# LSTM-based Mobility Prediction for Proactive Service Migration in Vehicular Edge Computing

Paulo J. Araújo
*Algoritmi Research Centre/LASI*
*University of Minho, Portugal*
paulo.araujo@algoritmi.uminho.pt

Helena Fernández López
*CESGA - Galicia Supercomputing Center*
*Santiago de Compostela, Spain*
hfernandez@cesga.es

Alexandre Santos
*Algoritmi Research Centre/LASI*
*University of Minho, Portugal*
alex@di.uminho.pt

***Abstract*—The increasing demand for mobile services requiring ultralow latency, high bandwidth, and reliable availability has positioned Multi-access Edge Computing (MEC) as a vital component of modern mobile networks. However, as users move across the network, orchestrating service placement and migration becomes essential to ensure this proximity remains an advantage. In this paper, we present a proactive orchestration framework for MEC-enabled vehicular environments, supported by a Long Short-Term Memory (LSTM) model that predicts the serving base station across multiple horizons. The framework is evaluated using a simulation setup built on Simu5G and SUMO, driven by augmented real-world vehicular traces processed through the full MEC stack. A data collection system interfacing with the Location and Radio Network Information services of each MEC Host feeds mobility and radio network information to the MEC Orchestrator, which coordinates application migration across MEC Hosts. Two orchestration strategies are evaluated: a reactive baseline triggered by serving cell changes and a proactive approach driven by ML-predicted user mobility. Results show that the LSTM model achieves high accuracy across different time steps, significantly outperforming a naive baseline, enabling proactive service migration. Incorporating the proactive strategy into the orchestrator yields a statistically significant Round-Trip Time (RTT) reduction over the reactive baseline under matched conditions. Despite this, results reveal that correctly predicted Base Station (BS) changes exhibit notably lower RTT, while incorrect predictions disproportionately degrade performance. This highlights prediction accuracy as the primary bottleneck and motivates the need for prediction-aware orchestration mechanisms.**

***Index Terms*—vehicular edge computing, machine learning, mobility prediction, service migration**

## I. Introduction

The growing complexity of mobile applications, combined with the rigorous latency and bandwidth requirements, has driven a fundamental shift in how mobile networks are designed. MEC has emerged as a key architectural solution to bridge the gap between User Equipment (UE)s — such as smartphones and connected vehicles — and centralized cloud infrastructures. By placing computational resources at the network edge, closer to end users, MEC enables latency-sensitive applications to execute with reduced delay while leveraging localized contextual information [1].

The vehicular domain stands out as a particularly compelling use case, where applications such as autonomous driving, cooperative perception, and intelligent traffic management demand both ultralow latency and high reliability, giving rise to the concept of Vehicular Edge Computing. However, as vehicles navigate the network, the proximity advantage that makes edge computing effective can degrade quickly. When a vehicle hands over to a base station associated with a different MEC Host (MEH), the application instance remains at its original location, forcing traffic through longer paths and increasing latency. Reactive approaches, where migration is triggered only after a handover is detected, leave the user with degraded Quality of Service (QoS) during the entire migration window. Proactive strategies that anticipate handovers and schedule migrations in advance can eliminate this penalty.

We present a predictive migration framework using an LSTM model to predict serving base stations across future horizons. The framework is evaluated via Simu5G and SUMO simulation, where vehicular traces are processed through the MEC stack, including Location and Radio Network Information Services. Results show that the LSTM model achieves high accuracy across all horizons, enabling statistically significant RTT reductions over the reactive baseline. Critically, the analysis reveals that mispredictions disproportionately degrade performance, highlighting prediction accuracy as the primary bottleneck and motivating the need for prediction-aware orchestration mechanisms.

The remainder of this paper is organized as follows. Section II examines the most recent machine-learning methods used for predicting handovers and forecasting mobility as well as their application in migrating applications within edge environments. The system model in Section III describes the vehicular edge computing architecture, data sources, and migration challenges from user mobility. Section IV details the predictive migration framework, which includes an LSTM-based serving-cell predictor across multiple time horizons and a proactive orchestration strategy that leverages these predictions to schedule migrations ahead of handover events.

This work has been supported by the European Union under the NextGenerationEU, through a grant of the Portuguese Republic's Recovery and Resilience Plan (PRR) Partnership Agreement, within the scope of the project BE.NEUTRAL – "Agenda da Mobilidade para a neutralidade carbónica das cidades", aiming at the development, industrialization and operation of new carbon-zero mobility cyber-physical products that will accelerate the transition to carbon-neutral cities by 2030 (Project ref. nr. 35 - C644874240-00000016; Total project investment: 221.376.867,63 Euros; Total Grant: 128.619.400,50 Euros).
ISBN 978-3-903176-82-9 © 2026 IFIP

Section V presents the simulation environment, evaluates the predictive model across multiple horizons, and analyzes the impact of the proactive migration strategy on end-to-end latency. Finally, Section VI concludes the paper and discusses future improvements to the current proactive orchestration strategy.

## II. Related Work

Seamless service migration in MEC-enabled vehicular environments remains an active research challenge [2]. Existing approaches range from simple yet effective reactive strategies to more sophisticated solutions that often incorporate mathematical models to either support or replace the MEC Orchestrator (MEO) decision process. These approaches can be generally divided into those that solely concentrate on forecasting user mobility or handover events and those that integrate such forecasts with an orchestration strategy to facilitate proactive service migration decisions.

Bernad et al. [3] proposed a stacked LSTM architecture capable of predicting serving cell assignments up to 7 seconds ahead. While the model achieved high accuracy, F1 scores were not reported, and no discussion was provided on how the predictions could be integrated into a practical orchestration strategy. A similar strategy was used by Belhadj et al. in [4], where an LSTM-based next-cell classification approach in a vehicular scenario generated with SUMO was applied. However, it uses only cell identity as the input feature and predicts a single time step ahead.

A two-stage approach combining Reference Signal Received Power (RSRP) regression and serving cell classification was proposed in [5], using LSTM and XGBoost models with transfer learning between simulation and a real 5G testbed, although no integration with a MEC orchestration strategy was considered. In [6], another two-stage framework combining XGBoost handover classification and a Deep Q-Network (DQN) agent was proposed to dynamically tune handover parameters in ultra-dense networks. The handover classification stage is particularly relevant because accurate handover type detection can serve as a valuable input signal for proactive MEC migration decisions.

