iHFL: intelligent Hierarchical Federated Learning Orchestration

Mattia Giovanni Spina
DIMES Dept
University of Calabria (Unical)
Rende (CS), Italy
mattiagiovanni.spina@dimes.unical.it

Gabriele Tassone
Spintel S.R.L.
Rende (CS), Italy
g.tassone@spintel.it

Gianluigi Greco
DeMaCS Dept.
University of Calabria (Unical)
Rende (CS), Italy
gianluigi.greco@unical.it

Floriano De Rango
DIMES Dept.
University of Calabria (Unical)
Rende (CS), Italy
derango@dimes.unical.it

***Abstract*—The advent of the next generation of networks is fostering the widespread distribution of massive and intelligent IoT deployments. In this evolving landscape, distributed Artificial Intelligence (AI) plays a pivotal role. Federated Learning (FL) and its hierarchical version, namely HFL, emerge as key enablers for effectively distributing intelligence across the device-edge-cloud continuum under stringent privacy and resource constraints. Although guaranteeing better scalability and robustness for the learning process, HFL suffers from system and data heterogeneity. Indeed, clients participating in the federated environment may exhibit different computational capabilities and, therefore, data sampling patterns. This negatively affects intermediate model updates, causing unstable and delayed convergence of the global model. To mitigate such issues and improve the learning process, this paper introduces an intelligent HFL (iHFL) framework in which edge servers are equipped with predictive orchestration capabilities. In iHFL, the statistical utility and resource availability of participating clients are characterized through a structured metadata vector. This vector is fed to an AI-based predictor placed at the edge, which predicts the expected behavior for the next learning rounds, enabling intelligent client selection. These components, then, aids to enable a filtering technique that reduces the propagation of intermediate weak updates across the hierarchy when the number of statistically useful participating clients is not sufficiently representative. The effectiveness of the proposal is demonstrated through an extensive experimental campaign which compared iHFL against existing solutions, showcasing its improvement in terms of faster convergence, better network (reduction of 31%) and resource utilization (reduced number of required active clients of about 40%), preserving the overall global model accuracy, even under heterogeneous conditions.**

***Index Terms*—Hierarchical Federated Learning, Internet of Things, Client Selection, Intelligent Hierarchy Orchestration**

I. Introduction

The fifth generation of networks (5G) paved the way for ubiquitous and massive IoT deployments, enabling intelligent services and a breakthrough form of interaction, such as holographic communication or tactile internet [1]. This rapidly determined an ever-increasing surge in the volume of generated and exchanged data. The upcoming 6G, fostering the tight integration of Artificial Intelligence (AI) in the network and combined with the network sofwarization process started with 5G, is expected to further exacerbate the challenges related to network and data management [2], [3]. This huge and increasing volume of data also represents a great challenge for the effective execution of traditional, centralized AI algorithms on constrained IoT devices [4] and remote powerful servers. In this setting, both communication cost and privacy concerns represent a primary issue. Distributed learning techniques constitute a viable way to mitigate such issues and design effective orchestration for AI pipelines in the network [5]. In this regard, **Federated Learning** (FL) has emerged as a promising approach enabling collaboration among training entities by distributing the overall training task and allowing them to share model updates, and not raw data, therefore reducing direct data exposure and alleviating part of the communication burden [6]. Nevertheless, despite its benefits, FL is also characterized by severe issues. First of all the communication overhead due to the frequent communication required between the client and the aggregation server on the far cloud. Additionally, traditional FL exhibits a unique point of failure: if the aggregation server goes down, the entire learning process stops [7], [8]. To mitigate such issues, the distribution of learning tasks is further improved by organizing the FL entities in a hierarchical fashion. In this direction, **Hierarchical FL** (HFL) introduces intermediate aggregation layers at the edge, reducing long-range communication overhead, improving scalability, and enabling part of the orchestration process to be moved closer to the data sources [9]. However, while improving traditional FL, HFL introduces several challenges, mainly coupled with system heterogeneity and data distribution, which must be handled jointly under dynamic and different operating conditions. In particular, upper-layer coordination and orchestration are often decoupled from client-relevant behavioral and reliability analysis, with weak local aggregations that can be propagated upward due to poor client statistical utility or if the set of clients is poorly representative or unstable [10]. Motivated by these challenges, this work improves on HFL by proposing an intelligent HFL (iHFL) framework, in which the edge servers closer to the FL clients are equipped with AI-based *predictive intelligence*. At the basis of iHFL lies the real-time characterization of the evolving statistical and resource availability of the participating clients through a structured metadata vector. This vector represents the input of an AI-based predictor which forecasts the expected client behavior for the next FL communication rounds, enabling an *intelligent*

979-8-3195-0167-7/26/$31.00 ©2026 IEEE

client selection which allows to exclude *stragglers* before the local aggregation. These components, then, converge into and enable an adaptive logic that allows edge servers to decide to perform the local aggregation or, instead, to deliver intermediate model weights upward in the hierarchy, avoiding the propagation of weak model updates due to a limited set of representative available participating clients. The effectiveness of the proposal is demonstrated through a comprehensive experimental campaign comparing the proposal against two recent existing solutions, namely HED-FL [11] and LA-FL [12], by *evaluating active client participation*, *accuracy*, *convergence time*, and *network overhead* under heterogeneous conditions. The remainder of this paper is organized as follows: Section II reviews recent literature works. Section III delves into the details of the proposal. Section IV showcases its benefits through the performance evaluation analysis and, finally, Section V concludes the manuscript.

II. Related Works

Recent literature witnesses the interest to address challenges related to both intelligent client selection and orchestration to improve the federated learning process in HFL.

