N_{BS}\times t_{unit}}, E_{\max}\right), 1\right)$
      Client i performs local training for E_i and sends w_i^t to ES.
    ES aggregates updates:
    $w_{\text{edge}}^{k,t} = \frac{1}{N_{\text{edge}}^k}\left(\sum_{i\in S_k} n_i w_i^t + \sum_{j\in A_k} \tau_j n_j w_j^t\right)$

```
/* Inter-Cluster Aggregation                                   */
```
Cloud server receives all updated cluster models $\{w_{\text{edge}}^1, \ldots, w_{\text{edge}}^K\}$
for *each cluster* $\mathcal{C}_k$, $k = 1, 2, \ldots, K$ **do**
  Compute weights: $\lambda_k = \frac{\omega_k^{\text{qua}} \cdot \omega_k^{\text{dis}}}{\sum_{j=1}^K \omega_j^{\text{qua}} \cdot \omega_j^{\text{dis}}}$
Aggregate cluster models: $w_{\text{g}} = \sum_{k=1}^K \lambda_k w_{\text{edge}}^k$
return w_g

3.5 Dynamic Distribution-Aware Inter-Cluster Aggregation at the Edge-Cloud Layer

At the edge-cloud layer, the central server performs a dynamic aggregation of the K cluster models received from

ESs. This aggregation considers both the data distribution similarity between clusters and the data quantity associated with each cluster, ensuring a more balanced and robust global model update.

The global model w_{g} is updated by aggregating the cluster models $\{w^1_{\mathrm{edge}}, w^2_{\mathrm{edge}}, \ldots, w^K_{\mathrm{edge}}\}$. The aggregation formula is given as:

$$w_{\mathrm{g}} = \sum_{k=1}^{K} \lambda_k w^k_{\mathrm{edge}} \tag{11}$$

where λ_k is the weight assigned to cluster k, reflecting its importance in the global aggregation. The weight λ_k for each cluster k is dynamically calculated based on two factors: the quantity and distribution similarity.

Clusters with more data contribute proportionally more to the global model. The quantity weight ω^{qua}_k is defined as:

$$\omega^{\mathrm{qua}}_k = \frac{N^k_{\mathrm{edge}}}{\sum_{k=1}^{K} N^k_{\mathrm{edge}}}, \tag{12}$$

Clusters whose data distribution aligns more closely with the global target distribution are given higher weights. The distribution weight ω^{dis}_k is calculated using a similarity metric such as KL divergence:

$$\omega^{\mathrm{dis}}_k = \frac{1}{1 + D_{\mathrm{KL}}(P_k \| P_{\mathrm{g}})}, \tag{13}$$

where $D_{\mathrm{KL}}(P_k \| P_{\mathrm{g}})$ is KL divergence between the distribution P_k of cluster d and the target global distribution P_{g}.

The final weight λ_k is computed as a combination of the two factors:

$$\lambda_k = \frac{\omega^{\mathrm{qua}}_k \cdot \omega^{\mathrm{dis}}_k}{\sum_{k=1}^{K} \omega^{\mathrm{qua}}_k \cdot \omega^{\mathrm{dis}}_k}. \tag{14}$$

4 Experiments

We consider an HFL system consisting of 200 clients, 5 ESs, and one cloud server, assuming each ES authorizes the same number of clients with the approximate amount of training data. We conducted experiments in two scenarios: static and high-mobility. In the static scenario, each client remains associated with one ES throughout the entire training process. In the high-mobility scenario, we follow the experimental framework in MACFL [Feng *et al.*, 2022], which establishes a theoretical foundation based on Markov chains for the dynamic transition of clients.

4.1 Experiment Settings

Datasets. We validate our proposed CSAHFL on three popular datasets: Fashion-MNIST [Xiao *et al.*, 2017], CIFAR10 [Krizhevsky *et al.*, 2009] and SVHN [Netzer *et al.*, 2011].

Non-IID setting. To create data with dual-layer non-IID, we considers the widely used label skew non-IID data distribution. We first randomly assign each ES x classes of the total labels and then each client randomly selects y classes from the labels assigned to its corresponding ES. We focus on analyzing the impact of Edge non-IID (ENIID) on model performance. Therefore, y is uniformly set to 20%, while x is configured under the following three settings.