A container migration solution was presented in [7] by Zhang et al., where the authors applied trajectory prediction with Recurrent Neural Networks (RNN), but only for the next time slot. Although a real bus dataset was used, it comprised only 84 routes and followed highly structured mobility patterns. Furthermore, container dependencies are modeled through Activity On Edge (AOE) networks and the placement problem is solved via a metaheuristic approach. A proactive stateful microservice migration framework was described in [8], where LSTM trajectory prediction feeds a DQN agent responsible for selecting the optimal destination edge node. However, the work remains at the design stage, presenting only an architectural proposal without experimental validation of the proposed Reinforcement Learning (RL) migration policy. Dalgkitsis et al. [9] integrate a Convolutional Neural Network (CNN) for predicting mobility with a service orchestration strategy enhanced by a Genetic Algorithm to realize the Follow-Me Cloud concept. A real taxi dataset from the San Francisco Bay Area was used.

Zhao et al. [10] propose MAPSM, combining ensemble mobility prediction with a proactive container pre-migration procedure to minimize end-to-end delay, though the mobility model relies on sparse social network check-ins rather than continuous vehicular traces. The algorithm proposed in [11] addresses the joint problem of migration and resource allocation in MEC by focusing on minimizing migration events rather than maximizing resource usage. However, mobility is simplified to predetermined routes towards known destinations, limiting applicability to scenarios where movement patterns are unpredictable.

Existing studies rarely detail how location and radio network information are collected and utilized within the MEC system, both for model training and deployment. Notably, works relying on real-world datasets often fail to account for the limited variability of such traces, which constrains the generalization capacity of the resulting models. Moreover, these datasets are generally limited in size, which makes it challenging to evaluate the performance of the proposed solutions under realistic network conditions with a large user base. In this work, traces based on a real mobility dataset are augmented and processed through a complete simulation environment that replicates the full MEC stack, including the MEO, network conditions, and service migration mechanics. The resulting prediction model is designed to be integrated directly into the existing MEO pipeline, enabling proactive service migration across multiple prediction horizons.

## III. System Model

The fifth generation of mobile networks (and beyond) introduced strict requirements for latency and reliability with use cases framed in the eMBB and URLLC categories. To satisfy these demands, MEC is considered a fundamental architectural paradigm that ensures resources are available as close as possible to end-users. In the Vehicular-to-Everything (V2X) paradigm, vehicles are constantly moving and rely on MEC Hosts to run latency-critical services. This section describes the system model for a vehicular edge computing environment, including the available data sources and the migration challenges that arise from user mobility.

### A. Vehicular Edge Computing Architecture

In this paper, we consider a vehicular edge computing environment consisting of $N$ base stations distributed across an urban area, each with a limited coverage area that overlaps with neighboring base stations to ensure seamless handovers. While a single MEC Host could theoretically be associated with multiple access points, potentially employing different technologies, it is assumed that each base station is associated with a MEH with limited computing resources, responsible for hosting latency-sensitive applications for nearby mobile users. In the context of this work, vehicles are considered as UEs of the network.

At the system level, the MEO, as defined by the ETSI MEC reference architecture, is responsible for managing the lifecycle of application instances across all MEHs, including their instantiation, migration, and removal. The MEO has access to the MEHs resource information as well as insights produced by platform-level services, which can be used to apply informed application placement and migration decisions.

Among these services, the Location Service (LS) and the Radio Network Information Service (RNIS), both defined under the ETSI MEC standard, are of particular relevance. These services provide real-time telemetry about connected users and network conditions, which, as detailed further, serve as the primary data sources for the proposed predictive migration framework. The produced data can be fed to external prediction systems operating at the system level, enabling data-driven decision-making for application lifecycle management.

### B. Data Sources

At regular $\Delta t$ intervals, services running at the MEH level collect an observation for each active UE connected to a base station within the considered environment. These observations are then made available to system-level entities. For a given UE $i$ at time $t$, the observation vector is defined as in (1). It comprises its spatial coordinates $(p_x, p_y)$, speed $v$, direction of movement relative to the north $\theta$, distance to the serving base station $d$BS, its identifier $\text{id}_{\text{BS}}$, and the Layer 2 uplink and downlink delay from the UE to its serving base station, denoted as $\tau_{\text{UL}}$ and $\tau_{\text{DL}}$, respectively.

$$\mathbf{x}_i(t) = \{(p_x, p_y),\, v,\, \theta,\, d_{\text{BS}},\, \text{id}_{\text{BS}},\, \tau_{\text{UL}},\, \tau_{\text{DL}}\} \tag{1}$$

Since mobility patterns and network conditions evolve dynamically, a single observation may not provide sufficient information to anticipate future handovers in the serving base station. For this reason, over the past $k$ reporting intervals, the system maintains a temporal observation sequence for each UE, as defined in (2). This sequential representation captures trends in speed, direction, and network conditions, providing a richer context for prediction compared to single-snapshot approaches.

$$\mathbf{X}_i = [\mathbf{x}_i(t-k+1),\, \ldots,\, \mathbf{x}_i(t)] \tag{2}$$

Additional features can be obtained from this initial set of observations by applying feature engineering to enrich the final input vector. The specific features used in this work are detailed in Section V.

### C. The Migration Problem

In vehicular environments, particularly in urban areas, vehicles and pedestrians are constantly moving across the infrastructure, frequently transitioning between the coverage areas of different base stations. When a transition results in a handover to a base station associated with a different MEC Host, the UE can still maintain its connection to the original host. However, traffic must now go through longer paths, either through the core network or via inter-base-station connections. This conflicts directly with the single-hop connectivity paradigm envisioned by the Follow-me Cloud concept, resulting in increased delay [12].

To restore proximity, the MEO must transfer application instances from the original host to the one associated with the most recently serving base station, a process known as service migration. This process includes multiple steps and, depending on the underlying technology, can take a variable amount of time to complete, denoted as $\Delta_{migration}$. We consider stateless applications, where migration involves instantiating the application at the target host and notifying the UE of the new endpoint address. The transfer of session data in stateful scenarios, typically handled by the applications themselves, is outside the scope of this work.

