Quality-Aware Personalized AI Service Provisioning in UAV-Assisted 6G Networks

Mohammad Farhoudi[1], Masoud Shokrnezhad[2], and Tarik Taleb[3]
[1] *Oulu University, Finland*; mohammad.farhoudi@oulu.fi
[2] *ICTFICIAL Oy, Espoo, Finland*; masoud.shokrnezhad@ictficial.com
[3] *Ruhr University Bochum (RUB), Germany*; tarik.taleb@rub.de

***Abstract*—In sixth-generation (6G) artificial intelligence (AI) services, two quality dimensions should be jointly addressed: conventional quality (e.g., latency) and Quality of AI Services (QoAIS; output fidelity, continuity, personalization). Existing methods emphasize conventional quality, while neglecting QoAIS, particularly for personalized outputs in dynamic aerial-terrestrial settings. This paper introduces HyPE, a hybrid predictive-in-context-learning framework for holistically quality-aware personalized AI service provisioning in Unmanned Aerial Vehicle (UAV)-assisted 6G networks. HyPE integrates: (i) mobility-aware prediction to forecast spatio-temporal request distributions, (ii) learning-augmented decision leveraging Large Language Model (LLM)-based reasoning to optimize UAV trajectories and inference assignments, and (iii) pre-/post-processing service placement and routing using heuristics. We formulate an optimization problem for joint trajectory planning, service placement, and routing, and present HyPE as a scalable alternative to intractable optimal solutions. Simulations with empirical mobility traces and heterogeneous AI workloads show near-optimal coverage, reduced end-to-end latency, sustained QoAIS-driven, and continuity-based service personalization versus optimization and state-of-the-art baselines. The results highlight the promise of predictive learning-augmented provisioning for elastic, user-centric AI in 6G.**

***Index Terms*—Service Provisioning, Quality of AI Services (QoAIS), Edge-Cloud Environment, Intelligent UAV, 6G Aerial-Terrestrial Networks.**

I. Introduction

With the rising demand for Artificial Intelligence (AI)-driven services, sixth-generation (6G) networks should pave the way to accommodate them. 6G is envisioned as an AI-native infrastructure, enabling adaptive service provisioning [1], [2]. It should jointly satisfy two service quality dimensions: conventional quality (End-to-End (E2E) latency) and Quality-of-AI-Services (QoAIS) (output fidelity and continuity-aided user-specific personalization) [3], [4]. Meeting these dual requirements calls for a heterogeneous model stack, where edge-cloud nodes host lightweight distilled generative models for low-latency responses alongside full-scale models for richer reasoning and QoAIS-driven personalization that adapts outputs to user history [5], [6]. To remain scalable under high load, this satisfaction should be modular, leveraging pluggable pre-processing (e.g., filtering, visual encoding) and post-processing (e.g., voice modulation, translation, formatting) stages that can be composed per request to balance E2E latency and QoAIS [7]. Additionally, resource-intensive inference pipelines should be orchestrated in dynamic networks, posing a central challenge for efficient provisioning under variable scenarios [8].

Furthermore, mobility adds complexity to service provisioning, as users expect reliable responses while moving across network regions [9]. 6G networks aim to provide ubiquitous access through AI-native infrastructure and intelligent edge capabilities [10]. To realize such pervasive access, 6G network architectures incorporate aerial platforms as mobile edge nodes that complement the static terrestrial infrastructure [11]. Equipped with computing resources and embedded AI models, Unmanned Aerial Vehicles (UAVs) deliver on-demand coverage and adaptive service provisioning, executing pre-/post-processing and inference tasks. Dynamic transitions of these tasks are therefore essential to maintain QoAIS across user sessions in highly mobile environments.

Recent research has investigated task offloading, service placement, and pipeline scheduling in edge-cloud environments under latency and accuracy constraints. For instance, Hao *et al.* [12] proposed a discrete-continuous Deep Reinforcement Learning (DRL) method based on latent space to optimize UAV trajectories and resource allocation for latency reduction, while Ding *et al.* [4] introduced a multi-objective scheduler, using genetic-based algorithms for QoAIS-aware provisioning to balance latency and accuracy. Raj *et al.* [13] focused on personalized offloading by deadline-aware and guaranteed experience heuristic methods to assist visually impaired users, whereas Jin *et al.* [14] proposed multi-tier architectures and game-theoretic offloading strategies that dynamically allocate resources to minimize latency under vehicular mobility. Yan *et al.* [15] presented Action-Decoupled Soft Actor-Critic (AD-SAC) for accuracy-aware Large Language Model (LLM) offloading in UAV-satellite networks, jointly optimizing trajectory and model placement to minimize total service latency. Similarly, Hu *et al.* [16] developed Joint AI Agent Placement with Deep neural network Deployment (JAAPD-D), combining multi-timescale Lyapunov optimization with AI model placement to balance latency and acceptance rates.

Despite these advances, several important gaps remain. Existing studies mainly fall into two separate lines: they either address generic AI task offloading, resource allocation, or service placement without explicitly supporting personalized inference, or they consider personalized AI-oriented services without jointly optimizing UAV trajectory control,

inference node assignment, and service continuity under mobility. Current solutions mostly overlook QoAIS, especially the joint interplay among E2E latency, model fidelity, and continuity-aware personalization in dynamic aerial-terrestrial 6G networks. In particular, limited attention has been paid to the spatio-temporal alignment between UAV movement and the availability of user-specific historical context required for personalized inference, as well as to the coordinated use of distilled and full-scale models across heterogeneous nodes to satisfy different latency-quality demands. These limitations call for an adaptive network-layer enabler that proactively tracks user mobility and requests; positions UAVs near users with relevant personalization history; assigns appropriate model variants according to latency and quality requirements; and distributes pre-processing, inference, and post-processing functions across heterogeneous aerial-terrestrial resources. In response, this paper introduces a mobility-aware, latency- and QoAIS-driven framework for personalized AI service provisioning with four key contributions:

- a modular service fabric that composes pluggable pre- and post-processing stages to elastically serve requests, enabling cross-modal latency and QoAIS co-optimization with on-the-fly model distillation;
- a joint optimization formulation that couples user mobility for UAV trajectory planning, function placement, inference assignment, and routing under quality and resource constraints, balancing E2E latency, output fidelity, and personalization continuity;
- a hybrid predictive-learning stack that fuses DRL-based spatio-temporal demand forecasting with structured LLM-guided online orchestration, and lightweight heuristics for constraint-aware routing and placement; and
- extensive evaluations on mobility traces and heterogeneous AI workloads showing near-optimal coverage, consistently low latency, and sustained QoAIS-driven personalization and output fidelity under high user density.

The remainder of the paper is organized as follows: Section II presents the system model, including network architecture and service pipeline; Section III formulates the problem; Section IV details the proposed hybrid framework; Section V provides performance evaluation; and Section VI concludes and outlines future research directions.

II. System Model

In this section, we detail a system designed to deliver AI services to mobile users over dynamic 6G networks.