- **EIID:** Assign each ES all classes. The datasets among ESs are IID.
- **ENIID50%:** Assign each ES a total of 50% classes. The datasets among ESs are non-IID.
- **ENIID30%:** Assign each ES a total of 30% classes. The degree of non-IID between ESs is the highest.

Baselines. We compare CSAHFL against the following set of baselines. (1) FedAVG [McMahan *et al.*, 2017] is a classical cloud-based FL. (2) FedProx [Li *et al.*, 2020] optimizes statistical heterogeneity and system heterogeneity by adding a proximal term, we perform it in the client-edge aggregation layer. For baselines that adopt different clustering strategies and communication mechanisms based on HFL. (3) HierFedAVG [Liu *et al.*, 2020] is a cloud-edge-client HFL that performs synchronous updates in both client-edge and edge-cloud aggregation. (4) FedAT [Chai *et al.*, 2021] combines synchronous intra-tier training and asynchronous cross-tier training, and conducts client clustering based on their latencies. (5) MACFL [Feng *et al.*, 2022] is a mobility-aware cluster algorithm by redesigning the local update rule and model aggregation. (6) HiFlash [Wu *et al.*, 2023] combines client-edge synchronous and edge-cloud asynchronous aggregation with adaptive staleness control and heterogeneity-aware client-edge association. (7) FedUC [Ma *et al.*, 2024b] proposes a time-sharing scheduling algorithm to minimize intra-cluster aggregation latency. Considering that the FedAVG and FedAT algorithms are not affected by mobile scenarios, we do not repeat their results in the experiments.

Metrics. We use two common metrics to measure performance: average test accuracy and training time to achieve target accuracy.

Client Heterogeneity and Mobility Setting. Each client runs on a different thread and has a random computation delay obeying a normal distribution[Zhang *et al.*, 2024](minimum=2s, maximum=128s, μ=63, σ=40). In the presence of user mobility, we assume all the users are uniformly distributed over the entire network at the beginning of time. Each user will stay or move to a neighboring ES according to staying probability p_s, denotes the probability that users staying at the current ES [Feng *et al.*, 2022]. p_s=1 represents a static scenario. For the mobile scenario, we set p_s=0.5, meaning the probability of a client staying at the current ES to participate in training is 0.5.

Parameter Settings. For each dataset, the number of samples selected by clients for one training session is B = 10, the number of local epoch is E = 10, the learning rate is η = 0.01, the optimization algorithm used for local training of clients is SGD, the number of rounds the ES aggregates its clients' local models is r_e = 3, and the number of rounds the cloud server aggregates ESs' edge models is r_c = 50. The total number of communication rounds equals $r_e \times r_c$.

4.2 Performance Analysis

We conducted experiments on three datasets. As shown in Tables 1 and 2, CSAHFL consistently outperforms all baseline methods across all datasets, ENIID settings, and mobility scenarios, highlighting its robustness to data heterogeneity