In a reactive approach, the orchestrator starts the migration process only after a handover has been detected. During the $\Delta_{migration}$ window, the UE is served by a MEC host no longer co-located with its serving gNB, forcing traffic through the core network and degrading the QoS. To address this limitation, the orchestrator may adopt a proactive approach, leveraging predictive techniques to anticipate upcoming handovers. With this information, migration can be initiated early enough to ensure the application is available at the target host by the time the UE completes its transition.

## IV. Predictive Migration Framework

Dynamic vehicular edge environments are characterized by the constant movement of entities across the infrastructure. In such settings, seamless service continuity is critical for maintaining high-quality network metrics and preventing connection loss. Although the MEO manages service instantiation based on direct knowledge of server availability, it natively lacks access to broader contextual data, including specific user mobility patterns and network health. This type of information can assist in managing edge resources and support the migration of applications and services across the infrastructured network. For mission-critical use cases, such as self-driving cars and vehicular safety applications, maintaining ultralow latency is strictly required, and the system must ensure that necessary applications seamlessly follow the user, always running on a nearby MEH.

To address this challenging problem, the new proposed framework combines a mobility prediction model with a proactive migration strategy.

### A. Serving Cell Prediction

The new prediction model, presented and analysed in this paper, estimates, for each UE, the base station that will best serve it at multiple points in the near future. Given the observation sequence $\mathbf{X}_i$ defined in Section III-B, the model produces predictions for $H$ future time horizons, as in (3). Each prediction corresponds to one of the $N$ base stations or an additional out-of-coverage class, representing UEs leaving the coverage area, at time $t + h \cdot \Delta t$.

$$P(\mathbf{X}_i) \rightarrow \{\hat{s}_{t+1},\, \hat{s}_{t+2},\, \ldots,\, \hat{s}_{t+H}\} \tag{3}$$

This constitutes a multi-output classification problem with $N+1$ possible classes for each of the $H$ output horizons, enabling the system to identify the specific timestep at which a change will likely occur and use this to schedule the migration process.

Since vehicles will move around the urban area over time, the prediction model must be able to capture temporal dependencies. For this reason, an LSTM network - a type of RNN able to retain information from previous time steps through its internal memory structure - is used as the core of our model. Similar architectures have been successfully applied to serving-cell prediction in simulated vehicular scenarios [3] [4].

As presented in Figure 1, the proposed model processes the input sequence $\mathbf{X}_i$ through a first LSTM layer that returns its output sequence to a second LSTM layer, producing a final hidden state representing a summary of the temporal dynamics inherent in the input. Furthermore, the resulting representation passes through a fully connected layer with ReLU activation, resulting in a shared representation for the final stage, which is formed by $H$ independent output heads, one for each future horizon. Each output head consists of a dense layer with softmax activation, returning a probability distribution over the $N+1$ possible classes, as previously described. Dropout layers were implemented across the model layers to avoid overfitting and improve generalization.

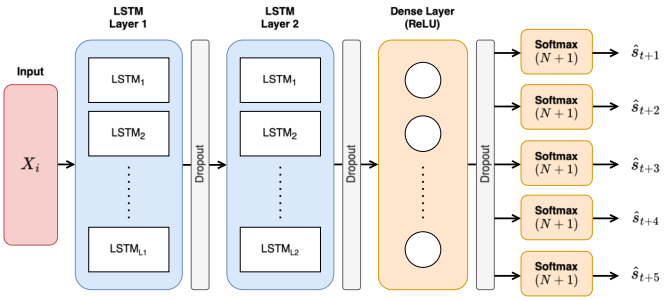



Fig. 1: Implemented model's architecture.

Finally, the model is trained end-to-end by minimizing the sum of the categorical cross-entropy losses across all $H$ output horizons, as shown in (4).

$$\mathcal{L} = \sum_{h=1}^{H} \mathcal{L}_{\text{CE}}(\hat{\mathbf{s}}_{t+h}, \mathbf{s}_{t+h}) \tag{4}$$

### B. Proactive Migration Strategy

Unlike reactive orchestration, which moves application instances only after being triggered, a proactive approach relies on predictive modeling to anticipate network changes and user trajectories. By forecasting exactly when a user will reach a new MEH, the system can start the migration of containers or virtual machines in advance, ensuring that the service is seamlessly available upon the users' arrival. Therefore, based on the prediction output defined in (3), the goal of our MEO is to proactively schedule migrations before handovers occur.

**Algorithm 1** Proactive Migration Scheduling

1: **for each** active UE $i$ **do**
2: $h^* \leftarrow \min\{h : \hat{s}_{i,t+h} \neq s_i(t) \wedge \hat{s}_{i,t+h} \leq N\}$
3: **if** no such $h^*$ exists **then**
4: **continue**
5: **end if**
6: $t_{trigger} \leftarrow \max(t, t + h^* \cdot \Delta t - \Delta_{migration})$
7: schedule MIGRATE$(i, m_{s_i(t)}, m_{\hat{s}_{i,t+h^*}})$ at $t_{trigger}$
8: **end for**

The general behavior is shown in Figure 2. For each active

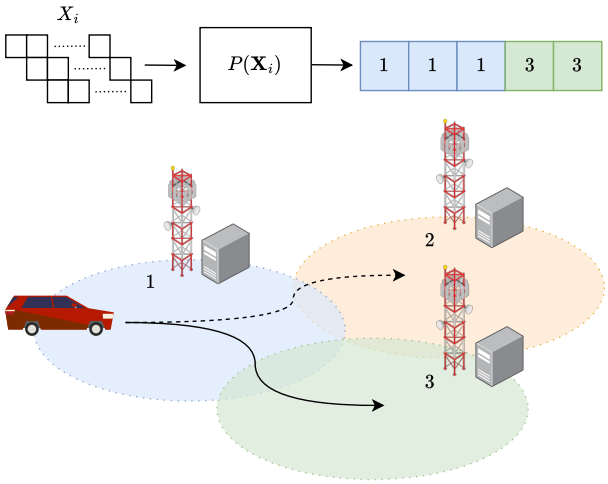



Fig. 2: Model predicting $H$ future time horizons.