A. Network Architecture

We model the system as a grid $\boldsymbol{\mathcal{A}}$ of discrete areas reflecting urban sectors or road segments. Area adjacency is captured by the binary predicate $\mathcal{A}_{a,a'}$, set to 1 if a and a' are neighbors and 0 otherwise. The network is modeled as a time-varying directed graph $\mathcal{G}(\mathcal{N},\mathcal{L},\mathcal{P})_t$, where $\boldsymbol{\mathcal{N}}$, $\boldsymbol{\mathcal{L}}_t$, and $\boldsymbol{\mathcal{P}}_t$ represent the set of network nodes, links, and candidate routing paths at time $t \in \boldsymbol{\mathcal{T}}$. The multi-tier network comprises cloud nodes, terrestrial edge servers (e.g., roadside units or base stations), and aerial UAV computing nodes. Each node $n \in \boldsymbol{\mathcal{N}}$ is characterized by its type, computing capacity $\widehat{\mathcal{C}}_n$, and model fidelity score $\overline{Q}_n$, capturing the quality of the AI model hosted at n. Communication links $l \in \boldsymbol{\mathcal{L}}_t$ describe feasible wired/wireless connections among nodes, characterized by capacity $\widehat{\mathcal{L}}_l$ and determined by network topology and aerial node positions. Depending on available links at each time frame, a set of paths $\boldsymbol{\mathcal{P}}_t$ defines feasible packet transmission routes, with $\mathcal{H}_{p,t}$ as well as $\mathcal{T}_{p,t}$ denoting head and tail nodes, and $\mathcal{J}_{p,l,t}$ indicating whether path p traverses link l at time t. Time is slotted, and each frame is long enough for one adjacency-constrained UAV move and service reconfiguration; thus, UAV movement time is absorbed into frame evolution.

B. AI Services

The system supports a catalog $\boldsymbol{\mathcal{S}}$ of AI services (e.g., voice command processing). Each service s with duration $\mathcal{T}_s^{du}$ is structured as a three-stage chain: (i) *Pre-processing* $\boldsymbol{\mathcal{F}}_s^{pr}$: lightweight functions with π_s^{pr} stages that prepare input for inference; (ii) *Inference*: reasoning using one AI model, with full generative models on edge-cloud nodes or distilled variants on UAVs; and (iii) *Post-processing* $\boldsymbol{\mathcal{F}}_s^{po}$: final transformations before serving the user, with π_s^{po} stages. This separation enables the selection of serving nodes with different fidelity levels under compute, communication, and latency constraints.

C. User and Request

Users $u \in \boldsymbol{\mathcal{U}}$ generate requests $r \in \boldsymbol{\mathcal{R}}_u$ - in which $\boldsymbol{\mathcal{R}}_u = \{r \in \boldsymbol{\mathcal{R}} \mid r \text{ is generated by user } u\}$ - over time horizon $\boldsymbol{\mathcal{T}}$, connecting through their Point of Attachment (PoA) to access the required services. Each request is represented by a tuple $(\mathcal{S}_r, \mathcal{T}_r^{pr}, \Delta_r, \mathcal{I}_{u_r,a,t}, \check{\mathcal{Q}}_r^{ou}, \check{\mathcal{Q}}_r^{la})$, specifying the requested service, start time, admissible request window $\Delta_r = [\mathcal{T}_r^{pr}, \mathcal{T}_r^{pr} + \mathcal{T}_{s_r}^{du}]$, user mobility path, minimum acceptable inference fidelity, and maximum tolerable E2E latency, respectively. Successful service delivery requires meeting both $\check{\mathcal{Q}}_r^{ou}$ and $\check{\mathcal{Q}}_r^{la}$. Additionally, personalization quality $\mathcal{Q}_u^{pe}$ is treated as a continuity proxy: repeated service at previously associated inference nodes improves context retention and response consistency. Each request r also requires minimum bandwidth $\check{\mathcal{L}}_r$ along with maximum packet size $\check{\mathcal{Z}}_{r,t}$, influencing transmission feasibility and latency, with latency modeled as path-based aggregation. Finally, requests' computational demands are defined in giga-floating point operations per second (GFLOPS) as $\check{\mathcal{C}}_{r,f}^{pr}, \check{\mathcal{C}}_r^{in}, \check{\mathcal{C}}_{r,f}^{po}$ for pre-, inference, and post-processing, respectively.

III. Problem Formulation

The optimization framework aims to jointly maximize the number of supported requests and their QoAIS while minimizing E2E latency for AI services in 6G networks. The objective function (OF) balances three terms: output fidelity weighted by α (reflecting node suitability), personalization quality scaled by β (promoting assignment continuity), and latency penalized by ζ (aggregated over routing paths). Here, α, β, and ζ are weighting coefficients that tune the relative importance of fidelity, personalization, and latency despite their different

numerical scales. Request acceptance is represented by the binary variable $\mathcal{X}_r$, with respect to $\mathcal{X}^{pr}_{r,f,t}, \mathcal{X}^{in}_{r,t}, \mathcal{X}^{po}_{r,f,t}$ denoting execution of pre-processing, inference, and post-processing functions, respectively. To unify notation across service stages, we use the stage-dependent variable $\mathcal{X}^{\phi}_{r,f,t}$. For pre- and post-processing stages, f indexes functions in F^{ϕ}_{s}, while for inference, we define a singleton set $\boldsymbol{\mathcal{F}}^{in}_{s} = \{\varnothing\}$, so that $\mathcal{X}^{in}_{r,f,t} = \mathcal{X}^{in}_{r,t}$. A request is considered accepted if its three service stages are completed within the prescribed temporal structure: all pre-processing functions are executed at the entry frame $\mathcal{T}^{pr}_{r}$, inference is executed during $\mathcal{T}^{in}_{r} = (\mathcal{T}^{pr}_{r}, \mathcal{T}^{pr}_{r} + \mathcal{T}^{du}_{s_r})$, and all post-processing functions are completed at the terminal frame $\mathcal{T}^{po}_{r} = \mathcal{T}^{pr}_{r} + \mathcal{T}^{du}_{s_r}$ (C1).

$$\max \sum_{\boldsymbol{\mathcal{R}}} \mathcal{X}_r + \alpha \cdot \sum_{\boldsymbol{\mathcal{R}}} \mathcal{Q}^{ou}_{r} + \beta \cdot \sum_{\boldsymbol{\mathcal{U}}} \mathcal{Q}^{pe}_{u} - \zeta \cdot \sum_{\boldsymbol{\mathcal{R}}} \mathcal{Q}^{la}_{r} \quad \textit{s.t.} \quad \text{C1 - C18} \quad \text{(OF)}$$

$$\mathcal{X}_r = \prod_{\phi \in \{pr,in,po\}} \Big(\sum_{F^{\phi}_{s_r}, \mathcal{T}^{\phi}_{r}} \mathcal{X}^{\phi}_{r,f,t} \Big) \qquad \forall\, r \in \boldsymbol{\mathcal{R}} \quad \text{(C1)}$$

A. Definition

The definition constraints govern activation rules, logical dependencies among functions, and their placement across the network. Function placements are binary variables $\mathcal{Y}^{pr}_{f,n,t}, \mathcal{Y}^{po}_{f,n,t}$ indicating that function f is deployed on node n at time t, and $\mathcal{Y}^{in}_{r,n,t}$ indicates that request r's inference task is served by node n at t. For notational uniformity, the same stage-indexed convention is used for placement and computation variables, with the inference stage treated as a singleton component. Constraint C2 ensures each request $r \in \mathcal{R}$ activates $\pi^{pr}_{s_r}$ pre-processing function $f \in \mathcal{F}^{pr}_{s_r}$ at entry time, and $\pi^{po}_{s_r}$ post-processing function $f \in \mathcal{F}^{po}_{s_r}$ at the terminal time. Constraint C3 binds inference execution variables $\mathcal{X}^{in}_{r,t}$ to deployment variables $\mathcal{Y}^{in}_{r,n,t}$, ensuring scheduled inference is served by a node. Finally, Constraint C4 guarantees that if a function f is demanded at time t, it should be deployed on at least one node.

