HiMAC: Hierarchical Federated Learning with Mobility-Aware Collaboration

Yibo Pan
SKLCCSE
Beihang University
Beijing, China
pysoible@buaa.edu.cn

Boyi Liu
SKLCCSE
Beihang University
Beijing, China
boyliu@buaa.edu.cn

Zimu Zhou
Department of Data Science
City University of Hong Kong
Hong Kong, China
zimuzhou@cityu.edu.hk

Fan Gao
SKLCCSE
Beihang University
Beijing, China
fan.gao@buaa.edu.cn

Tianyi Wang
SKLCCSE
Beihang University
Beijing, China
wangty000@buaa.edu.cn

Yi Xu
SKLCCSE
Beihang University
Beijing, China
xuy@buaa.edu.cn

Yongxin Tong
SKLCCSE
Beihang University
Beijing, China
yxtong@buaa.edu.cn

Abstract

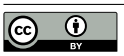
Mobility-aware hierarchical federated learning (MobiHFL) is an emerging paradigm to support urban applications where clients continuously move across edge-server coverage areas. Existing methods mainly treat mobility as a scheduling challenge and often train a single global model across all edge-servers. This overlooks the spatial non-IID characteristics of urban environments, where regions exhibit distinct yet correlated data distributions. Region-wise personalization is a natural remedy, but mobility makes it difficult. Each edge model is trained on its current client population, while future arriving and departing clients may follow different distributions, causing cold-start degradation and feature-classifier mismatch after handover. We propose HiMAC, a personalized MobiHFL framework that turns predicted client movement into proactive inter-edge collaboration. HiMAC uses per-client LSTM trajectory predictors to estimate future client flows and builds a round-level transition matrix to guide collaboration. With a decoupled backbone-classifier architecture, HiMAC prepares each edge for expected incoming clients through incoming-aware classifier initialization and improves compatibility with likely destination edges through outgoing-regularized backbone training. Extensive experiments on image classification and object detection benchmarks show improvements of up to 12.18 percentage points in image-classification accuracy and 6.50 points in mAP@50 over state-of-the-art baselines across varying mobility intensities.

CCS Concepts

• **Computing methodologies** → **Learning paradigms**.

Keywords

Personalized Federated Learning; Hierarchical Federated Learning; Mobility



CIKM '26, Rome, Italy.

ACM ISBN 979-8-4007-2539-5/2026/11
https://doi.org/10.1145/3799682.3840653

ACM Reference Format:
Yibo Pan, Boyi Liu, Zimu Zhou, Fan Gao, Tianyi Wang, Yi Xu, and Yongxin Tong. 2026. HiMAC: Hierarchical Federated Learning with Mobility-Aware Collaboration. In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management (CIKM '26), November 7–11, 2026, Rome, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3799682.3840653

1 Introduction

Mobile edge computing has made federated learning (FL) a natural paradigm for dynamic urban applications, including vehicular networks [5, 30, 44, 47], mobile sensing [27, 28], and urban traffic and perception [40, 43]. To reduce communication latency and align training with geographically distributed edge infrastructure, these systems often adopt hierarchical federated learning (HFL), where nearby mobile clients first coordinate with edge servers before cloud-level aggregation [5, 10]. In such settings, clients continuously move across edge-server coverage areas while collecting data, known as *mobility-aware HFL (MobiHFL)*. Compared with static FL, MobiHFL must handle unstable connectivity, limited dwell time, and changing local data distributions as clients move across regions [5, 30]. Accordingly, existing studies address mobility through client sampling [4, 44], dwell-time scheduling [3], trajectory-aware regional-model assignment and sampling [47], and device-edge-cloud collaboration [45].

Despite this progress, most MobiHFL methods optimize a *single global model*, while FedMG maintains multiple regional models and assigns them using predicted trajectories [47]. As illustrated in Fig. 1, regions can exhibit *distinct yet correlated* distributions because driving scenes and road environments vary geographically and urban clients have spatial dependencies [1, 24, 40]. This characteristic calls for *region-wise personalization*, where each edge maintains a model adapted to its local data characteristics instead of forcing all regions into one global model [35]. At the same time, correlations between nearby or frequently connected regions suggest that edges should collaborate selectively rather than train in isolation. However, existing MobiHFL methods [3, 4, 44, 45, 47] do not use predicted population flows to coordinate selective knowledge transfer among edge-specific models in both migration directions.

Personalized collaboration in MobiHFL is difficult because mobility makes each edge's data distribution nonstationary. An edge

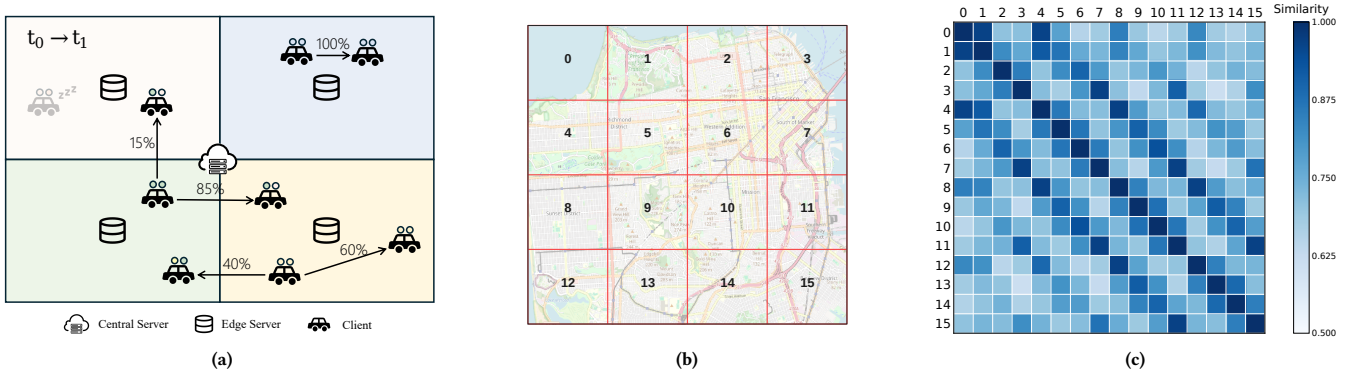


Figure 1: Mobility and spatial heterogeneity in MobiHFL. (a) Clients move across regions served by different edge servers. Their next regions can often be predicted from trajectory history. (b) San Francisco is partitioned into a 4×4 grid. Each cell represents an edge-server coverage area. (c) Pairwise similarity between region-level object-category distributions in BDD100K [43] shows that regions are distinct but correlated, motivating region-wise personalization with inter-region collaboration.

model is trained for the clients currently associated with that edge, but the clients that train or use the model in the next round may change. Clients migrating **into** an edge encounter a classifier calibrated to a population they did not help train, causing cold-start degradation. Clients migrating **out** will soon be served by destination classifiers shaped by different regional populations, causing feature-classifier incompatibility after handover. This **spatiotemporal distribution mismatch** makes collaboration reactive and quickly stale. Static collaboration-based personalized FL methods [11, 34, 38, 42], which assume fixed or slowly changing client-server associations, cannot maintain reliable collaboration graphs or cluster assignments under frequent migration.

We address this gap by exploiting the *predictability* of client mobility. Rather than treating mobility only as a source of instability, HiMAC uses trajectory regularities as proactive collaboration signals. Specifically, per-client LSTM trajectory predictors [14] estimate future edge associations, and their outputs are aggregated into a round-level stochastic transition matrix describing likely client flows between edges. This matrix induces a *dynamic* collaboration graph for mobility-aware personalization.

Built on this graph, we propose HiMAC, a *bidirectional mobility-aware personalized HFL* framework with a decoupled backbone-classifier architecture [2, 8]. The backbone captures shared representations and is coordinated through the cloud, while lightweight classifiers encode region-specific decision boundaries and are exchanged laterally among edges. This separation is essential to the bidirectional design: incoming adaptation can update region-specific decision boundaries without overwriting the shared representation, while outgoing adaptation can improve representation compatibility without homogenizing all regional classifiers. The transition matrix drives two complementary mechanisms. First, **incoming-aware classifier initialization** pre-aligns each edge classifier with the expected incoming client distribution, reducing cold-start degradation after client arrival. Second, **outgoing-regularized backbone training** aligns source-side features with likely destination edges, improving feature compatibility after client migration. Together, these mechanisms allow each edge to learn from where clients come from and prepare for where they will go.

The main contributions of this paper are as follows.

