Towards Seamless Hierarchical Federated Learning under Intermittent Client Participation: A Stagewise Decision-Making Methodology

Minghong Wu, Minghui Liwang, *Member*, *IEEE*, Yuhan Su, Li Li, *Member*, *IEEE*, Seyyedali Hosseinalipour, *Member*, *IEEE*, Xianbin Wang, *Fellow*, *IEEE*, Huaiyu Dai, *Fellow*, *IEEE*, Zhenzhen Jiao

Abstract—Federated Learning (FL) offers a pioneering distributed learning paradigm that enables devices/clients to build a shared global model that can be obtained through frequent model transmissions between clients and a central server, causing high latency, energy consumption, and congestion over backhaul links. To overcome these drawbacks, Hierarchical Federated Learning (HFL) has emerged, which organizes clients into multiple clusters and utilizes edge nodes (e.g., edge servers) for intermediate model aggregations between clients and the central server. Current research on HFL mainly focus on enhancing model accuracy, latency, and energy consumption in scenarios with a stable/fixed set of clients. However, addressing the dynamic availability of clients – a critical aspect of real-world scenarios – remains underexplored. This study delves into optimizing client selection and client-to-edge associations in HFL under intermittent client participation so as to minimize overall system costs (i.e., delay and energy), while achieving fast model convergence. We unveil that achieving this goal involves solving a complex NP-hard problem. To tackle this, we propose a stagewise methodology that splits the solution into two stages, referred to as Plan A and Plan B. Plan A focuses on identifying long-term clients with high chance of participation in subsequent model training rounds. Plan B serves as a backup, selecting alternative clients when long-term clients are unavailable during model training rounds. This stagewise methodology offers a fresh perspective on client selection that can enhance both HFL and conventional FL via enabling low-overhead decision-making processes. Through evaluations on diverse datasets, we show that our methodology outperforms existing benchmarks on crucial factors such as model accuracy and system costs.

Index Terms—Hierarchical federated learning, stagewise strategy, client selection, client-to-edge association, dynamic networks

arXiv:2502.09303v3 [cs.LG] 21 Sep 2025

1 Introduction

The proliferation of smart devices has led to the generation of massive amounts of data, fueling the rise of artificial intelligence (AI)/machine learning (ML) applications across Internet-of-Things (IoT) ecosystem [1]. Conventional AI/ML methods typically involve pooling the data of distributed IoT devices at a central location (e.g., a server) for model training. However, privacy regulations often restrict the transfer of devices' data across the network. This has triggered a paradigm shift from centralized to distributed AI/ML methods, with federated learning (FL) [3], [4], [43] emerging as a common approach. FL starts with a central server, e.g., a cloud server (CS), broadcasting a global model to clients (e.g., mobile devices). Subsequently, each client first synchronizes its local model with the global model and then trains its local model on its data (e.g., via gradient descent). Each client then uploads its local model parameters back to the server. The server finally aggregates the received models to update the global model (e.g., via weighted averaging). This process of local training and global model aggregations continues until reaching the desired performance [2].

FL naturally relies on frequent model transmissions between clients and the CS, which can incur high overhead (e.g., latency and energy) [5]–[7]. Hierarchical FL (HFL) is a well-recognized framework to address this challenge [8], [9]. HFL introduces an intermediate layer to the model training architecture of FL for intermediate model aggregations (a layer of edge servers (ESs)), that synchronize the model parameters of their associated clients frequently, and exchange model updates with the CS with a lower frequency [10].

1.1 Motivation and Research Direction

Although HFL has tremendous potentials in enhancing the time and energy efficiency of FL, its implementation in large-scale, dynamic networks (e.g., with mobile devices acting as clients and multiple ESs) faces unexplored challenges. We identify and study three major research questions: *i) How to actively select/recruit the dynamic and resource-constrained clients to achieve cost-efficiency (e.g., reducing the training*

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 62271424, 62401486, 62088101, 72171172; Shanghai Pujiang Programme under Grant no. 24PJD117; the Shanghai Municipal Science and Technology Major Project under Grant no. 2021SHZDZX0100; the Aeronautical Science Foundation of China under Grant no. 2023Z066038001; the Chinese Academy of Engineering, Strategic Research and Consulting Program under Grant no. 2023-XZ-65. *M. Wu (minghongwu@stu.xmu.edu.cn) is with the School of Informatics, Xiamen University, Fujian, China. M. Liwang (minghuiliwang@tongji.edu.cn) and L. Li (lili@tongji.edu.cn) are with the Department of Control Science and Engineering, and also with the Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai, China. Yuhan Su (ysu@xmu.edu.cn) is with the School of Electronic Science and Engineering, Xiamen University, Xiamen 361005, China. S. Hosseinalipour (alipour@buffalo.edu) is with Department of Electrical Engineering, University at Buffalo–SUNY, NY, USA. X. Wang (xianbin.wang@uwo.ca) is with the Department of Electrical and Computer Engineering, Western University, Ontario, Canada. H. Dai (hdai@ncsu.edu) is with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, USA. Z. Jiao (jiaozhenzhen@teleinfo.cn) is with the iF-Labs, Beijing Teleinfo Technology Co., Ltd., CAICT, China. Corresponding author: Minghui Liwang (minghuiliwang@tongji.edu.cn)*

delay and energy consumption)? Due to limited bandwidth availability, only a subset of clients can be selected/recruited for interactions with ESs [12]. The intermittent availability and heterogeneity of clients (i.e., varying computational and communication capabilities), as well as differences in their data qualities can further complicate the client selection [11]. *ii) How to associate the selected clients to proper ESs to achieve resource-efficiency and acceleration of model convergence?* Given a set of selected clients, different client-to-edge (C2E) association strategies can significantly influence the model convergence, due to non-independent and identically distributed (non-IID) nature of data across clients [13]. We hypothesize that optimal convergence occurs when the C2E association results in clusters of devices at each ES having a collective data distribution closely resembling the global dataset spread across all clients. This aligns with the intuition that effective device clustering should mitigate clients' local model biases during local model aggregations with ESs. This not only accelerates convergence but also diminishes the reliance on resource-intensive edge-to-cloud model aggregations, echoing observations noted in prior research [8], [10], [14]. *iii) How to expedite the overall decision-making process required for determining client selection and C2E association (i.e., reduce the decision-making overhead)?* The dynamics of client availability, varying local workloads, and individual selfishness necessitate frequent decision-making at the network operator level, such as at each global model training round in HFL. The requirement for constant decision-making regarding the client recruitment and C2E association can lead to significant overhead (i.e., increased latency and energy consumption). Thus, such frequent decision-making can disrupt the training process and complicate the necessary synchronizations within the training period.

1.2 Overview and Summary of Contributions

We study HFL over the device-edge-cloud continuum with intermittent client participation. Due to client availability dynamics, client selection and C2E association must be frequently updated. We address the underexplored challenge of reducing decision-making overhead by formulating a joint client selection and C2E association problem and proposing a low-overhead stagewise decision-making methodology. In essence, our approach shifts most decision-making from real time, as commonly done in FL and HFL, to long-term planning, thus accelerating training. To remain adaptive, it further incorporates lightweight real-time refinements. Our main contributions are:

• We are interested in addressing the challenge of joint client selection and C2E association for HFL under intermittent client participation, by formulating an optimization aimed at minimizing system costs, while maintaining a tolerable level of data heterogeneity among clients connected to each ES. We reveal that this problem is NP-hard, rendering its solution intractable, especially for large-scale networks.

• To manage the complexity, we implement a stagewise methodology that breaks the problem into two subproblems, solved at different points in the timeline: Plan A, deployed in advance to model training, and Plan B, implemented concurrently with model training. These stages function as asynchronous yet complementary. Plan A identifies clients who are likely to participate in future model training, while Plan B ensures seamless model training if clients selected under Plan A become unavailable.

• In Plan A, we introduce the concept of "client continuity" and derive closed-form constraints on tolerable data heterogeneity by modeling them as risks. To identify suitable long-term clients, we propose a strategy that combines gain-of-cost minimization-promoted C2E association and local iteration-based determination. This stage provides a novel pre-decision-making process where pre-selected clients can directly participate in each global training round as long as they are online, thus reducing the burden of real-time decision-making. In Plan B, we refine the decisions from Plan A in cases some long-term clients are unavailable (either dropped out or are offline) during a practical model training round. To address this, we introduce a cluster-based client update algorithm for the rapid exploration of suitable alternative clients, achieving an approximate optimal solution for real-time client recruitment.

• We conduct extensive experiments on various datasets (i.e., MNIST, Fashion-MNIST, CIFAR-10 and CIFAR-100, and ALI), verifying our superior performance as comparing with benchmark methods in terms of model accuracy, system cost, and time efficiency.

2 Literature Investigation

Numerous studies investigated balancing the model training performance (e.g., model accuracy) and cost-effectiveness for FL. For example, [15]–[17] study model compression, gradient quantization, and coding techniques to reduce the model aggregation overhead. Also, [18]–[20], [61] conduct communication/computation resource allocation for FL. Further, [21], [22] aim to reduce the required global aggregation rounds. These works study FL with direct client-to-cloud communications, which can cause increased latency, energy consumption, and congestion over the backhaul links.

To address the above discussed issues, HFL was proposed by *Liu et al.* [8], introducing an intermediate aggregation layer (e.g., formed by ESs) to reduce the frequency of direct communications between clients and the remote cloud. *Feng et al.* [24] further optimized subcarrier assignment, transmit power of clients, and computation resources in HFL. *Lin et al.* [25] proposed control algorithms for delay-aware HFL, obtaining policies to mitigate communication latency in HFL.

Considering the constraints of limited resources in wireless networks, developing the joint client selection and the C2E association strategy for HFL emerges as a viable approach to improve model training efficiency, which has attracted a great deal of research attention. For example, *Luo et al.* [5] conducted resource allocation and C2E association for HFL, enabling great potentials in low latency and energy-efficient FL. *Deng et al.* [10] minimized the total communication cost required for model learning with a target accuracy, by making decisions on edge aggregator selection and node-edge associations under the HFL framework. *Su et al.* [26] developed an online learning-based client selection algorithm for ESs, based on empirical learning results, reducing the cumulative delay on computation and communication. *Qu et al.* [27] observed the context of local computing and transmission of client-ES pairs, making client

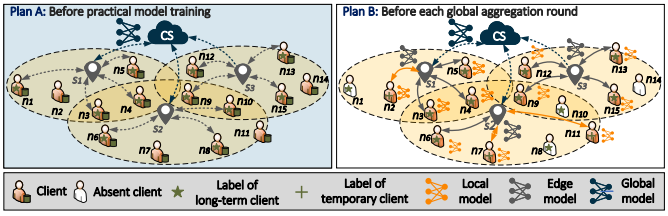


Fig. 1: A schematic of stagewise desicion-making for HFL over dynamic device-edge-cloud continuum with intermittent client participation.

selection decisions to maximize the network operator's utility, under a given budget.

Besides HFL, several novel architectures have been proposed to address the high communication overhead caused by frequent interactions with the remote server. For example, *Guo et al.* [53] proposed a hybrid local stochastic gradient descent (HL-SGD) algorithm, which leverages the availability of fast D2D links to accelerate convergence and reduce training time under non-IID data distributions. However, similar to traditional FL and HFL, the central entity in HL-SGD may still become a bottleneck and suffer from a single point of failure, raising concerns about fault tolerance and scalability. To tackle this issue, a cooperative federated edge learning (CFEL) [54], [55] framework was introduced, which exploits multiple aggregators and eliminates single points of failure, making it more scalable than prior FL frameworks. *Castiglia et al.* [54] proposed multi-level local SGD (MLL-SGD) in a two-tier communication network with heterogeneous workers with IID data distributions. *Zhang et al.* [55] designed a novel federated optimization method called cooperative edge-based federated averaging (CE-FedAvg), which efficiently learns a shared global model over the collective dataset with non-IID data of all edge devices, under the orchestration of a distributed network of cooperative edge servers.

Existing studies have advanced model training by improving client participation and optimizing network orchestration, laying a solid foundation for efficient FL. However, most assume persistent client availability or overlook the decision-making overhead of client recruitment and association with ESs, namely, factors critical in dynamic, resource-constrained environments. To this end, we explore an underexamined research angle and propose stagewise decision-making for HFL under intermittent client availability. Our method employs two complementary plans for client selection and C2E association at different timescales, ensuring seamless and efficient model training.

3 System Model

As shown in Fig. 1, we consider hierarchical federated learning (HFL) over a three-layer network hierarchy including *i)* a cloud server (CS) acting as the centralized aggregator, *ii)* several edge servers (ESs), denoted by $\mathcal{S} = \{s_1, \ldots, s_j, \ldots, s_{|\mathcal{S}|}\}$, acting as intermediate aggregators, and *iii)* a set of mobile devices (clients) $\mathcal{N} = \{\boldsymbol{n}_1, \ldots, \boldsymbol{n}_i, \ldots, \boldsymbol{n}_{|\mathcal{N}|}\}$. Each client $\boldsymbol{n}_i \in \mathcal{N}$ is described via a 4-tuple $\boldsymbol{n}_i = [\mathcal{D}_i, v_i, q_i, \xi_i]$, where $\mathcal{D}_i = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_k, y_k), \ldots, (\boldsymbol{x}_{|\mathcal{D}_i|}, y_{|\mathcal{D}_i|})\}$ denotes its local dataset and $\boldsymbol{x}_k$ and y_k denote the input feature and label of the k^{th} datapoint, respectively. Let $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_i, \ldots, \mathcal{D}_{|\mathcal{N}|}\}$ encapsulate the dataset profile of clients. Also, let v_i denote the computation capability of $\boldsymbol{n}_i$ (i.e., its CPU frequency), and q_i represent the transmit power of $\boldsymbol{n}_i$. Learning takes place through a sequence of global and local model aggregations, where the index of an arbitrary global model aggregation is denoted by g and the index of an arbitrary local aggregation is denoted by ℓ. We capture the intermittent client availability across global aggregations via $\xi_i^{(g)}$, which is an indicator function for describing the participation of client $\boldsymbol{n}_i \in \mathcal{N}$. In particular, $\xi_i^{(g)} = 1$ indicates that $\boldsymbol{n}_i$ takes part in the g^{th} global aggregation, and $\xi_i^{(g)} = 0$, otherwise ($\xi_i^{(g)}$ is detailed in Sec. 3.3).

Each global aggregation[1] of HFL starts with broadcasting the global model from CS to clients, with ESs acting as relays. Afterwards, clients synchronize their local models with the global model, train their local models on their datasets, and then upload their model parameters to their assigned ESs. After aggregating the received models from their clients (i.e., conducting local model aggregation), ESs have two options: *i)* broadcasting the aggregated edge model back to their clients for another round of local model training (i.e., conclusion of one local/edge aggregation round); *ii)* sending the aggregated edge model to the CS for global aggregation (i.e., conclusion of one global aggregation round). This procedure calls for addressing both client recruitment/selection (i.e., determining proper clients for training) and client-to-edge (C2E) association (i.e., assigning clients to proper ESs), which given their combinatorial natures are NP-Hard (discussed in Sec. 4). Thus, the overhead of solving these problems can become unacceptable in large-scale networks, affecting the sustainability and practicality of HFL.

Subsequently, our primary goal is *to recruit clients and assign them to appropriate ESs at each global aggregation round, while ensuring i) the reliability and performance of the system, meaning seamless execution of HFL with an adequate number of clients and fast convergence of the trained global model; ii) low decision-making overhead, which involves minimizing the time spent on real-time decision-making to maximize the time available for model training; iii) reduced costs (i.e., energy consumption and latency) of model training.* We achieve this goal via a stagewise decision-making process, consisting of Plan A and Plan B, where in Plan A, a set of long-term clients (often those with higher probabilities of participation in model training) are selected for future global model aggregations in a pre-decision manner. Also, Plan B involves real-time recruitment and orchestration of short-term clients to compensate for long-term ones that are unavailable at each practical global aggregation round. For instance, as illustrated in Fig. 1, consider the case where client $\boldsymbol{n}_1$ is pre-selected by Plan A as a long-term participant. Due to dynamism, $\boldsymbol{n}_1$ may become unavailable before the actual training starts, which may result in a partial loss of training data. To mitigate this, Plan B activates a replacement strategy by selecting client $\boldsymbol{n}_2$ as a temporary participant, thereby filling the data gap and preserving both the seamlessness and efficiency of HFL progress. Key notations are summarized in Appx. A.

1. "Global aggregation" and "global iteration" are exchangeably used.

3.1 Modeling of HFL Procedure of Our Interest

We introduce $\boldsymbol{A}^{(g)} = \left\{a_{i,j}^{(g)} \mid \boldsymbol{n}_i \in \mathcal{N}, s_j \in \mathcal{S}\right\}$, where $a_{i,j}^{(g)} = 1$ if client $\boldsymbol{n}_i$ is *selected/recruited and associated with ES* s_j; otherwise $a_{i,j}^{(g)} = 0$. We next formalize different HFL steps.