$$\sum_{\boldsymbol{\mathcal{F}}^{\phi}_{s_r}} \mathcal{X}^{\phi}_{r,f,\mathcal{T}^{\phi}_{r}} = \pi^{\phi}_{s_r} \qquad \forall\, r, \phi \in \boldsymbol{\mathcal{R}},\ \{pr, po\} \quad \text{(C2)}$$

$$\sum_{\boldsymbol{\mathcal{N}}} \mathcal{Y}^{in}_{r,n,t} > \sum_{\boldsymbol{\mathcal{R}}} \mathcal{X}^{in}_{r,t} \qquad \forall\, r, t \in \boldsymbol{\mathcal{R}}, \Delta_r \quad \text{(C3)}$$

$$\sum_{\boldsymbol{\mathcal{N}}} \mathcal{Y}^{\phi}_{f,n,t} > \frac{\sum_{\boldsymbol{\mathcal{R}}} \mathcal{X}^{\phi}_{r,f,t}}{\mathcal{R}} \qquad \forall f, t, \phi \in \bigcup (\boldsymbol{\mathcal{F}}^{\phi}_{s}), \Delta_r,\ \{pr, po\} \quad \text{(C4)}$$

B. Multi-time-frame Trajectory

To enable mobility-aware orchestration of aerial nodes, these constraints govern UAV spatial behavior and user interactions. UAV positions are captured by $\mathcal{S}_{n,a,t}$, which equals 1 if UAV n occupies an area a at time t, while edge nodes remain static but area-aware. Constraint C5 enforces location exclusivity, requiring each UAV to occupy exactly one area per time slot. Constraint C6 introduces adjacency-based transitions, allowing a UAV to move to an area a at time t only if it resided in an adjacent area a' at the preceding time $t-1$. Constraint C7 ensures a consistent user-node association $\mathcal{B}_{u,n,t}$ by restricting each user u to at most one binned node per time frame, and activating it only when the user and node share the same area.

$$\sum_{\boldsymbol{\mathcal{A}}} \mathcal{S}_{n,a,t} = 1 \qquad \forall\, n, t \in \boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{T}} \quad \text{(C5)}$$

$$\mathcal{S}_{n,a,t} \leq \sum_{a' \in \boldsymbol{\mathcal{A}}} \mathcal{A}_{a,a'} \cdot \mathcal{S}_{n,a',t-1} \qquad \forall\, n, a, t \in \boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{T}} \quad \text{(C6)}$$

$$\sum_{\boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{A}}} \mathcal{B}_{u_r,n,t} \cdot \mathcal{S}_{n,a,t} \cdot \mathcal{I}_{u_r,a,t} \leq 1 \qquad \forall\, r, t \in \boldsymbol{\mathcal{R}}, \Delta_r \quad \text{(C7)}$$

C. Dynamic Network Graph

These constraints govern the dynamic construction of time-varying network topology under mobility. Constraint C8 defines $\mathcal{L}_{n,n',t}$, a binary variable indicating the existence of a link between the nodes n and n' at time t, enabled only when nodes occupy adjacent or overlapping areas, thus reflecting nodes' coverage limits. Based on this, the feasible link set $\boldsymbol{\mathcal{L}}_t$ is constructed (C9), while the candidate path set $\boldsymbol{\mathcal{P}}_t$ is derived as sequences of active links (C10). Constraint C11 then enforces routing consistency by activating $\mathcal{J}_{p,l,t}$ only if link l is active in path p at t.

$$\mathcal{L}_{n,n',t} \leq \sum_{a,a' \in \boldsymbol{\mathcal{A}}} \mathcal{A}_{a,a'} \cdot \mathcal{S}_{n,a,t} \cdot \mathcal{S}_{n',a',t} \qquad \forall\, n, n', t \in \boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{T}} \quad \text{(C8)}$$

$$\boldsymbol{\mathcal{L}}_t \triangleq \big\{ (n, n') \in \boldsymbol{\mathcal{N}} \times \boldsymbol{\mathcal{N}} \ \big|\ \mathcal{L}_{n,n',t} = 1 \big\} \qquad \forall\, t \in \boldsymbol{\mathcal{T}} \quad \text{(C9)}$$

$$\boldsymbol{\mathcal{P}}_t \triangleq \{ p = (\mathcal{H}_{p,t}\ , \mathcal{T}_{p,t}) \mid p \subset \boldsymbol{\mathcal{L}}_t \} \qquad \forall\, t \in \boldsymbol{\mathcal{T}} \quad \text{(C10)}$$

$$\mathcal{J}_{p,l,t} \leq \sum_{\boldsymbol{\mathcal{N}}, \boldsymbol{\mathcal{N}}} \mathcal{L}_{n,n',t} \qquad \forall\, p, l, t \in \boldsymbol{\mathcal{P}}_t, \boldsymbol{\mathcal{L}}_t, \boldsymbol{\mathcal{T}} \quad \text{(C11)}$$

D. Path Selection

These constraints structure data flow across service stages, ensuring that requests are routed on feasible, coherent paths in dynamic topologies induced by UAV mobility. Routing is enforced across service stages. For each request r, time frame t, and stage ϕ, one feasible path is selected between the active source and destination endpoint (C12). Here, $\eta^{h,\phi}_{r,n,t}$ and $\eta^{t,\phi}_{r,n,t}$ denote the active head and tail nodes of the stage ϕ's route for request r at time t: for $\phi = pr$, they correspond to the user's PoA node and the selected pre-processing node; for $\phi = in$, to the previously selected pre-processing node and the selected inference node; and for $\phi = po$, to the selected inference node and the selected post-processing node. In this way, the path is created whenever two consecutive functions or inference nodes are active at frame t, which adapts to time-varying connectivity, resource availability, and user proximity.

$$\sum_{\substack{\boldsymbol{\mathcal{P}}_t,\ \boldsymbol{\mathcal{F}}^{\phi}_{s_r} \\ \mathcal{H}_{p,t} = \eta^{h,\phi}_{r,n,t}, \\ \mathcal{T}_{p,t} = \eta^{t,\phi}_{r,n,t}}} \mathcal{R}^{tr}_{r,p,t} \mathbb{1}(\mathcal{Y}^{\phi}_{f,n,t} == 1) = 1 \quad \forall r, t, \phi \in \boldsymbol{\mathcal{R}}, \mathcal{T}^{\phi}_{r}, \{pr, in, po\} \quad \text{(C12)}$$

E. QoS Constraints

These constraints ensure service delivery with sufficient quality. Constraint C13 models E2E latency $\mathcal{Q}^{la}_{r}$ as the cumulative sum of link latencies, each given by the ratio of the request's maximum packet size $\breve{\mathcal{Z}}_{r,t}$ to link capacity $\widehat{\mathcal{L}}_{l}$. The latency should follow the request's predefined tolerance $\breve{\mathcal{Q}}^{la}_{r}$.