- We formulate personalized MobiHFL and identify the spatiotemporal distribution mismatch that arises when region-wise personalization meets client mobility.
- We propose HiMAC, a bidirectional personalized MobiHFL framework that converts LSTM-predicted client transitions into proactive inter-edge collaboration over a decoupled backbone-classifier architecture.
- We design two transition-driven mechanisms, incoming-aware classifier initialization and outgoing-regularized backbone training, to address cold-start degradation and feature-classifier incompatibility during handovers.
- Experiments on image classification and object detection benchmarks show that HiMAC improves image-classification accuracy by up to 12.18 percentage points and improves mAP@50 by 6.50 points under high-mobility object detection, with robust gains across mobility intensities and system scales.

2 Related Work

Mobility-aware Hierarchical Federated Learning. Standard FL [26, 41] lets clients train locally and send updates to a central server without sharing raw data. In mobile environments, this flat architecture is fragile because clients may have short dwell times, unstable connectivity, and updates computed under stale global states, known as the *model shuffling* effect [30]. Hierarchical FL (HFL) [5, 10] introduces edge servers as regional aggregators. By aggregating nearby clients before cloud coordination, HFL reduces communication latency, stabilizes local participation, and better matches the spatial organization of mobile edge infrastructure. Recent studies extend HFL to explicitly handle mobility. Theoretical analyses show that mobility can both help and hurt learning,

by mixing data across regions while also shortening communication windows and changing local distributions [5, 30]. System-oriented methods address dynamic participation through client sampling [4, 44], dwell-time scheduling [3], device-to-device aggregation [45], and random-walk collaboration [29]. Closer to our setting, FedMG maintains regional models and uses predicted trajectories for assignment and sampling [47]. Complementary work uses asynchrony to benchmark heterogeneous availability [20] or balance multimodal training [25], while early-exit methods address heterogeneous computation or personalized adaptive inference [22, 32].

Yet none converts population-level transition probabilities into a dynamic inter-edge graph that separately adapts classifiers for arrivals and representations for departures, as HiMAC does.

Personalized Federated Learning. Personalized FL (PFL) [35] addresses statistical heterogeneity by adapting models to clients, clusters, or regions. At the global-model level, distribution regularization reduces cross-client discrepancy on non-IID data [36]; PFL instead preserves client- or region-specific objectives. Representative methods decouple models into shared backbones and personalized heads [2, 8], personalize a global model through local fine-tuning or regularization [9, 19], or cluster clients according to data or model similarity [21, 34]. Graph-based methods infer collaboration structures for selective aggregation [23, 38, 42] or progressively prune collaboration links [6], while spatial FL methods exploit dependencies among geographically distributed clients or urban regions [40, 46].

Most PFL methods assume relatively stable client populations and server associations. Their personalized heads, clusters, or collaboration graphs are learned from current or historical similarity, which can become outdated when clients frequently migrate across edge regions. Under MobiHFL, the relevant collaborators for an edge depend not only on present similarity but also on which clients are likely to arrive and where current clients are likely to go. In contrast, HiMAC treats client transition probabilities as a dynamic collaboration signal and updates both classifier initialization and backbone training according to predicted mobility. While decoupled personalization, graph collaboration, MMD, and trajectory prediction are established, HiMAC couples them through a round-varying transition matrix that predicts the next population and drives upstream classifier transfer for arrivals and downstream representation alignment for departures.

3 Problem Statement

Mobility-aware Hierarchical Federated Learning. In MobiHFL, mobile clients move across regions, collect data from their current locations, train locally, and upload updates to nearby edge servers. Edge servers aggregate client updates within their coverage areas and periodically coordinate with a central cloud server.

Consider one cloud server, m edge servers, and n mobile clients. At round t, each client c collects local data D_c^t and is associated with one edge server i, forming the client set $\mathcal{E}_i^t$. Let $f_c(\theta; D_c^t)$ be the empirical loss of model θ on client c's data. The edge-level objective for edge i is:

$$F_i(\theta; \tilde{D}_i^t) = \sum_{c \in \mathcal{E}_i^t} \beta_{ic}^t \, f_c(\theta; D_c^t) \tag{1}$$

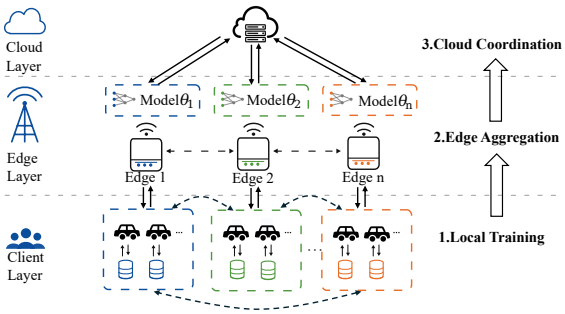


Figure 2: Standard personalized MobiHFL workflow.

where $\tilde{D}_i^t = \bigcup_{c \in \mathcal{E}_i^t} D_c^t$ is the local data pool at edge i, and $\beta_{ic}^t = |D_c^t| / \sum_{c \in \mathcal{E}_i^t} |D_c^t|$ is the client weight. The cloud server aggregates edge models to minimize the global objective:

$$F(\theta; D^t) = \sum_{i=1}^{m} \alpha_i^t \, F_i(\theta; \tilde{D}_i^t), \quad \alpha_i^t = \frac{|\tilde{D}_i^t|}{\sum_{j=1}^{m} |\tilde{D}_j^t|} \tag{2}$$

A defining characteristic of urban environments is *spatial non-IID*. Driving data distributions vary across regions with geographic context, road type, weather, and traffic conditions [1, 24]. As mobile clients collect data while moving across regions, such spatial heterogeneity propagates to edge-level updates [40, 47], creating gradient conflicts across edges and limiting the effectiveness of a single global model. This motivates region-wise personalization.

MobiHFL with Edge-wise Personalization. We extend personalized FL [35] to the edge level. Each edge server i maintains a dedicated model θ_i for its associated clients, replacing the shared global model θ with edge-specific parameters. Then the edge-level objective becomes:

$$F_i(\theta_i; \tilde{D}_i^t) = \sum_{c \in \mathcal{E}_i^t} \beta_{ic}^t \, f_c(\theta_i; D_c^t) \tag{3}$$

and the overall objective is

$$\min_{\{\theta_i\}} \sum_{i=1}^{m} \alpha_i^t \, F_i\left(\theta_i; \, \tilde{D}_i^t\right) \tag{4}$$

Without additional coupling, Eq. (4) reduces to independent local optimization at each edge. Hence, effective personalized MobiHFL requires collaboration that preserves regional specialization while sharing useful knowledge across related edges.

Fig. 2 illustrates a standard training round.

(1) *Local training.* Each client c trains on its local data D_c^t for E epochs and uploads the update to its associated edge server.
(2) *Edge aggregation.* Each edge server aggregates updates from $\mathcal{E}_i^t$ to update its edge model θ_i and exchanges information with neighboring edges.
(3) *Cloud coordination.* The cloud server collects parameters from all edges, performs personalized coordination to aggregate personalized models for all edge servers.

However, mobility makes solving Eq. (4) challenging. Each edge model is trained for the clients currently associated with that edge, while the clients that train or use the model in later rounds may

change. Clients migrating *into* an edge encounter a model calibrated to a population they did not help train. Clients migrating *out* will later be served by a destination edge whose model was shaped by another population. This *spatiotemporal distribution mismatch* turns the edge objectives $\{F_i(\theta_i; \tilde{D}_i^t)\}$ into moving targets. As client populations change, static collaboration strategies can quickly become obsolete [38, 42], while asynchrony-aware Co-PFL addresses model-age staleness rather than mobility-driven edge reassignment [23]. These challenges motivate mobility-aware collaboration that predicts client transitions before local training and adjusts inter-edge knowledge transfer accordingly.

Assumptions. We make three assumptions. *(i)* Each client is associated with exactly one edge server in each round and can communicate reliably with that edge. *(ii)* Clients are mobile while edge servers are stationary. *(iii)* Clients continuously collect data from their current regions, so their local data distributions change as they move. After migration, a client connects to the destination edge and uses that edge's current model. It does not carry a full source-edge model across regions.

4 Method

4.1 HiMAC Overview

Key Idea: Mobility-Conditioned Spatial Collaboration. Nearby regions often share feature geometry *e.g.* road structures, POI density, and travel modes, but differ in label distributions, *e.g.* traffic volume and congestion patterns. This motivates a *decoupled* model architecture. Each edge server i maintains $\theta_i = [\theta_i^s; \phi_i]$, where the backbone θ_i^s captures region-invariant representations and the classifier ϕ_i captures region-specific decision rules. For an input batch X, prediction is given by $f(X; \theta_i^s, \phi_i) = \phi_i(\theta_i^s(X))$.

