Reinforcement Learning Based Trajectory Optimization for MEC-UAV

Filipe Samuel da Silva, Renata Kellen Gomes dos Reis, and Andson Marreiros Balieiro

Abstract Unmanned Aerial Vehicles (UAVs) are envisioned to integrate terrestrial and aerial infrastructure in three-dimensional Sixth-Generation (6G) networks. When combined with MEC resources (MEC-UAV), they can expand the coverage and computational capacity of terrestrial infrastructures, enabling services such as computational offloading for users. However, as UAVs are battery-powered devices, energy-efficient management is essential. In this respect, UAV trajectory optimization plays a key role, as it impacts not only the system's operational lifetime without recharging, but also the quality of service provided by MEC-UAV systems. This work proposes a RL-based solution for MEC-UAV trajectory optimization, considering the return of remote processing results to users and the impact of the MEC-UAV trajectory on energy consumption, the proportion of admitted offloaded tasks, and the timing of result delivery to users. Our solution aims to maximize the number of tasks admitted for processing on the MEC-UAV while also increasing the proportion of successfully completed tasks and reducing energy consumption during flight operations. Results demonstrate that our approach achieves a better balance across the metrics, including low energy consumption, a high percentage of admitted and completed tasks, and a more consistent MEC-UAV trajectory, compared with existing approaches from the literature.

Filipe Samuel da Silva
Centro de Informatica (CIn), Universidade Federal de Pernambuco (UFPE), Recife - PE, e-mail: fss8@cin.ufpe.br

Renata Kellen Gomes dos Reis
Centro de Informatica (CIn), UFPE, Recife - PE e-mail: rkgr@cin.ufpe.br

Andson Marreiros Balieiro
Centro de Informatica (CIn), UFPE, Recife - PE e-mail: amb4@cin.ufpe.br

1 Introduction

The Fifth Generation (5G) of Mobile Networks introduced significant innovations to cellular networks, including the use of millimeter waves alongside sub-6 GHz frequencies and the adoption of a service-based architecture within a cloud-native core network [1]. Furthermore, 5G was designed to encompass a wide range of applications categorized as enhanced mobile broadband (eMBB), massive machine-type communication (mMTC), and ultra-reliable low latency communication (URLLC). Supporting such diverse services poses challenges, making Multi-Access Edge Computing (MEC) essential for meeting the requirements of applications with strict latency demands (e.g., URLLC) or those requiring intensive computation but operating with limited resources, such as certain Internet of Things (IoT) ones. MEC achieves this by providing cloud resources (computing, storage, or connectivity) closer to the end user while also reducing core network congestion [2].

MEC enables computational offloading services, where users send their tasks to be processed on edge servers, achieving low latency and conserving energy on user devices, which is essential for battery-powered devices such as IoT ones [3]. However, in disaster, remote or with high momentary demand (e.g., concerts and large sporting events) scenarios, terrestrial MEC infrastructures may fail to meet application requirements (e.g., latency), become unavailable (e.g., due to connectivity issues), or prove impractical to adopt (e.g., due to financial or physical deployment constraints). In such cases, Unmanned Aerial Vehicles (UAVs) provide a viable alternative to dynamically establish mobile MEC infrastructures (MEC-UAVs). UAVs offer flexible and cost-effective deployment options, along with line-of-sight connectivity to users, which improves channel gain and subsequently reduces the energy consumption of user devices during data offloading [8]. In addition, UAVs have been proposed to integrate terrestrial and aerial infrastructure in 5G Advanced and Sixth-Generation (6G) networks, forming three-dimensional networks [6]. When equipped with MEC resources, UAVs can extend the coverage and computational capacity of terrestrial infrastructures.

Since UAVs are battery-limited devices, adopting MEC-UAVs necessitates energy-efficient designs distinct from those used for terrestrial systems [11]. For instance, improving energy efficiency directly increases the amount of data that can be offloaded and processed on a MEC-UAV before it requires recharging. In addition to the energy spent on communication and processing, MEC-UAVs incur extra energy consumption to remain aloft and support mobility, which can exceed the power required for communication [11]. In this context, UAV trajectory optimization plays a key role in MEC-UAV systems, as it impacts not only the system's operational lifetime without recharging but also the quality of service provided by the MEC-NFV system. This includes aspects such as coverage (e.g., the number of users served), offloaded task response time, and successful task completion ratio.