Meanwhile, output quality $\mathcal{Q}_r^{ou}$ is captured as the cumulative fidelity contributed by the inference nodes serving r, while it should meet or exceed a required threshold $\check{\mathcal{Q}}_r^{ou}$ (C14). To foster personalization quality, Constraint C15 introduces a recency-weighted history variable $\mathcal{H}_{u,n,t}^{pa}$, quantifying the degree to which a node n has historically served user u up to time t, modulated by a decay factor δ. Building upon this, Constraint C16 defines $\mathcal{Q}_u^{pe}$ as the sum of such affinities across inference tasks, rewarding continuity-based responses.

$$\mathcal{Q}_r^{la} \triangleq \sum_{\boldsymbol{\mathcal{P}}_t,\boldsymbol{\mathcal{L}}_t,\boldsymbol{\mathcal{T}}} \mathcal{R}_{r,p,t}^{tr} \cdot \mathcal{J}_{p,l,t} \cdot \left(\check{\mathcal{Z}}_{r,t}/\widehat{\mathcal{L}}_l\right) \leq \check{\mathcal{Q}}_r^{la} \qquad \forall\, r \in \boldsymbol{\mathcal{R}} \quad \text{(C13)}$$

$$\mathcal{Q}_r^{ou} \triangleq \sum_{\boldsymbol{\mathcal{T}},\boldsymbol{\mathcal{N}}} \mathcal{Y}_{r,n,t}^{in} \cdot \overline{Q}_n \geq \check{\mathcal{Q}}_r^{ou} \qquad \forall\, r \in \boldsymbol{\mathcal{R}} \quad \text{(C14)}$$

$$\mathcal{H}_{u,n,t}^{pa} = \sum_{\mathfrak{T}=0}^{t-1} \delta^{t-\mathfrak{T}} \sum_{\boldsymbol{\mathcal{R}}_u} \mathcal{Y}_{r,n,\mathfrak{T}}^{in} \qquad \forall\, u,n,t \in \boldsymbol{\mathcal{U}},\boldsymbol{\mathcal{N}},\boldsymbol{\mathcal{T}} \quad \text{(C15)}$$

$$\mathcal{Q}_u^{pe} \triangleq \sum_{\boldsymbol{\mathcal{T}},\boldsymbol{\mathcal{N}}} \mathcal{H}_{u,n,t}^{pa} \sum_{\boldsymbol{\mathcal{R}}_u} \mathcal{Y}_{r,n,t}^{in} \qquad \forall\, u \in \boldsymbol{\mathcal{U}} \quad \text{(C16)}$$

F. Capacity Constraints

To ensure feasible service provisioning aligned with physical infrastructure limitations in the 6G aerial-terrestrial networks, capacity constraints bound both compute and communication resources. Computing load is limited by aggregating processing demands from all active pre-, inference, and post-processing tasks, requiring total utilization not exceeding the node's capacity $\widehat{\mathcal{C}}_n$ (C17). Communication is restricted by ensuring that cumulative bandwidth demands $\check{\mathcal{L}}_r$ from all routed requests using the link l do not surpass its capacity $\widehat{\mathcal{L}}_l$, thereby preventing congestion (C18).

$$\sum_{\boldsymbol{\mathcal{R}},\boldsymbol{\mathcal{F}}_{s_r}^{\phi}} \mathcal{X}_{r,f,t}^{\phi} \cdot \mathcal{Y}_{f,n,t}^{\phi} \cdot \check{\mathcal{C}}_{r,f}^{\phi} \leq \widehat{\mathcal{C}}_n \quad \forall\, n,t,\phi \in \boldsymbol{\mathcal{N}},\boldsymbol{\mathcal{T}},\{pr,in,po\} \quad \text{(C17)}$$

$$\sum_{\boldsymbol{\mathcal{R}},\boldsymbol{\mathcal{P}}_t} \mathcal{R}_{r,p,t}^{tr} \cdot \mathcal{J}_{p,l,t} \cdot \check{\mathcal{L}}_r \leq \widehat{\mathcal{L}}_l \qquad \forall\, l,t \in \boldsymbol{\mathcal{L}}_t,\boldsymbol{\mathcal{T}} \quad \text{(C18)}$$

IV. Proposed Method

The problem defined in Section III is NP-hard [17]. Hence, finding the solution of (OF) becomes computationally intractable in large-scale instances despite unrealistic assumptions that all system knowledge is available. To address this, we propose the hybrid predictive-in-context-learning (HyPE) framework, which integrates predictive modeling, learning-augmented decisions, and heuristics for scalable real-time service provisioning. HyPE operates in three phases: (i) mobility-aware prediction (MAP), which forecasts user request patterns to mitigate imperfect knowledge arising from user mobility; (ii) learning-augmented decision (LEAD), which leverages LLMs to jointly plan UAV trajectories and inference node assignments based on historical context and quality requirements; and (iii) service placement and routing (SET), which applies Proximity-Greedy Allocation (PGA) for pre/post-function placement and Latency-Biased Shortest Path (LBSP) for efficient route selection under dynamic topology. The E2E process of HyPE is shown in Algorithm 1.

MAP: In the first phase, HyPE employs a DRL predictor to address the challenges of forecasting user mobility and anticipating service demands in highly dynamic 6G environments. This design builds on prior DRL-based prediction methods for mobile service environments [18]. Unlike offline learning models, which cannot adapt to rapidly changing traffic patterns, we employ online DRL to continuously capture the spatio-temporal evolution of user behavior. Specifically, each PoA hosts a Dueling Double Deep Q-Learning (D3QL) agent with a Long Short-Term Memory (LSTM)-Convolutional Neural Network (CNN) architecture to process historical mobility and request traces. The LSTM captures temporal dependencies in user movement and service arrivals, while the CNN extracts local spatial correlations across neighboring areas. Based on these observations, the agent outputs the probability distribution that user u_r will appear in area a and generate request r in the next time frame. Unlike using prediction only for mobility estimation, MAP is extended to jointly anticipate both user location and service intent, and to produce a prioritized set of likely future requests, reflecting their probability of occurrence. The agent's reward function is defined by prediction accuracy, incentivizing reliable mobility awareness while ensuring responsiveness to evolving user contexts. This "MAP" does not output final UAV placement or inference decisions; rather, it provides probabilistic forecasts of future user locations and anticipated requests that guide subsequent phases in proactive service provisioning.

LEAD: Following user mobility and request predictions and to satisfy anticipated request demands, we leverage an LLM-driven planner as an adaptive decision engine. In this setting, the LLM is tasked with producing two critical outputs for each upcoming time frame: (i) the trajectory design $\mathcal{S}_{n,a,t+1}$, denoting the area assignment for each UAV, and (ii) the inference assignment $\mathcal{Y}_{r,n,t+1}^{in}$, specifying which network node is responsible for processing the inference stage of each predicted request. The LEAD phase is tasked to transform the raw predictions of the MAP phase into actionable orchestration decisions. These decisions are made while jointly considering latency constraints (C13), fidelity requirements (C14), and personalization objectives (C16), thereby ensuring that the orchestration remains both feasible and user-centric.

LLMs offer a compelling alternative by enabling adaptive, in-context learning-based decision-making. First, LLMs can directly process inputs and align with the nature of personalized AI service provisioning, where user intent and mobility patterns should be considered simultaneously. Second, unlike static optimization solvers, LLMs can be prompted with domain-specific rules to perform inference selection and trajectory planning without exhaustively enumerating all feasible configurations. This allows decisions to be generated in real-time, even under partial observability or incomplete mobility information. Third, the ability of LLMs to generalize from prior interactions and adapt to new mobility or service contexts introduces a form of transferable intelligence, thereby supporting more flexible decision generation across heterogeneous deployment scenarios [19]. Finally, the adoption of LLMs integrates naturally with the broader system vision of AI-native 6G networks, where network control and service

```
<Role>Node trajectory + Service provisioner</Role>

<Nodes>Nodes capacity, fidelity, & current location</Nodes>
<Users>Predicted user areas</Users>
<Request>Anticipated user requests</Requests>
<Areas>Area and adjacency info</Areas>
<InfMem>History of user-node inference assignments</InfMem>

<Constraints>
1. Move UAV for a user if its capacity ≥ requested capacity.
2. Users and nodes should be collocated for binnding.
3. Ensure connectivity of consecutive node assignments.
4. Edge nodes are static and do not move.
5. UAV nodes move based on area adjacency rules <Areas>.
</Constraints>
<Examples>Few-shot positive/negative</Examples>
<Reasoning_Guidance>
- Trade-off: latency vs. QoAIS (personalization & fidelity).
- Prefer nodes with <InfMem> when feasible.
- If no request, move UAVs to anticipate future demands.
</Reasoning_Guidance>
<Performance_Metrics>
- Fidelity score of selected nodes.
- Personalization score of selected nodes.
</Performance_Metrics>

Design <Nodes> next areas and select inference nodes for
<Requests> of <Users> subject to <Constraints> following
<Examples> as examples, considering <Reasoning_Guidance> to
improve <Performance_Metrics>.

<Output_Schema>
{
   time: [str],
   nodes: [list] = {id: [str], area: [str]},
   inference: [list] = {request: [str], node: [str]}
}
</Output_Schema>
```

Figure 1. Structured prompt design for LLM-based node trajectory and inference selection in the LEAD phase.

orchestration are increasingly entrusted to intelligent methods rather than rigid rule-based optimization.