Dataset		Fashion-MNIST			CIFAR10			SVHN		
ENIID setting		EIID	ENIID50%	ENIID30%	EIID	ENIID50%	ENIID30%	EIID	ENIID50%	ENIID30%
$p_s = 1$	FedAVG	87.50±1.01	83.26±0.97	83.08±1.32	48.52±0.98	48.45±0.34	47.74±1.02	84.58±0.26	83.78±0.55	83.83±0.56
	FedProx	87.93±0.92	82.96±0.74	80.97±0.74	45.28±0.56	45.28±0.78	40.86±0.84	85.04±0.93	82.65±1.30	78.65±0.95
	HierFedAVG	87.43±0.80	83.97±0.46	81.44±0.79	45.51±0.75	45.57±1.10	44.13±0.74	85.05±0.18	84.92±1.50	82.90±0.74
	FedAT	83.47±2.32	82.31±4.94	77.01±3.37	51.22±1.84	47.11±3.43	37.64±0.80	82.05±3.92	78.82±1.11	75.15±0.79
	MACFL	88.02±0.37	84.95±0.58	81.65±0.91	53.02±1.34	49.77±0.34	47.77±0.34	87.77±0.38	84.13±0.74	81.63±0.74
	HiFlash	89.12±1.35	85.24±0.29	81.48±0.52	65.08±0.42	60.58±1.02	49.37±0.64	88.49±0.42	84.92±0.67	80.59±0.37
	FedUC	88.14±1.17	83.97±1.32	82.44±1.45	60.98±1.19	58.94±0.74	54.70±0.85	86.62±1.22	83.79±0.64	82.38±1.38
	CSAHFL	**92.14±0.21**	**91.11±0.15**	**88.49±0.43**	**69.35±0.31**	**68.06±0.27**	**64.75±0.34**	**90.68±0.20**	**89.62±0.18**	**88.17±0.56**
$p_s = 0.5$	FedProx	86.96±1.01	82.27±2.63	79.86±0.98	45.64±2.78	46.63±1.52	43.32±0.65	83.96±1.48	82.68±1.65	81.57±0.33
	HierFedAVG	86.59±0.21	83.80±1.32	81.39±2.96	44.58±1.57	44.04±0.64	41.88±1.04	83.44±0.27	81.62±0.33	80.67±1.53
	MACFL	88.44±0.62	85.97±0.15	83.01±0.97	54.06±1.15	46.02±0.13	50.81±0.27	86.06±0.80	83.71±0.42	82.79±1.50
	HiFlash	87.71±0.87	87.13±0.23	85.09±1.12	64.83±0.59	63.47±0.44	62.97±0.13	86.92±0.63	86.20±0.24	83.54±0.29
	FedUC	86.70±0.98	83.36±1.56	82.57±1.42	60.51±1.86	56.34±1.30	54.85±1.23	84.19±0.92	82.03±0.55	80.97±2.05
	CSAHFL	**92.82±0.34**	**91.8±0.42**	**88.27±0.58**	**70.68±0.56**	**68.35±0.20**	**64.55±0.30**	**89.72±0.14**	**89.03±0.29**	**88.11±0.49**

Table 1: Test accuracy comparison across different datasets. For each baseline, the average of final local test accuracy over all clients is reported. We run each baseline 3 times. The top results are emphasized in bold.

ENIID setting		EIID		ENIID50%		ENIID30%	
Target		60%	85%	60%	82%	60%	80%
$p_s = 1$	FedAVG	823 (1.17×)	5043 (3.62×)	987 (1.37×)	8028 (6.46×)	1302 (1.85×)	6229 (3.59×)
	FedProx	1341 (1.90×)	5387 (3.87×)	1836 (2.56×)	10137 (8.16×)	1869 (2.65×)	11353 (6.54×)
	HierFedAVG	1780 (2.52×)	6712 (4.82×)	1811 (2.52×)	7258 (5.84×)	1811 (2.65×)	11464 (6.61×)
	FedAT	**706 (1×)**	10528 (7.56×)	**718 (1×)**	7134 (5.74×)	**705 (1×)**	-
	MACFL	1632 (2.31×)	6465 (4.64×)	1796 (2.50×)	6912 (5.57×)	1751 (2.48×)	5787 (3.32×)
	HiFlash	1400 (1.98×)	2775 (1.99×)	1407 (1.96×)	4299 (4.46×)	1380 (1.96×)	3197 (1.84×)
	FedUC	718 (1.02×)	3493 (2.50×)	730 (1.02×)	2665 (2.15×)	1283 (1.82×)	3250 (1.87×)
	CSAHFL	724 (1.03×)	**1392 (1×)**	743 (1×)	**1242 (1×)**	989 (1.40×)	**1735 (1×)**
$p_s = 0.5$	FedProx	1776 (2.39×)	7619 (5.12×)	2296 (3.23×)	11068 (8.93×)	2091 (2.81×)	10776 (5.86×)
	HierFedAvg	1819 (2.44×)	7164 (4.81×)	1776 (1.79×)	6958 (5.61×)	3202 (4.30×)	11438 (6.22×)
	MACFL	1795 (2.41×)	5909 (3.97×)	1807 (1.82×)	5089 (4.10×)	1735 (2.33×)	4374 (2.38×)
	HiFlash	882 (1.19×)	6497 (4.37×)	1093 (1.10×)	1733 (1.40×)	824 (1.11×)	2224 (1.21×)
	FedUC	853 (1.12×)	5012 (3.37×)	1065 (1.07×)	3793 (3.06×)	912 (1.22×)	3612 (1.96×)
	CSAHFL	**744 (1×)**	**1488 (1×)**	**991 (1×)**	**1240 (1×)**	**745 (1×)**	**1840 (1×)**