Li et. al., in [12], propose a management solution for HFL in which clients are selected estimating the experienced training latency and combining it with the time expected to upload the model updates upward in the hierarchy. In the same direction, the authors in [13] present a delay-aware HFL exploiting information relevant to the round-trip delay between the edge and the cloud during the training process. Nevertheless, these solution do not consider a multi-dimensional prospective other than training latency or network delay for the estimation, particularly limiting the ability of the orchestration process to distinguish the causes of client inefficiency and to take more informed selection and aggregation decisions. *Ali et al.* in [14] determine the number of global communication rounds based on the different delays experienced by the clients. However, none of the above works explore the nature of the delay, which may arise from system heterogeneity of the involved clients, rather than solely from network-relevant issues. The work proposed in [15] proposes a client selection technique that combines client statistical utility and trust score to exclude participants that may either exhibit malicious behavior or provide weak model updates. In [16] a two-step client selection procedure for HFL is introduced by leveraging computational resource indicators like CPU and RAM. However, this proposal does not consider situations in which extremely powerful clients may still exhibit poor statistical utility due to a limited set of data or due to imbalance in the label distribution. *HED-FL* is introduced in [11]. It proposes a dynamic and energy-driven heuristic orchestration strategy to adapt the number of aggregation, prior to send the model updates to the global aggregator server, to be performed on the different layers of the hierarchy balancing accuracy and energy consumption.
FedSAE is proposed in [17] to address system heterogeneity in HFL by selecting clients according to historical workload-completion information and predicting the affordable workload of each participant. However, it does not explicitly account for the current resource conditions of clients, which may reveal transient straggler behavior and affect the reliability of the selected updates. The work in [18] adopts a Pareto-Optimal Bi-Level Reinforcement Learning (HFL-PBRL) to optimize the frequency of the aggregation and the number of rounds, balancing accuracy, communication, and convergence time. A novel Dynamic Gradient Filtering (DGF) is proposed in [19] for a robust FL against Byzantine failures.

Main Contributions. Overall, the above literature analysis shows that HFL orchestration should not be limited to isolated optimization objectives. Rather, effective hierarchy management requires a broader and multi-dimensional perspective that jointly accounts for system heterogeneity, data distribution properties, client reliability, and the quality of the intermediate model updates. This is crucial to prevent weak or poorly representative local updates from propagating upward in the hierarchy, which can negatively affect the global model's convergence stability and speed. To address these gaps, *iHFL* provides an intelligent HFL orchestration framework to effectively orchestrate the learning process by combining *(i)* a structured metadata vector aimed at capturing the evolving behavior of participating clients in terms of statistical utility and available resource; *(ii)* an AI-based predictor that, fed with the metadata vector, forecasts the expected behavior of the clients to support an intelligent client selection that aims at reducing the negative impact of clients with poor statistical utility. Finally, *(iii)* to comprehensively orchestrate the hierarchy, edge servers implements an adaptive logic that allows them to avoid the propagation of weak model updates when the number of statistically significant clients is limited.

III. Design of an intelligent HFL (iHFL) Orchestration Framework

This section delves into the details of iHFL, presenting its main components: *(i)* metadata vector-aided client profiling, *(ii)* edge server intelligent prediction of clients' behavior, *(iii)* informed client selection technique, and *(iv)* edge-level filtering mechanism.

A. System Model

The framework considers an FL deployment organized in a hierarchy $H = \{h_0, h_2, ..., h_{l-1}\}$ composed of l levels.

- h_0 represents the hierarchy level in which FL clients (e.g., IoT devices) reside. Each client $c_i \in h_0$ performs a local training at round t on a private local dataset $D_{c_i}^{(t)}$. This results in a local model update denoted as $\theta_{c_i}^{(t)}$.
- h_1 denotes the set of edge servers $e_{1j} \in h_1$ closest to participating clients. These edge servers play a pivotal role. Let $e_j^{(1)} \in H_1$ denote a generic edge server, and let $C_{e_{1j}} \in \mathrm{h_0}$ be the set of clients associated with it. Unlike the upper-layer intermediate nodes, each $e_j^{(1)}$ is equipped with an AI-based predictive model $M(\cdot)$ (detailed in the upcoming section). With this model, the edge server can follow the behavioral evolution of the set of clients it controls $C_{e_{1j}}$, forecasting their expected behavior and