• ***Step 1***: Let $\boldsymbol{\omega}_i^{(g;\ell,t)}$ denote the local model of client $\boldsymbol{n}_i$, at edge aggregation round ℓ ($0 \leq \ell \leq \mathcal{L}-1$), where t is the index for local updates. After receiving the global model $\boldsymbol{\omega}^{(g)}$ broadcasted by CS at global aggregation g, each client $\boldsymbol{n}_i$ first synchronizes its local model as $\boldsymbol{\omega}_i^{(g;0,0)} \leftarrow \boldsymbol{\omega}^{(g)}$. Then, $\boldsymbol{\omega}_i^{(g;\ell,t)}$ is obtained via stochastic gradient descent (SGD) as

$$\boldsymbol{\omega}_i^{(g;\ell,t)} = \boldsymbol{\omega}_i^{(g;\ell,t-1)} - \eta \widetilde{\nabla} F_i(\boldsymbol{\omega}_i^{(g;\ell,t-1)}), \quad 1 \leq t \leq \mathcal{T}, \quad (1)$$

where η is the learning rate and $\widetilde{\nabla}$ denotes the stochastic gradient obtained over a mini-batch of datapoints selected uniformly at random. Also, F_i is the local loss function at $\boldsymbol{n}_i$ defined as $F_i(\boldsymbol{\omega}) = \frac{1}{d_i}\sum_{k=1}^{d_i} f_i(\boldsymbol{x}_k, y_k; \boldsymbol{\omega})$, where $d_i = |\mathcal{D}_i|$ represents the data size of client $\boldsymbol{n}_i$, and f_i is the loss function of the machine learning model (e.g., cross-entropy loss).

• ***Step 2***: Clients will upload their local model parameters to ESs after $\mathcal{T}$ local iterations, followed by each ES s_j aggregating these models via federated averaging [2] as

$$\boldsymbol{\omega}_j^{(g;\ell+1)} = \sum_{i=1}^{|\mathcal{N}|} a_{i,j}^{(g)} \xi_i^{(g)} d_i \boldsymbol{\omega}_i^{(g;\ell,\mathcal{T})} / D_j^{(g)}, \quad (2)$$

where $D_j^{(g)} = \sum_{i=1}^{|\mathcal{N}|} a_{i,j}^{(g)} \xi_i^{(g)} d_i$ captures the dataset size of all clients associated with ES s_j. The aggregated edge model $\boldsymbol{\omega}_j^{(g;\ell)}$ is sent to the CS if $\ell + 1 = \mathcal{L}$; otherwise it is returned to associated clients for the next local update.

• ***Step 3***: The CS aggregates the model parameters sent from the ESs to form a global model as

$$\boldsymbol{\omega}^{(g+1)} = \sum_{j=1}^{|\mathcal{S}|} D_j^{(g)} \boldsymbol{\omega}_j^{(g;\mathcal{L})} / D^{(g)}, \quad (3)$$

where $D^{(g)} = \sum_{j=1}^{|\mathcal{S}|} D_j^{(g)}$, which is broadcasted back to clients to initiate the next round of local model training.

3.2 Modeling of Computation and Communication

The latency and energy consumption at each client $\boldsymbol{n}_i$ for conducting one SGD update are $t_i^{\text{cmp}} = \frac{c_i \zeta_i d_i}{v_i}$ and $e_i^{\text{cmp}} = \alpha_i v_i^2 c_i \zeta_i d_i$, respectively, where $\zeta_i \in (0, 1]$ is the fraction of local dataset in SGD mini-batch. Here, c_i is the number of cycles of processor (e.g., CPUs and GPUs) required for processing one datapoint, α_i is the chipset capacitance of $\boldsymbol{n}_i$, and v_i is $\boldsymbol{n}_i$'s of processor frequency[2] [23].

We assume that each ES s_j allocates a fixed bandwidth B_j to each of its clients[3], and the available bandwidth can support M_j clients concurrently [10]. Accordingly, uplink data rate from client $\boldsymbol{n}_i$ to ES s_j can be computed as $r_{i,j}^{(g)} = B_j \log_2\left(1 + \frac{q_i h_{i,j}^{(g)}}{N_0 B_j}\right)$, where $h_{i,j}^{(g)}$ is the channel gain between $\boldsymbol{n}_i$ and s_j [28] during global aggregation g, and N_0 is the noise power spectral density. Let Ω denote the size (in bits) of the model parameters of clients [8]. The transmission delay and energy consumption between $\boldsymbol{n}_i$ and s_j during an edge aggregation in global aggregation round g are given by $t_{i,j}^{(g),\text{com}} = \Omega / r_{i,j}^{(g)}$ and $e_{i,j}^{(g),\text{com}} = q_i t_{i,j}^{(g),\text{com}}$, respectively. Subsequently, the overall delay and energy consumption of the interactions between $\boldsymbol{n}_i$ and s_j to conduct an edge aggregation during global aggregation g are $T_{i,j}^{(g)} = \mathcal{T} t_i^{\text{cmp}} + t_{i,j}^{(g),\text{com}}$, and $E_{i,j}^{(g)} = \mathcal{T} e_i^{\text{cmp}} + e_{i,j}^{(g),\text{com}}$, respectively.

We make a few standard assumptions to facilitate analysis. First, the computing time and energy consumption of ESs and CS for model aggregation are ignored (supported by [24]). Second, for the wired transmissions between ESs and the CS, the transmission delay $T_j^{(g),\text{com}}$ and energy consumption $E_j^{(g),\text{com}}$ for ES s_j are assumed to be known (as supported by [29], relying on the real-world simulations). Third, due to the large downlink bandwidth and transmit power of ESs, we neglect the downlink latency and energy consumption [24].

Consequently, the overall training delay $\mathbb{T}^{(g)}$, and energy consumption $\mathbb{E}^{(g)}$ of global aggregation g are given by

$$\mathbb{T}^{(g)} = \max_{s_j \in \mathcal{S}} \left\{ \mathcal{L} \max_{\boldsymbol{n}_i \in \mathcal{N}} \left\{ a_{i,j}^{(g)} \xi_i^{(g)} T_{i,j}^{(g)} \right\} + T_j^{(g),\text{com}} \right\}, \quad (4)$$

$$\mathbb{E}^{(g)} = \sum_{s_j \in \mathcal{S}} \left(\mathcal{L} \sum_{\boldsymbol{n}_i \in \mathcal{N}} a_{i,j}^{(g)} \xi_i^{(g)} E_{i,j}^{(g)} + E_j^{(g),\text{com}} \right). \quad (5)$$

3.3 Modeling of Client Participation

Given the stochastic nature of client availability, the status indicator of client $\boldsymbol{n}_i$ in the g^{th} global aggregation $\xi_i^{(g)}$ is probabilistic. To reflect this, we model it as a Bernoulli random variable (e.g., $\xi_i^{(g)} \sim \mathcal{B}(p_i)$), where $E[\xi_i^{(g)}] = p_i$. Note that p_i can be obtained via inspecting the client $\boldsymbol{n}_i$'s historical behavior[4]. In particular, let τ indicate the length of observation rolling window and K denote the total number of windows. We estimate each p_i according to the historical statistics of client n_i in executing model training tasks over the previous $K\tau$ global iterations. Formally, at the end of the K^{th} observation rolling window, we calculate the estimated online probability p_i for client n_i by combining these weighted observations as follows:

$$\widetilde{p_i} = \sum_{\kappa=1}^{K} \frac{2\kappa}{K(K+1)} \sum_{g=(\kappa-1)\tau+1}^{\kappa\tau} \frac{\mathbf{X}_i^{(g)}}{\tau}, \quad (6)$$

where $\mathbf{X}_i^{(g)}$ is the observed value of $\xi_i^{(g)}$ at the g^{th} time instant of window κ. Fresher historical records (e.g., the $(K-1)^{\text{th}}$

2. The processor frequency v_i is a virtualized concept designed to capture different processor capabilities. For example, a larger v_i may correspond to a GPU, while a smaller v_i reflects a CPU-only device, which naturally expresses client heterogeneity. Similarly, it is common in FL studies [10], [26], [50] to use such abstract parameters to represent client computing capacity rather than specifying processor types explicitly, thereby capturing device heterogeneity without binding to specific hardware.

3. We assume that orthogonal frequency-division multiple access (OFDMA) is used for clients to upload their models to ESs, allowing us to ignore the interference (similar to [30]). The C2E channels are assumed to be stable during each global aggregation duration [31], [32], [48].

4. Probabilistic methods (e.g., Poisson point processes) and channel models (e.g., Gauss-Markov) are widely used to characterize network behavior from historical data [45], [46]. Their popularity stems from the fact that network conditions often evolve gradually, allowing a finite history to reliably guide future decisions. Leveraging such models enables effective capture of network dynamics, leading to more accurate forecasting and informed decision-making.

window is closer to the present time than the $(K-2)^{\text{th}}$) are assigned with larger weights (i.e., $\frac{\kappa}{\sum_{\kappa=1}^{K}\kappa}=\frac{2\kappa}{K(K+1)}$) in (6).

During the training process, the online status of clients can be continuously updated and incorporated into the historical data, enabling real-time improvements in predicting their future online behaviors. Consequently, the estimation of a client's online probability and the implementation of Plan A are not limited to the initial phase of training but can also be performed repeatedly throughout the training period (an example is shown in Fig. 7, Appx. H). Also, please note that in the main text, for analytical simplicity, we assume that Plan A is implemented for once before the training process, namely, before the first global iteration starts.

4 Stagewise Client Selection and C2E Association over Device-Edge-Cloud Hierarchy

This section elaborates on our stagewise HFL mechanism, where our major goal is *to determine appropriate clients for each ES to accelerate the model training, while minimizing the training delay (i.e., the time spent on transmission and data processing) along with the energy costs (i.e., the energy consumption of clients induced by model parameter transfer and local training)*.

4.1 Primary Optimization Formulation

4.1.1 Key criteria in constraint design: reaching adequate and balanced data for model training

To ensure a desired convergence rate for the global model, it is essential to engage enough clients (corresponding to enough training data [20]) while keeping the heterogeneity of data of clients associated with each ES within a reasonable range, to avoid local model bias. To this end, we begin by introducing Kullback-Leibler divergence (KLD) [10], [47] as a measure to quantify the difference between the data distribution covered by each ES and a reference data distribution (e.g., the uniform distribution), attempting to assess the degree of data heterogeneity across ESs without exposing their data. Considering a supervised learning task where the input data is divided into $|\mathcal{Z}|$ classes based on the label set $\mathcal{Z}=\{z_1,\ldots,z_h,\ldots,z_{|\mathcal{Z}|}\}$. We define the set of datapoints in client $\boldsymbol{n}_i$'s dataset with label z_h as $\boldsymbol{y}_i(h)=\{(\boldsymbol{x}_k,y_k)\in\mathcal{D}_i\mid y_k=z_h\}$. Accordingly, the KLD of edge data at ES s_j can be defined as follows:

$$\mathrm{KLD}\left(\boldsymbol{P}_j^{(g)}||\boldsymbol{Q}\right)=\sum_{h=1}^{|\boldsymbol{\mathcal{Z}}|}\boldsymbol{P}_j^{(g)}(h)\log\frac{\boldsymbol{P}_j^{(g)}(h)}{\boldsymbol{Q}(h)},\forall s_j\in\mathcal{S},\quad(7)$$

where

$$\boldsymbol{P}_j^{(g)}(h)=\frac{\sum_{i=1}^{|\mathcal{N}|}a_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{D_j^{(g)}},\forall s_j\in\mathcal{S},\forall z_h\in\mathcal{Z}.\quad(8)$$

Specifically, $\boldsymbol{P}_j^{(g)}$ represents the data distribution of ES s_j at global aggregation g, while $\boldsymbol{Q}$ describes the reference data distribution shared to all clients [33]. Upon constraining the KLD acorss ESs, we can impose a high level of similarity across the associated clients, and thus a faster model convergence due to a lower model bias (i.e., constraint (9a) in the later formulation). Also, since having sufficient training data plays a crucial role in the model convergence, we consider a tolerable minimum data size $D_{\min}$ to be covered by each ES (i.e., constraint (9b) in the later formulation).

4.1.2 Optimization problem

We next formulate the joint problem of client selection and C2E association for each global aggregation g as $\boldsymbol{\mathcal{P}}_0^{(g)}$, with the goal of minimizing the weighted sum of delay $\mathbb{T}^{(g)}$ and energy cost $\mathbb{E}^{(g)}$, under KLD and data size constraints:

$$\boldsymbol{\mathcal{P}}_0^{(g)}:\ \min_{\boldsymbol{A}^{(g)}}\ \lambda_t\mathbb{T}^{(g)}+\lambda_e\mathbb{E}^{(g)}\quad(9)$$

$$\text{s.t.}\quad \mathrm{KLD}\left(\boldsymbol{P}_j^{(g)}||\boldsymbol{Q}\right)\le KLD_{\max},\forall s_j\in\mathcal{S}\quad(9a)$$

$$D_j^{(g)}\ge D_{\min},\forall s_j\in\mathcal{S}\quad(9b)$$

$$a_{i,j}^{(g)}\in\{0,1\},\forall\boldsymbol{n}_i\in\mathcal{N},\forall s_j\in\mathcal{S}\quad(9c)$$

$$\sum_{j=1}^{|\mathcal{S}|}a_{i,j}^{(g)}\le 1,\forall\boldsymbol{n}_i\in\mathcal{N}\quad(9d)$$

$$\sum_{i=1}^{|\mathcal{N}|}a_{i,j}^{(g)}\le M_j,\forall s_j\in\mathcal{S}\quad(9e)$$

$$a_{i,j}^{(g)}=0,\text{if }\xi_i^{(g)}=0,\forall\boldsymbol{n}_i\in\mathcal{N},\forall s_j\in\mathcal{S}\quad(9f)$$

where λ_t and λ_e are weighting coefficients, signifying the importance of energy consumption and delay. Constraint (9a) restricts the KLD of the data covered by each ES below a maximum threshold, denoted as $KLD_{\max}$, to ensure balanced training data across the ESs (which reflects in low model bias). Constraint (9b) ensures sufficient training data contributed by clients associated with each ES. Constraint (9c) forces binary values for $a_{i,j}^{(g)}$, constraint (9d) ensures that each client is linked to at most one ES, while constraint (9e) limits the number of clients that are covered by each ES s_j. Lastly, constraint (9f) ensures that only available/participating clients are assigned to the ESs.

Optimization $\boldsymbol{\mathcal{P}}_0^{(g)}$ is a 0-1 integer programming (01-IP), making it NP-Hard [36]. Specifically, as the number of clients increases (i.e., as $\xi_i^{(g)}$ grows), identifying and associating clients with proper ESs can become increasingly complex, which is why many studies in FL do not prioritize optimal solutions [50]–[52]. To mitigate the excessive solution overhead that could disrupt the continuity of model training, we reformulate $\boldsymbol{\mathcal{P}}_0^{(g)}$ into two subproblems, representing two asynchronous yet complementary stages of decision-making[5]. These two subproblems are addressed via two plans. *i) Plan A* entails a pre-decision-making process in advance to practical model training. It conducts long-term decision-making by relying on historical statistics of client participation. *ii) Plan B* is a real-time decision-making process that takes place at the beginning of each global iteration based on the present clients (i.e., those with $\xi_i^{(g)}=1$). Specifically, Plan A identifies long-term clients who are likely to participate in subsequent global

5. We solve problem $\boldsymbol{\mathcal{P}}_0^{(g)}$ from a unique angle by segmenting the into two distinct stages, diverging from the conventional approach of simplifying a complex problem, which often involves two easier subproblems. Consequently, while the subproblems in Plan A and Plan B may still be complex, they possess a reduced problem size.

training rounds shifting the overhead of real-time decision-making to the long-term phase. Since solely relying on Plan A[6] is impractical (e.g., due to the unpredictable absence of long-term works), Plan B serves as a contingency strategy, identifying necessary clients during each global aggregation to ensure a seamless model training. The overall workflow of our stagewise decision-making mechanism is summarized in Alg. 1 (provided in Appx. B), which explicitly illustrates how Plan A (pre-decision-making via Algs. 2–3 for solving $\boldsymbol{\mathcal{P}}_1^{(g)}$) and Plan B (real-time decision-making via Alg. 4 for solving $\boldsymbol{\mathcal{P}}_2^{(g)}$) are coordinated throughout the training process.

4.2 Problem Transformation

We next tackle $\boldsymbol{\mathcal{P}}_0^{(g)}$ by decoupling it into the two subproblems $\boldsymbol{\mathcal{P}}_1^{(g)}$ and $\boldsymbol{\mathcal{P}}_2^{(g)}$, addressed by Plan A and Plan B. In this section, we provide a practical and intuitive view for tackling the 0-1 integer programming problem, by leveraging clients' historical information through a two-stage segmentation, to accelerate model training while balancing accuracy and system cost.

4.2.1 Design of Plan A

Since Plan A requires preselecting some clients and assign them to the appropriate ESs before the start of model training, we begin by obtaining its intended subproblem $\boldsymbol{\mathcal{P}}_1^{(g)}$. This subproblem offers an unique long-term perspective, with the aim of minimizing the system cost by selecting relatively "stable" clients. In other words, these clients are more likely to participate in the subsequent global training rounds, thereby reducing the workload of Plan B, i.e., reducing the problem size and its search space through decreasing the number of clients that need to be considered.