Several solutions for UAV trajectory optimization have been proposed in the literature [10] [11] [5] [13] [12] [3] [7]. For instance, [10] addresses trajectory design and user-UAV assignment in a multi-UAV multi-user system, aiming to maximize user throughput, while [11] examines energy efficiency in UAV-assisted communi-

cation networks and proposes a trajectory strategy for UAVs hovering above a single user device. However, [10] [11] do not incorporate MEC services into their scopes. On the other hand, [5] proposes a Deep Reinforcement Learning (DRL)-based approach for energy consumption minimization by optimizing the UAV trajectory, task partition, and resource allocation in MEC-UAV systems, considering channel state information (CSI) imperfections and uncertainty regarding task complexity. Similarly, [13] adopts DRL to optimize UAV trajectory definition and users' offloaded task ratio scheduling, aiming to minimize the overall energy consumption. In [12], the authors minimize the energy consumption and task completion time as separate problems by jointly designing the computation offloading, resource allocation, and UAV trajectory. They employ successive convex approximation (SCA)-based algorithms to address these problems and evaluate their solution in terms of energy consumption and completion time. Furthermore, [3] addresses trajectory optimization and computing offloading decision problems using deep deterministic policy gradient and population diversity-binary particle swarm optimization algorithms to minimize energy consumption and delay. The authors in [7] employ a Deep Reinforcement Learning with Federated learning solution to control the UAV trajectory, user association, and resource allocation, aiming at minimizing the energy consumption across all user devices.

Although their studies provides valuable insights, [5] [13][12] [3] [7] overlook an essential aspect of practical applications: the return of processed data to the local device, which is typical in use cases like video processing (e.g., applying filters). Their frameworks consider only the uplink transmission and task processing stages in calculating task response time, neglecting the time required to send back the results, which is influenced by MEC-UAV movement. Consequently, they fail to account for the impact of MEC-UAV trajectory on reception time. Since users needs to be within the MEC-UAV's coverage to receive task results, UAV movement directly affects when users can receive these results, even after remote computation is completed. In addition, works, such as [3], assume that tasks admitted by the MEC-UAV will always be completed, regardless of the time required. This assumption does not hold for latency-sensitive applications with strict response time requirements, where delays can render the data obsolete or the application ineffective.

This work addresses these gaps by proposing a RL-based solution for MEC-UAV trajectory optimization, taking into account the return of remote processing results to users and the impact of the MEC-UAV trajectory on energy consumption, the proportion of admitted offloaded tasks, and the timing of result delivery to users. Therefore, our solution not only aims to maximize the number of tasks admitted for processing on the MEC-UAV but also to increase the proportion of successfully completed tasks, those processed and returned to users within their latency requirements, while reducing energy consumption in flight operations.The proposed solution is evaluated based on the percentage of admitted tasks, completed tasks, and energy consumption, and was compared with existing approaches from the literature. Results show that our solution achieves a better balance across the metrics, including low energy consumption, a high percentage of admitted and completed tasks, and a more consistent MEC-UAV trajectory.

The remainder of this paper is organized as follows. Section 2 describes a trajectory optimization problem and Section 3 presents the proposed Reinforcement Learning-based solution. Section 4 analyzes the simulation results, comparing our solution to three others from the literature. Finally, Section 5 provides concluding remarks and highlights future directions.

2 The Trajectory Optimization Problem

In a MEC-UAV environment supporting computational offloading service, mobile users offload their tasks to UAVs equipped with MEC capabilities. The MEC-UAVs process these tasks and return their results to the users, thereby extending the battery life of user devices and enabling the execution of high-resource-demand tasks that exceed the devices' capabilities. In addition, MEC-UAVs can move a 3D space toward users, providing flexible computing infrastructure within their coverage areas. This makes them a useful solution for scenarios such as natural disasters or high-demand events, where terrestrial infrastructure is limited [9]. However, the limited energy and computing resources of MEC-UAVs necessitate effective trajectory control and resource management in both self-controlled and externally guided scenarios.