UE, the system examines the sequence of predicted serving cells across the $H$ horizons to determine whether a handover is imminent. If confirmed, the earliest horizon at which it will occur is identified. The migration is then scheduled such that it is completed by the time the UE reaches the target base station.

The scheduling mechanism is formalized as follows. Let $U$ be the set of active UEs, with $i \in U$, and $s_i(t)$ be the serving cell of UE $i$ at time $t$. Furthermore, $m_s$ denotes the MEH associated with serving cell $s$ and $\Delta_{migration}$ the time it takes for a given application instance to change from $m_{s_i(t)}$ to $m_{\hat{s}_{i,t+h^*}}$, where $h^*$ is the earliest predicted horizon at which the handover is expected. The MEO computes a trigger time $t_{trigger}$ such that the migration is completed by the time the UE reaches the target base station, as defined in Algorithm 1. If the trigger time has already elapsed, the migration is initiated as soon as possible. The condition $\hat{s}_{i,t+h} \leq N$ in line 2 filters out-of-coverage predictions, as proactively removing an application while the UE is still communicating would disrupt ongoing sessions. UE departures are instead handled reactively via an inactivity timeout. If an out-of-coverage prediction occurs at an intermediate horizon, the algorithm skips it and continues searching for a valid handover target.

## V. Performance Evaluation

This section presents the experimental assessment of the proposed system. We start by outlining the simulation environment and the experimental setup, then proceed to discuss the training process and evaluate the performance of the predictive models, and finally, we analyze the outcomes of the proactive migration.

### A. Simulation Environment

The experimental setup was developed by integrating various modules from existing frameworks that utilize Omnet++. An example of this is Simu5G, which facilitates the simulation of intricate 5G network infrastructure [13]. The key elements of this ETSI MEC-compliant initiative include MEH, MEC Platform (MEP), Virtualisation Infrastructure Manager (VIM), and MEO. Although Simu5G provides a comprehensive simulation baseline, several extensions were required to support proactive orchestration [14]. Notably, the MEO lacked native application migration capabilities, which had to be implemented. Additionally, no built-in mechanism existed to retrieve Location Information or Radio Network Information at the system level. To address this, a data collection module was developed that interfaces with the LS and RNIS of each MEH, periodically gathering all available information on the UEs connected to the BS associated with that MEH. This information is then aggregated, processed, and exposed to other system components — in this case, the MEO – in a manner similar to that presented by Araújo et al. in [15].

The main simulation tool was complemented by three additional frameworks: INET, Veins, and SUMO. INET is a well-established open-source model library for OMNeT++ that provides implementations of a wide range of Internet protocols and network architectures, forming the communication foundation of the simulation. The Veins project builds upon this by enabling interaction with SUMO through its Traffic Control Interface (TraCI), allowing the framework to manage mobile nodes under realistic, dynamically evolving traffic conditions. SUMO is a dedicated traffic simulator designed to model the movement of individual vehicles within a defined road network.

The most important entities of the simulated system are depicted in Figure 3, together with their interconnections and respective versions. The components highlighted in red represent those most relevant to this study, as they required the most significant modifications. The MEO is responsible for controlling which application is instantiated at which MEH and at what time, ensuring that seamless migration between MEH is possible. The User Application LifeCycle Management Proxy (UALCMP) facilitates communication between the MEO and the UE, serving in this setup to inform the UE about updates to service endpoints during migration.

The migration process begins with the instantiation of the application at a new MEH, while the original instance remains active and continues serving the UE. Once the new instance is ready, the MEO notifies the UE of its new service endpoint, waits for confirmation, and subsequently terminates

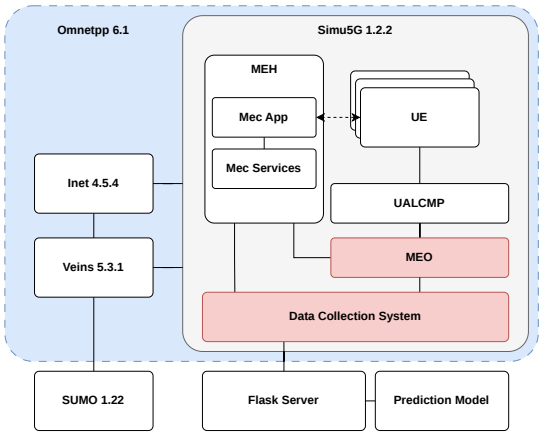


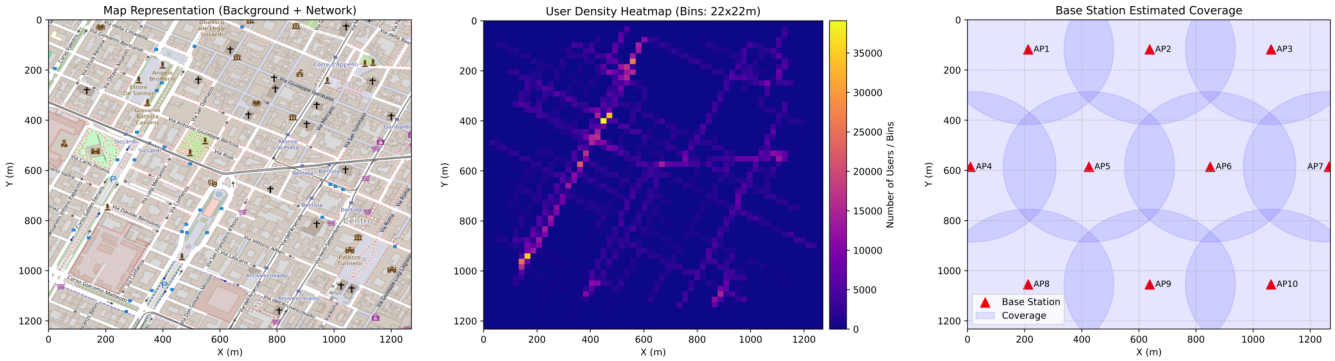

Fig. 3: Simulation tools used to produce the dataset and test the migration strategy.

the original instance at the source MEH. Throughout this process, UEs continuously exchange messages, measuring all components of the Round-Trip Time (RTT).