To develop Plan A, one naive approach to reach its long-term objective is to minimize the expectation of the overall system cost given by (9) under the distribution of clients participation (i.e., the expected value of $\lambda_t\mathbb{T}^{(g)} + \lambda_e\mathbb{E}^{(g)}$). Nevertheless, implementing this approach can inadvertently result in choosing clients with lower participation probabilities. Although this may reduce transmission delay and energy costs, it does not align with another key objective of Plan A, which is ensuring consistent and reliable client participation to reduce the overhead of real-time decision-making in Plan B. To cope with this, we first introduce a unique consideration by setting $\xi_i^{(g)} = 1$ ($\forall \boldsymbol{n}_i \in \mathcal{N}$) in both $\mathbb{T}^{(g)}$ and $\mathbb{E}^{(g)}$, updated by $\widehat{\mathbb{T}}^{(g)}$ and $\widehat{\mathbb{E}}^{(g)}$, indicating that all the clients are supposed to be online in Plan A. We define a matrix $\widehat{\boldsymbol{A}}^{(g)} = \left\{\widehat{a}_{i,j}^{(g)} \mid \boldsymbol{n}_i \in \mathcal{N}, s_j \in \mathcal{S}\right\}$ to differentiate from the C2E solution of $\boldsymbol{\mathcal{P}}_0^{(g)}$, where $\widehat{a}_{i,j}^{(g)} = 1$ indicates that $\boldsymbol{n}_i$ is selected and associated with s_j as a long-term client in Plan A. If, $\widehat{a}_{i,j}^{(g)} = 1$, client $\boldsymbol{n}_i$ will join each practical global iteration proactively as long as $\xi_i^{(g)} = 1$. To guarantee that clients with a higher chance of participation in the training process are identified in Plan A, we develop the concept of "client continuity". This concept is measured through the following metric, ensuring the consistency of participation among the selected clients in Plan A:

$$\mathbb{C}^{(g)} = \left(\prod_{i:\sum_{j=1}^{|\mathcal{S}|}\widehat{a}_{i,j}^{(g)}=1} p_i\right)^{\frac{1}{\sum_{j=1}^{|\mathcal{S}|}\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}}}. \tag{10}$$

In (10), $\mathbb{C}^{(g)}$ describes the geometric mean of participating probabilities of selected clients for global aggregation g. Specifically, the index term $i : \sum_{j=1}^{|\mathcal{S}|}\widehat{a}_{i,j}^{(g)} = 1$ counts the clients who have been selected and associated with ESs.

Apparently, Plan A employs pre-decisions, including either pre-decision steps made prior to the actual model training (standing for one primary focus of our paper) or additional adjustments implemented concurrently with the training process (e.g., estimation of online probability of clients). In doing so, directly obtaining the original constraints (9a) and (9b) presents a noteworthy challenge. For instance, the practical values of $\xi_i^{(g)}$ in each actual global aggregation round may be unknown. Consequently, we reformulate these constraints into probabilistic expressions, and express them as follows:

$$\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)} || \boldsymbol{Q}\right) > KLD_{\max} - \Delta_k\right) \leq \delta, \forall s_j \in \mathcal{S}, \tag{11a}$$

$$\Pr\left(\widehat{D}_j^{(g)} < D_{\min} + \Delta_d\right) \leq \varepsilon, \forall s_j \in \mathcal{S}. \tag{11b}$$

where $\widehat{\boldsymbol{P}}_j^{(g)}$ and $\widehat{D}_j^{(g)}$ are the updates after replacing $a_{i,j}^{(g)}$ in $\boldsymbol{P}_j^{(g)}$ and $D_j^{(g)}$ with $\widehat{a}_{i,j}^{(g)}$, respectively; while $\Pr(\cdot)$ represents the probability. In these updated constraints, we have introduced two constants Δ_k ($\Delta_k > 0$) and Δ_d ($\Delta_d > 0$) to tune the likelihood of the satisfaction of the original ones (i.e., (9a) and (9b)), aiming to mitigate the risk of obtaining unsatisfactory KLD and data size during practical model training, by ensuring these values remain below thresholds δ and ε[7]. Accordingly, the optimization subproblem $\boldsymbol{\mathcal{P}}_1^{(g)}$, which is intended to be solved in Plan A is given by

$$\boldsymbol{\mathcal{P}}_1^{(g)}: \min_{\widehat{\boldsymbol{A}}^{(g)}} \ \lambda_t\widehat{\mathbb{T}}^{(g)} + \lambda_e\widehat{\mathbb{E}}^{(g)} - \lambda_c\mathbb{C}^{(g)} \tag{11}$$

$$\text{s.t.} \quad \text{(11a), (11b)},$$

$$\widehat{a}_{i,j}^{(g)} \in \{0,1\}, \forall \boldsymbol{n}_i \in \mathcal{N}, \forall s_j \in \mathcal{S} \tag{11c}$$

$$\sum_{j=1}^{|\mathcal{S}|}\widehat{a}_{i,j}^{(g)} \leq 1, \forall \boldsymbol{n}_i \in \mathcal{N} \tag{11d}$$

$$\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)} \leq M_j, \forall s_j \in \mathcal{S} \tag{11e}$$

Constraints (11a) and (11b) in $\boldsymbol{\mathcal{P}}_1^{(g)}$ aim to mitigate the risk of obtaining unsatisfactory KLD and data size during

6. Plan A is flexible and can be executed multiple times alongside model training, allowing pre-decisions to adapt to varying channel conditions, client capacities, and thus online probabilities. For simplicity, we first analyze the case where these factors remain constant, so Plan A runs once before training. In Appx. H, we extend to fluctuating scenarios, where Plan A runs repeatedly, for example, once before training and then synchronized with subsequent training rounds, with each new global iteration reflecting updated client settings.

7. Although these probabilistic constraints might result in suboptimal client selection, they enhance the robustness and effectiveness of the preliminary decisions in Plan A by taking a long-term perspective on constraints related to KLD and data size. Therefore, these assumptions are more likely to hold in practice under dynamic conditions, thereby accelerating the convergence of the global model.

practical model training, by ensuring these values remain below thresholds δ, and ε. Constraint (11d) guarantees that each client can be connected to no more than one ES, while constraint (11e) caps the total number of clients that can be simultaneously connected to each ES during every edge aggregation. Note that obtaining the close form of (11a) and (11b) for arbitrary distribution of client participation is highly challenging. To tackle this challenge, we use the Markov inequality [44] to convert them into tractable forms, which are given by (11f) and (11g) below (details of derivations of these transformations can be found in Appx. C and Appx. D). In Appx. I and Appx. J, we have included more extended experimental evaluations, to further substantiate the theoretical soundness and practical efficacy of our proposed constraints.

$$\prod_{i:\widehat{a}_{i,j}^{(g)}=1}(1-p_i)+\Big(1-\prod_{i:\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\Big) \times\sum_{h=1}^{|\mathcal{Z}|}\frac{\boldsymbol{G}_j^{(g)}(h)}{KLD_{\max}-\Delta_k}\leq\delta,\forall s_j\in\mathcal{S} \quad (11f)$$

$$\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}p_i d_i\geq\left(D_{\min}+\Delta_d\right)(1-\varepsilon),\forall s_j\in\mathcal{S} \quad (11g)$$

where $\boldsymbol{G}_j^{(g)}(h)$ is a piecewise function related to the upper bound of $\mathrm{KLD}(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q})$. It can be construed that with replacing the above two constrains instead of (11a) and (11b) in $\boldsymbol{\mathcal{P}}_1^{(g)}$, $\boldsymbol{\mathcal{P}}_1^{(g)}$ will be a non-linear 01-IP, rendering the attainment of its optimal solution challenging [10], [37]. Also, the constraints on risks, specifically (11f) and (11g), add another layer of complexity. To accelerate reaching an approximate solution for $\boldsymbol{\mathcal{P}}_1^{(g)}$, we delve into designing two lightweight algorithms, which are gain-of-cost minimization-promoted C2E association (GoCMinC2EA) for addressing the C2E association problem, and local iteration-based long-term client determination (LILongClientD) for solving the long-term client selection problem. These two algorithms work iteratively until reaching the near optimum $\boldsymbol{\mathcal{P}}_1^{(g)}$. For brevity, we use $\widehat{F}^{(g)}=\lambda_t\widehat{\mathbb{T}}^{(g)}+\lambda_e\widehat{\mathbb{E}}^{(g)}-\lambda_c\mathbb{C}^{(g)}$ to denote the cost function in Plan A – to distinguish it from $F^{(g)}=\lambda_t\mathbb{T}^{(g)}+\lambda_e\mathbb{E}^{(g)}$.

A. Gain-of-cost minimization-promoted C2E Association (GoCMinC2EA) in Plan A

We outline our approach for addressing the C2E association problem associated with $\boldsymbol{\mathcal{P}}_1^{(g)}$, which can be described as: *how to assign clients within a specified long-term set, $\mathcal{N}^s$, to a ESs to minimize system latency and energy costs, while guaranteeing service continuity within an acceptable risk threshold?* To tackle this, we propose GoCMinC2EA detailed in Alg. 2.

Alg. 2 considers a set of long-term clients $\mathcal{N}^s$ (obtained later by Alg. 3). It first allocates clients exclusively covered by a single ES to that ES (lines 2-3). For clients covered by multiple ESs, we employ a greedy strategy to link each client $\boldsymbol{n}_i$ with one of its accessible ESs from the set $\mathcal{S}_i$ (lines 5-10), aiming to reduce the value of $\widehat{F}^{(g)}$. For unmatched clients, GoCMinC2EA checks the variation denoted by $\Delta\widehat{F}_{i,j}^{(g)}$: the difference of the value of $\widehat{F}^{(g)}$ before (i.e., $\widehat{F}_{(-)}^{(g)}$) and after (i.e., $\widehat{F}_{(+)}^{(g)}$) assigning client $\boldsymbol{n}_i\in\mathcal{N}^o$ to ES $s_j\in\mathcal{S}_i$ (line 6). By evaluating all potential mappings, we allocate a client to an ES that yields the minimum change in $\widehat{F}^{(g)}$, denoted as $\Delta\widehat{F}_{i^*,j^*}^{(g)}$, thereby optimizing the system's performance. Following this assignment, we update the set of not-associated clients $\mathcal{N}^o$ (lines 7-9). We repeat this procedure until every client in $\mathcal{N}^o$ has been assigned to an ES.

Constraints (11e)–(11g) are checked after all clients are associated with ESs. If the conditions are satisfied, we set $\widehat{F}\mathrm{min}^{(g)}=\widehat{F}^{(g)}(+)$ to indicate a feasible association; otherwise, the association is marked infeasible, and a backtracking algorithm is used to refine the greedy solution. In this process, clients are reconsidered one by one in reverse order. For instance, if assigning client n_1 to ES s_1 violates a constraint, we remove this mapping and try alternative ESs (e.g., s_2 or s_3). If no feasible option exists for n_1, we step back to the previous client (e.g., n_2), cancel its mapping, and attempt to reassign it to a different ES, until reaching a feasible solution.

Algorithm 2: Gain-of-cost minimization-promoted C2E association (GoCMinC2EA)

Input : the set of ES $\mathcal{S}_i$ that client $\boldsymbol{n}_i$ can associate with; a given long-term client set $\mathcal{N}^s$
Output: the C2E association result $\widehat{\boldsymbol{A}}^{(g)}=\left\{\widehat{a}_{i,j}^{(g)}\mid\boldsymbol{n}_i\in\mathcal{N},s_j\in\mathcal{S}\right\}$; the minimum value $\widehat{F}_{\min}^{(g)}$ of cost function $\widehat{F}^{(g)}$

1 Initialization: $\widehat{\boldsymbol{A}}^{(g)}\leftarrow\boldsymbol{0}$, $\Delta\boldsymbol{F}^{(g)}\leftarrow\boldsymbol{0}$, $\mathcal{N}^o\leftarrow\emptyset$, $\widehat{F}_{\min}^{(g)}\leftarrow INF$[8];
2 $\mathcal{N}^o\leftarrow\mathcal{N}^o\cup\{\boldsymbol{n}_i\mid|\mathcal{S}_i|>1\}$;
3 Associate each client $\boldsymbol{n}_i\in\mathcal{N}^s\backslash\mathcal{N}^o$ with its available ES and update $\widehat{\boldsymbol{A}}^{(g)}$;
4 **repeat**
5 **repeat**
6 Calculate all feasible cost function variations $\Delta\widehat{F}_{i,j}^{(g)}=\widehat{F}_{(+)}^{(g)}-\widehat{F}_{(-)}^{(g)}$ for associating client $\boldsymbol{n}_i\in\mathcal{N}^o$ with each ES $s_j\in\mathcal{S}_i$, and record them in $\Delta\boldsymbol{F}^{(g)}$;
7 $\Delta\widehat{F}_{i^*,j^*}^{(g)}\leftarrow\min\left\{\Delta\widehat{F}_{i,j}^{(g)}\mid\Delta\widehat{F}_{i,j}^{(g)}\in\Delta\boldsymbol{F}^{(g)},\Delta\widehat{F}_{i,j}^{(g)}>0\right\}$;
8 $\widehat{a}_{i^*,j^*}^{(g)}\leftarrow1$;
9 $\mathcal{N}^o\leftarrow\mathcal{N}^o\backslash\{\boldsymbol{n}_{i^*}\}$;
10 **until** $\mathcal{N}^o=\emptyset$;
11 **if** constraints (11e), (11f) and (11g) are satisfied **then**
12 $\widehat{F}_{\min}^{(g)}\leftarrow\widehat{F}_{(+)}^{(g)}$;
13 **else**
14 Disassociate and mark these associations as "infeasible association";
15 **until** $\widehat{F}_{\min}^{(g)}<INF$ or no feasible association can be found;
16 **return** $\widehat{\boldsymbol{A}}^{(g)}$, $\widehat{F}_{\min}^{(g)}$;

If the solution obtained from GoCMinC2EA either satisfies all the constraints or can be modified into a feasible solution with minimal backtracking, the computational complexity of Alg. 2 could closely approximate that of its greedy component (the best case), i.e., $\mathcal{O}(|\mathcal{N}^o|)$. This implies that the additional steps required for verification and adjustment may not increase the computations, especially if the need for backtracking is limited. However, a worst case may occur where extensive backtracking is necessary to identify a feasible solution, in which case the complexity of GoCMinC2EA will approximate that of the backtracking component. This

8. We define *INF* as a large number, ensuring that any value of $\widehat{F}_{\min}^{(g)}$ satisfying all constraints will be less than *INF*.

suggests that, in situations requiring significant adjustments to the initial mappings, the computations and time needed to reach a feasible solution could increase, aligning with the complexity of the backtracking, e.g., $\mathcal{O}(\prod_{i=1}^{|\mathcal{N}^o|}|\mathcal{S}_i|)$.

B. Local Iteration-based Long-term Client Determination (LILongClientD) in Plan B

We next delve into LILongClientD, which aims to recruit long-term clients that induce a low cost of model training. Since there are $2^{|\mathcal{N}|}$ combinations of clients, considering all combinations is impractical. To this end, we design a lightweight algorithm called LILongClientD (presented in Alg. 3) that updates the selection of clients in an iterative manner. This method optimizes the client selection by focusing on incremental performance gains rather than evaluating every possible combination. We also adopt the introduced GoCMinC2EA algorithm to determine the appropriate C2E associations, aiming to minimize the cost function $\widehat{F}^{(g)}$. In a nutshell, for a given set of clients, $\mathcal{N}^s$, Alg. 3 evaluates whether an alternative set of clients with a better solution than the present one exists. Specifically, in LILongClientD, we design three key operations: *Add* (lines 4-11), *Remove* (lines 12-19) and *Exchange* (lines 20-28), for refining the selected clients, with details moved to Appx. E.

Considering Alg. 3, by initializing a randomly selected client set, LILongClientD repeats the above three operations until no adjustments reduce $F'_{\min}$. According to the number of iterations within the internal loop on *Add*, *Remove* and *Exchange* and the complexity of Alg. 2, denoted by $\mathcal{O}(V)$ (where $\mathcal{O}(|\mathcal{N}^o|) \leq \mathcal{O}(V) \leq \mathcal{O}(\prod_{i=1}^{|\mathcal{N}^o|}|\mathcal{S}_i|)$), their computational complexities are $\mathcal{O}(V|\mathcal{N}\backslash\mathcal{N}^s|)$, $\mathcal{O}(V|\mathcal{N}^s|$ and $\mathcal{O}(V|\mathcal{N}||\mathcal{N}^s|$, respectively. Also, the complexity of Alg. 3 depends heavily on the number of repetitions[9] assumed to be β. In particular, the computation complexity of Alg. 3 is $\mathcal{O}(\beta V|\mathcal{N}||\mathcal{N}^s|)$.

Unlike prior exhaustive or greedy search methods [5], [10], [37], which suffer from heavy backtracking and poor scalability, we design a novel two-stage framework: Plan A conducts offline optimization to handle complex constraints, while Plan B performs lightweight online refinement during training. This design ensures that even as $|\mathcal{N}|$ grows, the real-time decision-making remains highly efficient, effectively mitigating the runtime bottleneck encountered in conventional single-stage decision-making approach.