In LLMs, a free-form text description is submitted to the language model via a prompt, and it is instructed to generate a structured output. To effectively harness the reasoning capacity of LLMs, we design a structured prompting strategy composed of three complementary mechanisms, depicted in Figure 1.

Role Prompting: Initially, we implement role prompting, a natural language processing technique that assigns specific roles, personas, or contexts to language models to elicit more specialized responses. This method leverages corresponding knowledge patterns, terminology, and reasoning approaches typical of the assigned role, which enhances contextual consistency and reduces ambiguity [20]. The LLM is designated as a "Node trajectory and service provisioner" for a grid-based aerial-terrestrial network, responsible for providing UAV areas and inference nodes. By assigning this specialized role, the LLM is guided to operate within the reasoning patterns and decision logic expected of a domain-specific planner, rather than generating unconstrained text. The prompt provides contextual blocks, including predicted user areas and requests, node capacities and qualities, prior inference-memory for personalization continuity $\mathcal{M}_t^{in}$, and domain-specific constraints, with the instruction that its output should consist of available node blocks while prohibiting the inclusion of its own suggestions. Thus, this approach effectively narrows the search space and grounds the LLM's reasoning in the operational state, thereby improving its accuracy and response time.

Contrastive Few-Shot Learning: To ensure adaptability to evolving user contexts, the prompt incorporates contrastive few-shot examples derived from before the task execution [21]. High-reward examples (i.e., past assignments that yielded low latency as well as high fidelity and personalization scores) and low-reward examples (i.e., assignments violating Quality of Service (QoS) or leading to infeasible placements) are embedded into the prompt. This contrastive structure enables the LLM to refine its reasoning by explicitly learning from both successes and failures, thereby improving generalization to unseen mobility patterns and heterogeneous service demands. Through this mechanism, the LLM avoids repeating detrimental provisioning strategies and adapts more robustly to the stochasticity of user mobility. What is more, we employ contrastive learning to ensure that the LLM remains responsive to dynamic contexts in which user requests evolve and require diverse contextual interpretations. Specifically, we retain the recorded inference selection outputs $\mathcal{Y}_{r,n,t}^{in}$ following the completion of request r. For subsequent user requests, we select $\mathcal{K}$ examples from the highest-reward outputs, referred to as $\mathbb{K}^+$, and $\mathcal{K}$ examples from the lowest-reward outputs, referred to as $\mathbb{K}^-$. This strategy provides the LLM with updated guidance on what to prioritize and what to avoid, ensuring its adaptability to shifting requirements, particularly those related to personalization.

Structure Enforcement: To guarantee seamless integration of LLM outputs into the orchestration pipeline, we enforce a structured response format. Specifically, the LLM is instructed to provide node area and inference node assignments in a machine-readable schema, validated automatically using the Pydantic library [22]. The schema fields directly map to the optimization variables: nodes candidates $\mathcal{S}_{n,a,t+1}$ and inference candidates $\mathcal{Y}_{r,n,t+1}^{in}$, while validated outputs are the only decisions passed to SET for $\mathcal{B}_{u,n,t+1}$, $\mathcal{Y}_{f,n,t+1}^{pr,po}$, and $\mathcal{R}_{r,p,t+1}^{tr}$. Furthermore, the structured format allows direct validation against feasibility constraints: UAV moves are checked against adjacency rules (C6), inference assignments are validated against node capacity constraints (C17), and the network graph is updated according to link feasibility (C9-C10). Any infeasible decisions are discarded, and the LLM is iteratively guided toward valid outputs.

The LEAD phase algorithm is shown in Algorithm 1, steps 2-8. Consider a network with one UAV, one edge node, and two users u_1 and u_2. At time t, the MAP phase predicts that u_1 will move to area a_2 and request the speech-to-text service, while u_2 will remain in area a_1 and request image captioning. The UAV is currently in area a_1, and the edge node is static in a_3. The LLM prompt contains: (1) $<Users>$: predicted areas and $<Requests>$ (u_1 in a_2, u_2 in a_1); (2) $<Nodes>$: UAV (capacity 100 GFLOPS, fidelity 0.6), edge (capacity 400 GFLOPS, fidelity 0.8); (3) $<InfMem>$: u_1 previously served by edge, u_2 by UAV; (4) $<Reasoning\_Guidance>$: maximum tolerable latency 80ms, minimum required fidelity 0.7. The LLM, acting as the UAV trajectory and service provisioner, produces: (1) UAV trajectory: candidate move of UAV from a_1 to a_2 ($\mathcal{S}_{uav,a_2,t+1} = 1$); (2) Inference assignment: candidate assignment of u_1's inference on the edge node ($\mathcal{Y}_{r_{u_1},edge,t+1}^{in} = 1$) to meet fidelity > 0.7, and keep u_2's inference on the UAV ($\mathcal{Y}_{r_{u_2},uav,t+1}^{in} = 1$) to minimize latency and increase personalization. After validation, these

assignments are feasible: UAV mobility satisfies adjacency (C6), inference node assignments respect node capacities (C17), and the updated graph reflects new links (C9-C10). By validating outputs against network constraints, the framework ensures that UAV trajectories and node selections are not only context-aware but also operationally feasible. Hence, LEAD is not treated as an exact solver; it serves as a structured policy prior that proposes high-utility candidate decisions, which are then completed by deterministic constraint validation and SET. Thus, the validated outputs feed into the subsequent SET phase, where pre/post-processing placement and routing decisions are finalized to complete the orchestration cycle.