Table 2: Training time to reach target test accuracy on Fashion-MNIST. "-" indicates the target accuracy was not reached.

and client mobility. Some representative training curves are illustrated in Figure 3.

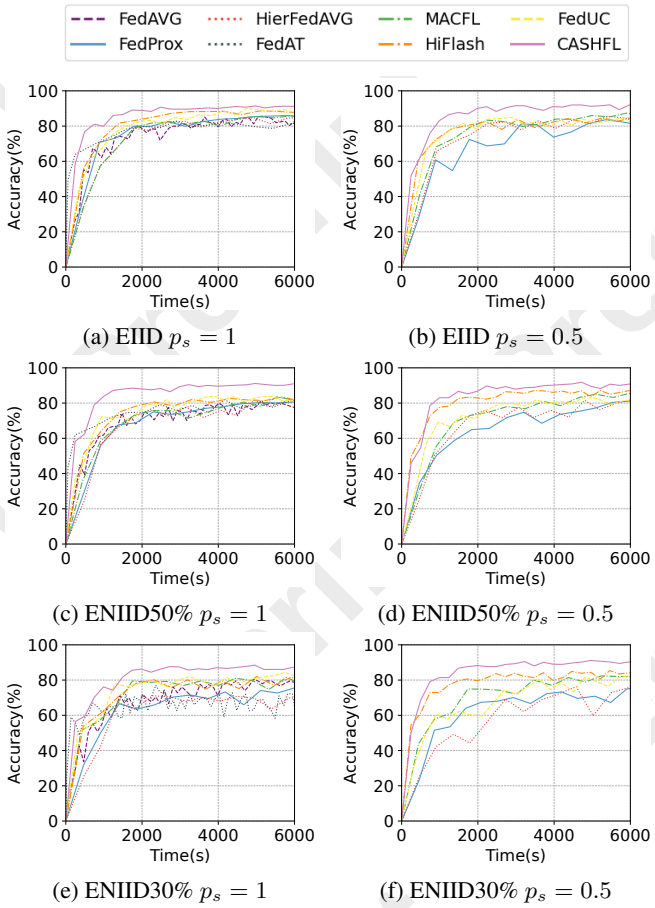
Impact of Edge Non-IID on Model Performance
The severity of ENIID directly correlates with a decline in model performance. Across all datasets, the model accuracy decreases as the data distribution transitions from EIID to ENIID30%. (1) In the EIID scenario, all methods perform relatively well due to the balanced data distribution across ESs. For example, On CIFAR10, CSAHFL achieves 69.35%, significantly outperforming FedAvg 48.52%. (2) In the ENIID50% scenario, as data heterogeneity increases, model performance begins to degrade. For example, on CIFAR10, FedAT's accuracy drops from 51.22% (EIID) to 47.11% (ENIID50%), while CSAHFL maintains a high accuracy of 68.06%. (3) Under the most extreme ENIID30% scenario, traditional methods experience significant performance degradation. On CIFAR10, FedProx achieves only 43.32%, while CSAHFL achieves a much higher accuracy of 64.55%.

Impact of Client Mobility on Model Performance
In high-mobility scenarios, client mobility introduces additional challenges, causing accuracy drops for all methods. The accuracy drops in high-mobility scenarios are primarily due to inconsistent data distributions. However, the accuracy improvement observed in HiFlash algorithms may be attributed to client mobility providing a greater variety of data sample choices for edge model updates. CSAHFL achieves the highest accuracy across all datasets and non-IID settings.