4.2.2 Design of Plan B

Apparently, implementing Plan A represents a coexistence of both risks and opportunities, as it replies on the estimation of uncertain online behaviors of clients. Although we have made certain efforts to control potential risks that can leave negative impacts on model training (e.g., by using (11a) and (11b)), one case may raise, in which some of the long-term clients are absent from the training process, further degrading the performance of Plan A. Thus, when long-term decision-making becomes ineffective due to the potential absence, we switch to real-time decision-making (Plan B), which is crucial to ensure a desired training performance. In Plan B, we recruit short-term clients at each global aggregation who have not been selected in Plan A with the aim to ensure a seamless training process through obtaining the solution for $\boldsymbol{\mathcal{P}}_2^{(g)}$ given by

$$\boldsymbol{\mathcal{P}}_2^{(g)}: \min_{\widetilde{\boldsymbol{A}}^{(g)}} \lambda_t \widetilde{\mathbb{T}}^{(g)} + \lambda_e \widetilde{\mathbb{E}}^{(g)} \tag{12}$$

$$\text{s.t.} \quad \text{KLD}\left(\widetilde{\boldsymbol{P}}_j^{(g)} || \boldsymbol{Q}\right) \leq KLD_{\max}, \forall s_j \in \mathcal{S} \tag{12a}$$

$$\widetilde{D}_j^{(g)} \geq D_{\min}, \forall s_j \in \mathcal{S} \tag{12b}$$

$$\widetilde{a}_{i,j}^{(g)} \in \{0,1\}, \forall \boldsymbol{n}_i \in \mathcal{N}, \forall s_j \in \mathcal{S} \tag{12c}$$

$$\widetilde{a}_{i,j}^{(g)} = 1, \text{if } \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} = 1, \forall \boldsymbol{n}_i \in \mathcal{N}, \forall s_j \in \mathcal{S} \tag{12d}$$

$$\sum_{j=1}^{|\mathcal{S}|} \widetilde{a}_{i,j}^{(g)} \leq 1, \forall \boldsymbol{n}_i \in \mathcal{N} \tag{12e}$$

$$\sum_{i=1}^{|\mathcal{N}|} \widetilde{a}_{i,j}^{(g)} \leq M_j, \forall s_j \in \mathcal{S} \tag{12f}$$

Algorithm 3: Local iteration-based long-term client determination (LILongClientD)

Input : the set of ES $\mathcal{S}$; the set of clients $\mathcal{N}$
Output: the C2E association result $\widehat{\boldsymbol{A}}^{(g)} = \left\{\widehat{a}_{i,j}^{(g)} \mid \boldsymbol{n}_i \in \mathcal{N}, s_j \in \mathcal{S}\right\}$; the minimum value $\widehat{F}_{\min}^{(g)}$ of the cost function $\widehat{F}^{(g)}$

1 Initialization: a feasible clients selection solution $\mathcal{N}^s$;
2 Obtain $\widehat{\boldsymbol{A}}^{(g)}$, $\widehat{F}_{\min}^{(g)}$ from GoCMinC2EA ($\mathcal{N}^s$);
3 **repeat**
4 //**Operation 1. Add**
5 $\mathcal{N}_0^s \leftarrow \mathcal{N}^s$;
6 **for** $\boldsymbol{n}_i \in \mathcal{N}\backslash\mathcal{N}_0^s$ **do**
7 Obtain $\widehat{\boldsymbol{A}}_{\cup\{\boldsymbol{n}_i\}}^{(g)}$, $\widehat{F}_{\cup\{\boldsymbol{n}_i\}}^{(g)}$ from GoCMinC2EA ($\mathcal{N}_0^s \cup \{\boldsymbol{n}_i\}$) in Alg. 2;
8 **if** $\widehat{F}_{\cup\{\boldsymbol{n}_i\}}^{(g)} < \widehat{F}_{\min}^{(g)}$ **then**
9 $\widehat{F}_{\min}^{(g)} \leftarrow \widehat{F}_{\cup\{\boldsymbol{n}_i\}}^{(g)}$;
10 $\mathcal{N}^s \leftarrow \mathcal{N}_0^s \cup \{\boldsymbol{n}_i\}$;
11 $\widehat{\boldsymbol{A}}^{(g)} \leftarrow \widehat{\boldsymbol{A}}_{\cup\{\boldsymbol{n}_i\}}^{(g)}$;
12 //**Operation 2. Remove**
13 $\mathcal{N}_0^s \leftarrow \mathcal{N}^s$;
14 **for** $\boldsymbol{n}_i \in \mathcal{N}_0^s$ **do**
15 Obtain $\widehat{\boldsymbol{A}}_{\backslash\{\boldsymbol{n}_i\}}^{(g)}$, $\widehat{F}_{\backslash\{\boldsymbol{n}_i\}}^{(g)}$ from GoCMinC2EA ($\mathcal{N}_0^s \backslash \{\boldsymbol{n}_i\}$) in Alg. 2;
16 **if** $\widehat{F}_{\backslash\{\boldsymbol{n}_i\}}^{(g)} < \widehat{F}_{\min}^{(g)}$ **then**
17 $\widehat{F}_{\min}^{(g)} \leftarrow \widehat{F}_{\backslash\{\boldsymbol{n}_i\}}^{(g)}$;
18 $\mathcal{N}^s \leftarrow \mathcal{N}_0^s \backslash \{\boldsymbol{n}_i\}$;
19 $\widehat{\boldsymbol{A}}^{(g)} \leftarrow \widehat{\boldsymbol{A}}_{\backslash\{\boldsymbol{n}_i\}}^{(g)}$;
20 //**Operation 3. Exchange**
21 $\mathcal{N}_0^s \leftarrow \mathcal{N}^s$;
22 **for** $\boldsymbol{n}_i \in \mathcal{N}_0^s$ **do**
23 **for** $\boldsymbol{n}_j \in \mathcal{N} - \mathcal{N}_0^s$ **do**
24 Obtain $\widehat{\boldsymbol{A}}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}^{(g)}$, $\widehat{F}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}^{(g)}$ from GoCMinC2EA ($\mathcal{N}_0^s \backslash \{\boldsymbol{n}_i\} \cup \{\boldsymbol{n}_j\}$) in Alg. 2;
25 **if** $\widehat{F}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}^{(g)} < \widehat{F}_{\min}^{(g)}$ **then**
26 $\widehat{F}_{\min}^{(g)} \leftarrow \widehat{F}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}^{(g)}$;
27 $\mathcal{N}^s \leftarrow \mathcal{N}_0^s \backslash \{\boldsymbol{n}_i\} \cup \{\boldsymbol{n}_j\}$;
28 $\widehat{\boldsymbol{A}}^{(g)} \leftarrow \widehat{\boldsymbol{A}}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}^{(g)}$;
29 **until** no adjustment can reduce $\widehat{F}_{\min}^{(g)}$;
30 **return** $\widehat{\boldsymbol{A}}^{(g)}$, $\widehat{F}_{\min}^{(g)}$;

9. Here, "repetition" refers to the executions in lines 4-28, Alg. 3, i.e., each repetition involves executing each of the three operations once.

Note that $\widetilde{a}_{i,j}^{(g)}$ denotes a binary indicator as shown by constraint (12c), describing the selection of proper online clients that are not long-term ones in Plan A. In particular, we use a set $\widetilde{\boldsymbol{A}}^{(g)}$ to collect $\widetilde{a}_{i,j}^{(g)}$, distinguishing them from the previous discussed $\boldsymbol{A}^{(g)}$ and $\widehat{\boldsymbol{A}}^{(g)}$. More importantly, when $\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}=1$, we have $\widetilde{a}_{i,j}^{(g)}=1$ to involve all the online long-term clients into the training, as depicted by constraint (12d). Similar to $\widehat{\boldsymbol{P}}_j^{(g)}$ and $\widehat{D}_j^{(g)}$, $\widetilde{\boldsymbol{P}}_j^{(g)}$ and $\widetilde{D}_j^{(g)}$ are the results after replacing $a_{i,j}^{(g)}$ in $\boldsymbol{P}_j^{(g)}$ and $D_j^{(g)}$ with $\widetilde{a}_{i,j}^{(g)}$, respectively. More importantly, in Plan B, during the actual training process, the constraints on KLD and data size are treated as standard inequality constraints (i.e., (12a) and (12b)), rather than in Plan A (they are formulated probabilistically with risk-control measures). This adjustment reflects the actual client participation status and facilitates more deterministic and reliable decisions for client selection and C2E association. Together, these complementary plans accelerate convergence by balancing the uncertainty managed in Plan A with the in-situ optimization provided by Plan B.

Algorithm 4: Cluster-based Client Update (CCU)

Input : the clients set $\mathcal{N}_j$ covered by each ES s_j; the C2E association result of Plan A $\widehat{\boldsymbol{A}}^{(g)}=\left\{\widehat{a}_{i,j}^{(g)}\mid \boldsymbol{n}_i\in\mathcal{N},s_j\in\mathcal{S}\right\}$; the clients' participation indicator $\boldsymbol{\xi}^{(g)}=\left\{\xi_1^{(g)},\ldots,\xi_{|\mathcal{N}|}^{(g)}\right\}$

Output: the C2E association result $\widetilde{\boldsymbol{A}}^{(g)}=\left\{\widetilde{a}_{i,j}^{(g)}\mid n_i\in\mathcal{N},s_j\in\mathcal{S}\right\}$; the minimum value of $\widetilde{F}_{\min}^{(g)}$ of $\lambda_t\widetilde{\mathbb{T}}^{(g)}+\lambda_e\widetilde{\mathbb{E}}^{(g)}$

1 Obtain the similarity matrix $\boldsymbol{\Psi}^{(g)}=\left\{\psi_{i,i'}^{(g)}\mid \boldsymbol{n}_i,\boldsymbol{n}_{i'}\in\boldsymbol{\mathcal{N}}\right\}$ via $\psi_{i,i'}^{(g)}=\boldsymbol{u}_i^{(g)}\cdot\boldsymbol{u}_{i'}^{(g)}/(\|\boldsymbol{u}_i^{(g)}\|\times\|\boldsymbol{u}_{i'}^{(g)}\|)$;
2 Use DBSCAN algorithm [38] to cluster $\mathcal{N}_j$ into clusters $\mathcal{C}_j$;
3 **for** $n_i\in\mathcal{N}$ **do**
4 $\widetilde{a}_{i,j}^{(g)}\leftarrow 1$, if $\widehat{a}_{i,j}^{(g)}=1,\forall s_j\in\mathcal{S}$
5 **if** constraint (12a) and (12b) are not satisfied **then**
6 Attempt to replace the selected offline clients with the online ones in the same cluster until constraint (12a) and (12b) are satisfied;
7 **if** someone belonging to the noise dropouts or the backup cannot be found to meet the constraints **then**
8 Apply Algs. 2 and 3 to these clients for iterative optimization;
9 Calculate $\widetilde{F}_{\min}^{(g)}$ via (4), (5) and (12);
10 **retrun** $\widetilde{\boldsymbol{A}}^{(g)}$, $\widetilde{F}_{\min}^{(g)}$;

To tackle $\boldsymbol{\mathcal{P}}_2^{(g)}$, we introduce a cluster-based client update (CCU) algorithm, referred to as Plan B, for timely on-site decision-making. This method is outlined in Alg. 4. In a nutshell, the method starts with obtaining the similarity evaluation function $\psi_{i,i'}^{(g)}=\frac{\boldsymbol{u}_i^{(g)}\cdot\boldsymbol{u}_{i'}^{(g)}}{\|\boldsymbol{u}_i^{(g)}\|\times\|\boldsymbol{u}_{i'}^{(g)}\|}$ to determine the similarity matrix $\boldsymbol{\Psi}^{(g)}=\left\{\psi_{i,i'}^{(g)}\mid \boldsymbol{n}_i,\boldsymbol{n}_{i'}\in\mathcal{N}\right\}$, where $\psi_{i,i'}^{(g)}$ is the cosine similarity between $\boldsymbol{u}_i^{(g)}$ and $\boldsymbol{u}_{i'}^{(g)}$, and $\boldsymbol{u}_i^{(g)}=\left[d_i,T_{i,j}^{(g)},E_{i,j}^{(g)}\right]$ is the vector characterized by data size, delay and energy consumption of client $\boldsymbol{n}_i$ when associated with ES s_j. Given a similarity threshold $\psi_{\min}$ of a *neighborhood* (clients $\boldsymbol{n}_{i'}$ is considered as one of the *neighbors* within the *neighborhood* of client $\boldsymbol{n}_i$ if $\psi_{i,i'}^{(g)}\geq\psi_{\min}$) and a density threshold $P_{\min}$ for the *neighborhood* (there are at least $P_{\min}$ points/clients in the *neighborhood* of a *core point*[10]), we use the density-based spatial clustering of applications with noise (DBSCAN) [38] to cluster $\mathcal{N}_j$ into clusters $\mathcal{C}_j$. While DBSCAN-based clustering is generally effective, it may occasionally fail to form a valid cluster if all candidate clients under an ES significantly differ from the offline long-term client in terms of data characteristics, latency, or energy consumption.

As shown in (12d), we copy $\widehat{\boldsymbol{A}}^{(g)}$ derived from Plan A to $\widetilde{\boldsymbol{A}}^{(g)}$ (lines 3-4). According to the similarity matrix $\boldsymbol{\Psi}^{(g)}$, the clients in the neighborhood of each offline long-term client are sorted in an increasing order of similarity $\psi_{i,i'}^{(g)}$. For each ES, it first selects the same number of offline long-term clients according to $\boldsymbol{\Psi}^{(g)}$ in the corresponding cluster (line 6). Should the constraints remain unmet or if any participant falling into the category of *noise point* dropouts thus no valid cluster can be formed, a fallback mechanism is triggered. Specifically, we first attempt to randomly select an equivalent number of clients as backups to replace the offline long-term clients for each ES. Following this, Algs. 2 and 3 are applied to these backup clients for local iterative optimization (lines 7-8 of Alg. 4)[11].

5 Evaluations

This section presents experiments on both real-world data, EUA dataset[12] (see Sec. 5.1), and numerical simulation data (outlined in Sec. 5.2) to evaluate our proposed StagewiseHFL framework. Simulations were conducted using MATLAB R2021a and Python 3.9 with PyTorch 1.13 on a workstation equipped with an Intel Core i9-13900K @ 3.00 GHz processor, 32 GB memory, and an NVIDIA GeForce RTX 4080 GPU. More details can be found in Appx. F. We focus on three metrics: *i)* model test accuracy, *ii)* the overall cost of reaching a target learning accuracy, calculated by the cost function $\sum_g F^{(g)}=\sum_g(\lambda_t\mathbb{T}^{(g)}+\lambda_e\mathbb{E}^{(g)})$ (the sum of the objective of $\boldsymbol{\mathcal{P}}_0^{(g)}$), and *iii)* running time, reflecting the overhead caused by decision-making, i.e., the time spent on making decisions of C2E association and client selection. The considered benchmarks inspired by [5], [10] are detailed below.

- *OrigProbSolver*: This method directly solves the original problem $\boldsymbol{\mathcal{P}}_0^{(g)}$ by iteratively using Algs. 2 and 3, before the start of each global iteration.
- *KLDMinimization*: This method minimizes the averaged KLD of data between ESs via Algs. 2 and 3, before each global iteration, i.e., to reach the most approximately edge-IID data.
- *ClientSelOnly*: This method iteratively optimizes the selected clients to solve $\boldsymbol{\mathcal{P}}_0^{(g)}$ via Alg. 3, while associating them randomly to ESs, before the start of each global iteration.
- *C2EAssocOnly*: This method randomly selects a set of clients [2] and optimizes C2E associations to solve $\boldsymbol{\mathcal{P}}_0^{(g)}$ via Alg. 2, before the start of each global iteration.

10. For more details about DBSCAN algorithm, please refer to [38].

11. In extreme scenarios where a large number of long-term clients dropout under the same ES, even the fallback may fail to fully satisfy data constraints. Possible remedies include discarding the ES result, reweighting its contribution in global aggregation, or temporarily reassigning its long-term clients to other ESs. We leave these strategies as potential extensions for future work.

12. https://github.com/swinedge/eua-dataset

• *C2EGreedyAssoc*: This method iteratively optimizes the selected clients to solve $\mathcal{P}_0^{(g)}$ via Alg. 3, while greedily associate each client $\boldsymbol{n}_i$ to an ES s_j with the minimum transmission latency $t_{i,j}^{com}$, before each global iteration.
• *FedCS* [56]: This method selects as many clients as possible for each ES via Alg. 3, before the start of each global iteration.

Note that, methods ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, and FedCS, which focus solely on optimizing either the C2E association or the client selection problem, face challenges in simultaneously meeting the constraints related to KLD and data size. Thus, we exclude the KLD as a constraint for these benchmark methods in our evaluation.

5.1 Real-world EUA Dataset-driven Experiments

We first use the real-world EUA dataset, which has been widely employed in edge computing and FL environments. This dataset includes geographic locations of 1,464 base stations (BSs) and 174,305 end-users within the Melbourne metropolitan area, Australia. We selected 93 users as clients and 4 BSs as ESs with a 500m $\times$ 500m region. The dataset provides latitude and longitude of these entities, enabling us to calculate the distances between clients and ESs accordingly.

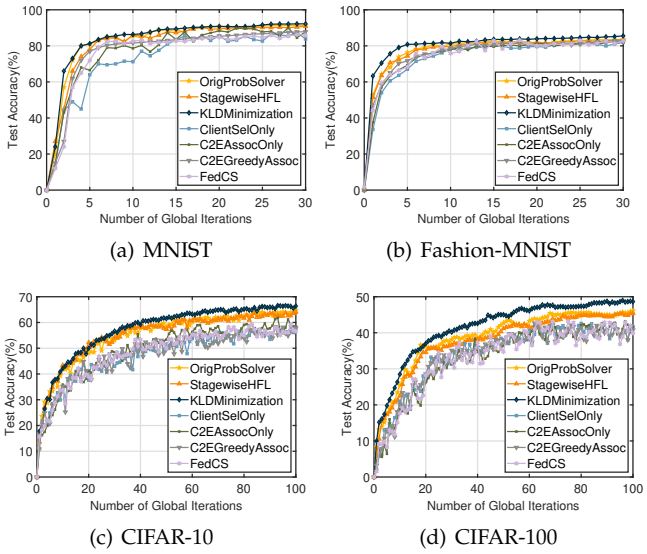
We evaluate on five datasets and models: *i)* MNIST [39] with CNN, *ii)* Fashion-MNIST [41] with LeNet5 [39], *iii)* CIFAR-10 [40] with CNN, *iv)* CIFAR-10 with ResNet18 [42], and *v)* ALI [65], [66] with VGG [67]. Due to space limitation, we report details for the first four in the main document, with extended analysis and results in Appx. G and Appx. L. While existing methodologies fields such as computer vision have attained high test accuracies for these datasets [57]–[60], our primary concern in this paper is to assess the performance of various methods in terms of time and energy efficiency against a predetermined target accuracy, rather than striving for the highest possible accuracy. This also aligns with the rationale of several HFL studies such as [10], [11], [24].