Migration decisions are triggered by the Data Collection System, which supports two operating modes. The first, named MigrateOnChange (MoC), is reactive: it continuously monitors UE location and issues a migration notification to the MEO upon detecting a serving cell change. To prevent ping-pong handover (PPHO), it incorporates a configurable time-to-trigger (TTT) mechanism analogous to those used in standard handover procedures, introducing a stabilization period before any notification is dispatched. The second mode, MigrateOnPrediction (MoP), is proactive: rather than reacting to observed changes, it delegates the migration decision to an external inference server that anticipates serving cell changes in advance, as detailed in Section IV. Additionally, a data logging mode is available, allowing the system to record all collected information to files for offline analysis or model training.

### B. Experimental Scenario

The experimental scenario was designed to enable the implementation and evaluation of the proposed proactive orchestration strategy while also facilitating its benchmarking against alternative strategies. To ensure realistic and reproducible results, a well-known simulation scenario was used as the mobility based on our tests, namely the Turin SUMO Traffic (TuST) simulation scenario [16]. The original mobility traces were derived from a subsection of the TuST scenario, cropped to an area of approximately 1.2×1.2 km, depicted in Figure 4a. To increase scenario diversity, 75 individual runs were generated using SUMO scripting tools combined with manually adjusted vehicle density scaling factors, following a normal distribution (mean = 0.3). Of the 75 generated runs, 45 were used to train, validate, and test the predictive models, whereas the remaining 30 were reserved for end-to-end system evaluation, ensuring independence between model development and system-level performance assessment.




(a) Snapshot of the considered area. (b) User density for 45 simulation runs. (c) Position of the Base Stations in the map.

Fig. 4: From map to density prediction.

Given the high traffic volume in the region, the simulation window was narrowed from a full day (86,400 s) to a 3,600 s interval spanning the 7–8 AM peak hour, resulting in approximately 1300 vehicles per run. Figure 4b illustrates the aggregated mobility heatmap of the 45 training runs, which allows the identification of the most critical areas in terms of UE density.

To maintain continuous coverage across the entire area, 10 base stations were strategically placed, as depicted in Figure 4c. Following the 3GPP TR 38.901 Urban Macro (UMa) propagation model, each base station achieves a signal range of approximately 350 m. Network delays between system elements were also manually configured to reflect realistic mobile network conditions. Since migration is only worthwhile if the nearest MEH offers a clear latency advantage, the delay model was parameterized so that proximity to a MEH translates into a meaningful and immediate reduction in response time, giving the MEO a strong incentive to relocate service instances closer to the UE. The adopted delay values are shown in Figure 5. A BS and its co-located MEH have a delay of 2 ms between them, whereas accessing a non-co-located MEH experiences a minimum delay of 22 ms, making the locality advantage explicit.

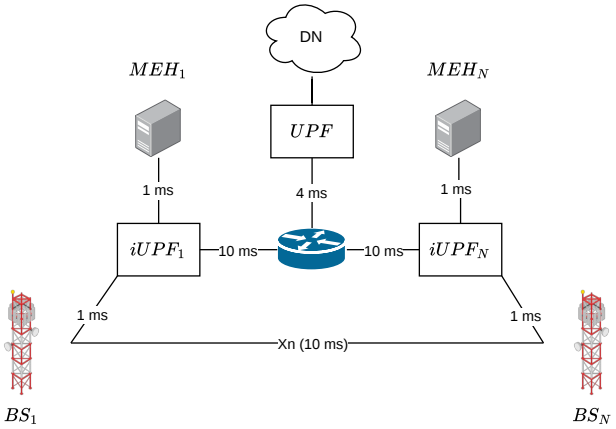



Fig. 5: Delays between elements of the scenario.

The time required to transfer an instance from one host to another varies significantly based on the data size and the technology employed, whether it be Virtual Machine (VM)s, containers, or containers over VMs. To streamline the simulation process, we assumed a migration time ($\Delta_{\text{migration}}$) of 12 s. This value was chosen based on the available literature, as the average maximum value for a container running in a VM to move to another VM, as in [17]. This value also falls within the prediction horizon of the proposed model, which anticipates mobility changes up to 5 time steps $\Delta t$ of 3 s ahead (15 s total). The relevant simulation parameters are listed in Table I.

TABLE I
Relevant simulation parameters per run

| Parameter | Value |
|---|---|
| Simulation time (s) | 3600 |
| Total number of vehicles | ≈1300 |
| $\Delta t$ (s) | 3 |
| $\Delta_{\text{migration}}$ (s) | 12 |
| TTT ($\Delta t$) | 3 |
| Number of MEHs/BSs | 10 |
| Considered area (km$^2$) | ≈1.44 |

### C. Model Training and Performance

The dataset was generated by executing 45 simulation runs while the Data Collection System was set to Data Saving mode, mirroring the exact conditions of the live system. Each run provided a comprehensive record of UE mobility and network interactions, with measurements captured at 3 s intervals.

These runs were randomly partitioned at the run level into 31 runs for training, 7 for validation, and 7 for testing. The raw and processed datasets are publicly available [18]. This division resulted in approximately 1.5 million training sequences, 339,000 validation sequences, and 346,000 test sequences. To validate the choice of LSTM as the recurrent architecture, we also evaluated a variant using Gated Recurrent Unit (GRU) cells, which offer a simpler gating mechanism with fewer parameters. Additionally, a naive persistence baseline that always predicts the current serving cell for all horizons is included as a performance lower bound.

Starting from the base features described in Section III-B, additional features were engineered to enrich the input of the model. Specifically, 5 lagged time steps of position, speed, bearing, and serving cell were computed per UE to provide trajectory history, and the Euclidean distance from the UE to each of the 10 access points was calculated to provide spatial context. Categorical features were one-hot encoded, resulting in a total of 96 features per timestep (36 continuous and 60 binary). The target labels were generated by shifting the serving cell forward by 1 to 5 steps, enabling the model to predict the next access point across 5 simultaneous horizons ($t$+3 s to $t$+15 s).