4.3 Hyperparameter Analysis

We investigate the effect of the aggregation interval factor α, the number of max local epoch E_{max}, and the clustering threshold β on system performance. The results are summarized as follows:

Interval factor α and max local epoch E_{max}. We conduct a combined analysis of α and E_{max} to study their impact on training time. As shown in α is tuned across the set {0.5, 1.0, 1.5, 2.0}, while E_{max} is varied across {5,10,20,30}. Smaller α values result in tighter aggregation intervals, allowing fewer clients with longer computation times to complete training and participate in aggregation. Larger α extends the aggregation interval, enabling more clients to complete full training rounds with E_{max}, but at the cost of increased overall training time. On the other hand, weaker clients with lower computational capacity reduce their local epochs E to finish training within the aggregation interval. This dynamic adjustment helps maintain their participation in the aggregation process while keeping their training times manageable. As

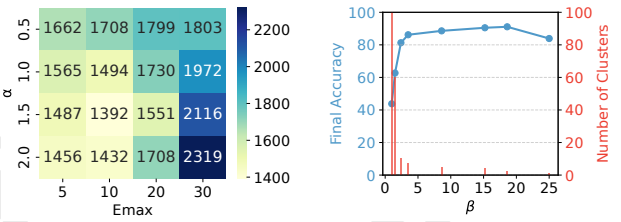


(a) EIID $p_s = 1$ (b) EIID $p_s = 0.5$

(c) ENIID50% $p_s = 1$ (d) ENIID50% $p_s = 0.5$

(e) ENIID30% $p_s = 1$ (f) ENIID30% $p_s = 0.5$

Figure 3: Comparative analysis of accuracy curves on Fashion-MINST datasets.



(a) (b)

Figure 4: Hyperparameter analysis on Fashion-MNIST with ENIID50. (a) Heatmap of Training Time with Varying α and $E_{\max}$. (b)Impact of β on Final Accuracy and Number of Clusters.

ENIID Setting	Method	$p_s = 1$		$p_s = 0.5$	
		ACC	Target Time	ACC	Target Time
EIID	w/o C	90.12	1432	89.65	1497
	w/o SA	91.53	5210	91.41	5108
	w/o D	91.21	1398	90.89	1537
ENIID50%	w/o C	86.34	1503	85.92	1607
	w/o SA	88.72	5419	89.03	6238
	w/o D	89.06	1408	88.74	1718
ENIID30%	w/o C	82.2	1545	81.87	1633
	w/o SA	85.17	6193	81.64	6501
	w/o D	84.69	1473	83.87	1829

Table 3: Performance Comparison under Ablation Study

shown in Figure 4(a), the optimal balance is achieved when α=1.5 and E_{max}=10, which balances training efficiency and sufficient client participation. which minimizes training time while allowing sufficient clients to participate with E_{max}.

Clustering threshold β. The results suggest that β has a significant impact on both the number of clusters and model performance. At lower values of β ($\beta < 5$), the final accuracy increases rapidly, indicating faster model convergence due to fewer clusters and more concentrated client collaboration. As shown in Figure 4(b), a moderate value $\beta \approx 20$ achieves the best trade-off, maximizing model accuracy while maintaining a reasonable number of clusters.

4.4 Ablation Study

To validate the effectiveness of the CSAHFL framework, we designed three different ablation experiment methods. (1) w/o C (without clustering): the client-edge clustering mechanism is removed, and clients are randomly assigned to edge servers without considering data distribution similarity. (2) w/o SA (without Adaptive Semi-Asynchronous Aggregation): Semi-asynchronous aggregation is replaced with a synchronous strategy, requiring all clients to complete local training before aggregation. (3) w/o D (Without Distribution-Aware Inter-Cluster Aggregation): The inter-cluster aggregation at the edge-cloud level is replaced by uniform weighting, ignoring data distribution differences across edge servers.