TABLE 1: Simulation setting [5], [24]

Parameter	Value
Online probability, p_i	[0.5,1)
Capacitance coefficient, α_i	10^{-28}
Number of CPU cycles required for processing one sample data, c_i	[30,100] cycles/bit
Clients' CPU frequency, f_i	[1,10] GHz
Allocated bandwidth for clients, B_j	1 MHz
Maximum number of clients that can access to ESs, M_j	[8,12]
Clients' transmit power, q_i	[200,800] mW
Noise power spectral density, N_0	-174 dBm/Hz
Transmission delay for ESs uploading the edge model to the CS, T_j^{com}	[160,200] ms

To emulate non-IID data, we allocate datapoints from only 1 to 3 labels (out of 10) to each client for MNIST, Fashion-MNIST and CIFAR-10, while 10 to 30 labels (out of 100) for CIFAR-100. Each client holds a data quantity within the range of [255, 1013] for MNIST and Fashion-MNIST, and [206, 863] for CIFAR-10 and CIFAR-100. The participation of client n_i follows a Bernoulli distribution $\mathcal{B}(1, p_i)$, where p_i is randomly chosen from the interval [0.5,1). The minimum tolerable data size $D_{\min}$ in OrigProbSolver and StagewiseHFL is set as 2500, while $KLD_{\max}$ is set by 0.2. Other parameters are summarized in Table 1.

5.1.1 Evaluation of test accuracy



(a) MNIST (b) Fashion-MNIST

(c) CIFAR-10 (d) CIFAR-100

Fig. 2: Performance comparison in terms of test accuracy.

Fig. 2 shows the performance in terms of test accuracy on MNIST, Fashion-MNIST, CIFAR-10 and CIFAR-100, where $\mathcal{T}$ and $\mathcal{L}$ are set to 5 and 3, respectively [10], [49]. As shown in Fig. 2(a), our StagewiseHFL outperforms ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc and FedCS by achieving higher test accuracy by 3.12%, 7.11%, 3.03%, and 5.03%, respectively, within 30 global iterations on MNIST. Comparing to OrigProbSolver, our StagewiseHFL obtains a slightly lower test accuracy (around 0.34% lower). This is because our proposed Plan A relies on historical information, where the gap between the estimation of historical records and the actual network condition (i.e., actual attendance of clients) can lead to performance degradation. We will later show that this slight performance edge of OrigProbSolver comes with the drawback of significant running time, rendering it inapplicable for large scale systems (later depicted in Fig. 5). Similarly, our StagewiseHFL exhibits 1.31% lower test accuracy than KLDMinimization method. This is raised from that KLDMinimization solely focuses achieving data homogeneity (i.e., IID data) across the ESs, while ignoring the cost (i.e., latency and energy consumption) induced by such C2E assignment. As a result, as later shown, KLDMinimization suffers from a prohibitively high cost F and running time. Notably, KLDMinimization, which directly reduces the average KLD across ESs, achieves higher accuracy and faster convergence than competing baselines, while OrigProbSolver and StagewiseHFL, both incorporating KLD constraints into their optimization, can consistently outperform other methods that lack explicit heterogeneity control. Together, the superior performance of KLDMinimization, OrigProbSolver, and StagewiseHFL underscores a key insight: as the distribution of data across ESs becomes more uniform, the convergence of the model can be greatly improved as reflected in the reduced number of global rounds to reach the target accuracy.

We can observe a similar trend, where our proposed StagewiseHFL demonstrates outstanding performance in terms of model accuracy and convergence speed, as shown

in Figs. 2(b)-2(d). For example, considering CIFAR-10, our StagewiseHFL outperforms ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc and FedCS on CIFAR-10 by 6.12%, 5.52%, 6.25% and 7.47% in test accuracy, while being only 0.71% lower than OrigProbSolver and 2.33% lower than KLDMinimization after 100 global iterations, as shown in Fig. 2(c). We next show that our method, while showing slightly lower prediction performance than the best benchmark, exhibits a notable performance gap when other metrics are considered.

5.1.2 Evaluation of delay, energy cost, and running time

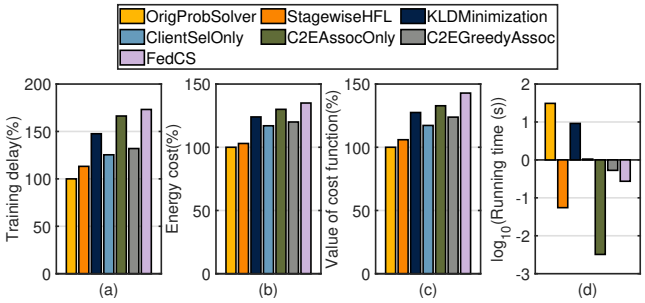


Fig. 3: Performance on training delay, energy consumption, overall cost, and running time upon having real-world dataset.

Fig. 3 illustrates the training delay, energy consumption, the overall cost (i.e., $\sum_g F^{(g)}$), and the time consumed for inference decision, to reach over 90% test accuracy on MNIST. We set $\mathcal{T} = 5$, $\mathcal{L} = 3$, and regard OrigProbSolver as the reference line, to better show the gap among different methods in Figs. 3(a)-3(d). In Fig. 3(a), the weighting factors λ_t (for delay) and λ_e (for energy) are assigned values of 1 and 0, respectively. This indicates that we put full emphasis on minimizing the training delay. Comparing to OrigProbSolver which optimizes problem $\boldsymbol{\mathcal{P}}_0^{(g)}$ at the beginning of each global iteration, different benchmark methods can introduce various levels of increase on the training delay due to the complications embedded in their designs: 13.2%, 47.6%, 25.4%, 66.3%, 32.0% and 73.2% increase for StagewiseHFL, KLDMinimization, ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, FedCS, respectively. Our StagewiseHFL stands out as the superior choice among the evaluated methods. In Fig. 3(b), we showcase the performance metrics related to energy consumption, by setting $\lambda_t = 0$ and $\lambda_e = 1$. Our StagewiseHFL, in comparison to OrigProbSolver, incurs only a marginal performance loss of 3.0%. This slight decrease demonstrates StagewiseHFL's superior efficiency, highlighting its capability in balancing energy conservation and performance[13]. Without loss of generality, when it comes to randomly assigning weighting factors of training delay and energy consumption such that $\lambda_t + \lambda_e = 1$ (λ_t, $\lambda_e \in (0, 1)$) in Fig. 3(c), our StagewiseHFL only incurs a 5.9% loss compared to OrigProbSolver, while achieving performance improvements of up to 21.6%, 11.3%, 26.9%, and 17.9%, and 36.9% in comparison with KLDMinimization, ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, and FedCS, respectively. We next show that the above-discussed close performance of our method to the best baseline comes with a notable gap in terms of running time.

To illustrate the real-time decision-making overhead of different methods, i.e., the time spent on selecting clients and associating them with ESs, Fig. 3(d) employs a logarithmic scale to represent the average decision-making time per global training round. Our StagewiseHFL shows the closest performance to OrigProbSolver in terms of cost savings (see Figs. 3(a)-3(c)), while offering an average inference time of less than 100ms, which is order of magnitude lower than that of OrigProbSolver. This verifies the effectiveness of our approach in terms of achieving a reasonable value of resource consumption (i.e., energy and delay) under a notably low decision-making overhead.

5.2 Numerical Data-driven Experiments

To further evaluate our approach's decision-making overhead and model convergence, we conduct a complementary set of experiments. These experiments aim to investigate the impact of various factors, such as the number of clients (i.e., $|\mathcal{N}|$) and ESs (i.e., $|\mathcal{S}|$), the minimum acceptable data size for model training (i.e., $D_{\min}$), and the likelihood of client participation (i.e., p_i), on the performance.

TABLE 2: Parameter setting used in Sec. 5.2

* $\mathcal{I}_1$: [0.4, 1), $\mathcal{I}_2$: [0.45, 1), $\mathcal{I}_3$: [0.5, 1), $\mathcal{I}_4$: [0.6, 1), $\mathcal{I}_5$: [0.7, 1)

	$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{S}\rvert$	$D_{\min}$	p_i
Set#1	75, 80, . . ., 120	4	2500	$\mathcal{I}_3$
Set#2	100	3, 4, . . .,7	2500	$\mathcal{I}_3$
Set#3	100	4	2000, 2500, . . ., 4000	$\mathcal{I}_3$
Set#4	100	4	2500	$\mathcal{I}_1$, $\mathcal{I}_2$, . . ., $\mathcal{I}_5$

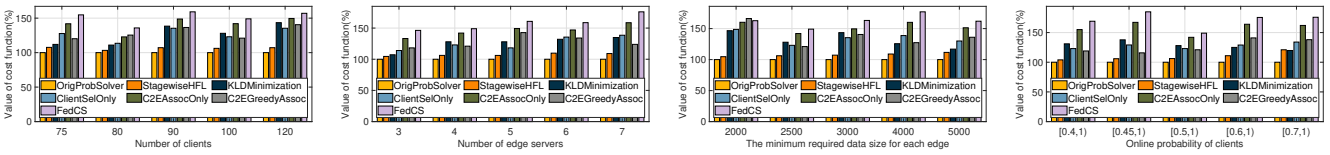
5.2.1 Evaluation of the overall cost

We first evaluate the induced cost[14] to achieve a 90% accuracy on MNIST dataset under various parameters: the number of clients $|\mathcal{N}|$, the number of ESs $|\mathcal{S}|$, the tolerable minimum data size $D_{\min}$ for each ES, and the participation probability p_i for each client $\boldsymbol{n}_i$. For illustrations, we conduct the next experiments under 4 sets of parameters detailed in Table 2.

We present a performance comparison on the value of cost function F in Fig. 4(a), using OrigProbSolver as the reference line. Taking into account varying numbers of clients (75 to 120), our StagewiseHFL experiences a performance gap of 7.6%, 3.3%, 7.1%, 6.2%, and 7.2% as compared to OrigProbSolver. This is because Plan A in StagewiseHFL relies on historical data, introducing risks of inaccurate estimates in client selection and C2E association (the drawbacks associated with running time of OrigProbSolver will be analyzed in Fig. 5). Our StagewiseHFL exhibits cost-effectiveness when $|\mathcal{N}|$ ranges from 75 to 120 compared to KLDMinimization, ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc and FedCS. This is attributed to our optimized decision-making process for both client selection and C2E association. For instance, upon having $|\mathcal{N}| = 80$ in Fig. 4(a), StagewiseHFL achieves

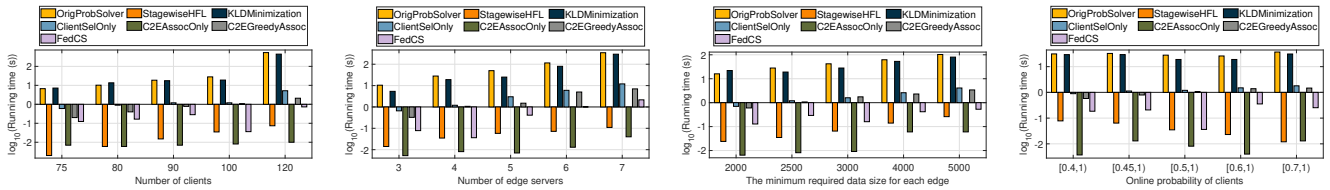
13. Reported energy values are normalized, as commonly done in FL [5], [56], to avoid hardware dependence and ensure reproducibility. Since all methods run under identical conditions, these values still offer a fair and meaningful comparison.

14. The reported cost metrics are virtualization-based, derived from analytical models and empirical measurements [23], [28], and abstracted from specific hardware, ensuring general applicability of our results.



(a) Cost vs. $|\mathcal{N}|$ (Set#1) (b) Cost vs. $|\mathcal{S}|$ (Set#2) (c) Cost vs. $D_{\min}$ (Set#3) (d) Cost vs. p_i (Set#4)

Fig. 4: Performance on the overall system cost within numerical dataset.



(a) Running time vs. $|\mathcal{N}|$ (Set#1) (b) Running time vs. $|\mathcal{S}|$ (Set#2) (c) Running time vs. $D_{\min}$ (Set#3) (d) Running time vs. p_i (Set#4)

Fig. 5: Performance on running time within numerical dataset.

a reduction in system cost by 7.7%, 10.26%, 19.5%, 22.2% and 32.6% as compared to KLDMinimization, ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, and FedCS, respectively. A similar trend, where our StagewiseHFL demonstrates cost-saving performance that ranks second with a small margin only to OrigProbSolver, is seen in Figs. 4(b)-4(d).

5.2.2 Evaluation of running time

Fig. 5 evaluates the performance on time efficiency reflected by running time, upon having different values of $|\mathcal{N}|$, $|\mathcal{S}|$, $D_{\min}$, and p_i. Our StagewiseHFL significantly improves time efficiency compared to the benchmark methods under consideration, with the exception of C2EAssocOnly in some certain scenarios. This is because C2EAssocOnly does not prioritize optimizing client selection, thereby streamlining the decision-making process (however, it suffers from a large cost shown in Fig. 4).

Inspecting Figs. 5(a)-5(c), varying $|\mathcal{N}|$, $|\mathcal{S}|$ and $D_{\min}$ changes the problem scale, which in turn affects the complexity of algorithms used for client selection and C2E association. For instance, in Fig. 5(a), the delay on decision-making of ClientSelOnly increases from 600ms to 9.8s with $|\mathcal{N}|$ rises from 70 to 120. Note that although OrigprobSolver outperforms our StagewiseHFL in terms of cost savings (see Fig. 4), its inference delay becomes prohibitively high as the problem size increases (with the rise of $|\mathcal{N}|$, $|\mathcal{S}|$ and $D_{\min}$). For example, its decision-making delay varies from 10.44s to 346s as $|\mathcal{S}|$ increases from 3 to 7, shown in Fig. 5(b). While the time cost does escalate with an increase in problem size, our approach continues to be viable across various environments compared to all benchmarks. This superior performance is primarily due to our strategically designed Plan A, which pre-selects a specific number of long-term clients. This significantly mitigates the problem size encountered in Plan B, thereby accelerating the real-time decision-making process for client selection and C2E associations. A similar trend is found in Fig. 5(c), where rising tolerable minimum data size $D_{\min}$ results in a growth of inference time, e.g., the inference time increases from 22s to 80s for KLDMinimization while from 24ms to 260ms for our StagewiseHFL when $D_{\min}$ varies from 2000 to 5000. Also, as the client participation probability rises, our StagewiseHFL increasingly relies less on Plan B during actual training (since more selected long-term clients will participate in training), resulting in reduced complexity and further enhancing inference speeds. As illustrated in Fig. 5(d), having the client online probability range of [0.7, 1), StagewiseHFL achieves an average of only 12ms per global aggregation round, which is even faster than C2EAssocOnly.

In summary, Fig. 5 reveals that the running time of StagewiseHFL consistently remains below 500ms across various problem scales, which offers a commendable reference for future low-overhead HFL design in dynamic environments.

5.3 Analysis of the Impact of Key Parameters

Since improper parameter settings can adversely model training performance, we further evaluate various combinations of $KLD_{\max}$ and $D_{\min}$, as well as $\mathcal{T}$ and $\mathcal{L}$ in this section, to show how they affect the system while identifying the best combinations.

5.3.1 Analysis of the impact of risk-related parameters

Our StagewiseHFL introduces distinctive considerations on risk evaluation in Plan A, representing one of our contributions, which sets this work apart from existing studies. As two key parameters that affect the risks are KLD and data size, we first illustrate how $KLD_{\max}$ and $D_{\min}$ impact the overall system cost and the required global aggregation rounds to reach a target test accuracy in Table 3. To achieve 90% accuracy on MNIST within $\mathcal{T} = 5$ and $\mathcal{L} = 3$, Table 3 shows that lowering the KLD for edge data can reduce the number of global aggregations. Considering $D_{\min} = 2500$, setting the value of $KLD_{\max}$ from 0.5 to 0.2 reduces global iterations from 29 to 19. However, under non-IID data, setting the value of $KLD_{\max}$ too low often leads to the selection of more clients for participation, increasing system costs. For example, when $KLD_{\max} = 0.1$, the system costs are higher than those with $KLD_{\max} = 0.2$ for all $D_{\min}$ settings. Extended evaluation in Appx. K further examine the relationship between $KLD_{\max}$ and model accuracy on

CIFAR-100 and Tiny-ImageNet, as shown in Tables 8 and 9. These results confirm that the positive effect of KLD control on accuracy also extends to more complex datasets. For the tolerable data size, although raising $D_{\min}$ can help improve model accuracy, when $D_{\min}$ is set to a higher value, incorporating an excessive amount of data into training may decelerate convergence, thereby necessitating an increased number of global aggregations. For instance, upon having $KLD_{\max} = 0.4$, increasing $D_{\min}$ from 2000 to 4000 increases the required global aggregation rounds from 27 to 36. In sum, the careful selection of $D_{\min}$ and $KLD_{\max}$ is crucial for optimizing the overall performance.