We assume an edge-served handover protocol. After migration, a client connects to its destination edge and uses that edge's current model pair $[\theta_k^s; \phi_k]$, rather than carrying the full source-edge model across regions. Under this design, backbones are shared *vertically* through the cloud to maintain stable global representations, while classifiers are exchanged *laterally* across edges to exploit regional similarity without sacrificing local specialization. This separation also reduces handover overhead, since lateral exchanges involve only classifier heads, which account for about 5-10% of the full model in our evaluated architectures, together with compact feature prototypes used for outgoing regularization. Decoupling assigns the two mobility mismatches to different components: incoming collaboration changes the regional boundary through ϕ_i, whereas outgoing regularization changes feature geometry through θ_i^s; a monolithic exchange would entangle both operations and risk losing local specialization.

Principle: Mobility Matters. As discussed in Sec. 3, client mobility induces a *spatiotemporal distribution mismatch*. Under the decoupled architecture, this mismatch appears in two forms:

- **Incoming distributional mismatch.** Clients arriving at edge i reflect the distributions of their origin regions. The destination classifier ϕ_i, calibrated to the current local population, may be misaligned with the incoming one, leading to cold-start degradation.
- **Outgoing representational incompatibility.** Clients leaving edge i will later be served by a destination classifier ϕ_k. If the source backbone θ_i^s produces features that are poorly matched to ϕ_k, prediction quality degrades after handover.

Static collaboration strategies [38, 42] cannot address these issues because inter-edge similarity itself evolves with client movement. HiMAC instead grounds collaboration in a transition matrix $\mathcal{P} \in \mathbb{R}^{m \times m}$, where $\mathcal{P}_{j \to i}^{(t)}$ denotes the probability that clients move from edge j to edge i in round t. This matrix provides a dynamic basis for bidirectional adaptation.

Mobility-Aware Round Workflow. HiMAC restructures the *reactive* MobiHFL workflow in Sec. 3 by moving mobility prediction and classifier preparation *before* local training, as illustrated in Fig. 3. Each round consists of four stages:

1. **Mobility Prediction and Transition Matrix Construction** (Sec. 4.2). Each client runs a personalized LSTM to predict its next edge association, and the server aggregates these predictions into the transition matrix $\mathcal{P}^{(t)}$.
2. **Incoming-Aware Classifier Initialization** (Sec. 4.3). Before local training, each edge i initializes its classifier by aggregating classifiers from top-K upstream neighbors selected according to incoming transition probabilities $\mathcal{P}_{j \to i}^{(t)}$.
3. **Local Training with Outgoing Regularization** (Sec. 4.4). Each edge trains on its local client data while regularizing backbone features toward downstream feature distributions, weighted by outgoing probabilities $\mathcal{P}_{i \to k}^{(t)}$.
4. **Edge/Cloud Backbone Aggregation** (Sec. 4.5). After local training, edges upload updated backbones to the cloud, which aggregates the next-round global backbone while leaving classifiers local.

This bidirectional design reflects the causal role of mobility. Incoming collaboration accounts for *where clients come from*, while outgoing regularization anticipates *where they will go*.

4.2 Transition Matrix Generation

Principle. The transition matrix $\mathcal{P}^{(t)} \in \mathbb{R}^{m \times m}$ provides the mobility signal to guide HiMAC's collaboration. Unlike static similarity metrics, *e.g.* gradient cosine similarity or distribution overlap, $\mathcal{P}^{(t)}$ captures how client populations are expected to move between edges. A large entry $\mathcal{P}_{j \to i}^{(t)}$ indicates that many clients currently at edge j are likely to appear at edge i in the next round, making edge j a relevant source of knowledge for edge i. This allows HiMAC to prepare for distribution shifts before they occur.

Personalized Trajectory Prediction. Each client c maintains a personalized LSTM predictor $\mathcal{M}_c$ trained on its own mobility history. Given a trajectory sequence $\mathcal{T}_c(t) = \{p_c(t-(L-1)\Delta t), \ldots, p_c(t)\}$ of length L, the predictor estimates the next position as

$$\hat{p}_c(t + \Delta t) = \mathcal{M}_c(\mathcal{T}_c(t)). \tag{5}$$

We use personalized predictors because client mobility often follows individual routines, such as commuting patterns and repeated visits, which are difficult to capture with a single shared predictor. Raw trajectories remain local; clients share only predicted destination-region indices and use a self-transition until L observations are available.

Matrix Construction. The predicted positions are mapped to edge-region indices by $\text{map}(\cdot)$ and aggregated into a frequency matrix $\mathbf{F} \in \mathbb{N}_0^{m \times m}$. Each entry $\mathbf{F}_{jk}$ counts the number of clients predicted to

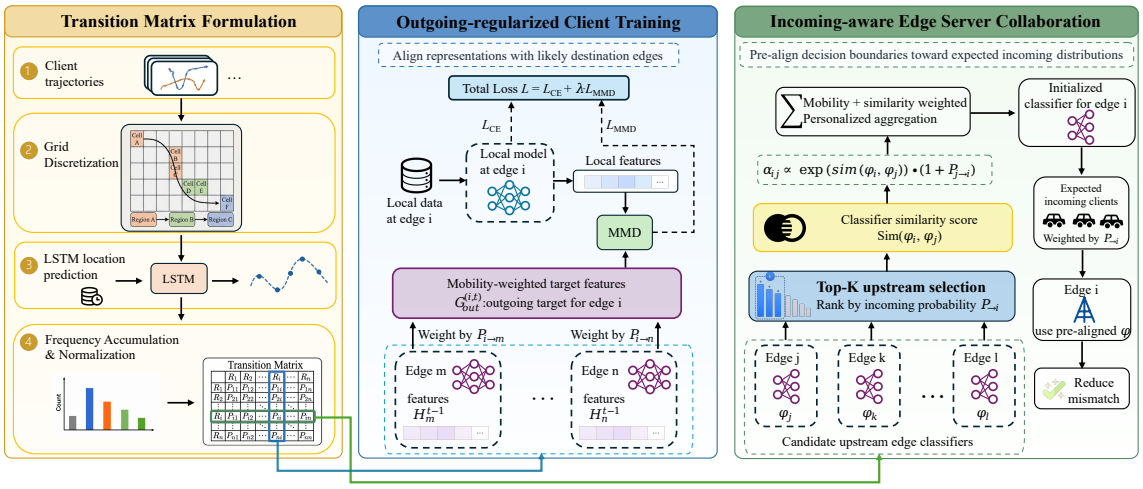


Figure 3: HiMAC framework overview. It follows four stages: transition matrix construction, incoming-aware classifier initialization, outgoing-regularized local training, and edge/cloud backbone aggregation.

Algorithm 1: Transition Matrix Construction

Input: Client trajectories $\{p_c(t)\}_{c=1}^{n}$, LSTM models $\{\mathcal{M}_c\}_{c=1}^{n}$, prediction horizon Δt
Output: Stochastic transition matrix $\mathcal{P}(t) \in \mathbb{R}^{m\times m}$

Initialize frequency matrix $\mathbf{F} \leftarrow \mathbf{0}_{m\times m}$;
for *each client* c **do**
  Extract trajectory sequence $\mathcal{T}_c \leftarrow \{p_c(t - k\Delta t)\}_{k=0}^{L-1}$;
  Predict future position $\hat{p}_c(t + \Delta t) \leftarrow \mathcal{M}_c(\mathcal{T}_c)$;
  $j \leftarrow \text{map}(p_c(t))$; // Current region via grid mapping
  $k \leftarrow \text{map}(\hat{p}_c(t + \Delta t))$; // Predicted region
  if $0 \le j < m$ ***and*** $0 \le k < m$ **then**
    $\mathbf{F}_{jk} \leftarrow \mathbf{F}_{jk} + 1$; // Count valid transition
for *each region* $j = 0, \ldots, m-1$ **do**
  $S_j \leftarrow \sum_{l=0}^{m-1} \mathbf{F}_{jl}$;
  if $S_j > 0$ **then**
    $\mathcal{P}_{j\cdot}(t) \leftarrow \mathbf{F}_{j\cdot}/S_j$; // Row-wise normalization
  else
    $\mathcal{P}_{jj}(t) \leftarrow 1$; // Identity fallback for empty rows
return $\mathcal{P}(t)$ *with* $\mathcal{P}(t)\mathbf{1} = \mathbf{1}$ *and* $\mathcal{P}(t) \ge \mathbf{0}$

move from edge j to edge k. We obtain a row-stochastic transition matrix by normalizing each row:

$$\mathcal{P}_{j\to k}(t) = \frac{\mathbf{F}_{jk}}{\sum_{l=0}^{m-1} \mathbf{F}_{jl}}, \quad \forall\, j,k \in \{0,\ldots,m-1\} \tag{6}$$

If no outgoing transition is observed for edge j, we set $\mathcal{P}_{j\to j}^{(t)} = 1$ and all other entries in row j to zero. The matrix is refreshed once per communication round with a prediction horizon Δt aligned with the communication round, keeping the collaboration graph current while avoiding overly noisy updates.