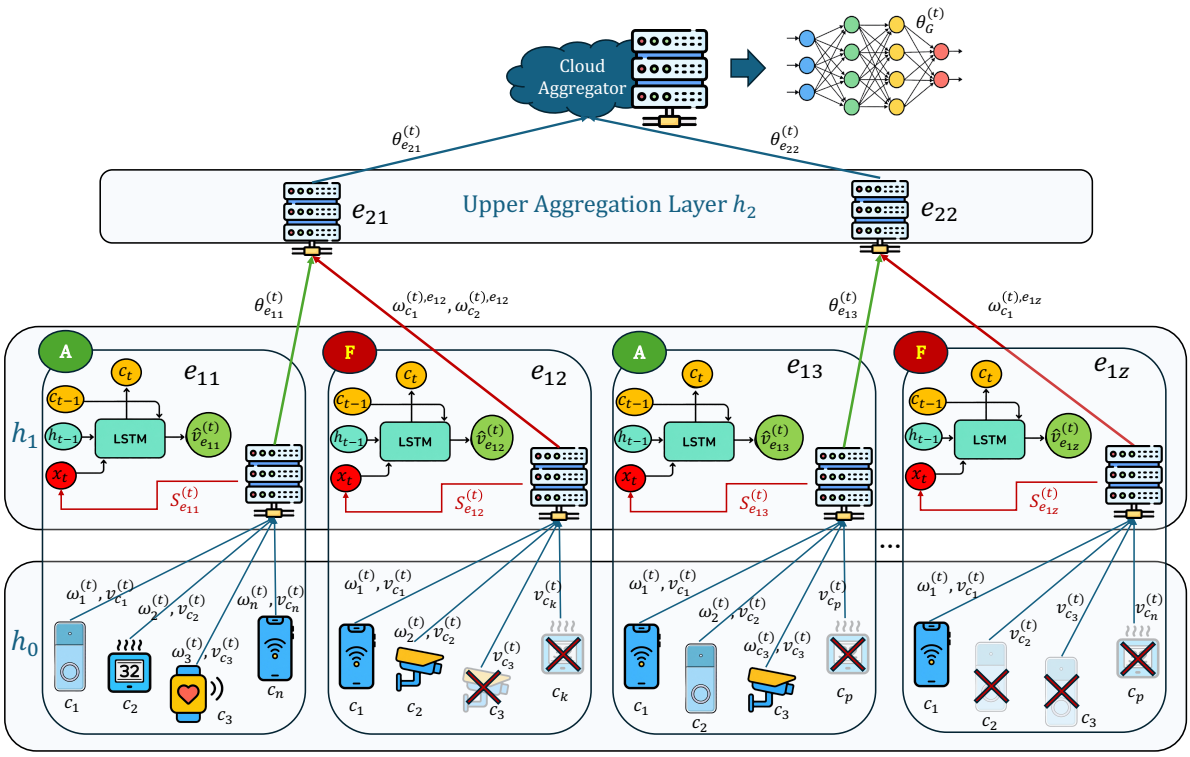


Fig. 1. Graphical Representation of the Proposed iHFL Orchestration framework on a hierarchy composed of four layers.

supporting the design of an intelligent client selection. This intending to filter clients with poor statistical utility or clients whose (energy or computing) resource are about to be depleted or over utilized (i.e., *straggler* clients). Most importantly, such approach can anticipate the transition of clients to *straggler* state promptly filtering them and reducing their negative impact on the global model convergence. Finally, the entities in h_1, like e_{1j}, determine the set of admitted clients for the local aggregation, namely $C_{e_{1j},sel} \subseteq C_{e_{1j}}$.

- The remainder intermediate entities in the hierarchy $h_{1<i<l-1}$ operate as aggregation-only entities. These nodes are not directly connected to clients and are not enhanced with predictive capability. Their role is limited to receiving the outputs generated at the lower layer, combining them according to the hierarchical aggregation process, and forwarding the resulting aggregate toward the global coordinator.
- Finally, the global server that aggregates all the contributions that come from the entities of the lower layers in the hierarchy, and computes the global model θ^G.

The proposed hierarchy equips with predictive intelligence only the edge servers closest to the clients. This choice is motivated by their finer-grained observability over client behavior, while also avoiding unnecessary communication overhead toward upper hierarchy layers. Fig.1 graphically depicts the described hierarchical organization.

B. AI-based Predictive Intelligence over Edge Server

To enhance the edge servers $e_{1j} \in h_1$ with the AI-based prediction model $M(\cdot)$, a structured metadata vector characterizing the client operational and statistical utility profile needs to be defined. Specifically, let $v_{c_i}^{(t)}$ the metadata vector associated with client c_i. The metadata vector is, then, defined as follows:

$$\mathbf{v}_{c_i}^{(t)} = \left[d_{c_i}^{(t)},\ r_{c_i}^{(t)},\ b_{c_i}^{(t)},\ tr_{c_i}^{(t)},\ \psi_{c_i}^{(t)}\right]^\top \in \mathbb{R}^5 \qquad (1)$$

In particular,

- $d_{c_i}^{(t)}$ represents the **local data volume**, measuring the amount of training data currently available at the client. Low values may indicate that the local update is poorly representative and more sensitive to noise.
- $r_{c_i}^{(t)}$ is the **residual energy** of the device. This prevents repeatedly selecting clients that are close to energy depletion.
- $b_{c_i}^{(t)}$ the **available uplink bandwidth**, reflecting the communication capability of the client during the current round. Low values may delay update transmission and introduce bottlenecks-
- $tr_{c_i}^{(t)}$ the **expected local training time**, used as an indicator of computational load and potential *straggler* behavior.
- $\psi_{c_i}^{(t)}$ the **class-distribution entropy**, used to quantify the statistical quality of the local dataset and the degree of class imbalance. Particularly, the Shannon entropy is adopted to compute such metric. Low entropy values

indicate highly skewed local label distributions, whereas larger values correspond to more balanced local data.

This vector is transmitted by the clients to the edge servers in h_1 along with their local model updates. Each component of v captures a crucial aspect of the current state of the clients in terms of available resource, utility, system heterogeneity and data distribution. To capture the temporal evolution of each client profile, each edge server $e_{1j} \in h_1$ builds an observation window $W_{c_i}^{(t)}$ of length k associated with each participating client c_i and defined as follows:

$$W_{c_i}^{(t)} = \left[v_{c_i}^{(t-k+1)}, v_{c_i}^{(t-k+2)}, \ldots, v_{c_i}^{(t)}\right] \tag{2}$$

The edge server e_{1j} aggregates the $W_{c_i}^{(t)}$ of each client $c_i \in C_{e_j}$ into the following metadata vector:

$$S_{e_{1j}}^{(t)} = \left\{W_{c_i}^{(t)} : c_i \in C_{e_{1j}}\right\} \tag{3}$$

This represents the input vector of the AI-based prediction model $M(\cdot)$, whose computation can be denoted as follows:

$$\hat{v}_{e_{1j}}^{(t)} = M(S_{e_{1j}}^{(t)}) \tag{4}$$

It is worth noticing that $\hat{v}_{e_{1j}}^{(t)}$ represents the expected behavior of clients, on the basis of the behavioral profile observed in round t, for the next round $t+1$. Given the nature of the data that compose the metadata vector and that must capture the evolving temporal behavioral profile of the participating clients, a recurrent neural network has been adopted. Specifically, a Long-Short Term Memory (LSTM) network is implemented. Its memory mechanism allows it to capture historical (long- and short-term) dependencies in sequential data, enabling the edge server to anticipate client degradation or variations in statistical utility. At a high level of abstraction, the LSTM can be represented using a tuple (hs, R, W) in which hs is the hidden-state dimensionality, R is the number of stacked recurrent layers, W is the sequence length of the input observation window. Based on the edge-level observation window $S_{e_{1j}}^{(t)}$, the LSTM recursively updates hidden and cell memory states through standard gating mechanism, capturing both short-term variations and long-term behavioral patterns. The resulting representation is finally mapped into the predicted edge-level metadata profile expected for the next round, which is represented by $\hat{v}_{e_{1j}}^{(t)}$. This information is then used to support informed, statistical utility- and resource-aware client scoring and selection, as it is explained in the following paragraph.