This paper considers a MEC-UAV environment comprising n_t mobile users randomly distributed over an area of A m^2. At each time slot t, each user i, located at coordinates $(x_i[t], y_i[t])$ in the plane, has the probability P_{it} of sending a task to the MEC-UAV node for processing. A task is sent only if user i is within the coverage radius R of the MEC-UAV, which is positioned at coordinates $(X[t], Y[t], H)$, where H denotes its fixed height. Consequently, to enable task offloading, the Euclidean distance $(d_i[t])$ between node i and the MEC-UAV at slot t must not exceed R, as shown in Eq. 1 .

$$d_i[t] = \sqrt{(X[t] - x_i[t])^2 + (Y[t] - y_i[t])^2} \leq R \tag{1}$$

Given the MEC-UAV's movement from $(X[t-1], Y[t-1], H)$ to $(X[t], Y[t], H)$ between two consecutive time slots t and $t-1$, the energy consumed for flying is given by Eq. 2, as in [4], where P_f represents the flight power and V is the MEC-UAV's speed.

$$E_{\text{fly}}[t] = P_f \left((X[t] - X[t-1])^2 + (Y[t] - Y[t-1])^2 \right)^{1/2} V^{-1} \tag{2}$$

Once the user i is within the coverage radius of the MEC-UAV, it can offload its tasks for remote processing, which incurs transmission and processing costs. The energy cost for transferring and processing the task of user i on the UAV is given by Eq. 3, where P_h means the hover power, and T_{it}^{trans} denotes the task transmission time, as expressed in Eq. 4. This time is determined by the task size (D_i), measured in bits and the transmission rate (R_{it}) between user i and the MEC-UAV. The rate R_{it}, given by Eq. 5, is computed based on Shannon's Law, where β is the bandwidth

assigned to the i-MEC-UAV transmission, ρ denotes the user's transmission power, σ_2 represents the additive white Gaussian noise power, and h refers to the channel power gain considering the direct line of sight between the MEC-UAV and user i at time t. This gain is provided by Eq. 6.

$$E_{\mathrm{hov}}[t] = P_h T_{\mathrm{i}}^{\mathrm{trans}}[t] \tag{3}$$

$$T_i^{trans} = \frac{D_i[t]}{R_i[t]} \tag{4}$$

$$R_i[t] = \beta log_2(1 + \frac{\rho h}{\sigma_2}) \tag{5}$$

$$h_i[t] = \beta_0 (d_i[t])^{-2} \tag{6}$$

Let $K_p[t]$ denote the number of tasks processed by the MEC-UAV at time slot t, and N_a represent the number of users offloading tasks for remote processing, the total energy consumed by the MEC-UAV during its operation is given by Eq. 7.

$$E_{total} = \sum_t (E_{fly}[t] + E_{hov}[t] K_p[t] + \sum_{n=1}^{N_a} E_{hov}[t]) \tag{7}$$

Considering that each user i's task has a response waiting time constraint l_i, which denotes the maximum time that the UAV has to return the results to the user, expressed time slots k, the ratio of unreturned tasks L during the MEC-UAV operation is given by Eq. 8. This ratio is the complement of the ratio between the number of successfully returned tasks τ_{suc} and the number of admitted tasks τ_{adm} by the MEC-UAV for processing. Moreover, the ratio of users with requests accepted by the UAV for task processing A is given by Eq. 9, which accounts τ_{adm} and the total number of requests for task processing τ_{req}. Since the MEC-UAV moves to serve users, task offloading depends on the MEC-UAV's coverage. As a result, users may experience delays before they can begin sending their tasks for remote processing. Thus, we denote by W the average wait time experienced by admitted tasks.