4.3 Incoming-Aware Classifier Initialization

Principle. At the beginning of round t, edge i receives the latest global backbone $\theta_{global}^{(t)}$ and keeps its local classifier $\phi_i^{(t)}$. Before local training, the classifier should be prepared for clients that are likely to arrive. The incoming transition probability $\mathcal{P}_{j\to i}^{(t)}$ provides this signal. If many clients are predicted to move from edge j to edge i, then edge j's classifier $\phi_j^{(t)}$ provides a useful prior for adapting $\phi_i^{(t)}$ to the expected incoming distribution.

Incoming Collaboration. Edge i first selects the top-K upstream neighbors with the largest incoming probabilities:

$$\mathcal{N}_{in}^{(i)} = \text{argtop-K}_j\left\{\mathcal{P}_{j\to i}^{(t)}\right\} \tag{7}$$

Mobility determines which sources are relevant, while model similarity controls how much each source should influence the classifier. Specifically, edge i initializes a temporary classifier by

$$\tilde{\phi}_i^{(t)} \leftarrow \sum_{j\in\mathcal{N}_{in}^{(i)}\cup\{i\}} \frac{\gamma_j\, e^{\text{Sim}(\phi_i^{(t)},\phi_j^{(t)})}\,(1+\mathcal{P}_{j\to i}^{(t)})}{\sum_{k\in\mathcal{N}_{in}^{(i)}\cup\{i\}} \gamma_k\, e^{\text{Sim}(\phi_i^{(t)},\phi_k^{(t)})}\,(1+\mathcal{P}_{k\to i}^{(t)})}\,\phi_j^{(t)} \tag{8}$$

where $\tilde{\phi}_i^{(t)}$ is the temporary classifier at edge i before local training, and $\text{Sim}(\cdot,\cdot)$ is cosine similarity. The scaling factor γ_j is set to $\gamma_i = \gamma > 1$ for the self-node and $\gamma_j = 1$ for all neighbors. Thus, the initialization favors edges with high incoming mobility and compatible classifiers, while the amplified self-weight preserves edge i's local specialization and reduces negative transfer.

4.4 Outgoing-Regularized Local Training

Principle. After incoming-aware initialization, edge i trains on its current client data. At the same time, it should prepare clients that are likely to migrate to downstream edges. Under the edge-served handover protocol, a migrating client does not carry the source-edge model. After migration, it uses the destination edge's classifier on top of the globally coordinated backbone. Hence, the key challenge

is *representational compatibility*. The backbone updated at edge i should produce features that remain compatible with the classifiers of likely destination edges. To this end, HiMAC regularizes local backbone training using feature prototypes from downstream edges, weighted by outgoing transition probabilities $\mathcal{P}^{(t)}_{i\to k}$.

Outgoing-Regularized Objective. Let $\theta^s_i(X)$ be the feature distribution produced by edge i's backbone. Each edge k maintains a compact prototype buffer $\mathcal{G}^{(t)}_k$, which summarizes representative feature embeddings from its local clients. Each round, the edge refreshes this bounded buffer from its latest local features and shares at most Bd_h scalars with selected downstream neighbors, where B is the fixed capacity and d_h is the feature dimension. For edge i, we define the downstream neighbor set as $\mathcal{N}^{(i)}_{out} = \{k : \mathcal{P}^{(t)}_{i\to k} > 0\}$. Then we construct the outgoing target distribution by a mobility-weighted mixture of downstream prototypes:

$$\mathcal{G}^{(i),(t)}_{out} = \sum_{k\in\mathcal{N}^{(i)}_{out}} \mathcal{P}^{(t)}_{i\to k}\,\mathcal{G}^{(t)}_k \tag{9}$$

Edge i then minimizes

$$\min_{\theta^s_i,\phi_i} \underbrace{\mathcal{L}_{CE}\left(\tilde{D}^t_i;\,\theta^s_i,\phi_i\right)}_{\text{local task loss}} + \underbrace{\lambda\,\mathrm{MMD}^2\left(\theta^s_i(X),\;\mathcal{G}^{(i),(t)}_{out}\right)}_{\text{outgoing feature regularization}} \tag{10}$$

where $\mathcal{L}_{\mathrm{CE}}$ is the local task loss, λ controls the strength of outgoing regularization, and $\mathrm{MMD}^2(\cdot,\cdot)$ is the squared Maximum Mean Discrepancy [12] under an RBF kernel. The regularization term aligns source-side features with the feature distributions expected by likely destination edges, with larger transition probabilities imposing stronger alignment.

4.5 Edge/Cloud Backbone Aggregation

Principle. After outgoing-regularized local training, each edge uploads only its updated backbone to the cloud. Classifiers remain local to preserve region-specific decision boundaries. The cloud aggregates active edge backbones by sample-weighted averaging:

$$\theta^{(t+1)}_{global} = \sum_{i=1}^{m} \frac{n_i}{n_{total}}\,\theta^{s,(t+1)}_i \tag{11}$$

where $n_i = |\tilde{D}^t_i|$ is the local sample count at edge i. The resulting backbone $\theta^{(t+1)}_{global}$ initializes all edge backbones in the next round. This maintains a shared representation space across the federation, while leaving region-specific adaptation to local classifiers.

4.6 Discussions

Why Bidirectional Collaboration. Algorithm 2 summarizes the procedure of HiMAC. Incoming-aware initialization and outgoing-regularized training address complementary sides of mobility-induced mismatch. Using only incoming collaboration would prepare an edge for arriving clients but ignore the clients that are about to leave. Using only outgoing regularization would improve handover compatibility but fail to adapt the classifier to the next incoming population. By coupling both directions, HiMAC forms a mobility-conditioned consistency cycle, where each edge absorbs knowledge from likely source regions and aligns its representations with likely destination regions.

Algorithm 2: Mobility-Aware Hierarchical FL (round t)

Input: Client trajectories, LSTM predictors, global backbone $\theta^{(t)}_{global}$, neighbor budget K, self-factor γ, reg weight λ

Output: Next-round global backbone $\theta^{(t+1)}_{global}$; local classifiers $\{\phi^{(t+1)}_i\}$

; /* Mobility prediction */

Construct $\mathcal{P}^{(t)}$ using Algorithm 1;

; /* Server broadcast */

Broadcast $\theta^{(t)}_{global}$ and $\mathcal{P}^{(t)}$ to sampled edges;

for *each sampled edge* i **do**

  // Incoming classifier initialization

  Set $\theta^s_i \leftarrow \theta^{(t)}_{global}$; set $\phi_i \leftarrow \phi^{(t)}_i$;

  Select $\mathcal{N}^{(i)}_{in} \leftarrow$ top-$K(\mathcal{P}^{(t)}_{j\to i})$;

  $\phi_i \leftarrow \sum_{j\in\mathcal{N}^{(i)}_{in}\cup\{i\}} \frac{\gamma_j\, e^{\mathrm{Sim}(\phi^{(t)}_i,\phi^{(t)}_j)}(1+\mathcal{P}^{(t)}_{j\to i})}{\sum_{k\in\mathcal{N}^{(i)}_{in}\cup\{i\}}\gamma_k\, e^{\mathrm{Sim}(\phi^{(t)}_i,\phi^{(t)}_k)}(1+\mathcal{P}^{(t)}_{k\to i})}\,\phi^{(t)}_j$;

  // Local training with outgoing regularization

  for $e = 1$ **to** E **do**

    Sample minibatch (X, y) on clients of edge i;

    $\mathcal{L}_{task} \leftarrow \mathcal{L}_{CE}(f(X;\theta^s_i,\phi_i), y)$;

    $\mathcal{G}^{(i),(t)}_{out} \leftarrow \sum_k \mathcal{P}^{(t)}_{i\to k}\,\mathcal{G}^{(t)}_k$; // aggregate downstream features

    $\mathcal{L}_{out} \leftarrow \lambda\,\mathrm{MMD}^2\left(\theta^s_i(X),\;\mathcal{G}^{(i),(t)}_{out}\right)$; // feature alignment

    Update (θ^s_i,ϕ_i) by SGD on $\mathcal{L}_{task} + \mathcal{L}_{out}$;

  Set $\theta^{s,(t+1)}_i \leftarrow \theta^s_i$; set $\phi^{(t+1)}_i \leftarrow \phi_i$;

  Upload $\theta^{s,(t+1)}_i$ to server; keep $\phi^{(t+1)}_i$ local;

; /* Cloud coordination */

$\theta^{(t+1)}_{global} \leftarrow \sum_{i=1}^{m} \frac{n_i}{n_{total}}\,\theta^{s,(t+1)}_i$;

This design also avoids indiscriminate full-mesh collaboration. Naively averaging classifiers across all edges would homogenize region-specific decision boundaries and weaken personalization. In contrast, HiMAC builds a sparse collaboration graph from transition probabilities and model similarity. Incoming initialization implicitly aligns an edge classifier with likely upstream classifiers, while outgoing regularization explicitly constrains backbone features toward likely downstream feature distributions. Together, they encode a lightweight bidirectional handover protocol within model initialization and local training, without requiring full-model synchronization across edges.