SET: This phase integrates (i) MAP's predicted user areas and requests, (ii) LEAD's inference-node and trajectory decisions, and (iii) residual capacities to finalize user-node bindings, placing pre-/post-processing functions, and selecting routing paths, as detailed in Algorithm 1, steps 9-25. First, binned node ($\mathcal{B}_{u,n,t+1}$) is selected for each active user u at time $t+1$, as required by C7. Let $\boldsymbol{\mathcal{N}}^{bi}_{u_r,t+1}$ be the candidate collocated nodes for u, as defined in Eq. (1). The binding is selected by minimizing latency to n' from u's PoA ($\mathrm{L}_{u\to n'}$) while incorporating historical affinity $\widehat{\mathcal{H}}_{u,n,t+1}$ (normalized history value from $\mathcal{H}^{pa}_{u,n,t+1}$), weighted by κ to balance recency with latency, as shown in Eq. (2).

$$\boldsymbol{\mathcal{N}}^{bi}_{u,t+1} = \{n \in \mathcal{N} \mid \exists a : \mathcal{I}_{u,a,t+1} = 1 \ \wedge \ \mathcal{S}_{n,a,t+1} = 1\}, \quad (1)$$

$$\mathcal{B}_{u,n,t+1}=\begin{cases}1, & \text{if } n=\underset{n'\in\boldsymbol{\mathcal{N}}^{bi}_{u,t+1}}{\operatorname{argmin}} \Big(\mathrm{L}_{u\to n'}+\kappa\,(1-\widehat{\mathcal{H}}_{u,n',t})\Big).\\ 0, & \text{otherwise.}\end{cases} \quad (2)$$

Next, the PGA method is used for function placement ($\mathcal{Y}^{pr,po}_{f,n,t+1}$) that assigns each required function $f \in \mathcal{F}_{t+1}$ to the node that minimizes aggregated latency for its requesting users $\boldsymbol{\mathcal{R}}_f$. Functions are prioritized by latency tightness ($\sum_{\boldsymbol{\mathcal{R}}_f} r$). For each $f \in \mathcal{F}_{t+1}$, we define the candidate nodes set $\mathcal{N}^{fe}_f$ that satisfy Eq. (3), where $p \cdot \widehat{\mathcal{C}}_n$ denotes node n's residual compute capacity. If no candidate node exists ($\mathcal{N}^{fe}_f = \varnothing$), f's requests ($\mathcal{R}_f$) are partitioned across replicas on multiple nodes based on spillover replication, which assigns replicas greedily until the subset of nodes that satisfies (Eq. 3). In other words, we find two or more nodes to satisfy the function f, and each node serves a subset of requests. Afterward, for each feasible node, the total latency is $\sum_{\boldsymbol{\mathcal{R}}_f} \mathrm{L}_{u_r\to n}$, where $\mathrm{L}_{u_r\to n}$ is computed over feasible paths in $t+1$. The placement node $n^\star = \operatorname{argmin}_{n\in\boldsymbol{\mathcal{N}}^{fe}_f} \sum_{\boldsymbol{\mathcal{R}}_f} \mathrm{L}_{u_r\to n}$ is then selected, and capacities are updated via Eq. (4). The process repeats until all $\mathcal{F}_{t+1}$ functions are placed or resources are exhausted.

$$\sum_{\boldsymbol{\mathcal{R}}_f} (\check{\mathcal{C}}^{pr}_{r,f} + \check{\mathcal{C}}^{po}_{r,f}) \ \le \ p \cdot \widehat{\mathcal{C}}_n, \quad (3)$$

$$p \cdot \widehat{\mathcal{C}}_{n^\star} = p \cdot \widehat{\mathcal{C}}_{n^\star} - \Big(\sum_{\boldsymbol{\mathcal{R}}_{f_{pr}}} \check{\mathcal{C}}^{pr}_{r,f} + \sum_{\boldsymbol{\mathcal{R}}_{f_{po}}} \check{\mathcal{C}}^{po}_{r,f}\Big). \quad (4)$$

After binding and placement, routing is performed on the updated graph $\mathcal{G}_{t+1}$ using LBSP, which selects low-latency paths while ensuring link capacity feasibility (C18). For each request r and transfer step (user→pre, pre→inference, inference→post), a feasible path $\mathcal{R}^{tr}_{r,p,t+1}$ is selected. Each

Algorithm 1: HyPE service provisioning

Input: $\boldsymbol{\mathcal{N}}_t$ (state, capacity, output fidelity)
Output: $\mathcal{S}_{n,a,t+1}$,$\mathcal{Y}^{in}_{r,n,t+1}$,$\mathcal{Y}^{pr,po}_{f,n,t+1}$,$\mathcal{B}_{u,n,t+1}$,$\mathcal{R}^{tr}_{r,p,t+1}$

1 Use DRL method to predict $\mathcal{I}_{u,a,t+1}$, $\boldsymbol{\mathcal{R}}_{t+1}$
2 Create $\mathcal{M}^{in}_t$ & block structured prompt with examples
3 Query LLM & receive candidates $\{\mathcal{S}_{n,a,t+1}, \mathcal{Y}^{in}_{r,n,t+1}\}$
4 **foreach** *UAV* $n \in \boldsymbol{\mathcal{N}}_t$ **do**
5 | Validate adjacency constraint (C6)
6 **foreach** $r \in \boldsymbol{\mathcal{R}}_{t+1}$ **do**
7 | Validate node (C17) & fidelity/latency (C13-C14)
8 Update network graph $\mathcal{G}_{t+1}$ & $\mathcal{M}^{in}_{t+1}$
9 **foreach** $r \in \boldsymbol{\mathcal{R}}_{t+1}$ **do**
10 | Compute candidate set $\boldsymbol{\mathcal{N}}^{bi}_{u,t+1}$ (Eq. (1))
11 | Compute $\mathrm{L}_{u\to n}$ & $\mathcal{B}_{u,n^\star,t+1}$ based on Eq. (2)
12 Sort $\mathcal{F}_{t+1}$ by latency tightness $\sum_{\boldsymbol{\mathcal{R}}_f} r | \boldsymbol{\mathcal{R}}_f \leftarrow$ f's requests
13 **foreach** $f \in \mathcal{F}_{t+1}$ **do**
14 | Compute candidate nodes $\boldsymbol{\mathcal{N}}^{fe}_f$ that meet Eq. (3)
15 | **if** $\boldsymbol{\mathcal{N}}^{fe}_f = \varnothing$ **then**
16 | | Spillover_replication (partition $\boldsymbol{\mathcal{R}}_f$)
17 | $n^\star = \operatorname{argmin}_{n\in\boldsymbol{\mathcal{N}}^{fe}_f} \sum_{\boldsymbol{\mathcal{R}}_f} \mathrm{L}_{u_r\to n}$
18 | $\mathcal{Y}^{pr,po}_{f,n^\star,t+1} \leftarrow 1$ & Update $\widehat{\mathcal{C}}_{n^\star}$ (Eq. (4))
19 **foreach** $r \in \boldsymbol{\mathcal{R}}_{t+1}$ **do**
20 | Determine r's stage & Find previous node
21 | **foreach** $l \in \boldsymbol{\mathcal{L}}_t$ **do**
22 | | Compute $w_{l,r,t+1}$ based on Eq. (5)
23 | | **while** *found* **do**
24 | | | Find $p^\star$ via Eq. (6)
25 | | $\mathcal{R}^{tr}_{r,p^\star,t+1} \leftarrow 1$ & Update $u_{l,t+1}\ \forall\, l \in p^\star$

request imposes latency based on $\check{\mathcal{Z}}_{r,t}$ on link l, while load $u_{l,t+1}$ derived from LEAD or earlier SET steps. For each request, the per-link weight $w_{l,r,t+1}$ is defined via Eq. (5), where γ tunes congestion avoidance (practical tuning knob). The final path $p^\star$ minimizes latency while satisfying capacity constraints via Eq. (6). If the shortest path is infeasible, LBSP iteratively tests alternatives; if none exist, the request is delayed (if Δ_r allows) or rejected.

$$w_{l,r,t+1} = \frac{\check{\mathcal{Z}}_{r,t}}{\widehat{\mathcal{L}}_l} + \gamma \cdot \frac{u_{l,t+1}}{\widehat{\mathcal{L}}_l}, \quad (5)$$

$$p^\star = \underset{p\in\boldsymbol{\mathcal{P}}_{t+1}}{\operatorname{argmin}} \sum_{l\in p} w_{l,r,t+1} \text{ s.t. } \sum_{\tilde{r},\tilde{p}\ni l} \mathcal{R}^{tr}_{\tilde{r},\tilde{p},t+1} \cdot \check{\mathcal{L}}_{\tilde{r}} + \check{\mathcal{L}}_r \le \widehat{\mathcal{L}}_l \ \ \forall l \in p^\star. \quad (6)$$

V. Performance Evaluation

Our simulation environment is designed to reproduce the computational and networking characteristics of 6G networks. Table I lists the key parameters used in the simulation environments. To capture spatial dynamics, we map real-world user mobility traces from Zenodo [23] to our grid-based system that reflects real-world trajectories. Complementing mobility, requests are derived from the MMMU-Pro dataset [24], which provides diverse AI tasks. By probabilistically associating tasks to users over time, we reproduce the heterogeneous, unpredictable workloads typical of 6G scenarios.