C. Intelligent Client Selection via client behavioral prediction

The expected behavioral profile that the AI-based prediction model forecasted for the next round through $\hat{v}_{e_{1j}}^{(t)}$ is leveraged to compute a score associated with each participating client that enables the design of an intelligent client selection mechanism. Specifically, the rationale is the following: a client that observed metadata remains close to the predicted edge-level profile $\hat{v}_{e_{1j}}^{(t)}$ is more likely to behave consistently with the expected operating conditions of the local cluster, whereas large deviations may reveal unstable or unfavourable behavior. In this direction, let $v_{e_{1j}}^{(t)}$ be the observed profile of client $c_i \in C_{e_{1j}}^{(t)}$, at round t. The edge server e_{1j} assigns to client c_i the score $s_{c_i}^{(t)}$ as follows:

$$s_{c_i}^{(t)} = 1 - \sum_{u=1}^{k=5} \alpha_u \left| v_{c_i,u}^{(t)} - \hat{v}_{e_{1j},u}^{(t)} \right|, \qquad \sum_{u=1}^{k=5} \alpha_u = 1 \tag{5}$$

The Eq.5 represents a weighted deviation from the expected behaviour, whereas the term $1-$ is applied to map small deviations (the expected behavior aligns with the observed one) to scores close to 1 and large deviations (the predicted observed metadata vectors exhibit a large difference) to scores close to 0, meaning that the client does not align with the predicted expected behavior. Additionally, since different metadata components could have different operational relevance, the formulation introduces non-negative weights $\{\alpha_j\}_{u=1}^{5}$ satisfying $\sum_{u=1}^{k=5} \alpha_u = 1$ assigning larger weights to critical dimensions (e.g., battery level and training time) and smaller, yet non-zero, weights to degrading dimensions (e.g., data volume, entropy, network quality). Given a threshold $\gamma \in [0,1]$, the set of clients selected by the edge server e_{1j} to be part of the local aggregation process in the next round (and the current) is:

$$\mathcal{C}_{e_{1j},sel}^{(t|t+1)} = \left\{c_i \in C_{e_{1j}}^{(t)} : s_{c_i}^{(t)} \geq \gamma\right\} \tag{6}$$

It is worth noticing that the client selection does not block the communication of the metadata vector, meaning that each of the clients controlled by a given edge server e_{1j} still have to send their behavioral profile, so also client $c_i \notin \mathcal{C}_{e_{1j}}^{(t|t+1)}$. While such clients are not allowed to send their updated model weights, they still have to send their behavioral metadata vector to correctly capture possible variations in their pattern (e.g., over time they acquire more data, producing a more accurate model update, and therefore their statistical utility increases). The clients $c_i \notin \mathcal{C}_{e_{1j}}^{(t|t+1)}$ will, therefore, be stopped for round $t+1$.

D. Preventing weak local update propagation

The intelligent client selection presented in previous sections contributes to filters out clients that do not provide significant contribution to the training process, either due to their poor statistical utility or computational/network resource. Nevertheless, the selection may results in communication rounds in which the number of participating clients in each edge server e_{1j} is particularly limited. In such cases, performing local aggregation over a very limited set of updates may generate statistically weak intermediate model and propagate noisy or biased information toward the upper layers of the hierarchy and, eventually, undermining the effectiveness of the global model. To address this issue, that can nullify the advances introduced through the proposal, in iHFL the edge server in h_1 are also characterized by an adaptive behavioral modality which is aimed at preventing the propagation of weak model update under the aforementioned conditions. Particularly, an edge server $e_{1j} \in h_1$ can dynamically change its behavior

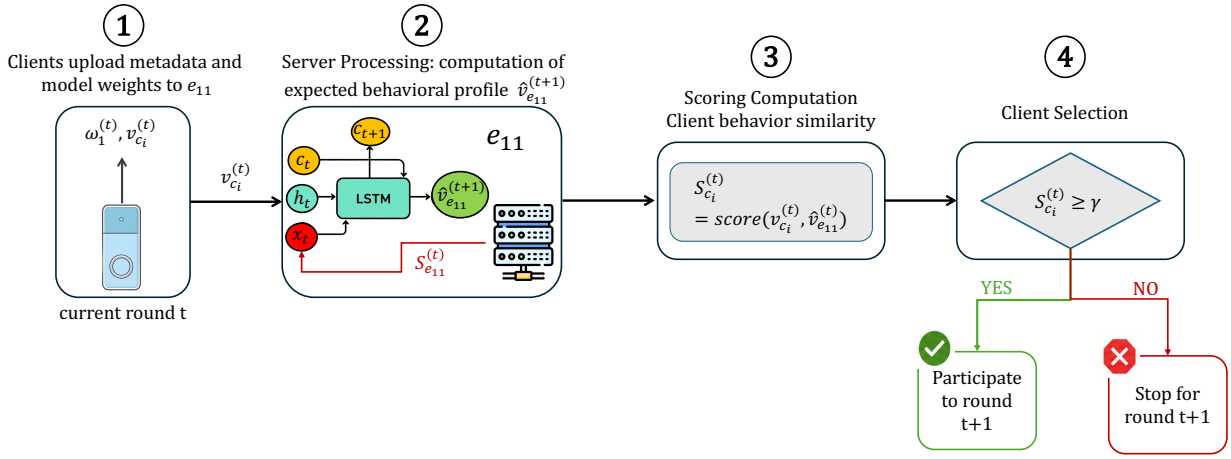


Fig. 2. Illustration of the decision process applied by the edge server for a single client. Specifically, at current round t, the score of the client c_1, namely $s_{c_1}^{(t)}$, is computed using Eq. 5. The score is then used to determine if such a client is allowed to continue for the round $t+1$ or not.