Communication Cost. HiMAC confines lateral inter-edge exchange to classifier heads and compact feature prototypes. Since classifier heads account for only about 5-10% of the full model in our evaluated architectures, each neighbor exchange is substantially smaller than full-model transfer. The overall communication cost

still depends on the model architecture, edge topology, and neighbor budget K, because cloud-edge backbone aggregation remains unchanged and each edge exchanges information with its selected neighbors. The top-K transition graph further bounds lateral overhead by keeping collaboration sparse. With K neighbors, lateral payload is bounded by $K(d_\phi + Bd_h)$ scalars per refresh; destination indices add scalar metadata for constructing the $m \times m$ transition matrix.

Computational Cost. Incoming-aware initialization requires computing similarities between an edge classifier and its top-K candidate upstream classifiers, with cost $O(Kd_\phi)$, where d_ϕ is the classifier parameter dimension. Outgoing regularization adds the cost of computing the MMD penalty between mini-batch features and downstream feature prototypes. If d_h is the feature dimension and K downstream prototype buffers are used, this adds an extra $O(Kd_h)$ alignment cost per update, up to kernel-evaluation constants. Since K is small and both classifier heads and prototype buffers are lightweight, the added overhead is modest compared with local training. Measured wall-clock time and client-side energy remain outside the current evaluation.

5 Experiments

5.1 Experimental Setup

Environment. All experiments are implemented in PyTorch and run on a server with dual Intel Xeon Gold CPUs and NVIDIA V100 GPUs with 40GB memory.

Datasets and Models. We evaluate HiMAC on image classification and object detection tasks. For image classification, we use MNIST [18], EMNIST [7], CIFAR-10, and CIFAR-100 [17]. To simulate spatial heterogeneity, we partition each dataset into region-level subsets using Dirichlet non-IID sampling [15], with $\alpha = 1.0$ for moderate non-IID and $\alpha = 0.1$ for strong non-IID. For object detection, we use BDD100K [43] with YOLOv5s [16], and partition samples by geographic region using the location metadata.

After partitioning, each client draws data only from its current region. We use a three-layer CNN for MNIST, EMNIST, and CIFAR-10, ResNet-18 [13] for CIFAR-100, and YOLOv5s for BDD100K.

Mobility Simulation. We simulate client mobility using Cabspotting [31], which contains 536 real-world taxi trajectories in San Francisco. The city is divided into a 4×4 grid over a $40\,\text{km} \times 40\,\text{km}$ area, where each grid cell corresponds to one edge region. Each client trains a personalized LSTM [14] with 128 hidden units on its trajectory history to predict its position one hour ahead.

Baselines. We compare HiMAC with four mobility-aware HFL methods: Mob-HierFAVG [5], MACH [44], FedMG [47], and MIDDLE [45]. We also adapt representative collaboration-based personalized FL methods to the hierarchical setting, including IFCA [11], ICFL [39], pFedGraph [42], FedSaC [38], and FedSoft [33]. For hierarchical adaptation, each baseline applies its original clustering, graph, or personalized aggregation rule over edge-level models before cloud coordination. All methods share the task model, data, trajectories, rounds, local epochs, batch size, and learning rate unless the original algorithm requires otherwise; Co-PFL baselines receive no future mobility. Because MobiHFL baselines do not all decouple personalization, we use Co-PFL comparisons and ablations to help distinguish personalization from transition-conditioned collaboration.

Configuration and Metrics. Unless otherwise specified, we use $m = 16$ edge servers, $n = 100$ mobile clients, $T = 300$ communication rounds, $E = 5$ local epochs per round, batch size 32, and learning rate 0.01. For HiMAC, we set the outgoing regularization weight to $\lambda = 0.05$, the neighbor budget to $K = 5$, and the self-weight factor to $\gamma = 3$.

For image classification, we report edge-local test accuracy and convergence over communication rounds. For object detection, we report mAP@50. All classification entries are means over five independent runs, with standard deviations reported in Table 1.

5.2 Main Results

5.2.1 Accuracy Improvements. Table 1 reports classification accuracy under moderate and strong spatial non-IID settings. HiMAC achieves the best or near-best performance across MNIST, EMNIST, CIFAR-10, and CIFAR-100. The gains are most notable on more challenging datasets. For example, on CIFAR-10 Dir(0.1), HiMAC reaches 82.82% accuracy, outperforming the strongest baseline by 12.18 percentage points. On CIFAR-100, HiMAC also improves over the best baseline by up to 1.89 points. These results show that mobility-aware personalization is especially beneficial when regional distributions are highly skewed and visually complex.

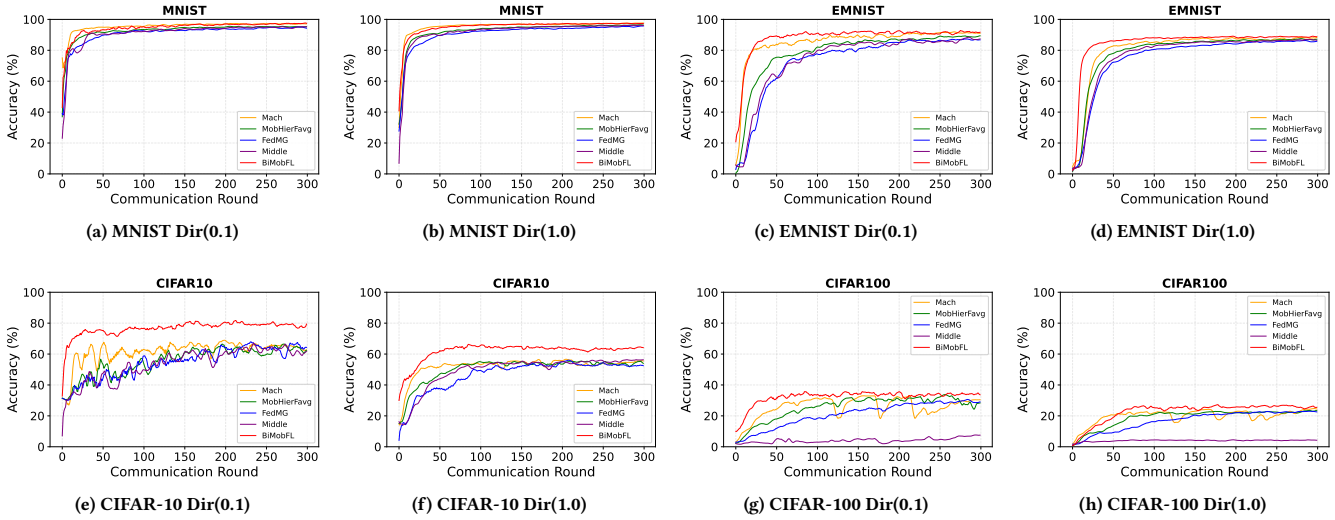
- *Comparison with MobiHFL Baselines.* Mobility-aware HFL baselines (*e.g.* FedMG and MACH) improve training stability under mobile participation by optimizing sampling, scheduling, or aggregation. However, they still mainly pursue a shared regional or global model, which limits their ability to fit region-specific distributions. For example, the strongest MobiHFL baseline on CIFAR-10 Dir(0.1), FedMG, achieves 70.64%, while HiMAC reaches 82.82%. The improvement comes from maintaining edge-specific classifiers for local specialization and using predicted mobility only to guide selective collaboration.
- *Comparison with Co-PFL Baselines.* Collaboration-based PFL baselines (*e.g.* FedSoft and pFedGraph) handle statistical heterogeneity through personalized models, clusters, or collaboration graphs. Their limitation is that these structures are built from current or historical similarity and become stale when clients frequently migrate across edges. For instance, pFedGraph falls behind HiMAC by 12.30 percentage points on CIFAR-10 Dir(0.1). HiMAC avoids this mismatch by rebuilding the collaboration topology every round from predicted client transitions, allowing edge models to adapt before the next client population arrives.

5.2.2 Convergence Curves. Fig. 4 shows the test accuracy over communication rounds. Across most settings, HiMAC converges faster in the early rounds and reaches a higher final accuracy. The advantage is especially clear on CIFAR-10 and CIFAR-100, where stronger visual complexity and non-IID partitions create greater regional specialization.