TABLE 3: Impact of $D_{\min}$ vs. $KLD_{\max}$ for reaching the accuracy of 90%

Cost/ Aggregations		$KLD_{\max}$ 0.1	0.2	0.3	0.4	0.5
$D_{\min}$	2000	18.09/23	11.02/23	9.65/22	11.29/27	11.89/28
	2500	17.31/22	**9.11/19**	10.91/26	13.15/29	13.18/29
	3000	16.53/21	10.09/20	14.18/30	17.48/35	21.75/47
	4000	17.37/22	14.08/24	11.80/22	18.79/36	20.00/36
	5000	16.58/21	14.75/22	14.36/23	14.73/23	15.52/25

5.3.2 Analysis of the impact of training configuration

We next analyze the impact of $\mathcal{T}$ and $\mathcal{L}$ on the overall system cost. Although $\mathcal{T}$ and $\mathcal{L}$ could ideally be optimized alongside other parameters to minimize overall system costs, our experiments are conducted under constraints including access limitations M_j of ESs, and fixed data volumes. As a result, the number of clients selected in each global iteration remains relatively consistent, enabling us to identify a practical and broadly applicable combination of $\mathcal{T}$ and $\mathcal{L}$. To keep the problem tractable and to ensure fair comparisons across different experimental settings, we follow existing works [5], [8], [10], [49], [61], which similarly adopt fixed settings of $\mathcal{T}$ and $\mathcal{L}$[15]. Nevertheless, $\mathcal{T}$ and $\mathcal{L}$ are not arbitrarily chosen. Through extensive tests over a wide range of candidate values, we verified that the adopted configuration consistently provides competitive performance, making it a reasonable and representative choice across our experiments. As shown in Table 4, when $\mathcal{T}$ is relatively small (e.g., $\mathcal{T} = 5$ and $\mathcal{L} = 10$), increasing the value of $\mathcal{L}$ can effectively decrease the number of required global iterations, thereby enhancing the model's convergence speed. At a specific intermediate value (e.g., $\mathcal{L} = 3$), we achieve a joint minimum for both the system cost and the required number of global aggregations. However, with larger values of $\mathcal{T}$, the number of global aggregations needed for achieving the target accuracy may not necessarily reduce; in fact, it could result in inferior performance outcomes. Due to the imbalanced data distribution across clients and a lower edge aggregation frequency (that is, executing more local iterations within a single round of edge iteration), the local models may become biased, preventing the global model from convergence. Particularly if $\mathcal{L} = 4$ and $\mathcal{L} = 5$ when $\mathcal{T} = 50$, the global model never reaches the accuracy of 90% and thus the training cost is denoted by infinity.

15. Note that the primary decision variable in our formulation is the client-to-edge (C2E) association matrix $\boldsymbol{A}$. Fixing $\mathcal{T}$ and $\mathcal{L}$ allows us to better highlight the core contribution of this work—namely, the establishment of a novel two-stage optimization paradigm for HFL, which has rarely been explored in the literature. While many factors in HFL can indeed be optimized, our focus here is on client association and selection, as they are most aligned with the proposed paradigm. The optimization of $\mathcal{T}$ and $\mathcal{L}$ will be considered as an interesting direction for future research.

TABLE 4: Impact of $\mathcal{T}$ vs. $\mathcal{L}$ for reaching the accuracy of 90%

Cost/ Aggregations		$\mathcal{L}$ 1	2	3	4	5
$\mathcal{T}$	5	10.32/52	11.06/28	**9.11/19**	12.58/16	15.69/16
	10	12.03/41	13.45/23	14.21/16	18.90/16	16.23/11
	20	13.86/28	15.83/16	16.30/11	21.70/11	27.11/11
	30	31.16/23	15.25/11	22.83/11	30.41/11	37.99/11
	50	17.46/16	23.95/11	35.88/11	∞	∞

Overall, these empirical results indicate that although $\mathcal{T}$ and $\mathcal{L}$ are treated as fixed hyper-parameters, their settings are carefully tuned to ensure that the reported results reflect near-optimal system performance under different scenarios. In fact, the comparisons presented in Table 4 implicitly serve as an ablation study, as varying $\mathcal{T}$ and $\mathcal{L}$ allows us to examine their influence on convergence speed, training cost, and final accuracy.

5.4 Threats to Validity

Since this work tries the first attempt to having two complementary plans for client selection, this section is dedicated to studying the validity of StagewiseHFL through inspecting factors that can compromise its effectiveness. Our goal is to highlight the benefits of our method from a rational perspective.

5.4.1 Threats to internal validity

The first threat to the validity of our StagewiseHFL is that the experimental environment may have favored it in the previous results. To fairly compare the performance of different methods, we change the settings of different parameters, e.g., $D_{\min}$, and p_i, while we also perform 100 experiments for diverse parameter settings (Table 1). Note that when clients have a high online probability, e.g., an extreme case in which all the clients will join in each global training rounds ($p_i = 1$, $\forall \boldsymbol{n}_i \in \mathcal{N}$), our design of pre-selection and pre-association of clients in Plan A may be no longer needed. The reason is that our plan A may result in a high overlap in the selected long-term clients, during each global iteration, potentially decreasing the diversity of training data while slowing down the convergence speed of the global model. As shown in Fig. 6(a), when the interval of p_i varies from [0.75, 1) to [0.95, 1), our StagewiseHFL incurs performance losses of 26.5%, 39.4%, 33.6%, 23.2%, 28.0% compared to OrigprobSolver in terms of cost-savings. It may also perform worse in comparison with C2EGreedyAssoc, and even worse than the ClientSelOnly and KLDMinimization in some cases. Such observations indicate that our StagewiseHFL should be used in scenarios with intermittent client participation.

5.4.2 Threats to external validity

We next acknowledge that the scalability of the long-term decision-making component in our method may be affected by the network size. As discussed in Sec. 4.2.1, the computational complexity of Plan A grows with the number of clients $|\mathcal{N}|$, potentially leading to long decision-making

times when $|\mathcal{N}|$ is large. As illustrated in Fig. 6(b), when $|\mathcal{N}|$ increases from 75 to 120, the runtime of Plan A rises from 8.3s to 855.8s. This observation highlights an inherent trade-off in StagewiseHFL, where a certain amount of long-term optimization time must be sacrificed to enable smoother subsequent training.

Nevertheless, this limitation does not affect practicality: Plan A runs once before training and later in parallel, so its overhead does not hinder real-time decisions. Results in Sec. 5 show that despite this cost, our two-stage framework significantly improves scalability, efficiency, and deployment feasibility, highlighting the necessity of the hybrid strategy.

6 Conclusion and Future Work

This paper we studied client selection and C2E association for dynamic HFL network, for which we investigate a stagewise decision-making methodology with two stages, namely Plan A and Plan B. In particular, in Plan A, we introduced the concept of "continuity" of clients, strategically determining long-term clients and associating them with appropriate ESs. In Plan B, we propose a method for rapidly identifying backup clients in case those recruited by Plan A were unavailable in practical model training rounds. Comprehensive simulations on different datasets and models demonstrated the commendable performance of our method compared to other benchmark methods.

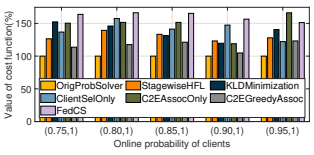
This work is the first to systematically explore stagewise decision-making in FL and HFL, showing its potential to accelerate training while reducing overhead. The promising results and clear rationale are expected to inspire further research on its theoretical optimality. Future research includes modeling uncertainties in client computation and communication, optimizing when to trigger Plan A in dynamic environments, and studying the impact of hardware accelerators (e.g., GPUs/TPUs) in heterogeneous settings. Another promising direction is leveraging D2D communications to improve local aggregation under intermittent client availability.

References

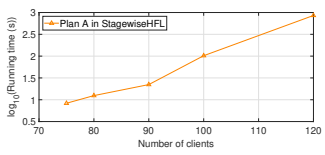
[1] Z. Zhao et al., "DeepThings: Distributed adaptive deep learning inference on resource-constrained IoT edge clusters", *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst*, vol. 37, no. 11, pp. 2348-2359, 2018.

[2] B. McMahan et al., "Communication-efficient learning of deep networks from decentralized data," *Proc. Int. Conf. Artif. Intell. Stat.*, pp. 1273–1282, 2017.

[3] Y. Zhou et al., "The role of communication time in the convergence of federated edge learning," *IEEE Trans. Veh. Technol.*, vol. 71, no. 3, pp. 3241–3254, 2022.



(a) Cost vs. p_i



(b) Running time vs. $|\mathcal{N}|$ for Plan A in StagewiseHFL

Fig. 6: Invalidity of StagewiseHFL.

[4] D. C. Nguyen et al., "Federated learning for internet of things: A comprehensive survey," *IEEE Commun. Surveys Tut.*, vol. 23, no. 3, pp. 1622-1658, 2021.

[5] S. Luo et al., "HFEL: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6535-6548, 2020.

[6] Z. Jiang et al., "Computation and communication efficient federated learning with adaptive model pruning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 3, pp. 2003-2021, 2024.

[7] Y. Tian et al., "Hierarchical federated learning with adaptive clustering on Non-IID data," *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, pp. 5063–5068, 2022

[8] L. Liu et al., "Client-edge-cloud hierarchical federated learning," *IEEE Int. Conf. Commun. (ICC)*, pp. 1-6, 2020.

[9] M. S. H. Abad et al., "Hierarchical federated learning across heterogeneous cellular networks," *IEEE Int. Conf. Acoust. Speech Signal Process Proc. (ICASSP)*, pp. 8866-8870, 2020.

[10] Y. Deng et al., "A communication-efficient hierarchical federated learning framework via shaping data distribution at edge," *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2600-2615, 2024.

[11] W. Mao et al., "Joint client selection and bandwidth allocation of wireless federated learning by deep reinforcement learning," *IEEE Trans. Serv. Comput.*, vol. 17, no. 1, pp. 336-348, 2024.

[12] S. Fu et al., "Joint optimization of device selection and resource allocation for multiple federations in federated edge learning," *IEEE Trans. Serv. Comput.*, vol. 17, no. 1, pp. 251-262, 2024.

[13] K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, pp. 1175–1191, 2017.

[14] H. Saadat et al., "RL-assisted energy-aware user-edge association for IoT-based hierarchical federated learning," *Int. Wireless. Commun. Mob. Comput. (IWCMC)*, pp. 548-553, 2022.

[15] J. Konečný et al., "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2017.

[16] S. Caldas et al., "Expanding the reach of federated learning by reducing client resource requirements," *arXiv:812.07210*, 2019.

[17] D. Alistarh et al., "QSGD: Communication-efficient SGD via gradient quantization and encoding," *Adv. neural inf. proces. syst.*, pp. 1707–1718, 2017.

[18] X. Zhang et al., "Vehicle selection and resource allocation for federated learning-assisted vehicular network", *IEEE Trans. Serv. Comput.*, vol. 23, no. 5, pp. 3817-3829, 2024.

[19] J. Zheng, "Federated learning for online resource allocation in mobile edge computing: A deep reinforcement learning approach," *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, pp. 1-6, 2023.

[20] Z. Yang et al., "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, 2021.

[21] W. Wu et al., "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1539-1551, 2021.

[22] K. Yang et al., "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022-2035, 2020.

[23] T. D. Burd et al., "Processor design for portable systems", *J. Signal Process. Syst.*, vol. 13, no. 2–3, pp. 203–221, 1996.

[24] J. Feng et al., "Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 11, pp. 2687-2700, 2022.

[25] F. P. -C. Lin et al., "Delay-aware hierarchical federated learning," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 2, pp. 674-688, 2024.

[26] L. Su et al., "Low-latency hierarchical federated learning in wireless edge networks," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6943-6960, 2024.

[27] Z. Qu et al., "Context-aware online client selection for hierarchical federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4353-4367, 2022.

[28] T.T.Vu et al., "Cell-free massive MIMO for wireless federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6377-6392, 2020.

[29] X. Xia et al., "Formulating cost-effective data distribution strategies online for edge cache systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4270–4281, 2022.

[30] Y. Sun et al., "Uplink interference mitigation for OFDMA femtocell networks," *IEEE Trans. Wireless Commun.*, vol. 11, no. 2, pp. 614-625, 2012.

[31] X. Cao et al., "Transmission power control for over-the-air federated averaging at network edge," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1571–1586, 2022.

[32] D.-J. Han, "FedMes: Speeding up federated learning with multiple edge servers," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3870–3885, 2021.

[33] Y. Jiao "Toward an automated auction framework for wireless federated learning services market," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 3034-3048, 2021.

[34] D. Chen et al., "Matching-theory-based low-latency scheme for multitask federated learning in MEC networks," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11415-11426, 2021.

[35] L. Wang et al., "Towards class imbalance in federated learning," *arXiv preprint arXiv:2008.06217*, 2020.

[36] M. Pal et al., "Facility location with nonuniform hard capacities," *Annu Symp Found Comput Sci Proc*, pp. 329–338, 2001.

[37] Z. Cheng et al., "Joint client selection and task assignment for multi-task federated learning in MEC networks," *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, pp. 1–6, 2021.

[38] M. Ester et al., "A density-based algorithm for discovering clusters in large spatial databases with noise," *Int. Conf. Knowl. Discov. Data Min.*, pp. 226–231, 1996.

[39] Y. Lecun et al., "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[40] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Tech. Rep.*, 2009.

[41] H. Xiao et al., "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," arXiv:1708.07747, 2017.

[42] K. He et al., "Deep residual learning for image recognition," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pp. 770–778, 2016.

[43] L. U. Khan et al., "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Commun. Mag.*, vol. 58,no. 10, pp. 88–93, 2020.

[44] W. Feller, "Markov chains," *An introduction to probability theory and its applications*, vol. 1, 3rd ed., Wiley, 1968.

[45] M. Perlmutter et al.,"Scattering statistics of generalized spatial poisson point processes," *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, pp. 5528-5532, 2022.

[46] R. He et al., "Non-stationary mobile-to-mobile channel modeling using the Gauss-Markov mobility model," *Int. Conf. Wirel. Commun. Signal Process. (WCSP)*, pp. 1-6 2017.

[47] H. Wang et al., "Optimizing federated learning on Non-IID data with reinforcement learning," *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, pp. 1698-1707, 2020.

[48] W. Sunet al., "Accelerating Convergence of Federated Learning in MEC With Dynamic Community," *IEEE Trans. Mobile Comput.*, vol. 23, no. 2, pp. 1769-1784, 2024.

[49] Y. Deng et al., "SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning," *IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, pp. 24-34, 2021.

[50] Z. Xu et al., "Energy or accuracy? Near-optimal user selection and aggregator placement for federated learning in MEC," *IEEE Trans. Mobile Comput.*, vol. 23, no. 3, pp. 2470-2485, 2024.

[51] B. Luo et al., "Cost-effective federated learning design," *IEEE Conf. Comput. Commun.*, pp. 1-10, 2021.

[52] H. T. Nguyen et al., "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 201-218, 2021.

[53] Y. Guo et al., "Hybrid local SGD for federated learning with heterogeneous communications," *Int. Conf. Learn. Represent.(ICLR)*, pp. 1-42, 2022.

[54] T. Castiglia et al., "Multi-level local SGD: Distributed SGD for heterogeneous hierarchical networks," *Int. Conf. Learn. Represent.(ICLR)*, pp. 1-36, 2021.

[55] Z. Zhang et al., "Scalable and low-latency federated learning with cooperative mobile edge networking," *IEEE Trans. Mobile Comput.*, vol. 23, no. 1, pp. 812–822, 2024.

[56] T. Nishio et al., "Client selection for federated learning with heterogeneous esources in mobile edge," *IEEE Int. Conf. Commun. (ICC)*, pp. 1–7, 2019.

[57] A. Byerly et al., "No routing needed between capsules," *Neurocomputing*, pp. 545-553, 2021.

[58] P. Gavrikov et al., "CNN filter DB: An empirical investigation of trained convolutional filters," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR)*, pp. 19044-19054, 2022.

[59] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[60] P. Foret et al., "Sharpness-aware minimization for efficiently improving generalization," *Int. Conf. Learn. Represent.(ICLR)*, pp. 1-20, 2021.

[61] X. Chen et al., "Toward dynamic resource allocation and client scheduling in hierarchical federated learning: A two-phase deep reinforcement learning approach," *IEEE Trans. Commun.*, vol. 72, no. 12, pp. 7798–7813, 2024.

[62] B. Gong et al. "Towards Hierarchical Clustered Federated Learning With Model Stability on Mobile Devices," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7148-7164, 2024.

[63] S. Yue et al., "AugFL: Augmenting Federated Learning with Pretrained Models," *IEEE/ACM Trans. Netw.*, vol. 33, no. 4, pp. 1870-1885, 2025.

[64] Y. Cheng et al., "SnapCFL: A Pre-Clustering-Based Clustered Federated Learning Framework for Data and System Heterogeneities," *IEEE Trans. Mobile Comput.*, vol. 24, no. 6, pp. 5214-5228, 2025.