We evaluate HyPE against state-of-the-art, optimization-based, and random baselines. As strong comparators, we

Table I
SIMULATION PARAMETERS AND SYSTEM CONFIGURATIONS

Parameter	Range / Value	Unit
Service duration ($\mathcal{T}_s^{du}$)	2-6	Time Frames
User request bandwidth ($\check{\mathcal{L}}_r$)	100 – 1000	Mbps
Pre-/Post-processing capacity ($\check{\mathcal{C}}_{r,f}^{\phi}$)	1 – 5	GFLOPS
Inference capacity ($\check{\mathcal{C}}_r^{in}$)	30 – 100	GFLOPS
Pre-/post-process step (π_s^{ϕ})	1-2	-
Personalization continuity factor (δ)	0.65 – 0.9	-
Minimum output quality ($\check{\mathcal{Q}}_r^{ou}$)	0.5 – 0.8	-
Latency requirement ($\check{\mathcal{Q}}_r^{la}$)	50 – 100	ms
Max packet size ($\check{\mathcal{Z}}_{r,t}$)	256 – 2048	Bytes
Node quality ($\overline{Q}_n$)	0.6 - 1	-
UAV compute capacity ($\widehat{\mathcal{C}}_n^{ca}$)	80 – 150	GFLOPS
Cloud/Edge compute capacity ($\widehat{\mathcal{C}}_n^{ca}$)	500–1000	GFLOPS
Link bandwidth capacity ($\widehat{\mathcal{L}}_l$)	100 – 2000	Mbps
MAP learning rate (D3QL)	10^{-4} – 10^{-3}	-
MAP replay buffer size	1k – 10k	transitions
LEAD LLM setup	Gemini-2.5-Flash; 1 API/frame; temp. 0.1	-
LEAD LLM cost	\$0.30,2.50/M	in,out
LLM prompt length	1k-4k	tokens
LLM few-shot examples ($\mathcal{K}$)	1-3	-
Break weight / LBSP bias (κ, γ)	0.1 – 1.0	-
Areas ($\boldsymbol{\mathcal{A}}$) / Edge-cloud/UAV nodes	5x5 / 15 / 5	-

include AD-SAC [15], a hybrid offloading/power-allocation method that minimizes latency under accuracy constraints via accuracy-aware inference offloading in UAV-satellite networks, and JAAPD-D [16], which jointly balances latency, acceptance rate, and resource orchestration with objectives overlapping ours. For a fair comparison, all methods are executed under the same constraints, such as mobility traces, request realizations, and latency/fidelity demands. To integrate them into our setting, AD-SAC is used for UAV trajectory and inference assignments under its original latency/fidelity design, while JAAPD-D is adapted to the same aerial-terrestrial network as a demand-driven placement baseline. We also benchmark an optimization-based formulation (OF) solved via Gurobi and a random strategy that randomizes UAV trajectories, function placements, and inference assignments.

Our experiments assess how key 6G capabilities are sustained as the density grows [25]. The first scenario examines coverage, wherein parts of the area is served by fixed infrastructure, requiring UAV repositioning for service delivery. As shown in Fig. 3, the Optimize oracle (full demand knowledge) achieves 100% acceptance at light loads and 75% at heavy loads, exposing UAV unreachability due to adjacency-limited movements. The Random baseline performs worst as UAVs fail to track moving users, causing a sharp degradation. HyPE matches optimal acceptance with 5 users and sustains 61% under heavy load, a strong suboptimal result driven by resource saturation and MAP prediction errors. AD-SAC drops to 53% because its agent optimizes UAV placement without explicit allocation or per-request admission, inducing contention and latency violations. JAAPD-D fares lower (42%) as its demand-driven heuristic lacks foresight for proactive UAV repositioning, leading to longer paths and reduced acceptance. By contrast, HyPE's history-driven LEAD and prediction-aware planning better accommodate requests under load.

The second scenario examines how HyPE balances E2E latency and AI-enabled capabilities [25] as demand increases, comparing Optimize (oracle), Random, AD-SAC, JAAPD-D, and HyPE. Fig. 2.a reports latency distributions (min/max/median with means) for accepted requests: Optimize achieves the lowest latency via full demand knowledge and coordinated UAV control; Random shows deceptively lower mean latency than AD-SAC and JAAPD-D as it accepts few requests (mostly from fixed nodes), biasing results; HyPE balances latency and QoAIS in which at light load, LEAD and SET favor high-fidelity edge inference, slightly increasing latency (92 ms vs. 86 ms for AD-SAC at 5 users), while at higher loads LBSP shortens paths and PGA shifts tasks to UAVs near users, reducing latency with modest, acceptable quality loss; AD-SAC minimizes latency at low load via smaller UAV-hosted models but escalates under heavy load due to lack of bandwidth control, whereas LEAD's prediction-aware trajectories keep HyPE within bounds; JAAPD-D starts slightly lower through dynamic allocation but degrades without proactive UAV repositioning as density rises.

Fig. 2.b evaluates AI-enabled personalization (continuity of user experience) and per-request output fidelity. Optimize leads by jointly optimizing placement, UAV movement, and historical affinity, while Random fails on both metrics due to unguided mobility and placement. HyPE achieves robust personalization and competitive fidelity by combining demand prediction with LEAD pre-positioning and PGA's continuity-aware placement toward spatially proximal nodes; personalization increases by 19% under load, while fidelity decreases (yet acceptable) by 45% as tasks shift to UAVs (intended trade-off given δ's personalization weighting). Under heavier load, HyPE shifts a larger fraction of inference tasks to distilled models to preserve service continuity and timely response delivery when fixed edge resources become saturated. Rather than a collapse in service quality, this is a controlled shift in which HyPE maintains interactive latency and personalization while still retaining competitive fidelity. Compared to AD-SAC, HyPE attains higher QoAIS by embedding personalization into reward and placement logic; AD-SAC degrades on both due to smaller UAV LLMs and lack of affinity modeling. JAAPD-D likewise underperforms HyPE (personalization 12 vs. 26) because personalization is absent from its objectives.