The faster convergence is enabled by the two proactive mechanisms. Incoming-aware initialization gives each edge a better classifier before training with newly arriving clients, while outgoing

Table 1: Test accuracy (%) across datasets and non-IID settings. Means and standard deviations reported over 5 runs.

Type	Methods	MNIST (%)		EMNIST (%)		CIFAR-10 (%)		CIFAR-100 (%)	
		$Dir(0.1)$	$Dir(1.0)$	$Dir(0.1)$	$Dir(1.0)$	$Dir(0.1)$	$Dir(1.0)$	$Dir(0.1)$	$Dir(1.0)$
MobiHFL	Mob-HierFAVG [5]	95.74±0.00	96.63±0.03	89.92±0.14	87.67±0.01	69.46±1.50	56.52±0.69	33.49±0.83	24.08±0.19
	MACH [44]	97.57±0.05	97.64±0.08	92.14±0.02	86.18±2.92	70.35±0.42	57.37±0.27	33.92±0.31	25.16±0.27
	FedMG [47]	95.34±0.11	95.56±0.22	87.63±0.06	86.42±0.08	70.64±1.63	55.89±0.52	30.46±0.49	23.24±0.00
	MIDDLE [45]	95.35±0.03	96.33±0.05	88.93±0.05	87.39±0.05	68.03±0.20	56.60±0.23	7.21±0.60	5.37±0.69
Co-PFL	IFCA [11]	91.08±2.98	95.5±0.34	21.29±6.03	6.25±0.17	37.40±0.74	25.55±0.32	6.22±0.28	4.02±0.02
	ICFL [39]	96.69±0.25	93.58±3.26	91.35±0.52	88.73±0.15	70.37±0.49	58.95±0.85	23.07±11.24	25.69±0.32
	pFedGraph [42]	95.69±0.06	96.67±0.04	89.95±0.23	87.89±0.00	70.52±0.35	56.73±0.18	32.62±0.44	23.99±0.24
	FedSaC [38]	88.59±0.27	89.45±0.54	80.39±1.79	78.55±0.34	57.98±1.31	51.83±0.83	29.34±0.35	16.95±0.41
	FedSoft [33]	96.74±0.02	93.79±3.20	90.93±0.13	88.27±0.15	69.55±1.21	57.90±0.58	37.28±0.01	19.17±0.12
Ours	HiMAC	**98.03±0.15**	**97.70±0.02**	**92.85±0.82**	**89.52±0.13**	**82.82±0.16**	**67.49±0.63**	**38.80±0.23**	**27.58±0.03**



(a) MNIST Dir(0.1) (b) MNIST Dir(1.0) (c) EMNIST Dir(0.1) (d) EMNIST Dir(1.0)

(e) CIFAR-10 Dir(0.1) (f) CIFAR-10 Dir(1.0) (g) CIFAR-100 Dir(0.1) (h) CIFAR-100 Dir(1.0)

Figure 4: Convergence curves across different datasets and non-IID settings. HiMAC generally converges faster and reaches higher final accuracy than the baselines.

regularization reduces feature mismatch for clients that later migrate to other edges. Together, they reduce the need for reactive correction after mobility-induced distribution shifts occur.

5.3 Micro-Benchmarks

5.3.1 Component-Wise Ablation. We first remove key components from HiMAC to quantify their contributions. Table 2 reports classification accuracy under the strong non-IID setting. "w/o Collaboration" disables incoming initialization, "w/o Regularization" sets $\lambda = 0$, and "w/o Mobility Prediction" replaces predicted transitions with a static, uniformly weighted spatial-neighbor graph while retaining the other operations.

The full model performs best across all datasets. Removing mobility prediction causes the largest degradation, especially on EMNIST and CIFAR-100, where accuracy drops by 7.87 and 6.37 points. Removing incoming collaboration or outgoing regularization also hurts performance, with clearer losses on EMNIST and CIFAR-10.

Table 2: Ablation study across datasets under strong non-IID settings. Accuracy (%) reported.

Dataset	MNIST	EMNIST	CIFAR-10	CIFAR-100
HiMAC (Full)	98.33	91.97	82.98	28.66
w/o Collaboration	97.46	88.72	81.04	27.89
w/o Regularization	97.60	87.78	81.28	28.18
w/o Mobility Prediction	97.27	84.10	78.88	22.29

These results confirm that the three components play complementary roles. Mobility prediction provides the routing signal for targeted collaboration. Incoming classifier collaboration prepares each edge for expected arriving clients. Outgoing regularization improves representation compatibility with likely destination edges. Removing any component weakens the ability to handle spatiotemporal distribution mismatch.

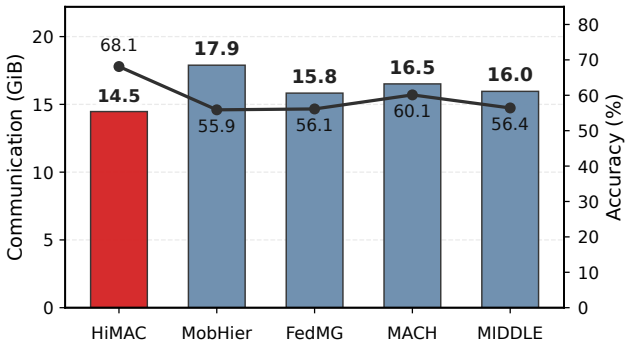


Figure 5: Estimated communication payload and accuracy on CIFAR-10 Dir(1.0). Payloads are counted using method-aware transmission patterns: full-model HFL for Mob-HierFAVG, sampling/scheduling-adjusted full-model traffic for FedMG and MACH, amortized edge-cloud synchronization for MIDDLE, and decoupled backbone/classifier communication for HiMAC. Both axes start from zero.

5.3.2 Communication Overhead. We further estimate the communication payload on CIFAR-10 Dir(1.0) to check whether the mobility-aware gains require heavy network traffic. The accounting covers 300 rounds and transmitted model parameters; scalar scheduling metadata and destination indices are excluded from the plot. We use method-aware accounting rather than assigning the same payload to all baselines. Mob-HierFAVG follows standard full-model HFL. FedMG and MACH still transmit full models, but their mobility/resource-aware scheduling and device sampling reduce the effective number of client-edge transfers. MIDDLE amortizes edge-cloud synchronization by its cloud interval, while its device-edge traffic remains full-model. For HiMAC, we count backbone synchronization and classifier-head exchange; thus, the figure reports model-parameter payload rather than complete traffic, with prototypes adding at most KBd_h scalars per edge refresh.

Fig. 5 shows that HiMAC uses 14.47 GiB. The baselines use 17.89 GiB (Mob-HierFAVG), 16.51 GiB (MACH), 15.96 GiB (MIDDLE), and 15.83 GiB (FedMG). The reduction over standard full-model HFL is 19.1%. This gap is intentionally moderate because the backbone still needs to be synchronized. The saving mainly comes from using compact classifier heads for mobility-driven inter-edge collaboration. At the same time, HiMAC reaches 68.12% accuracy, compared with 60.11% for MACH, 56.42% for MIDDLE, 56.14% for FedMG, and 55.89% for Mob-HierFAVG.

5.3.3 Impact of Mobility Intensity. We next evaluate robustness as client migration becomes more frequent. Table 3 reports CIFAR-10 Dir(0.1) accuracy under different mobility intensities, measured by transitions per client per 100 rounds.

All methods degrade as mobility becomes stronger, but HiMAC remains consistently ahead. Under high mobility, HiMAC reaches 74.88%, outperforming the strongest baseline by 8.25 points. Under very high mobility, it still maintains 72.47%, while the best baseline drops to 62.88%. Higher mobility causes more frequent client-edge reassignment, making static or reactive aggregation less reliable.

Table 3: Impact of mobility intensity on CIFAR-10 Dir(0.1). Migration frequency measured as transitions per client per 100 rounds.