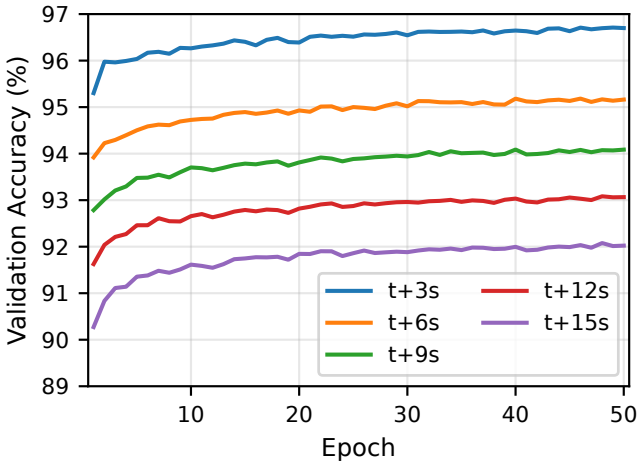



Fig. 6: Per-horizon validation accuracy.

Hyperparameters were selected via Keras Tuner's Random Search over 15 configurations, varying layer sizes, dropout rates, and learning rate, each trained for up to 30 epochs with early stopping (patience of 5), using minimum validation loss as the selection criterion. The best configuration, summarized in Table II, was retrained for up to 50 epochs with a patience of 6. As shown in Figure 6, the validation accuracy improves rapidly in the first 15 epochs and stabilizes thereafter, with shorter horizons consistently achieving higher accuracy as expected.

TABLE II
LSTM model hyperparameters

| Hyperparameter | Used Value(s) |
|---|---|
| RNN Layer 1 (units) | 128 |
| RNN Layer 2 (units) | 192 |
| Dropout 1 | 0.2 |
| Dropout 2 | 0.3 |
| Dense (units) | 128 |
| Dropout 3 | 0.35 |
| Learning rate | 0.0005 |

On the test set, both the LSTM and GRU models significantly surpass the naive baseline, which assumes that no UE will change BSs, particularly at longer horizons where the gap reaches approximately 20%, as shown in Figure 7. The first slightly outperforms the second by around 0.4%, suggesting that the added complexity provides limited benefit for this task.

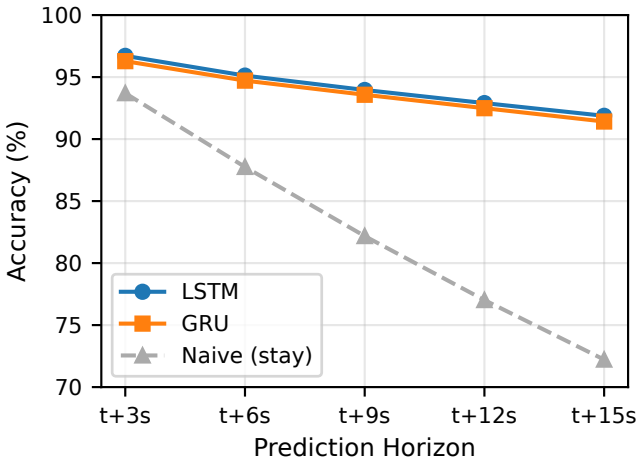



Fig. 7: Test accuracy across prediction horizons.

Table III shows the per-class F1 score across all prediction horizons. Most classes maintain high performance, with Base Stations 2, 5, and 6 exceeding 95% even at t+15s. Base Stations 4 and 7, highlighted in bold, consistently underperform, dropping from above 90% at t+3s to around 80% and 83% at t+15s, due to the lower UE density in their coverage areas, which reduced the number of available training samples. The Exit class has the lowest F1 at approximately 78%, since each UE exits the scenario only once per trip, whereas a typical UE undergoes two or three serving cell changes, making exit events significantly underrepresented in the training data. Despite this, its performance remains nearly constant across horizons, which may indicate that exit prediction is less sensitive to temporal uncertainty than serving cell prediction.

TABLE III
Per-class F1 score (%) across prediction horizons.

| Class | t+3s | t+6s | t+9s | t+12s | t+15s |
|---|---|---|---|---|---|
| BS 1 | 96.6 | 94.6 | 93.1 | 91.9 | 90.7 |
| BS 2 | 97.7 | 96.8 | 96.1 | 95.4 | 94.7 |
| BS 3 | 97.7 | 96.5 | 95.6 | 94.6 | 93.5 |
| **BS 4** | 90.3 | 85.6 | 84.1 | 82.5 | 80.0 |
| BS 5 | 97.8 | 96.9 | 96.5 | 96.1 | 95.6 |
| BS 6 | 97.9 | 96.9 | 96.2 | 95.6 | 95.0 |
| **BS 7** | 92.7 | 90.3 | 87.8 | 84.8 | 82.6 |
| BS 8 | 95.7 | 94.2 | 93.7 | 93.5 | 92.9 |
| BS 9 | 95.7 | 93.7 | 92.3 | 91.3 | 90.1 |
| BS 10 | 97.6 | 95.8 | 93.9 | 92.1 | 90.6 |
| **Exit** | 77.9 | 78.5 | 78.3 | 78.0 | 78.0 |

### *D. Proactive Migration Results*

In order to assess the RTT, two applications were created and implemented in the simulation environment. A client application running on each UE periodically sends requests to its associated server-side application at a fixed interval of 500 ms. This sampling frequency was chosen to provide sufficient

granularity for capturing latency variations during migration events without introducing unnecessary load on the simulated network. The server-side application receives each request and integrates the segmented delay components directly into the response, allowing the client to independently decompose and log each contribution into the total RTT. This design ensures that the measured RTT accurately reflects end-to-end network conditions as experienced by the UE throughout the simulation.

Using this setup, we compared the two orchestration strategies introduced in Section V-A: the reactive MoC and the proactive MoP. Each strategy was evaluated over 30 independent simulation runs, each using a different random seed, resulting in over 8.7 million application-layer messages per strategy. To ensure a fair statistical comparison, we first compute a summary metric per run, resulting in 30 independent values per strategy.