$$\mathrm{L} = 1 - \frac{\tau_{suc}}{\tau_{adm}} \tag{8}$$

$$\mathrm{A} = \frac{\tau_{adm}}{\tau_{req}} \tag{9}$$

Proper MEC-UAV resource and trajectory planning must consider multiple objectives, which can be formulated as an optimization problem, as shown in Eq. 10. Given a set of users with tasks to be remotely processed at the MEC-UAV node, determine the sequence of MEC-UAV positions (X[t], Y[t]) that minimizes total energy consumption during operation, non-admitted request ratio, unreturned task ratio, and the user wait time before task transmission, with each objective weighted by λ_1, λ_2, λ_3, and λ_4, respectively.

$$\mathrm{P}: \min_{\{X[t],Y[t]\}} \lambda_1 E_{total} + \lambda_2 W + \lambda_3 (1-A) + \lambda_4 L \tag{10}$$

3 The Reinforcement Learning-Based Solution

To address the problem defined in Eq. 10, we propose a reinforcement learning-based solution. The approach divides time into discrete intervals, referred to as time slots, each accommodating a maximum number of tasks to be processed by the MEC-UAV. Each slot is further subdivided into subslots, enabling task processing over time. Since tasks may span multiple subslots, a multi-slot processing strategy is required.

Based on variables such as user locations, workload, and energy costs, the reinforcement learning solution aims to determine the optimal MEC-UAV movement at each time slot. To do so, it employs Deep Reinforcement Learning (DRL), leveraging deep neural networks for decision-making and strategy adjustment in response to changes in user and MEC-UAV states, providing rewards, and updating actions accordingly. DRL optimizes energy consumption and task execution by dynamically adapting to network conditions [13].

The neural model, based on Deep Q-Networks (DQN), utilizes environmental insights to define UAV routes while addressing constraints such as battery capacity, processing limitations, and task queue management [13]. The training process follows Algorithm 1, which adjusts the network's weights based on rewards or penalties resulting from MEC-UAV actions. The algorithm also stores past experiences in a replay buffer for iterative updates. The proposed deep neural network combines a Convolutional Neural Network (CNN) with a Long Short-Term Memory (LSTM) network. The CNN processes an environmental map as an image, structured as a matrix containing information about users, their positions, and states. Meanwhile, the LSTM accounts for user context, including positions and distances to the MEC-UAV, allowing the model to process both spatial and sequential information. Figure 1 illustrates the network architecture.

The CNN layers extract spatial features using 3x3 filters and a pooling layer for dimensionality reduction. A flattening process converts the feature map into a one-dimensional vector, which passes through a fully connected layer for feature reduction and refinement. The resulting vector, combined with system state data, is fed into the LSTM, which then processes this data to obtain context information over time. The LSTM concludes with a intermediate linear layer and an output layer that defines six-movement options. Let $o[n]$ represent the network output, the movement direction is given by Eq. 11, speed by Eq. 12, and the subsequent position by Eq. 13. Using this approach, we aim to optimize MEC-UAV movements to maximize the number of served users, minimize response time, and reduce energy consumption, all while navigating the environment and processing user tasks within the coverage area.

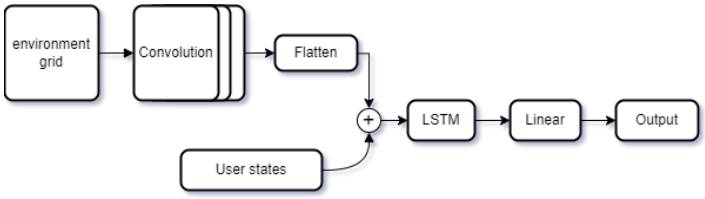


Fig. 1: Neural network architecture layers.