VI. CONCLUSION

This paper addressed personalized AI services in UAV-assisted 6G networks under mobility and constrained edge resources. We introduced HyPE, a hybrid predictive-in-context-learning framework that integrates mobility-aware demand prediction, LLM-guided decision-making, and heuristic function placement and routing to maintain stringent latency and QoAIS guarantees. Empirically, HyPE delivers elastic provisioning across the three axes of 6G capabilities: trajectory-managed UAVs extend coverage to 91% of the oracle while complementing fixed edges; accepted-request E2E latency remains within 6G-class interactive bounds; and distributed aerial-terrestrial inference attains 94% of

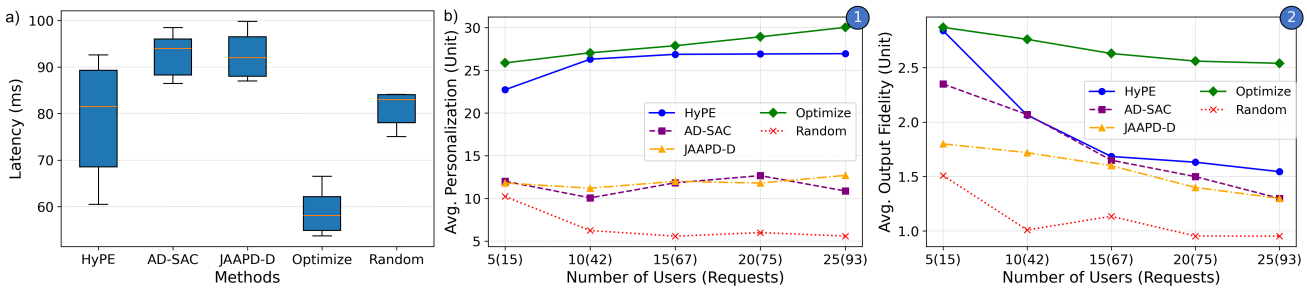

Figure 2. Comparison of HyPE with Optimize (OF), AD-SAC [15], JAAPD-D [16], and random methods in terms of a) latency, and b) AI-enabled capabilities (QoAIS as Personalization and Fidelity) as the number of users/requests (connection density) expands. The lower bounds in (a) mainly correspond to light-load periods with fewer users and requests, reflecting underutilized resources and minimal contention.

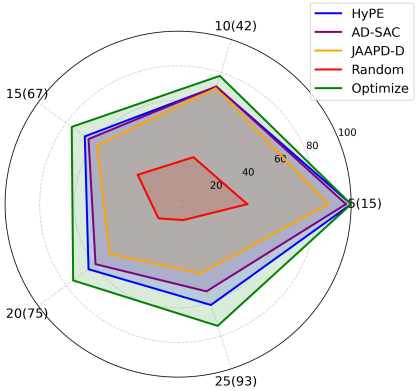

Figure 3. Coverage comparison in terms of accepted requests as the number of users (requests) increases.

optimal output fidelity. By eschewing combinatorial enumeration, HyPE achieves polynomial per-frame complexity $\mathcal{O}\big(\mathcal{T}(\mathcal{NU}+\mathcal{NUF}+\mathcal{UN}^2\log\mathcal{N})\big)$ trading worst-case optimality for real-time scalability. Future work includes energy-aware UAV trajectory planning, cross-domain orchestration, and continual learning for long-term personalization. As HyPE leverages pre-trained LLMs, future deployment should account for inference costs on-device or in federated setups.

Acknowledgment

The research work is supported in part by the Federal Ministry of Research, Technology, and Space (BMFTR), Germany, through the Project 6GEM+ under Grant 16KIS2411; by the European Union's Horizon Europe research and innovation programme under the 6G-Path project (Grant No. 101139172); and the Research Council of Finland 6G Flagship Programme under Grant No. 369116.

References

[1] M. Farhoudi, M. Shokrnezhad, and T. Taleb, "Service registration, indexing, discovery, and selection: An architectural survey toward a GenAI-driven future," *IEEE Access*, vol. 13, pp. 209 680–209 722, 2025.

[2] H. Mazandarani *et al.*, "Adaptive multiple access and service placement for generative diffusion models," in *Proc. IEEE Global Telecommun. Conf.*, Taipei, Taiwan, 2025, p. 5.97.

[3] H. ang Gao *et al.*, "A survey of self-evolving agents: On path to artificial super intelligence," 2025.

[4] T. Chen *et al.*, "Optimization of quality of AI service in 6G native AI wireless networks," *Electronics*, vol. 12, no. 15, 2023.

[5] Q. Zhang *et al.*, "A survey of graph retrieval-augmented generation for customized large language models," 2025.

[6] Y. Chen, J. Zhao, and H. Han, "A survey on collaborative mechanisms between large and small language models," 2025. [Online]. Available: https://arxiv.org/abs/2505.07460

[7] H. Hao, J. Han, C. Li, Y.-F. Li, X. Yue *et al.*, "Rap: Retrieval-augmented personalization for multimodal large language models," 2025.

[8] J. Wu *et al.*, "Personalized multimodal large language models: A survey," 2024.

[9] M. Farhoudi, M. Shokrnezhad *et al.*, "Discovery of 6G services and resources in edge-cloud-continuum," *IEEE Netw.*, vol. 39, no. 3, pp. 223–232, 2025.

[10] T. Taleb *et al.*, "6G system architecture: A service of services vision," *in ITU journal on future and evolving technologies*, vol. 3, no. 3, pp. 710–743, Dec. 2022.

[11] H. Mazandarani *et al.*, "Semantic-aware dynamic and distributed power allocation: a multi-UAV area coverage use case," 2025.

[12] H. Hao, C. Xu, W. Zhang, S. Yang, and G.-M. Muntean, "Joint task offloading, resource allocation, and trajectory design for multi-UAV cooperative edge computing with task priority," *IEEE Trans. Mobile Comput.*, vol. 23, no. 9, pp. 8649–8663, 2024.

[13] S. Raj, R. Mittal, H. Gupta, S. Yogesh SimmhanRaj *et al.*, "Adaptive heuristics for scheduling DNN inferencing on edge and cloud for personalized UAV fleets," *Future Generation Computer Systems*, vol. 173, p. 107874, Dec 2025.

[14] L. Jin *et al.*, "Adaptive task offloading and resource management for vehicular edge computing," *IEEE Trans. Veh. Technol.*, pp. 1–14, 2025.

[15] H. Yan *et al.*, "Accuracy-aware MLLM task offloading and resource allocation in UAV-assisted satellite edge computing," *Drones*, vol. 9, no. 7, 2025.

[16] Y. Hu *et al.*, "AI service deployment and resource allocation optimization based on human-like networking architecture," *IEEE Internet Things J.*, vol. 11, no. 14, pp. 24 795–24 813, 2024.

[17] M. Farhoudi, M. Shokrnezhad, S. Kianpisheh, and T. Taleb, "Deep learning based service composition in integrated aerial-terrestrial networks," in *International Conf. on Net. Softwarization*, 2025, pp. 204–208.

[18] M. Farhoudi *et al.*, "QoS-aware service prediction and orchestration in cloud-network integrated beyond 5G," in *Proc. IEEE Global Telecommun. Conf.*, Dec. 2023, pp. 369–374.

[19] K. C. Wibisono and Y. Wang, "From unstructured data to in-context learning: Exploring what tasks can be learned and when," in *Advances in Neural Information Processing Systems*, 2024, pp. 16 369–16 405.

[20] A. Kong *et al.*, "Better zero-shot reasoning with role-play prompting," *arXiv preprint arXiv:2308.07702*, 2023.

[21] M. Shokrnezhad *et al.*, "An autonomous network orchestration framework integrating large language models with continual reinforcement learning," *IEEE Commun. Mag.*, vol. 63, no. 8, pp. 78–84, 2025.

[22] S. Colvin *et al.*, "Pydantic: Data validation using python type hints," MIT License. [Online]. Available: https://github.com/pydantic/pydantic

[23] T. Yabe *et al.*, "Metropolitan scale and longitudinal dataset of anonymized human mobility trajectories," 2023. [Online]. Available: https://arxiv.org/abs/2307.03401

[24] X. Yue *et al.*, "Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark," 2025.

[25] R. I. R. Sector, "M.2160-0: Framework and overall objectives of the future development of IMT for 2030 and beyond," Nov. 2023, available: https://www.itu.int/rec/RREC-M.2160/en.