from a traditional *local aggregator* to a *forwarder* of raw weights from the client model. The rationale is: if the set of selected clients is not enough to ensure a robust model and fast convergence, the local server does not aggregate but forwards the raw model weights to upper layers to which the weights from other edge servers may converge and be comprehensively aggregated. In such a way, the aggregation takes place with a large number of contributions, which are also characterized by a significant statistical utility, being generated by the clients chosen with the previously described intelligent client selection mechanism.

Let $C_{e_{1j}}^{(t)}$ be the set of clients available at round t and let $\mathcal{C}_{e_{1j},sel}^{(t|t+1)} \subseteq C_{e_{1j}}^{(t)}$ be the subset of selected nodes using Eq.6. It is possible to define the participation ratio measured at the edge server $e_{1j} \in h_1$ as follows:

$$p_{e_{1j}}^{(t)} = \frac{|\mathcal{C}_{e_{1j},sel}^{(t|t+1)}|}{|C_{e_{1j}}^{(t)}|} \tag{7}$$

Based on $p_{e_{1j}}^{(t)}$, the edge server a generic server in h_1, namely e_{1j}, can behave in one of the following modes:

$$\begin{cases} \text{aggregate}, & \text{if } p_{e_{1j}}^{(t)} \geq \rho, \text{or} \\ \text{forward}, & \text{if } p_{e_{1j}}^{(t)} < \rho. \end{cases} \tag{8}$$

Particularly, if $p_{e_{1j}}^{(t)} \geq \rho$, the edge server performs the local aggregation over the selected client updates. This produces the local update $\theta_{e_{1j}}^{(t+1)}$ obtained via *FedAvg* aggregation function as follows:

$$\theta_{e_{1j}}^{(t)} = \text{FedAvg}(\theta_{c_i}^{(t)} \mid c_i \in \mathcal{C}_{e_{1j},sel}^{(t|t+1)}) \tag{9}$$

Otherwise, it sends the client model weights vector $\omega_{c_i}^{(t)} \; \forall \; c_i \in \mathcal{C}_{e_{1j},sel}^{(t|t+1)}$(no local aggregation is performed) to the upper layer. This design ensures that upper layers can combine the forwarded client updates with both locally aggregated models received from other edge servers and additional non-aggregated client updates relayed by edge servers operating in switching mode, thus benefiting from a broader and more reliable pool of contributions. Alg. 1 formally summarizes the operational steps of iHFL.

IV. Performance Evaluation

This sections is aimed at showcasing the benefits of iHFL, comparing it with existing recent literature works, namely *HED-FL* [11] and the work proposed in [12], referred to as *LA-FL*. *HED-FL* addresses the same objective of balancing model accuracy and energy consumption across the hierarchy determining whether updates are locally aggregated and then directly sent to the aggregation server to obtain the global model or they have to be re-transmitted to the clients for further training rounds. But it remains agnostic to system heterogeneity and does not include predictive client evaluation and selection, as well as effective hierarchy orchestration through adaptable edge server behavior. The comparison against *LA-FL* complements the performance evaluation by shifting the focus on the adoption of an informed client selection technique based on the estimation of the training latency of the participating clients. The objective of the experimental campaign is to prove that the proposal can reach comparable *accuracy* by reducing the *amount of active clients* required to participate during the training process, ensuring a *faster convergence* by selecting the most statistically useful clients and consequently reducing the amount of generated *network traffic*.

A. Experimental Setup

The proposal has been implemented by integrating the Flower Simulator, used to manage the federated learning process, with Containernet, adopted to emulate the networked and container-based execution environment. The experiments run on a server equipped with an Intel Xeon Gold 6252N CPU, 192 cores, and 528 GB of RAM. The considered hierarchy includes 80 clients at level (h_0), evenly distributed among four edge servers in h_1, two intermediate nodes in h_2, and

Algorithm 1 iHFL Edge-Side Operational Steps

Require: Set of clients $C_{e_{1j}}^{(t)}$ associated with edge server e_{1j}; predicted metadata profile $\hat{v}_{e_{1j}}^{(t)}$; thresholds γ and ρ; weights $\{\alpha_u\}_{u=1}^{5}$.

Initialize $\mathcal{C}_{e_{1j},sel}^{(t|t+1)} \leftarrow \emptyset$
for all $c_i \in C_{e_{1j}}^{(t)}$ **do**
  Receive $\theta_{c_i}^{(t)}$ and $v_{c_i}^{(t)}$
  Compute:

$$s_{c_i}^{(t)} = 1 - \sum_{u=1}^{5} \alpha_u \left| v_{c_i,u}^{(t)} - \hat{v}_{e_{1j},u}^{(t)} \right|$$

  if $s_{c_i}^{(t)} \geq \gamma$ **then**
    $\mathcal{C}_{e_{1j},sel}^{(t|t+1)} \leftarrow \mathcal{C}_{e_{1j},sel}^{(t|t+1)} \cup \{c_i\}$
  else
    Exclude c_i from the current local aggregation step
  end if
end for
Compute:

$$p_{e_{1j}}^{(t+1)} = \frac{\left|\mathcal{C}_{e_{1j},sel}^{(t|t+1)}\right|}{\left|C_{e_{1j}}^{(t)}\right|}$$

if $p_{e_{1j}}^{(t)} \geq \rho$ **then**
  Perform local aggregation:

$$\theta_{e_{1j}}^{(t)} = \text{FedAvg}\left(\theta_{c_i}^{(t)} \mid c_i \in \mathcal{C}_{e_{1j},sel}^{(t|t+1)}\right)$$

  Forward $\theta_{e_{1j}}^{(t)}$ to the upper hierarchy layer
else
  Activate forwarding mode
  for all $c_i \in \mathcal{C}_{e_{1j},sel}^{(t|t+1)}$ **do**
    Forward $\theta_{c_i}^{(t)}$ to the upper hierarchy layer
  end for
end if
Update the metadata history for the next prediction step

the global aggregator as the root.