[65] G. S. Xia et al., "AID: A Benchmark Data Set for Performance Evaluation of Aerial Scene Classification", *IEEE Trans. Geosci. Remote Sens.*,vol. 55, no. 7, pp. 3965-3981, 2017.

[66] Y. Yang et al., "Efficient UAV Swarm-Based Multi-Task FederatedLearning with Dynamic Task Knowledge Sharing", *arXiv preprint arXiv:2503.09144*, 2025.

[67] K. Simonyan et al., "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Int. Conf. Learn. Represent.(ICLR)*, pp. 1-14, 2015.

APPENDIX A
LIST OF NOTATIONS

TABLE 5: List of Notations

Notation	Definition	Notation	Definition
$\mathcal{N}/\boldsymbol{n}_i$	Set of clients/the i^{th} client	$\mathcal{S}/s_j$	Set of ESs/the j^{th} ES
$\mathcal{Z}/z_h$	Label set/the h^{th} label	$\mathcal{D}_i$	Dataset of client $\boldsymbol{n}_i$
$d_i^{(g)}$	Data size of client $\boldsymbol{n}_i$	$D_j^{(g)}$	Data size of all clients associated with ES s_j
$D^{(g)}$	Data size of all clients that take part in the training process	$\boldsymbol{x}_k/y_k$	Input vector/labeled output of the k^{th} data
v_i	CPU frequency of client $\boldsymbol{n}_i$	q_i	Transmission power of client $\boldsymbol{n}_i$
$\xi_i^{(g)}$	Binary indicator describing the participation of a client $\boldsymbol{n}_i$ in g^{th} global iteration	$\boldsymbol{A}^{(g)}$	Association matrix indicating whether client $\boldsymbol{n}_i$ is associated with ES s_j or not
$\boldsymbol{\omega}_i^{(g;\ell,t)}$	Local model parameter of client $\boldsymbol{n}_i$ in t^{th} local update during ℓ^{th} edge aggregation round	$\boldsymbol{\omega}_j^{(g;\ell)}$	Edge model parameter of ES s_j in ℓ^{th} edge aggregation round
$\boldsymbol{\omega}^{(g)}$	Global model parameter	η	Learning rate
$\mathcal{T}/\mathcal{L}$	Edge/global aggregation frequency	c_i	Number of CPU cycles required for processing one sample data on $\boldsymbol{n}_i$
α_i	Effective capacitance coefficient of $\boldsymbol{n}_i$'s computing chipset	B_j	Allocated bandwidth for each client associated with s_j
M_j	Maximum number of clients that can access to s_j	$h_{i,j}^{(g)}$	Channel gain of between $\boldsymbol{n}_i$ and s_j
N_0	Noise power spectral density	$R_{i,j}^{(g)}$	Data transmission rate of the uplink from client $\boldsymbol{n}_i$ to ES s_j
Ω	Data size of the model parameters uploaded by clients	p_i	Participation probability of $\boldsymbol{n}_i$
$t_i^{\text{cmp}}/e_i^{\text{cmp}}$	Computation latency/energy consumption for a client $\boldsymbol{n}_i$ to perform one local training iteration	$t_{i,j}^{(g),\text{com}}/e_{i,j}^{(g),\text{com}}$	Transmission delay/energy consumption between $\boldsymbol{n}_i$ and s_j during one edge iteration
$T_{i,j}^{(g)}/E_{i,j}^{(g)}$	Delay/energy consumption caused by the interaction between $\boldsymbol{n}_i$ and s_j during one edge iteration	$T_j^{(g),\text{com}}/E_j^{(g),\text{com}}$	Transmission delay/energy consumption for ES s_j uploading the edge model to the CS
$\mathbb{T}^{(g)}/\mathbb{E}^{(g)}$	Overall training delay/energy consumption during g^{th} global iteration	$\mathbb{C}^{(g)}$	System "continuity" describing the geometric mean of online probability of each selected client
$\boldsymbol{y}_i$	Vector representing the data distribution of client $\boldsymbol{n}_i$	$\boldsymbol{P}_j^{(g)}/\boldsymbol{Q}$	Data distribution of ES s_j/reference data distribution
$KLD_{\max}$	Upper bound of KLD the covered data associated with each ES	$D_{\min}$	lower bound of size of the covered data associated with each ES

APPENDIX B
PSEUDOCODE OF ALG. 1

Algorithm 1: Stagewise Hierarchical Federated Learning (StagewiseHFL)

Input : the set of ES $\mathcal{S}$; the set of clients $\mathcal{N}$

Output: the long-term C2E association decision $\widehat{\boldsymbol{A}}^{(g)}$; the real-time C2E association decision $\widetilde{\boldsymbol{A}}^{(g)}$

%**Long-term decision-making stage (Plan A)**

Estimate p_i ($\forall \boldsymbol{n}_i \in \mathcal{N}$) before practical model training;

Solve $\boldsymbol{\mathcal{P}}_1$ through Algs. 2-3 and obtain the long-term C2E association decision $\widehat{\boldsymbol{A}}^{(0)}$;

%**Real-time decision-making stage (Plan B)**

for $g = 1$ to $g_{\max}$ **do**

  $\widehat{\boldsymbol{A}}^{(g)} = \widehat{\boldsymbol{A}}^{(0)}$;

  Solve $\boldsymbol{\mathcal{P}}_2$ in the g^{th} global aggregation through Algs. 4 and obtain the real-time C2E association decision $\widetilde{\boldsymbol{A}}^{(g)}$;

return $\widehat{\boldsymbol{A}}^{(g)}$ and $\widetilde{\boldsymbol{A}}^{(g)}$;

APPENDIX C
DERIVATION OF (11f)

By substituting the definition of $\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)$ from (7) and (8) into (11a), we can obtain

$$\Pr\left(\sum_{h=1}^{|\mathcal{Z}|}\frac{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}d_i}\log\frac{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{\boldsymbol{Q}(h)\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}d_i} > KLD_{\max}-\Delta_k\right)\leq\delta,\forall s_j\in\mathcal{S}. \tag{13}$$

Note that the fractions in the above (13) have no meaning when $\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}=0$, i.e., all clients selected and associated with ES s_j in plan A are offline, then, we consider $\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)$ to be $+\infty$ in such a case. Given the law of total probability, we can decompose the left side of (11a) as:

$$\begin{aligned}
&\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\right)=\Pr\left(\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}=0\right)\times\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}=0\right)\\
&+\Pr\left(\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq 0\right)\times\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq 0\right)\\
&=\prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)+\left(1-\prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\right)\times\Pr\left(\sum_{h=1}^{|\mathcal{Z}|}\frac{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}d_i}\log\frac{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{\boldsymbol{Q}(h)\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}d_i}\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq 0\right).
\end{aligned}\tag{14}$$

For the case where $\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq 0$, let $\boldsymbol{R}_j^{(g)}(h)=\frac{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\boldsymbol{y}_i(h)}{\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}d_i}$, and define $N_j^{(g)}=\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}$ as the number of clients associated with ES s_j. Then, we have the following scenarios.

When $N_j^{(g)}=1$, we have $\boldsymbol{R}_j^{(g)}(h)=\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}}{d_1\xi_1^{(g)}}=\frac{\boldsymbol{y}_1(h)}{d_1}$.

When $N_j^{(g)}=2$, we have $\boldsymbol{R}_j^{(g)}(h)=\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}+\boldsymbol{y}_2(h)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}}=\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}+\boldsymbol{y}_1(h)\frac{d_2}{d_1}\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}}+\frac{\left(\boldsymbol{y}_2(h)-\boldsymbol{y}_1(h)\frac{d_2}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}}=\frac{\boldsymbol{y}_1(h)}{d_1}+\frac{d_2\left(\frac{\boldsymbol{y}_2(h)}{d_2}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}}$.

When $N_j^{(g)}=3$, we have $\boldsymbol{R}_j^{(g)}(h)=\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}+\boldsymbol{y}_2(h)\xi_2^{(g)}+\boldsymbol{y}_3(h)\xi_3^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+d_3\xi_3^{(g)}}=\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}+\boldsymbol{y}_1(h)\frac{d_2}{d_1}\xi_2^{(g)}+\boldsymbol{y}_1(h)\frac{d_3}{d_1}\xi_3^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+d_3\xi_3^{(g)}}+\frac{\left(\boldsymbol{y}_2(h)-\boldsymbol{y}_1(h)\frac{d_2}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+d_3\xi_3^{(g)}}+\frac{\left(\boldsymbol{y}_3(h)-\boldsymbol{y}_1(h)\frac{d_3}{d_1}\right)\xi_3^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+d_3\xi_3^{(g)}}=\frac{\boldsymbol{y}_1(h)}{d_1}+\frac{d_2\left(\frac{\boldsymbol{y}_2(h)}{d_2}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+d_3\xi_3^{(g)}}+\frac{d_3\left(\frac{\boldsymbol{y}_3(h)}{d_3}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_3^{(g)}}{d_1\xi_1^{(g)}+d_3\xi_2^{(g)}+d_3\xi_3^{(g)}}$.

Therefore, when $N_j^{(g)}$ takes a general positive integer, we can get

$$\begin{aligned}
\boldsymbol{R}_j^{(g)}(h)=&\frac{\boldsymbol{y}_1(h)\xi_1^{(g)}+\boldsymbol{y}_2(h)\xi_2^{(g)}+\ldots+\boldsymbol{y}_{N_j^{(g)}}(h)\xi_{N_j^{(g)}}^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}=\frac{\boldsymbol{y}_1(h)}{d_1}+\frac{d_2\left(\frac{\boldsymbol{y}_2(h)}{d_2}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}\\
&+\ldots+\frac{d_{N_j^{(g)}}\left(\frac{\boldsymbol{y}_{N_j^{(g)}}(h)}{d_{N_j^{(g)}}}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_{N_j^{(g)}}^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}.
\end{aligned}\tag{15}$$

Note that when $\frac{\boldsymbol{y}_1(h)}{d_1}=\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, all but the first term $\frac{\boldsymbol{y}_1(h)}{d_1}$ in the above equation are less than 0, i.e., $\boldsymbol{R}_j^{(g)}(h)=$

$\frac{\boldsymbol{y}_1(h)}{d_1}+\frac{d_2\left(\frac{\boldsymbol{y}_2(h)}{d_2}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}+\ldots+\frac{d_{N_j^{(g)}}\left(\frac{\boldsymbol{y}_{N_j^{(g)}}(h)}{d_{N_j^{(g)}}}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_{N_j^{(g)}}^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}\leq\frac{\boldsymbol{y}_1(h)}{d_1}$. Similarly, when $\frac{\boldsymbol{y}_1(h)}{d_1}=\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, $\boldsymbol{R}_j^{(g)}(h)=\frac{\boldsymbol{y}_1(h)}{d_1}+\frac{d_2\left(\frac{\boldsymbol{y}_2(h)}{d_2}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_2^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}+\ldots+\frac{d_{N_j^{(g)}}\left(\frac{\boldsymbol{y}_{N_j^{(g)}}(h)}{d_{N_j^{(g)}}}-\frac{\boldsymbol{y}_1(h)}{d_1}\right)\xi_{N_j^{(g)}}^{(g)}}{d_1\xi_1^{(g)}+d_2\xi_2^{(g)}+\ldots+d_{N_j^{(g)}}\xi_{N_j^{(g)}}^{(g)}}\geq\frac{\boldsymbol{y}_1(h)}{d_1}$.

Accordingly, we can easily derive that $\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\leq\boldsymbol{R}_j^{(g)}(h)\leq\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$.

To facilitate the analysis, we construct a function $\mathrm{U}(\cdot)$ on $\boldsymbol{R}_j^{(g)}(h)$ such that $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)=\boldsymbol{R}_j^{(g)}(h)\log\left(\frac{\boldsymbol{R}_j^{(g)}(h)}{\boldsymbol{Q}(h)}\right)$. This function is evidently convex in the domain $(0,+\infty)$, and it attains a local minimum at $\boldsymbol{R}_j^{(g)}(h)=\frac{\boldsymbol{Q}(h)}{e}$. Based on the relationship between the local minimum point $\frac{\boldsymbol{Q}(h)}{e}$ and the values of $\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$ and $\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, we can determine the upper bound of $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)$ when $\boldsymbol{R}_j^{(g)}(h)$ lies in $\left[\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\},\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right]$.

Specially, it can be categorized into the following three cases: *i)* when $\frac{\boldsymbol{Q}(h)}{e}\geq\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, we have $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)\leq\mathrm{U}\left(\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right)$; *ii)* when $\frac{\boldsymbol{Q}(h)}{e}\leq\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, we have $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)\leq\mathrm{U}\left(\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right)$; *iii)* when $\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\leq\frac{\boldsymbol{Q}(h)}{e}\leq\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}$, we have $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)\leq\max\left\{\mathrm{U}\left(\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right),\mathrm{U}\left(\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right)\right\}$.

Let us define a function

$$\boldsymbol{G}_j^{(g)}(h)=\begin{cases}\mathrm{U}\left(\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right) & ,\frac{\boldsymbol{Q}(h)}{e}\geq\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\\ \mathrm{U}\left(\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right) & ,\frac{\boldsymbol{Q}(h)}{e}\leq\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\\ \max\left\{\mathrm{U}\left(\min_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right),\mathrm{U}\left(\max_{i|\widehat{a}_{i,j}^{(g)}=1}\left\{\frac{\boldsymbol{y}_i(h)}{d_i}\right\}\right)\right\} & ,\text{otherwize.}\end{cases}\tag{16}$$

as the upper bound of $\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)$. We can then obtain the upper bound of $\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\mid\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq0\right)$ according to the Markov inequality[16] as follows:

$$\begin{aligned}&\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq0\right)\leq\frac{\mathbf{E}\left[\sum_{h=1}^{|\mathcal{Z}|}\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq0\right]}{KLD_{\max}-\Delta_k}\\&=\sum_{h=1}^{|\mathcal{Z}|}\frac{\mathbf{E}\left[\mathrm{U}\left(\boldsymbol{R}_j^{(g)}(h)\right)\Big|\sum_{i=1}^{|\mathcal{N}|}\widehat{a}_{i,j}^{(g)}\xi_i^{(g)}\neq0\right]}{KLD_{\max}-\Delta_k}\leq\sum_{h=1}^{|\mathcal{Z}|}\frac{\boldsymbol{G}_j^{(g)}(h)}{KLD_{\max}-\Delta_k},\end{aligned}\tag{17}$$

where $\mathbf{E}[\cdot]$ is the expectation operator. Finally, we can get the upper bound of $\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right)>KLD_{\max}-\Delta_k\right)$ as

16. It is worth noting that the above derivation relies on the use of Markov's inequality to upper-bound the violation probability. Although such a bound may be conservative in practice, it provides a tractable closed-form surrogate that guarantees the feasibility of the probabilistic constraint in (11a). This theoretical guarantee motivates the adoption of the Markov and union bounds in our formulation, even at the cost of potential looseness in the resulting constraints.

follows:

$$\Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right) > KLD_{\max} - \Delta_k\right) = \prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i) + \left(1 - \prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\right)$$

$$\times \Pr\left(\mathrm{KLD}\left(\widehat{\boldsymbol{P}}_j^{(g)}||\boldsymbol{Q}\right) > KLD_{\max} - \Delta_k \Big| \sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} \neq 0\right) \leq \prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\left(1 - \prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\right) \times \sum_{h=1}^{|\mathcal{Z}|} \frac{\boldsymbol{G}_j^{(g)}(h)}{KLD_{\max} - \Delta_k}. \quad (18)$$

Therefore, the constraint (11a) is tightened to (11f) in Sec. 4.2.1, which is

$$\prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i) + \left(1 - \prod_{i|\widehat{a}_{i,j}^{(g)}=1}(1-p_i)\right) \times \sum_{h=1}^{|\boldsymbol{\mathcal{Z}}|} \frac{\boldsymbol{G}_j^{(g)}(h)}{KLD_{\max} - \Delta_k} \leq \delta, \forall s_j \in \mathcal{S}.$$

APPENDIX D
DERIVATION OF (11g)

By substituting the definition of $D_j^{(g)}$ into the above (11b), we can obtain that

$$\Pr\left(\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} d_i < D_{\min} + \Delta_d\right) \leq \varepsilon, \forall s_j \in \mathcal{S}. \quad (19)$$

Base on the Markov inequality, we can obtain the lower bound of the left side of the above inequality as follows:

$$\Pr\left(\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} d_i < D_{\min} + \Delta_d\right) \geq 1 - \frac{\mathbf{E}\left[\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} d_i\right]}{D_{\min} + \Delta_d}, \forall s_j \in \mathcal{S}. \quad (20)$$

Further, according to the linearity of expectation, we can get

$$\mathbf{E}\left(\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \xi_i^{(g)} d_i\right) = \sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} \mathbf{E}\left(\xi_i^{(g)}\right) d_i = \sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} p_i d_i, \forall s_j \in \mathcal{S}. \quad (21)$$

We can thus relax the constraint (11b) to

$$1 - \frac{\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} p_i d_i}{D_{\min} + \Delta_d} \leq \varepsilon, \forall s_j \in \mathcal{S}, \quad (22)$$

which results in (11g):

$$\sum_{i=1}^{|\mathcal{N}|} \widehat{a}_{i,j}^{(g)} p_i d_i \geq (D_{\min} + \Delta_d)(1-\varepsilon), \forall s_j \in \mathcal{S}.$$

APPENDIX E
DEATAILS OF OPERATIONS IN ALG. 3

• ***Operation 1. Addition of clients (Add)***: We test a new client from the not-selected clients set (i.e., $\mathcal{N} \backslash \mathcal{N}^s$) by adding it into the selected client set and run Alg. 2 to calculate $\widehat{F}^{(g)}$, denoted by $\widehat{F}^{(g)}_{\cup\{\boldsymbol{n}_i\}}$. If this operation satisfies all constraints and reduces $\widehat{F}^{(g)}$, we update $\widehat{F}^{(g)}_{\min}$, the corresponding selected client set $\mathcal{N}^s$ and association matrix $\widehat{\boldsymbol{A}}^{(g)}$. This continues until we evaluate all the not-selected clients, identifying and incorporating the additions that contribute to the reduction of the cost function. If no such additional improvement is found, we retain the original set of clients (lines 4-11).