Method	Low (5)	Medium (15)	High (30)	Very High (50)
Mob-HierFAVG [5]	65.41	63.87	63.93	62.35
FedMG [47]	63.08	60.41	62.09	60.18
MIDDLE [45]	65.29	64.57	63.67	62.88
MACH [44]	70.29	68.08	66.63	58.40
HiMAC	**80.28**	**77.61**	**74.88**	**72.47**

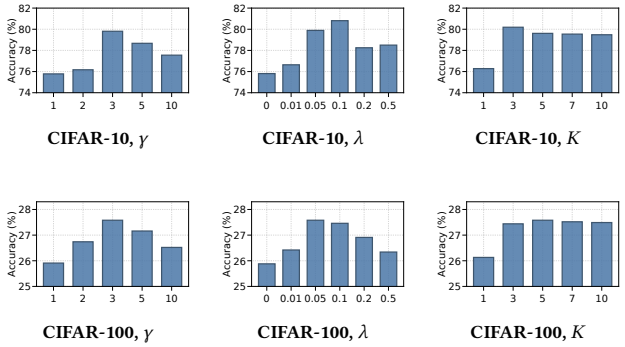


Figure 6: Hyperparameter sensitivity analysis on CIFAR-10 Dir(0.1) and CIFAR-100 Dir(1.0). Each panel label indicates the varied hyperparameter, the x-axis lists its candidate values, and the y-axis reports test accuracy (%). Moderate values perform best: $\gamma = 3$ and λ around 0.05–0.1, while accuracy becomes similar once K reaches 5.

HiMAC anticipates these changes through the transition matrix, so classifiers and representations are adjusted before the next client population arrives.

5.3.4 Hyperparameter Sensitivity. We study the sensitivity of HiMAC to the self-weight factor γ, outgoing regularization weight λ, and neighbor budget K. Fig. 6 reports results on CIFAR-10 Dir(0.1) and CIFAR-100 Dir(1.0).

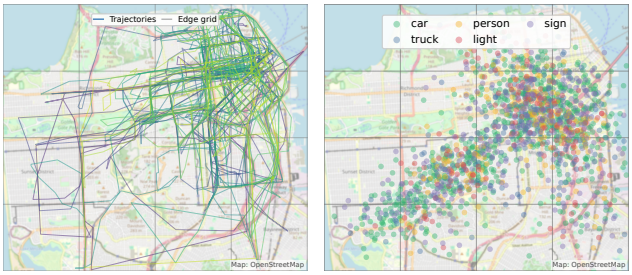
Moderate collaboration is preferable in both settings. Accuracy peaks at $\gamma = 3$, improves as λ increases from zero to a moderate range, and remains stable once K reaches 5. This reflects the trade-off between local specialization and knowledge transfer. Insufficient collaboration leaves edges unprepared for incoming clients, while excessive collaboration may introduce negative transfer from weakly related regions. A moderate self-weight and sparse top-K neighborhood preserve local identity while still exploiting mobility-guided correlations.

5.3.5 Effectiveness of Mobility Prediction. We evaluate whether the LSTM predictor provides a reliable transition matrix by comparing the actual and predicted transition matrices at $t = 300$.

The predicted matrix captures the main mobility structure. Both matrices show diagonal dominance, indicating that many clients

Table 4: Accuracy comparison of actual vs. predicted transition matrices under Dir(0.1). Accuracy (%) reported.

Transition matrix	MNIST	EMNIST	CIFAR-10	CIFAR-100
Actual (oracle)	98.19	93.06	83.21	39.15
Predicted (LSTM)	98.03	92.85	82.82	38.80
Gap	0.16	0.21	0.39	0.35



(a) Cabspotting trajectories. **(b) BDD100K object observations.**

Figure 7: Visualization for object detection case study. The left subfigure shows sampled Cabspotting trajectories over the 4×4 edge grid. The right subfigure shows spatially distributed observations for representative BDD100K categories. Both use OpenStreetMap as the geographic background.

remain in their current regions, and spatial locality, where off-diagonal transitions concentrate around nearby regions. The element-wise difference between predicted and actual matrices is below 0.05 for 92% of entries. Table 4 further shows that using predicted transitions is within 0.39 accuracy points of the oracle setting.

This result suggests that HiMAC does not require perfect individual trajectory prediction. It only needs reliable edge-level estimates of client flows. Self-transitions preserve local knowledge, while off-diagonal probabilities identify which neighboring edges should exchange information.

5.4 Case Study on Real-World Object Detection

Experiment Setup. We further evaluate HiMAC on BDD100K object detection [43]. BDD100K contains driving scenes with location metadata, which we use to construct region-specific object observations in San Francisco. Client mobility is modeled using Cabspotting taxi trajectories [31]. Together, these datasets simulate a realistic mobile edge setting where clients move across regions with different visual distributions. Fig. 7 visualizes the mobility traces and spatially distributed object observations.

Results. Table 5 reports the best mAP@50 within 20 communication rounds under different mobility intensities. HiMAC consistently outperforms all MobiHFL baselines. In the static setting, it improves over the strongest baseline, MACH, by 2.21 mAP@50 points. As mobility increases, the gap widens. Under high mobility, HiMAC reaches 24.32%, while MACH drops to 17.82%, giving a 6.50-point improvement in mAP@50.

Table 5: Object detection performance (mAP@50 %) on YOLO-BDD100K under varying mobility intensities. Migration frequency is measured as transitions per client per 100 rounds.

Method	Static (0)	Low (5)	Medium (15)	High (30)
Mob-HierFAVG [5]	25.42	21.44	18.25	14.12
FedMG [47]	25.85	22.30	19.55	15.68
MIDDLE [45]	26.12	22.84	20.12	16.54
MACH [44]	26.55	23.78	20.48	17.82
HiMAC	**28.76**	**27.18**	**25.86**	**24.32**

Object detection magnifies spatiotemporal mismatch because clients observe region-specific road layouts, backgrounds, and object co-occurrences. When clients move, they may train or evaluate under edge models shaped by different visual distributions. HiMAC transfers well to this setting because its decoupled backbone-classifier design and transition-driven collaboration are not tied to classification. They align shared representations and region-specific prediction heads before mobility-induced shifts become severe.

5.5 Limitations

Although raw trajectories remain local, destination indices and prototypes may leak mobility or representation information. Federated trajectory matching protects distributed locations via geo-indistinguishability and secure computation [37]; extending such guarantees to iterative transition and prototype sharing through secure aggregation, perturbation, or private exchange remains open. Our benchmarks decouple Dirichlet vision data from taxi mobility and report best rather than per-class or post-handover performance; robustness to corrupted transitions and wall-clock and client-side energy costs also remains future work.

6 Conclusion

We presented HiMAC, which uses predicted mobility to drive bidirectional collaboration among personalized edge models. It improves image-classification accuracy by up to 12.18 percentage points and high-mobility object-detection mAP@50 by 6.50 points.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China (NSFC) (Grant Nos. 62425202, 62336003), the Beijing Natural Science Foundation (Z230001), the Fundamental Research Funds for the Central Universities No. JKF-2026007844235, and the State Key Laboratory of Complex & Critical Software Environment (SKLCCSE). This work is partially supported by the RGC of Hong Kong SAR, China (Project No. CityU 11206425). Yi Xu and Yongxin Tong are the corresponding authors.

GenAI Usage Disclosure

The authors used generative AI tools only for language polishing, grammar checking, and editing suggestions. They were not used to generate results or data, conduct evaluation, or replace scientific reasoning. The authors verified and finalized all technical content, experimental design, results, and claims.

References

[1] Hidehisa Arai, Keita Miwa, Kento Sasaki, Yu Yamaguchi, Kohei Watanabe, Shunsuke Aoki, and Issei Yamamoto. 2025. CoVLA: Comprehensive Vision-Language-Action Dataset for Autonomous Driving. In *WACV*. 1933–1943.

[2] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. 2019. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818* (2019).

[3] Luca Ballotta, Nicolò Dal Fabbro, Giovanni Perin, Luca Schenato, Michele Rossi, and Giuseppe Piro. 2025. VREM-FL: Mobility-Aware Computation-Scheduling Co-Design for Vehicular Federated Learning. *IEEE Transactions on Vehicular Technology* 74, 2 (2025), 3311–3326. doi:10.1109/TVT.2024.3479780

[4] Xing Chang, Mohammad S. Obaidat, Jingxiao Ma, and Xiaoping Xue. 2025. Grid-Assisted Federated Learning in Vehicular Crowdsensing: A Mobility-Aware and Probabilistic Vehicle Selection Approach. *IEEE Transactions on Vehicular Technology* 74, 7 (2025), 11264–11280. doi:10.1109/TVT.2025.3543906

[5] Tan Chen, Jintao Yan, Yuxuan Sun, Sheng Zhou, Deniz Gündüz, and Zhisheng Niu. 2025. Mobility Accelerates Learning: Convergence Analysis on Hierarchical Federated Learning in Vehicular Networks. *IEEE Transactions on Vehicular Technology* 74, 1 (2025), 1657–1673. doi:10.1109/TVT.2024.3466299

[6] Siyao Cheng, Boyi Liu, Zimu Zhou, Zheng Wu, and Yongxin Tong. 2026. Progressive Collaboration Pruning for Federated Learning. In *CIKM*.

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *IJCNN*. IEEE, 2921–2926.

[8] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2021. Exploiting shared representations for personalized federated learning. In *ICML*. 2089–2099.