Configuration of h_0. Clients train a lightweight Convolutional Neural Network on the MNIST dataset. The CNN is composed of two convolutional layers for feature extraction and two fully connected layers for classification. The hidden layers utilize the ReLU activation function, while the output layer applies a LogSoftmax function to yield log-probabilities across the 10 target classes. Specifically, the feature extraction backbone consists of two sequential convolutional blocks with 32 and 64 filters (3×3 kernel size), respectively. Each block incorporates 2×2 max pooling and a 0.2 dropout rate to enhance generalization. The resulting feature maps are flattened and processed by a 128-neuron dense layer, regularized with a 0.5 dropout rate, before being fed to the final output layer. System heterogeneity is introduced by assigning different CPU capacities, i.e., (0.2,0.4,0.6,0.8) cores, to emulate resource-constrained participants.

TABLE I
PREDICTION ERROR OF THE SELECTED LSTM CONFIGURATIONS.

ID	(hs, R, W)	MAE	RMSE
C1	(16,1,5)	0.145	0.168
C2	**(64,1,5)**	**0.135**	**0.154**
C3	(32,1,10)	0.154	0.177

Predictive Intelligence in h_1. The predictive model $M(\cdot)$ deployed on the edge server in h_1 is implemented through an LSTM network defined as (hs, R, W). To evaluate the trade-off between prediction accuracy and computational complexity, three representative configurations are considered: **C1**=(16,1,5), which represents a lightweight lower-bound architecture; **C2**=(64,1,5), which increases the recurrent representation while preserving a single-layer design; and **C3**=(32,1,10), which evaluates the impact of a longer temporal observation window. In all cases, the last hidden state is mapped through a dense layer with five output neurons, followed by a Sigmoid activation to keep predictions consistent with the normalized metadata range ([0,1]). The configurations are evaluated in terms of MAE and RMSE. This comparison will reveal that the most suitable configuration is **C2**=(64,1,5). Finally, the weights $\{\alpha_i\}_{u=1}^{5}$ are uniformly set to 0.2, avoiding introducing any preference for the dimension considered for the metadata vector.

TABLE II
IMPACT OF THE PARTICIPATION-RATIO THRESHOLD ρ ON ROUND TIME AND EDGE SERVER IN h_1 OPERATIVE MODE CHANGE EVENTS.

Threshold	Avg. per-round time [s]	#Operative Mode Changes
$\rho = 0.1$	202.47	70
$\rho = 0.2$	197.92	84
$\rho = 0.5$	196.42	88
$\rho = 0.8$	297.03	117

B. Experimental Results

LSTM ablation and tuning: Tab.I shows the results relevant to the comparison of the LSTM architectures considered, in terms of RMSE and MAE. These metrics provide complementary insights, since MAE captures the average prediction deviation, while RMSE gives higher weight to larger errors. The results show that C2 achieves the best predictive performance, reducing MAE by about (6.9%) and (12.3%) with respect to C1 and C3, respectively. A similar trend is observed for RMSE, where C2 reduces the prediction error by approximately (8.3%) compared to C1 and (13.0%) compared to C3. Conversely, C1 provides intermediate results due to its lightweight design, while C3 exhibits the highest error, suggesting that extending the observation window does not necessarily improve the forecasting quality. Therefore, C2 is selected as the reference LSTM configuration for the proposed framework.

Thresholds Analysis: The iHFL orchestration framework introduces two parameters: ρ and γ. On the one side, ρ guides the edge server behavioral switch, from local aggregator to forwarding entities. On the other side, γ guides the client

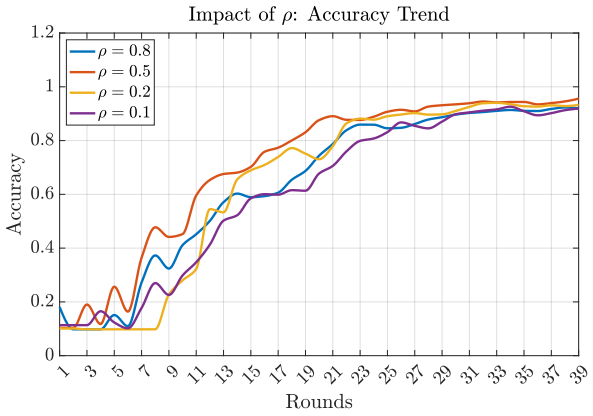


Fig. 3. Accuracy trend under different participation-ratio thresholds.

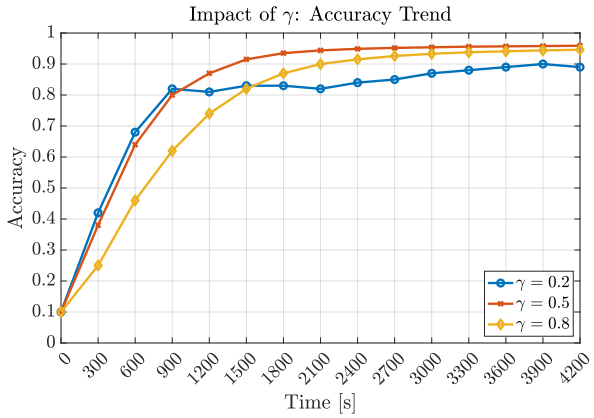


Fig. 4. Accuracy trend under different client-selection thresholds.