As shown in Table 3, removing the clustering module leads to a significant accuracy drop in non-IID scenarios, especially under ENIID30% (82.20% vs. 90.12%). This demonstrates that clustering clients based on data similarity effectively mitigates data heterogeneity at the client-edge layer. Without semi-asynchronous aggregation, the framework experiences slower convergence, particularly in dynamic scenarios ($p_s = 0.5$). This highlights the importance of adaptive client participation for reducing idle time and improving efficiency. Uniform aggregation weights at the edge-cloud layer reduce the framework's ability to handle inter-edge data heterogeneity, as evidenced by the lower accuracy in ENIID50% (89.03% vs. 88.74%).

5 Conclusions and Future Work

This paper presents the CSAHFL framework, which effectively addresses dual-layer non-IID challenges and client heterogeneity through privacy-preserving clustering, adaptive semi-asynchronous intra-cluster aggregation, and dynamic distribution-aware inter-cluster aggregation. Experiments demonstrate its superior performance in both static and high-mobility scenarios. However, our approaches lack adaptability to dynamic data changes. Future work will investigate personalized federated learning to enhance adaptability in evolving environments.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (2023YFF0725103), National Natural Science Foundation of China (U22B2038,62192784).

References

[Chai *et al.*, 2021] Zheng Chai, Yujing Chen, Ali Anwar, Liang Zhao, Yue Cheng, and Huzefa Rangwala. Fedat: A high-performance and communication-efficient federated learning system with asynchronous tiers. In *Proceedings of the international conference for high performance computing, networking, storage and analysis*, pages 1–16, 2021.

[Cui *et al.*, 2022] Yangguang Cui, Kun Cao, Junlong Zhou, and Tongquan Wei. Optimizing training efficiency and cost of hierarchical federated learning in heterogeneous mobile-edge cloud computing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(5):1518–1531, 2022.

[Deng *et al.*, 2024] 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*, 2024.

[Elbir *et al.*, 2021] Ahmet M Elbir, Anastasios K Papazafeiropoulos, and Symeon Chatzinotas. Federated learning for physical layer design. *IEEE Communications Magazine*, 59(11):81–87, 2021.

[Feng *et al.*, 2022] Chenyuan Feng, Howard H Yang, Deshun Hu, Zhiwei Zhao, Tony QS Quek, and Geyong Min. Mobility-aware cluster federated learning in hierarchical wireless networks. *IEEE Transactions on Wireless Communications*, 21(10):8441–8458, 2022.

[Huang *et al.*, 2022] Wei Huang, Tianrui Li, Dexian Wang, Shengdong Du, Junbo Zhang, and Tianqiang Huang. Fairness and accuracy in horizontal federated learning. *Information Sciences*, 589:170–185, 2022.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[Li *et al.*, 2020] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[Li *et al.*, 2022] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 965–978. IEEE, 2022.

[Liu *et al.*, 2020] Lumin Liu, Jun Zhang, SH Song, and Khaled B Letaief. Client-edge-cloud hierarchical federated learning. In *ICC 2020-2020 IEEE international conference on communications (ICC)*, pages 1–6. IEEE, 2020.

[Luo *et al.*, 2020] Siqi Luo, Xu Chen, Qiong Wu, Zhi Zhou, and Shuai Yu. Hfel: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning. *IEEE Transactions on Wireless Communications*, 19(10):6535–6548, 2020.

[Ma *et al.*, 2024a] Qianpiao Ma, Yang Xu, Hongli Xu, Jianchun Liu, and Liusheng Huang. Feduc: A unified clustering approach for hierarchical federated learning. *IEEE Transactions on Mobile Computing*, 2024.

[Ma *et al.*, 2024b] Qianpiao Ma, Yang Xu, Hongli Xu, Jianchun Liu, and Liusheng Huang. Feduc: A unified clustering approach for hierarchical federated learning. *IEEE Transactions on Mobile Computing*, 23(10):9737–9756, 2024.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[Morell Martínez *et al.*, 2022] José Ángel Morell Martínez, Enrique Alba-Torres, et al. Dynamic and adaptive fault-tolerant asynchronous federated learning using volunteer edge devices. 2022.

[Netzer *et al.*, 2011] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, 2011.