• ***Operation 2. Removal of clients (Remove)***: We remove a client (i.e., move it from the long-term client set to the not-selected set) and run Alg. 2 to calculate $\widehat{F}^{(g)}$, denoted by $\widehat{F}^{(g)}_{\backslash\{\boldsymbol{n}_i\}}$. If this removal satisfies all constraints and reduces $\widehat{F}^{(g)}$, we update $\widehat{F}^{(g)}_{\min}$, $\mathcal{N}^s$ and $\widehat{\boldsymbol{A}}^{(g)}$. This continues until checking all selected clients, seeking the removals that offer reduction in the cost function. If no such removal are found, we preserve the initial selection of clients (lines 12-19).

• ***Operation 3. Exchange of clients (Exchange)***: We execute a swap by removing a currently selected client and adding another (i.e., exchanging a selected client with a not-selected one). Following this exchange, we run Alg. 2 to calculate $\widehat{F}^{(g)}$, denoted by $\widehat{F}^{(g)}_{\backslash\{\boldsymbol{n}_i\}\cup\{\boldsymbol{n}_j\}}$. If this exchange satisfies all constraints and decreases $\widehat{F}^{(g)}$, we update $\widehat{F}^{(g)}_{\min}$, $\mathcal{N}^s$ and $\widehat{\boldsymbol{A}}^{(g)}$. This is carried out until examining all clients, identifying the exchanges that reduce $\widehat{F}^{(g)}$. If no such exchange is found, we retain the original set of clients (lines 20-28).

APPENDIX F
DETAILED INFORMATION OF EXPERIMENTAL TESTBED

All simulations were conducted on a workstation equipped with an Intel Core i9-13900K @ 3.00 GHz CPU, 32 GB memory, and an NVIDIA GeForce RTX 4080 GPU, using MATLAB R2021a and Python 3.9 with PyTorch 1.13. The decision-making modules (Plan A and Plan B) were implemented in MATLAB as single-threaded processes on the CPU, without employing GPU acceleration, parallel computing toolboxes, or commercial solvers. The model training and federated learning procedure were simulated on the above testbed using PyTorch. We note that such single-machine simulation of distributed training is a widely adopted and validated practice in both FL and HFL research [62]–[64], as it provides a fair and reproducible environment for comparing algorithmic performance without introducing confounding factors from heterogeneous hardware or networking infrastructures. Therefore, the adopted testbed setup is both consistent with existing literature and sufficient to demonstrate the effectiveness of the proposed framework.

APPENDIX G
DATASETS AND MODELS

To ensure reproducibility and fair comparison with prior works, we adopt four widely used benchmark datasets, each coupled with a representative deep learning model. The details are summarized as follows:

• **MNIST dataset** [39]: A standard handwritten digit recognition dataset consisting of 60,000 training and 10,000 testing grayscale images of size 28×28. We employ a standard PyTorch CNN model with 21,840 parameters.

• **Fashion-MNIST dataset** [41]: A dataset containing 60,000 training and 10,000 testing grayscale images of 10 fashion product categories. We adopt the classical LeNet-5 model [39], which contains 29,034 parameters.

• **CIFAR-10 dataset** [40]: A color image dataset with 50,000 training and 10,000 testing samples across 10 object categories. We use a PyTorch CNN model with 576,778 parameters.

• **CIFAR-100 dataset** [40]: A more challenging extension of CIFAR-10, containing 100 classes with 600 images each. For this dataset, we employ the ResNet-18 model [42], which has 11,220,196 parameters.

• **Aerial Landscape Images (ALI) dataset** [65], [66]: A dataset constructed from the publicly available AID and NWPU-Resisc45 datasets, containing 15 aerial scene categories (e.g., rivers, mountains, forests, farmland, etc.) under diverse aerial conditions. To evaluate model performance, we employ a lightweight VGG network [67] with 60,074 parameters.

These datasets cover a broad spectrum of task complexities, from simple grayscale digits to high-resolution colored images, while the selected models range from lightweight CNNs to deeper architectures (e.g., ResNet18). This diverse setup provides a comprehensive evaluation of the proposed framework under different learning scenarios.

APPENDIX H
EXTENDED EVALUATIONS I

To better capture realistic and dynamic scenarios, we set the channel gain between clients and ESs to fluctuate within $[10^{-9}, 10^{-8}]$, and client CPU frequencies to vary within $[1, 10]$ GHz in each global iteration (unlike the fixed settings in Sec. 5). Under such dynamics, multiple executions of Plan A are necessary to accurately estimate clients' online probabilities. In practice, we set Plan A to run once every 10 global rounds, thereby adapting to environmental variations. As shown in Fig. 7, StagewiseHFL achieves over 90% test accuracy on MNIST while incurring only modest increases of 13.2%, 3.0%, and 6.0% in training delay, energy consumption, and overall cost, respectively, compared with OrigProbSolver, while still outperforming other baselines. For decision-making overhead, StagewiseHFL requires an average of only 51ms, demonstrating strong effectiveness and time efficiency even in highly dynamic environments. This efficiency is largely due to the fact that additional executions of Plan A can be performed in parallel with the training process, ensuring that repeated runs do not introduce noticeable real-time decision-making delay. Moreover, the frequency of triggering Plan A can be flexibly adjusted to balance decision accuracy and runtime efficiency. For example, Plan A may be reactivated when: *i)* unacceptable system costs (e.g., excessive training delay or energy consumption) are observed, *ii)* channel conditions or clients' computing capabilities change significantly, or *iii)* a predefined number of global rounds has been completed. Although a more comprehensive exploration of these adaptive strategies is left for future work, the above results confirm that parallel execution of Plan A enables StagewiseHFL to remain scalable and practical for real-world deployments.

APPENDIX I
EXTENDED EVALUATIONS II

To verify the reliability of the designed constraints (11a) and (11b) in Plan A, we conducted experiments under various values of $D_{\min}$ and $KLD_{\max}$, evaluating both the cost-saving performance of StagewiseHFL and its approximation to OrigProbSolver. As shown in Fig. 8, in most cases, we only have slight gaps in terms of value of cost function, in comparison with OrigProbSolver. However, when Plan A selects a large number of long-term clients, the performance of StagewiseHFL may degrade. For instance, when $D_{\min} = 5000$ and $KLD_{\max} = 0.1$, the overall cost of StagewiseHFL is 11.6% higher in terms of time and 9.6% higher in terms of energy compared with OrigProbSolver. This degradation is mainly due to the reduced diversity of training data, which requires more global rounds to reach the target accuracy, thereby increasing both time

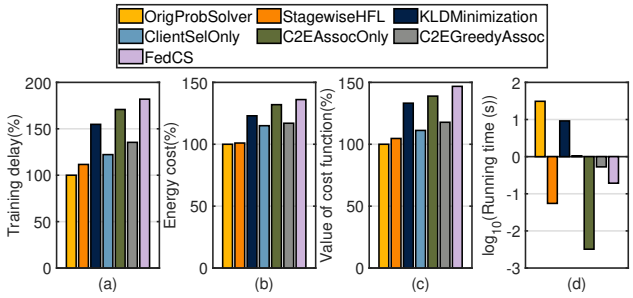
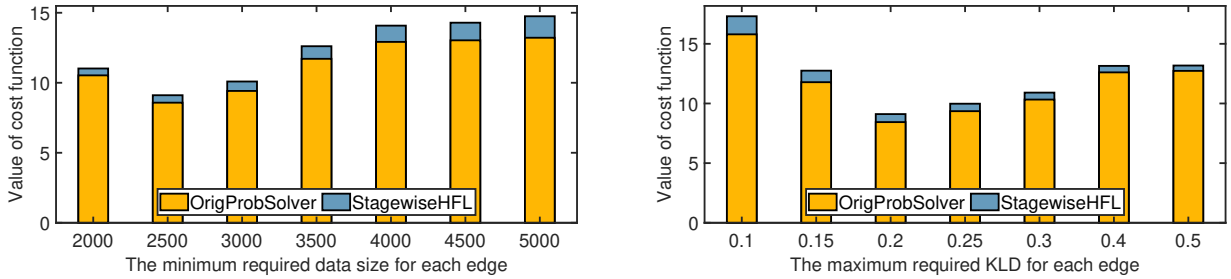


Fig. 7: Performance on training delay, energy consumption, overall cost, and running time under varying channel conditions and clients' computational capabilities.

and energy consumption. Nevertheless, with appropriate choices of $D_{\min}$ and $KLD_{\max}$, StagewiseHFL maintains reliable performance in practice. Moreover, the observed cost gaps remain acceptable, particularly considering that OrigProbSolver involves heavy decision-making overhead (see Fig. 5), making it impractical for dynamic real-world networks. From another view, these small gaps can verify the approximation of constraints (11a)) and (11b) against their originals, (9a)) and (9b), respectively.

It is worth noting that although the use of Markov inequality and the union bound introduces a certain degree of constraint relaxation during the transformation, the results still show that StagewiseHFL achieves substantial cost reduction and maintains robustness across different settings, and further substantiates the theoretical soundness and practical efficacy of our proposed constraints (11f) and (11g).



(a) Cost vs. $D_{\min}$

(b) Cost vs. $KLD_{\max}$

Fig. 8: Performance on the overall system cost for OrigProbSolver and StagewiseHFL.

APPENDIX J
EXTENDED EVALUATIONS III

To further validate the practical effectiveness of our proposed constraints (11f) and (11g), we conducted extensive Monte Carlo simulations to quantitatively demonstrate the relationship between the theoretical thresholds (δ, ϵ) and their corresponding empirical violation rates ($\widehat{\delta}$, $\widehat{\epsilon}$), which are summarized in Tables 6 and 7 below.

TABLE 6: Empirical violation rate $\widehat{\delta}$ vs. theoretical threshold δ

$\widehat{\delta}$		$KLD_{\max}$				
		0.1	0.2	0.3	0.4	0.5
δ	0.15	0.31	0.29	0.26	0.24	0.23
	0.20	0.36	0.32	0.31	0.32	0.29
	0.25	0.37	0.34	0.31	0.32	0.28
	0.30	0.38	0.33	0.30	0.28	0.29
	0.35	0.41	0.37	0.35	0.34	0.33

TABLE 7: Empirical violation rate $\widehat{\epsilon}$ vs. theoretical threshold ϵ

$\widehat{\epsilon}$		$D_{\min}$				
		2000	2500	3000	4000	5000
ϵ	0.15	0.27	0.26	0.22	0.23	0.21
	0.20	0.32	0.30	0.26	0.24	0.23
	0.25	0.33	0.31	0.28	0.26	0.25
	0.30	0.35	0.32	0.27	0.28	0.26
	0.35	0.38	0.34	0.33	0.29	0.27

As shown in Tables 6 and 7, the empirical violation rates $\widehat{\delta}$ and $\widehat{\epsilon}$ are generally close to their theoretical thresholds δ and ϵ, respectively. Although in certain configurations the empirical values are noticeably higher—for example, $\widehat{\delta} = 0.31$ when $KLD_{\max} = 0.1$ and $\delta = 0.15$—the deviations remain bounded and fall within an acceptable range for practical applications. This is consistent with the conservative nature of the Markov and union bounds, and is further influenced by the introduction of two constants $\Delta_k > 0$ and $\Delta_d > 0$, which were designed to tune the satisfaction probability of the original constraints (9a) and (9b) and to mitigate the risk of violating KLD and data size thresholds during practical training. The results confirm that our method offers a balanced and robust trade-off between theoretical rigor and practical feasibility.

APPENDIX K
EXTENDED EVALUATIONS IV

To further validate the effect of $KLD_{\max}$ on model accuracy, , building upon the observations in Sec. 5.3.1 we extend our experiments to two more challenging datasets: CIFAR-100 (trained with ResNet-18) and Tiny-ImageNet (trained with ResNet-34), where Tiny-ImageNet is a downsampled variant of the ImageNet dataset, consisting of 200 classes, each with 500 training images and 50 validation images. All images are resized to 64×64 pixels. Compared with CIFAR-100, Tiny-ImageNet is more complex both in terms of class granularity and visual diversity. In the HFL setup with 100 clients, CIFAR-100 is partitioned such that each client owns 10–30 classes, while in Tiny-ImageNet each client is assigned 20–60 classes. We compare the performance of OrigProbSolver and StagewiseHFL under different values of $KLD_{\max}$. For each setting, we report the final test accuracy within 100 global iteration.

As shown in Tables 8 and 9, imposing more restrictive $KLD_{\max}$ values (i.e., enforcing lower data heterogeneity across ESs) leads to higher test accuracy for both methods. For example, on CIFAR-100, reducing KLDmax from 1.0 to 0.3 improves OrigProbSolver's accuracy from 35.34% to 55.12%, indicating that stricter distributional constraints effectively enhance model performance by mitigating data heterogeneity. While OrigProbSolver consistently achieves slightly higher accuracy than StagewiseHFL, the latter remains highly competitive, with only a marginal gap, and offers substantially better scalability. These results reaffirm the trend observed on MNIST, demonstrating that controlling KLD positively impacts model accuracy and that this effect extends to larger and more complex datasets.

TABLE 8: Test accuracy vs. $KLD_{\max}$ on CIFAR-100

Test accuracy (%)		$KLD_{\max}$				
		0.3	0.4	0.5	0.7	1.0
Method	OrigProbSolver	55.12	47.77	43.27	36.49	35.34
	StagewiseHFL	54.37	46.43	43.66	36.38	35.41

TABLE 9: Test accuracy vs. $KLD_{\max}$ on Tiny-ImageNet

Test accuracy (%)		$KLD_{\max}$				
		0.3	0.5	0.7	1.0	1.5
Method	OrigProbSolver	45.21	41.42	36.26	32.49	31.34
	StagewiseHFL	43.64	39.94	35.79	32.38	31.41

APPENDIX L
EXTENDED EVALUATIONS V

To further bridge the gap between simulation-based experiments and real-world deployment, we conduct supplementary evaluations on the ALI dataset, from which we select 10 representative categories that closely emulate realistic UAV deployment scenarios (see Fig. 9).



(a) Agriculture (b) Airport (c) Beach (d) City (e) Desert (f) Forest

Fig. 9: Different categories of images in ALI

As illustrated in Fig. 10, after 100 global iterations on ALI, StagewiseHFL achieves higher final test accuracy than ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, and FedCS by 2.68%, 2.26%, 1.48%, and 3.29%, respectively, while remaining only 1.12% and 1.59% lower than OrigProbSolver and KLDMinimization. Regarding cost efficiency on the ALI dataset (Fig. 11(c)), to reach over 85% test accuracy, StagewiseHFL incurs merely a 5.3% loss compared to OrigProbSolver, yet outperforms KLDMinimization, ClientSelOnly, C2EAssocOnly, C2EGreedyAssoc, and FedCS by 19.6%, 21.0%, 18.9%, 24.6%, and 31.5%, respectively. Importantly, our method achieves an average inference time of less than 100 ms (Fig. 11(d)), which is an order of magnitude faster than OrigProbSolver, highlighting its suitability for real-time deployment. Overall, these results consistently demonstrate that StagewiseHFL delivers superior performance in model accuracy, cost efficiency, and decision-making speed, even under large-scale and more challenging scenarios.

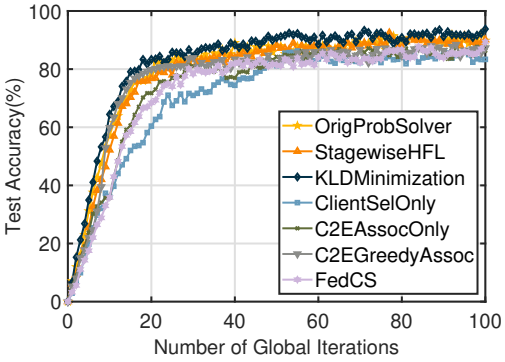


Fig. 10: Performance comparison in terms of test accuracy on ALI

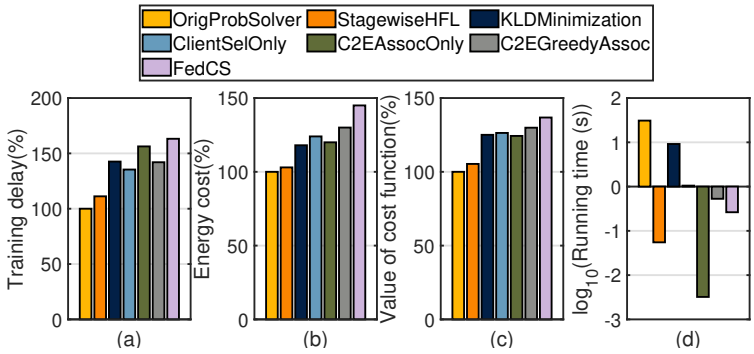


Fig. 11: Performance on training delay, energy consumption, overall cost, and running time on ALI