TABLE III
AVERAGE PERCENTAGE OF ACTIVE CLIENTS PER ROUND.

Approach	Avg. active clients [%]
HED-FL	100
LA-FL	57
iHFL	41

selection technique. A proper analysis of these parameters, is crucial to provide a comprehensive evaluation of iHFL. The impact of the participation-ratio threshold ρ is evaluated by testing four values, i.e., ($\rho \in \{0.1,0.2,0.5,0.8\}$). This parameter determines whether an edge server performs local aggregation or forwards the selected client updates upward, depending on the proportion of selected clients. As can be noted in Fig.3 and Tab.II, large values of ρ make the aggregation condition harder to satisfy, increasing the number of forwarding events and the overall execution time. Conversely, small values make the edge server more permissive, reducing switching overhead but increasing the risk of aggregating weak or poorly representative local updates. The results show that $\rho = 0.5$ achieves the best balance, preserving good accuracy while avoiding both excessive forwarding and unreliable local aggregations. Concerning the impact of γ, three values have been considered ($\gamma \in \{0.2,0.5,0.8\}$. As shown in Fig. 4, γ directly affects the accuracy trend by regulating how selective the edge server is when admitting clients to the local aggregation step. With $\gamma = 0.2$, the selection is more permissive and involves about (75%) of clients; however, the inclusion of less stable or weakly representative updates leads to a less regular accuracy evolution, requiring around 2100 s to reach 93.5% accuracy. Conversely, $\gamma = 0.8$ provides a stricter selection, reducing active clients to about 29%, but this overly selective behavior slows down convergence, requiring around 3000 s to reach the same accuracy. The best trade-off is obtained with $\gamma = 0.5$, which shows the most balanced accuracy trend in Fig. 4, selects about 52% of clients, and reaches 93.5% accuracy in approximately 1800 s. Therefore, $\gamma = 0.5$ and $\rho = 0.5$ are adopted in the remaining experiments.

iHFL Comparison Results: : The final experimental campaign compares iHFL against HED-FL and LA-FL to assess the benefits of the proposed predictive and multi-dimensional orchestration strategy. The results in Table III show that iHFL significantly reduces the average number of active clients per round, selecting only (41%) of participants, compared to (57%) for LA-FL and full participation for HED-FL. This confirms that the proposed client selection process is able to involve fewer, but more informative and reliable, participants without degrading the learning process. Fig.5 shows that reducing of the involved participating clients – excluding the *stragglers* and selecting the ones that show the best trade-off between utility and resource availability – leads the federated environment to a faster convergence time: the proposal allows the system to reach the maximum value of accuracy of *HED-FL* after 2000s of training, while *HED-FL* requires more than 4000 s. Putting it in terms of measured rounds, the proposal is capable of halving the amount of required rounds. A direct consequence of the reduced amount of required active clients, and faster convergence is the reduction of network overhead imposed on the communication channel. Indeed, iHFL reduces the generated network traffic by about (31.25%) with respect to HED-FL and by about (25.89%) with respect to LA-FL.

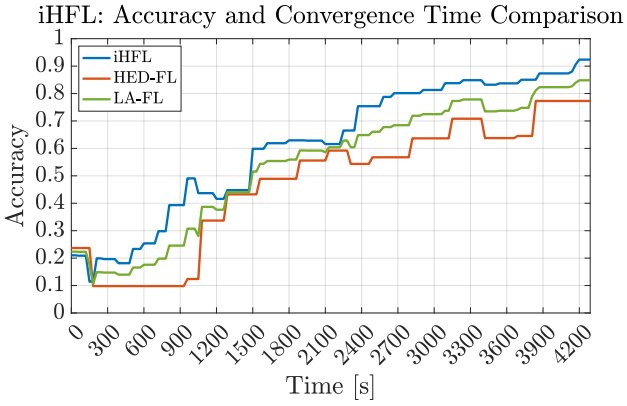


Fig. 5. Comparison of iHFL against state of the art approaches, HED-FL and LA-FL.

To assess the robustness of iHFL under heterogeneous and unreliable conditions, different client unavailability ratios

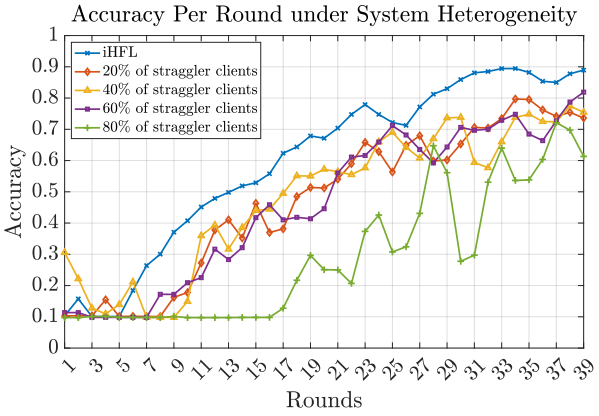


Fig. 6. Effectiveness of the effective iHFL edge-side orchestration and dynamic behavioral mode switching.

are considered, i.e., 20%, 40%, 60%, and 80%. This setting emulates scenarios in which clients may become slow or unavailable due to limited energy, reduced bandwidth, or constrained computational capabilities, thus delaying or weakening local aggregation. Under these conditions, each edge server is allowed to dynamically adapt its behavior: when the selected clients are sufficiently representative, local aggregation is performed; otherwise, the selected model updates are forwarded upward, enabling higher-layer entities to aggregate them with broader contributions. The comparison is conducted against HED-FL, which represents the most suitable baseline for this test since it also addresses hierarchy-level aggregation decisions, whereas LA-FL mainly focuses on client selection. As shown in Fig. 6, accuracy is better preserved by iHFL as the percentage of unavailable clients increases. In contrast, HED-FL remains agnostic to the actual number and status of participating clients and may perform local aggregation even when the available set is too limited. These results confirm the ability of iHFL to adapt to heterogeneous operating conditions while avoiding the propagation of weak intermediate models.