[9] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. 2020. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *NeurIPS* 33 (2020), 3557–3568.

[10] Chenyuan Feng, Howard H. Yang, Deshun Hu, Zhiwei Zhao, Tony Q. S. Quek, and Geyong Min. 2022. Mobility-Aware Cluster Federated Learning in Hierarchical Wireless Networks. *IEEE Transactions on Wireless Communications* 21, 10 (2022), 8441–8458. doi:10.1109/TWC.2022.3166386

[11] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. 2020. An efficient framework for clustered federated learning. *NeurIPS* 33 (2020), 19586–19597.

[12] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A Kernel Two-Sample Test. *Journal of Machine Learning Research* 13, 25 (2012), 723–773. https://jmlr.org/papers/v13/gretton12a.html

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. doi:10.1162/neco.1997.9.8.1735

[15] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335* (2019).

[16] Glenn Jocher. 2020. *Ultralytics YOLOv5*. doi:10.5281/zenodo.3908559

[17] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images*. Technical Report. University of Toronto. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 2002. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (2002), 2278–2324.

[19] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and robust federated learning through personalization. In *ICML*. 6357–6368.

[20] Boyi Liu, Shuyuan Li, Zimu Zhou, Shuo Kang, Yiming Ma, and Yongxin Tong. 2025. Poster: Asynchronous Federated Learning Library and Benchmark with AFL-Lib. In *MobiCom*. 1281–1283. doi:10.1145/3680207.3765660

[21] Boyi Liu, Yiming Ma, Zimu Zhou, Yexuan Shi, Shuyuan Li, and Yongxin Tong. 2024. CASA: Clustered Federated Learning with Asynchronous Clients. In *SIGKDD*. 1851–1862. doi:10.1145/3637528.3671979

[22] Boyi Liu, Zimu Zhou, Cheng Fang, and Yongxin Tong. 2026. Federated Personalization of Early-Exit Networks. In *CIKM*.

[23] Boyi Liu, Zimu Zhou, Pengfei Gao, Shuo Kang, and Yongxin Tong. 2026. Towards Asynchronous Client Collaboration in Personalized Federated Learning. In *INFOCOM*. 1–10. doi:10.1109/INFOCOM59046.2026.11571282

[24] Mingyu Liu, Ekim Yurtsever, Jonathan Fossaert, Xingcheng Zhou, Walter Zimmer, Yuning Cui, Bare Luka Zagar, and Alois C. Knoll. 2024. A Survey on Autonomous Driving Datasets: Statistics, Annotation Quality, and a Future Outlook. *IEEE Transactions on Intelligent Vehicles* 9 (2024), 7138–7164. doi:10.1109/TIV.2024.3394735

[25] Yiming Ma, Boyi Liu, Zimu Zhou, Yanfeng Wang, and Yongxin Tong. 2026. Harnessing Asynchrony to Balance Modalities in Multi-Modal Federated Learning. In *DASFAA*. 455–471. doi:10.1007/978-981-92-0369-7_29

[26] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*. 1273–1282.

[27] Xiaomin Ouyang, Xian Shuai, Yang Li, Li Pan, Xifan Zhang, Heming Fu, Sitong Cheng, Xinyan Wang, Shihua Cao, Jiang Xin, et al. 2024. ADMarker: A Multi-Modal Federated Learning System for Monitoring Digital Biomarkers of Alzheimer's Disease. In *MobiCom*. 404–419.

[28] Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2021. ClusterFL: A Similarity-Aware Federated Learning System for Human Activity Recognition. In *MobiSys*. 54–66.

[29] Ziba Parsons, Fei Dou, Houyi Du, Zheng Song, and Jin Lu. 2023. Mobilizing personalized federated learning in infrastructure-less and heterogeneous environments via random walk stochastic ADMM. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Article 1598, 12 pages.

[30] Yan Peng, Xiaogang Tang, Yiqing Zhou, Yuenan Hou, Jintao Li, Yanli Qi, Ling Liu, and Hai Lin. 2023. How to Tame Mobility in Federated Learning Over Mobile Networks? *IEEE Transactions on Wireless Communications* 22, 12 (2023), 9640–9657. doi:10.1109/TWC.2023.3272920

[31] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. 2009. CRAWDAD Dataset epfl/mobility (v. 2009-02-24). Downloaded from https://crawdad.org/epfl/mobility/20090224. doi:10.15783/C7J010

[32] Lehao Qu, Shuyuan Li, Zimu Zhou, Boyi Liu, Yi Xu, and Yongxin Tong. 2025. DarkDistill: Difficulty-Aligned Federated Early-Exit Network Training on Heterogeneous Devices. In *SIGKDD*. 2374–2385. doi:10.1145/3711896.3736902

[33] Yichen Ruan and Carlee Joe-Wong. 2022. FedSoft: Soft Clustered Federated Learning with Proximal Local Updating. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 7 (2022), 8124–8131. doi:10.1609/aaai.v36i7.20785

[34] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. 2021. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems* 32, 8 (2021), 3710–3722. doi:10.1109/TNNLS.2020.3015958

[35] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2023. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems* 34, 12 (2023), 9587–9603. doi:10.1109/TNNLS.2022.3160699

[36] Yansheng Wang, Yongxin Tong, Zimu Zhou, Ruisheng Zhang, Sinno Jialin Pan, Lixin Fan, and Qiang Yang. 2023. Distribution-Regularized Federated Learning on Non-IID Data. In *ICDE*. 2113–2125. doi:10.1109/ICDE55515.2023.00164

[37] Yuxiang Wang, Yuxiang Zeng, Yi Xu, Zimu Zhou, and Yongxin Tong. 2024. Efficient and Private Federated Trajectory Matching. *IEEE Transactions on Knowledge and Data Engineering* 36, 12 (2024), 8079–8092. doi:10.1109/TKDE.2024.3424411

[38] Kunda Yan, Sen Cui, Abudukelimu Wuerkaixi, Jingfeng Zhang, Bo Han, Gang Niu, Masashi Sugiyama, and Changshui Zhang. 2024. Balancing Similarity and Complementarity for Federated Learning. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*. PMLR, 55739–55758. https://proceedings.mlr.press/v235/yan24a.html

[39] Yihan Yan, Xiaojun Tong, and Shen Wang. 2024. Clustered federated learning in heterogeneous environment. *IEEE Transactions on Neural Networks and Learning Systems* 35, 9 (2024), 12796–12809. doi:10.1109/TNNLS.2023.3264740

[40] Linghua Yang, Wantong Chen, Xiaoxi He, Shuyue Wei, Yi Xu, Zimu Zhou, and Yongxin Tong. 2024. FedGTP: Exploiting Inter-Client Spatial Dependency in Federated Graph-Based Traffic Prediction. In *KDD*. 6105–6116. doi:10.1145/3637528.3671613

[41] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology* 10, 2, Article 12 (2019), 19 pages. doi:10.1145/3298981

[42] Rui Ye, Zhenyang Ni, Fangzhao Wu, Siheng Chen, and Yanfeng Wang. 2023. Personalized Federated Learning with Inferred Collaboration Graphs. In *Proceedings of the 40th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 202)*. PMLR, 39801–39817. https://proceedings.mlr.press/v202/ye23b.html

[43] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. 2018. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling. *CoRR* abs/1805.04687 (2018). arXiv:1805.04687 http://arxiv.org/abs/1805.04687

[44] Songli Zhang, Zhenzhe Zheng, Qinya Li, Fan Wu, and Guihai Chen. 2024. Mobility-Aware Device Sampling for Statistical Heterogeneity in Hierarchical Federated Learning. In *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*. 656–667. doi:10.1109/ICDCS60910.2024.00067

[45] Songli Zhang, Zhenzhe Zheng, Fan Wu, Bingshuai Li, Yunfeng Shao, and Guihai Chen. 2023. Learning From Your Neighbours: Mobility-Driven Device-Edge-Cloud Federated Learning. In *Proceedings of the 52nd International Conference on Parallel Processing (ICPP '23)*. ACM, 462–471. doi:10.1145/3605573.3605643

[46] Tianlong Zhang, Xiaoxi He, Yuxiang Wang, Yi Xu, Rendi Wu, Zhifei Wang, and Yongxin Tong. 2025. FedMetro: Efficient Metro Passenger Flow Prediction via Federated Graph Learning. In *SIGKDD*. 5215–5224. doi:10.1145/3711896.3737218

[47] Xinmin Zhang and Jie Wang. 2025. FedMG: Vehicular Edge Federated Learning for Mobile Scenarios with Geo-Dispersed Data. *IEEE Transactions on Vehicular Technology* 74, 1 (2025), 1520–1533. doi:10.1109/TVT.2024.3455333