$$c[n+1] = \begin{cases} c[n], & \text{if } o[n] \in 0,1,2,3,(velocity) \\ (c[n] - \frac{\pi}{4}) \mod 2\pi, & \text{if } o[n] = 4,(left) \\ (c[n] + \frac{\pi}{4}) \mod 2\pi, & \text{if } o[n] = 5,(right) \end{cases} \tag{11}$$

$$v[n+1] = \begin{cases} 0, & \text{if } o[n] = 0, \\ v[n] - 1, & \text{if } o[n] = 1, \\ v[n], & \text{if } o[n] = 2, \\ max(1 + v[n], 5), & \text{if } o[n] = 3, \\ v[n], & \text{if } o[n] \in 4,5, \end{cases} \tag{12}$$

$$\begin{aligned} x[n+1] &= x[n] + v[n] \cdot \cos(d[n]) \\ y[n+1] &= y[n] + v[n] \cdot \sin(d[n]) \end{aligned} \tag{13}$$

4 Result Analysis

To evaluate our solution, we conducted 10 simulation instances within a grid-shaped environment covering an area of 100m × 100m. Each grid segment measures 1 meter, allowing users to be positioned in any segment. We adopted a time slot duration of 1 second, with the MEC-UAV allowed to be placed in any segment, and a maximum speed (V_{max}) of 5 segments per time slot (5 seg/slot). Users can be served by the MEC-UAV when they are within its coverage radius. Table 1 summarizes the parameters adopted for the simulations and neural network training.

The training of our solution considered 1500 episodes, each comprising 1000 time slots. The learning rate was set to 0.004, and the training utilized an experience replay memory of up to 100000 entries, storing experiences to train the model. The training batch size was set to 4000, with a random sample from this memory selected at each time slot to improve training performance. Moreover, at each time slot, the rate of random movements ε decays of ε_{decay}. Additionally, for the first 160 episodes, the ε value is reduced by a factor f .

At each time slot, users within the MEC-UAV's coverage radius and with tasks to be processed, offload them and wait for the processing response. Users have a

Algorithm 1 Deep Q-Network (DQN) for model training

Initialize replay memory D to capacity N
Initialize action-value model with random weights θ
Set learning rate α and discount factor γ
Set exploration rate ε (decay factor $\varepsilon_{\text{decay}}$)
for each episode **do**
  Initialize state s_1 with environment setup
  for each step of episode **do**
    Get current system state s_t from environment
    With probability ε select a random action a_t
    otherwise select $a_t = \arg\max_a Q(s_t, a; \theta)$
    Execute action a_t, receive reward r_t, and observe new state s_{t+1}
    Store transition $(s_t, a_t, r_t, s_{t+1}, \text{done})$ in D
    Sample random mini-batch from D
    for each transition in mini-batch **do**
      Compute target y_j
      Set $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \theta) & \text{otherwise} \end{cases}$
      Perform a gradient descent step on $(y_j - Q(s_j, a_j; \theta))^2$ to update θ
    end for
    Update exploration rate $\varepsilon \leftarrow \varepsilon \cdot \varepsilon_{\text{decay}}$
  end for
end for

Table 1: Simulation and Training Parameters

Parameter	**Value**
Number of episodes	1500
Number of time slots per episode	1000
Initial number of users	25
Probability of user appearance	0.04
Probability of user leaving	0.004
Probability of task request (P_t)	0.015
Time slot duration	1s
Grid size	[100m, 100m]
Segment size	1m
Coverage radius (r_c)	15m
Learning rate lr	0.002
Discount rate γ	0.90
Maximum replay memory	100.000
Batch size	4000
Initial ε	0.9
ε_{decay}	0.9995
ε subtraction factor f per episode	0.005

Table 2: Energy Cost Parameters

Parameter	**Value**
P_h	0.08
P_f	0.11
D_m	10,000,000
V_{max}	5 seg/slot
β	20Mhz
β_0	1
ρ	0.1
σ_2	1×10^{-9}

Table 3: User Task Processing Cost

Parameter	**Value**
Task processing cost (k)	$1 \leq k \leq 25$
Max. user wait time for processing (l_i)	6 time slots
Parallel processing capacity of the MEC-UAV (K_p)	5

maximum wait time for task completion/processing (l_i), as set in Table 3. When the maximum parallel processing of the MEC-UAV (K_p) per slot is reached, new tasks will be rejected (lost). The simulations assume $K_p = 5$. Since tasks may have different complexities, their execution time requirements (k), measured in the number of time slots, are diverse. To represent this, we adopt the interval [1 25]. The parameters and values used to compute the energy costs for movement and processing are summarized in Table 2. At each time slot, the MEC-UAV energy consumption is influenced by its movement and the number of users (tasks) being served.