V. Conclusion

Hierarchical FL effectively improve the robustness and scalability of traditional FL. However, system heterogeneity of the clients that participate in the federation process constitutes a crucial obstacle to guarantee the benefits of HFL. To address this issue, this work proposes *iHFL*, an intelligent HFL orchestration framework that combines: *(i)* a structured metadata vector capturing the evolving statistical utility and resource availability of clients; *(ii)* an AI-based predictor that forecasts client behavior to support intelligent client selection and reduce the impact of statistically weak clients; and *(iii)* an adaptive edge-side operational mode logic that prevents the propagation of weak model updates when the number of statistically significant clients is limited. The proposed framework is evaluated under heterogeneous conditions against recent literature solutions, namely *HED-FL* [11] and *LA-FL* [12]. Experimental results demonstrate that *iHFL* reduces the number of active clients, accelerates convergence, and lowers communication overhead by approximately 31%, while preserving the global model accuracy. These results highlight the effectiveness of adaptive hierarchical coordination based on multi-dimensional client evaluation, participation control, and intelligent aggregation-level mechanisms.

Acknowledgments

This work was partially supported by Project PON SUNS - Secure Ubiquitous Networks and Services financed by MISE, (Azione 1.1.2) – CUP B29H26000360007. Period:2026-2028.

References

[1] M. A. Ferrag *et al.*, "Edge Learning for 6G-Enabled Internet of Things: A Comprehensive Survey of Vulnerabilities, Datasets, and Defenses," *IEEE Commun. Surv. Tutorials*, no. 4, pp. 2654–2713, Sep. 2023.

[2] F. Guo *et al.*, "Enabling Massive IoT Toward 6G: A Comprehensive Survey," *IEEE IoT J.*, vol. 8, no. 15, pp. 11 891–11 915, Mar. 2021.

[3] M. Tropea *et al.*, "Federated vs Transfer Learning Technique for Traffic Sign Recognition in Internet of Vehicles," in *2025 International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*. IEEE, 2025, pp. 27–31.

[4] W. Issa *et al.*, "Blockchain-Based Federated Learning for Securing Internet of Things: A Comprehensive Survey," *ACM Comput. Surv.*, vol. 55, no. 9, pp. 1–43, Jan. 2023.

[5] M. Le, T. Huynh-The, T. Do-Duy, T.-H. Vu, W.-J. Hwang, and Q.-V. Pham, "Applications of Distributed Machine Learning for the Internet-of-Things: A Comprehensive Survey," *IEEE Commun. Surv. Tutorials*, vol. 27, no. 2, pp. 1053–1100, Jul. 2024.

[6] E. Dritsas and M. Trigka, "Federated Learning for IoT: A Survey of Techniques, Challenges, and Applications," *Journal of Sensor and Actuator Networks*, vol. 14, no. 1, Jan. 2025.

[7] S. Ferretti *et al.*, "Decentralized coordination for resilient federated learning: A blockchain-based approach with smart contracts and decentralized storage," *Comput. Commun.*, vol. 236, p. 108112, Apr. 2025.

[8] F. Colosimo and F. De Rango, "Median-Krum: A Joint Distance-Statistical Based Byzantine-Robust Algorithm in Federated Learning," in *ACM Conferences*. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 61–68.

[9] S. S. Shahid *et al.*, "Hierarchical Federated Learning: Approaches, Applications, and Open Challenges," *IEEE Access*, vol. 13, pp. 190 333–190 353, Nov. 2025.

[10] A. Sacco *et al.*, "Dealing With Challenged IoT Networks in Hierarchical Federated Learning," *IEEE IoT J.*, vol. 12, no. 18, pp. 36 979–36 992, Jun. 2025.

[11] F. De Rango, A. Guerrieri, P. Raimondo, and G. Spezzano, "HED-FL: A hierarchical, energy efficient, and dynamic approach for edge Federated Learning," *Pervasive Mob. Comput.*, vol. 92, p. 101804, May 2023.

[12] J. Li, H. Wang, K. Cao, Y. Cui, T. Liu, and Z. Liu, "Latency-Aware Client Selection and Energy Management for Hierarchical Federated Learning," *IEEE IoT J.*, vol. 12, no. 24, pp. 52 964–52 978, Oct. 2025.

[13] F. P.-C. Lin *et al.*, "Delay-Aware Hierarchical Federated Learning," *IEEE Trans. Cognit. Commun. Networking*, vol. 10, no. 2, pp. 674–688, Nov. 2023.

[14] A. Ali and A. Arafa, "Delay Sensitive Hierarchical Federated Learning With Stochastic Local Updates," *IEEE Trans. Cognit. Commun. Networking*, vol. 11, no. 5, pp. 3412–3424, Jan. 2025.

[15] F. Colosimo, M. G. Spina, and F. De Rango, "FedAECS: a Federated Learning Adversary-aware and Efficient Client Selection," in *2025 IEEE 36th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, pp. 01–04.

[16] S. Trindade *et al.*, "Client Selection in Hierarchical Federated Learning," *IEEE IoT J.*, vol. 11, no. 17, pp. 28 480–28 495, May 2024.

[17] L. Li *et al.*, "Federated learning with workload-aware client scheduling in heterogeneous systems," *Neural Networks*, pp. 560–573, Oct. 2022.

[18] O. Manjang, Y. Zhai, J. Shen, A. Sarwar, X. He, and H. Wang, "Efficient Hierarchical Federated Learning With Pareto-Optimal Bi-Level Reinforcement Learning," *IEEE IoT J.*, vol. 12, no. 15, pp. 30 710–30 724, May 2025.

[19] F. Colosimo and F. De Rango, "Dynamic gradient filtering in federated learning with Byzantine failure robustness," *Future Gener. Comput. Syst.*, vol. 160, pp. 784–797, Nov. 2024.