[Pandya *et al.*, 2023] Sharnil Pandya, Gautam Srivastava, Rutvij Jhaveri, M Rajasekhara Babu, Sweta Bhattacharya, Praveen Kumar Reddy Maddikunta, Spyridon Mastorakis, Md Jalil Piran, and Thippa Reddy Gadekallu. Federated learning for smart cities: A comprehensive survey. *Sustainable Energy Technologies and Assessments*, 55:102987, 2023.

[Pervej *et al.*, 2024] Md Ferdous Pervej, Richeng Jin, and Huaiyu Dai. Hierarchical federated learning in wireless networks: Pruning tackles bandwidth scarcity and system heterogeneity. *IEEE Transactions on Wireless Communications*, 2024.

[Prigent *et al.*, 2024] Cédric Prigent, Alexandru Costan, Gabriel Antoniu, and Loïc Cudennec. Enabling federated learning across the computing continuum: Systems, challenges and future directions. *Future Generation Computer Systems*, 2024.

[Qian *et al.*, 2004] Gang Qian, Shamik Sural, Yuelong Gu, and Sakti Pramanik. Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237, 2004.

[Qin *et al.*, 2021] Zhijin Qin, Geoffrey Ye Li, and Hao Ye. Federated learning and wireless communications. *IEEE Wireless Communications*, 28(5):134–140, 2021.

[Rani *et al.*, 2023] Sita Rani, Aman Kataria, Sachin Kumar, and Prayag Tiwari. Federated learning for secure iomt-applications in smart healthcare systems: A comprehensive review. *Knowledge-based systems*, 274:110658, 2023.

[Shahid *et al.*, 2021] Osama Shahid, Seyedamin Pouriyeh, Reza M Parizi, Quan Z Sheng, Gautam Srivastava, and Liang Zhao. Communication efficiency in federated learning: Achievements and challenges. *arXiv preprint arXiv:2107.10996*, 2021.

[Singh *et al.*, 2022a] Parminder Singh, Gurjot Singh Gaba, Avinash Kaur, Mustapha Hedabou, and Andrei Gurtov. Dew-cloud-based hierarchical federated learning for intrusion detection in iomt. *IEEE journal of biomedical and health informatics*, 27(2):722–731, 2022.

[Singh *et al.*, 2022b] Pushpa Singh, Murari Kumar Singh, Rajnesh Singh, and Narendra Singh. Federated learning: Challenges, methods, and future directions. In *Federated Learning for IoT Applications*, pages 199–214. Springer, 2022.

[Tian *et al.*, 2022] Chunlin Tian, Li Li, Zhan Shi, Jun Wang, and ChengZhong Xu. Harmony: Heterogeneity-aware hierarchical management for federated learning system. In *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 631–645. IEEE, 2022.

[Vahidian *et al.*, 2023] Saeed Vahidian, Mahdi Morafah, Weijia Wang, Vyacheslav Kungurtsev, Chen Chen, Mubarak Shah, and Bill Lin. Efficient distribution similarity identification in clustered federated learning via principal angles between client data subspaces. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 10043–10052, 2023.

[Wang *et al.*, 2022] 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.

[Wen *et al.*, 2023] 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.

[Wu *et al.*, 2023] Qiong Wu, Xu Chen, Tao Ouyang, Zhi Zhou, Xiaoxi Zhang, Shusen Yang, and Junshan Zhang. Hiflash: Communication-efficient hierarchical federated learning with adaptive staleness control and heterogeneity-aware client-edge association. *IEEE Transactions on Parallel and Distributed Systems*, 34(5):1560–1579, 2023.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Yang *et al.*, 2022] Lei Yang, Yingqi Gan, Jiannong Cao, and Zhenyu Wang. Optimizing aggregation frequency for hierarchical model training in heterogeneous edge computing. *IEEE Transactions on Mobile Computing*, 22(7):4181–4194, 2022.

[Zhang *et al.*, 2024] Ruizhuo Zhang, Wenjian Luo, Yongkang Luo, Hongwei Zhang, and Jiahai Wang. Afl-dcs: An asynchronous federated learning framework with dynamic client scheduling. *Engineering Applications of Artificial Intelligence*, 133:107927, 2024.