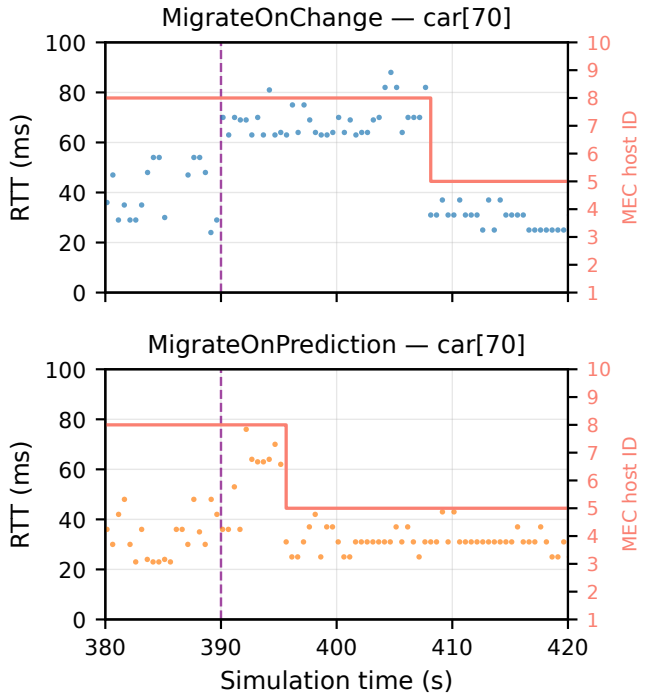



Fig. 8: RTT values and MEH during a period close to a handover.

To illustrate the impact of each strategy at the UE level, Figure 8 shows the response time and MEH assignment for a single vehicle (car[70]) during a handover event. The dashed line marks the approximate time of the radio-layer handover at the 390 s mark. In the reactive strategy, shown in the upper half, the MEC migration occurs about 20 seconds after the handover, resulting in a period where the UE is served by a non-local host. This delay is the result of the TTT plus the $\Delta_{\text{migration}}$. In the proactive strategy, shown in the bottom half, the migration is triggered ahead of time based on the predicted target cell. Depending on the prediction quality, migration may be completed before or after the actual handover. Generally, this results in less time spent on a non-ideal MEH, leading to reduced RTT. As can be seen, the number of RTT samples between the actual handover and the completed migration is significantly lower under MoP than under MoC. At the aggregate level, Figure 9 compares the overall RTT distributions across all runs. MoP achieves a mean RTT of 62.0 ms compared to 63.3 ms for MoC (p =0.023), a modest but statistically significant improvement of 2.4%.

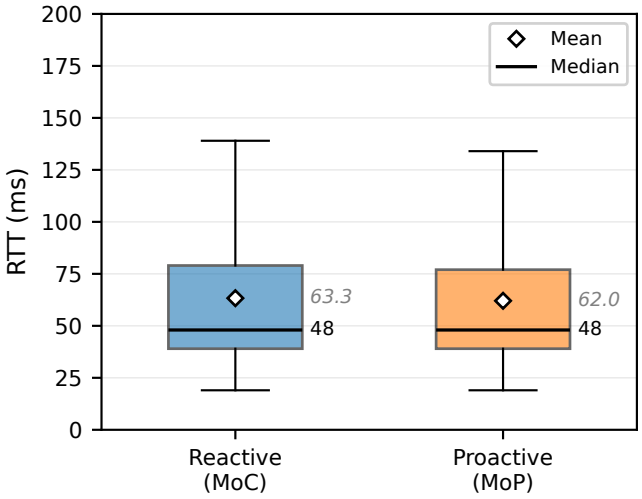



Fig. 9: RTT distribution for the two different migration approaches.

In order to understand the reasons behind the modest overall improvement, we examined the prediction quality associated with the MoP strategy. Each UE was classified based on whether all of its migration events targeted the correct destination cell. Specifically, a prediction is considered correct when the MEC migration destination matches the actual handover target. Based on this criterion, 82.4% of all migrations targeted the correct destination. At the UE level, 32,148 UEs had all migrations classified as correct, while the remaining 6,928 experienced at least one incorrect prediction. This is notably lower than the standalone model accuracy of 91 to 96% reported in Section V-C. This gap is largely explained by the fact that, once a migration is triggered, the service remains at the target host for a minimum time of 12 s, which means that an incorrect decision cannot be quickly corrected.

Figure 10 shows the RTT distribution for correctly and incorrectly predicted UEs under the MoP strategy. Correctly predicted UEs achieve substantially lower mean and median RTT values than incorrectly predicted ones, with differences of 6.9 ms and 8 ms, respectively. These findings indicate that the main limitation of the proactive approach is the accuracy of predictions: while accurate handover predictions lead to significant reductions in latency, incorrect predictions considerably reduce the overall improvement to just 2.4%. Finally, the current orchestration framework handles every prediction equally, with no mechanism to check the confidence of a given prediction. This means that a single bad prediction has a negative impact for at least the duration of the $\Delta_{migration}$.

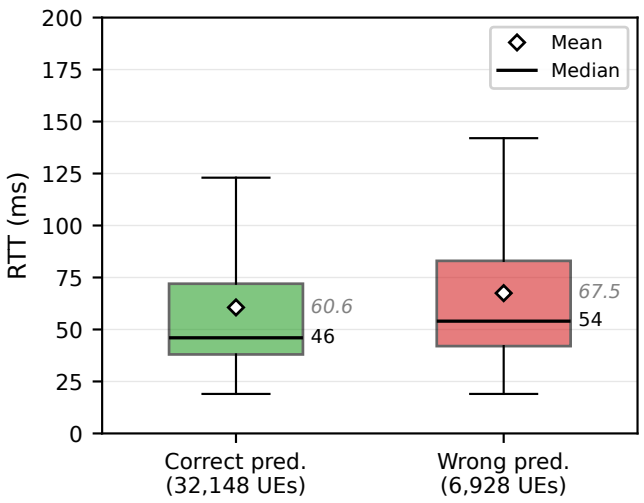



Fig. 10: RTT distribution for MoP with prediction distinction.

## VI. Conclusions and Future Work

In this paper, we presented a proactive orchestration framework for vehicular edge computing environments, combining an LSTM-based serving cell predictor with a proactive migration strategy that anticipates handover events and migrates individual applications in advance. The framework was evaluated within a simulated environment constructed using a series of well-known simulation tools, where augmented real-world vehicular traces were processed through an entire MEC stack.

The LSTM model showed high accuracy across all prediction horizons, ranging from 96% at t+3 s to 91% at t+15 s, outperforming the GRU variant and the naive persistence baseline. When integrated into the MEO, the MoP approach resulted in a statistically significant 2.4% reduction in RTT compared to the reactive baseline MoC ($p = 0.023$), based on data from 8.7 million application-layer messages per strategy across 30 separate simulation runs. The qualitative analysis of the deployed model revealed that the overall improvement is primarily given by accurately predicted handovers, while wrong predictions disproportionately degrade performance.

Several directions for future work emerge from these findings. Incorporating prediction confidence into the orchestration logic would enable the system to determine when to trust the model's output. Additionally, alternative model architectures and input representations should be explored, such as a cascading approach where a density prediction model feeds into the serving cell predictor. Finally, planned extensions include a Deep Reinforcement Learning-based migration agent, an oracle baseline, sensitivity analysis over $\Delta_{migration}$ and TTT, feature ablation, and evaluation on different topologies.

## References


[1] Q.-V. Pham, F. Fang, V. N. Ha, M. J. Piran, M. Le, L. B. Le, W.-J. Hwang, and Z. Ding, "A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art," *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020. https://doi.org/10.1109/ACCESS.2020.3001277

[2] S. R. Alkaabi, M. A. Gregory, and S. Li, "Multi-access edge computing handover strategies, management, and challenges: A review," *IEEE Access*, vol. 12, pp. 4660–4673, 2024. https://doi.org/10.1109/ACCESS.2024.3349587

[3] C. Bernad, A. Dedinec, K. Gilly, S. Filiposka, and A. Mishev, "Towards smart 6G: Mobility prediction for dynamic edge services migration," *Expert Syst. Appl.*, vol. 297, p. 129348, 2026. https://doi.org/10.1016/j.eswa.2025.129348

[4] A. Belhadj, K. Akilal, S. Bouchelaghem, M. Omar, and S. Aissani, "Next-cell prediction with LSTM based on vehicle mobility for 5G mc-IoT slices," *Telecommun. Syst.*, vol. 87, no. 3, pp. 809–833, 2024. https://doi.org/10.1007/s11235-024-01214-6

[5] N. Uniyal, A. Bravalheri, X. Vasilakos, R. Nejabati, D. Simeonidou, W. Featherstone, S. Wu, and D. Warren, "Intelligent mobile handover prediction for zero downtime edge application mobility," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6. https://doi.org/10.1109/GLOBECOM46510.2021.9685282

[6] K. Sun, Q. Han, Z. Yang, W. Huang, H. Zhang, and V. C. M. Leung, "Proactive handover type prediction and parameter optimization based on machine learning," *IEEE Trans. Wireless Commun.*, vol. 24, no. 4, pp. 3515–3528, 2025. https://doi.org/10.1109/TWC.2025.3531702

[7] W. Zhang, J. Luo, L. Chen, and J. Liu, "A trajectory prediction-based and dependency-aware container migration for mobile edge computing," *IEEE Trans. Services Comput.*, vol. 16, no. 5, pp. 3168–3181, 2023. https://doi.org/10.1109/TSC.2023.3290023

[8] E. Dritsas, K. Ramantas, and C. Verikoukis, "A mobility-aware reinforcement learning proactive solution for state data migration in edge computing," in *Proc. IEEE 29th Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, 2024, pp. 1–6. https://doi.org/10.1109/CAMAD62243.2024.10943072

[9] A. Dalgkitsis, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Data driven service orchestration for vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4100–4109, 2021. https://doi.org/10.1109/TITS.2020.3011264

[10] X. Zhao, Y. Shi, S. Chen, J. Liu, B. Ji, and S. Mumtaz, "MAPSM: Mobility-aware proactive service migration framework for mobile-edge computing in consumer internet of vehicles," *IEEE Trans. Consum. Electron.*, vol. 71, no. 2, pp. 3753–3766, 2025. https://doi.org/10.1109/TCE.2025.3563627

[11] A. Mukhopadhyay, G. Iosifidis, and M. Ruffini, "Migration-aware network services with edge computing," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 2, pp. 1458–1471, 2022. https://doi.org/10.1109/TNSM.2021.3139857

[12] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 369–382, 2019. https://doi.org/10.1109/TCC.2016.2525987

[13] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G–an OMNeT++ library for end-to-end performance evaluation of 5G networks," *IEEE Access*, vol. 8, pp. 181 176–181 191, 2020. https://doi.org/10.1109/ACCESS.2020.3028550

[14] P. J. Araújo, "Simu5G fork with proactive MEC orchestration extensions," [Online]. Available: https://github.com/paulojna/Simu5G/, 2025, accessed: Apr. 2026.

[15] P. J. Araújo, H. F. López, and A. Santos, "Efficient mobility management for MEC orchestration in vehicular scenarios," in *Proc. 20th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, 2024, pp. 383–390. https://doi.org/10.1109/WiMob61911.2024.10770413

[16] M. Rapelli, C. Casetti, and G. Gagliardi, "Vehicular traffic simulation in the city of Turin from raw data," *IEEE Trans. Mobile Comput.*, vol. 21, no. 11, pp. 4189–4204, 2022. https://doi.org/10.1109/TMC.2021.3075985

[17] T. V. Doan, G. T. Nguyen, H. Salah, S. Pandi, M. Jarschel, R. Pries, and F. H. P. Fitzek, "Containers vs virtual machines: Choosing the right virtualization technology for mobile edge cloud," in *Proc. IEEE 2nd 5G World Forum (5GWF)*, 2019, pp. 46–52. https://doi.org/10.1109/5GWF.2019.8911715

[18] P. J. Araújo, H. F. López, and A. Santos, "RAVENS mobility dataset for MEC orchestration in vehicular scenarios — Turin (TuST)," Zenodo, 2026, accessed: Apr. 2026. https://doi.org/10.5281/zenodo.19683780