We evaluated our solution based on several metrics, including the percentage of admitted tasks, percentage of successfully completed tasks, energy cost for MEC-UAV movement, and percentage of remaining battery energy. Additionally, we compared our proposal with three approaches from the literature: an MLP-based model, a K-means strategy that moves the MEC-UAV towards the closest user cluster, and a random movement solution.

Fig. 2 presents the average results obtained by the evaluated solutions in terms of admitted tasks, completed tasks, and remaining battery energy, expressed as percentages (%). For the first metric, our RL-based solution achieved a performance of 73%, whereas the MLP model admitted only 58% of tasks and the random approach got 63.40%. The lower percentage of admitted tasks allowed the MLP and Random models to achieve higher completion ratios; however, this came at the cost of processing fewer overall tasks. Compared with the K-means, our solution showed superior performance regarding the percentage of completed tasks. Although the K-means approach guided the MEC-UAV to admit more tasks, it resulted in higher energy consumption compared to other approaches, leaving the remaining battery energy at only 63.68% by the end of the simulation. In contrast, our solution achieved a higher remaining battery energy of 73.16%.

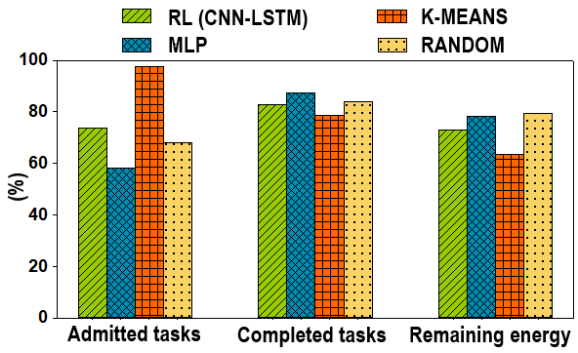


Fig. 2: Average percentage of admitted and completed tasks, and remaining battery energy.

Analyzing the product of the percentage of admitted tasks and completed tasks alongside the remaining energy, we observe that the K-means approach handled 76.92% of all tasks but consumed 36.32% of the MEC-UAV's energy. On the other hand, our solution addressed 67.43% of the tasks while consuming only 26.84% of

the energy. The difference between the respective products and energy consumptions could suggest similar performance for these approaches. However, when considering the energy cost of MEC-UAV movements, shown in Fig. 3, we observe that our solution achieved significant savings, consuming 28.21% less energy compared to the K-means solution, which expended 39.84 J in flight operation. This outcome highlights a more effective use of the MEC-UAV's energy operating with the proposed solution, efficiently processing tasks while avoiding excessive energy waste in movements.

Although the random approach consumed less energy by focusing on specific areas and limiting its movement region (see Fig. 4d), it admitted and processed fewer tasks compared to our solution (see Fig. 4a). Consequently, the energy savings achieved in flight operations were not utilized to enhance service performance, processing more tasks. A similar behavior was observed with the MLP solution, which also limited its movement scope (see Fig. 4b), resulting in lower product admitted (58.20%) and completed (87.35%) tasks despite reduced energy consumption.

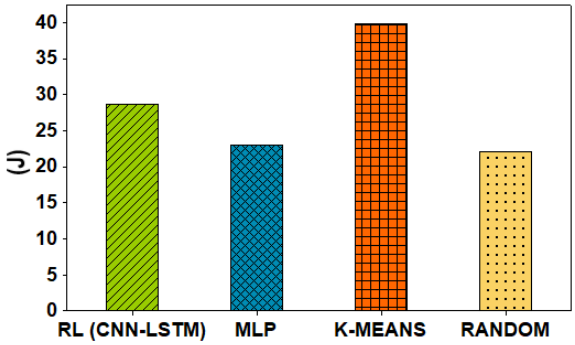
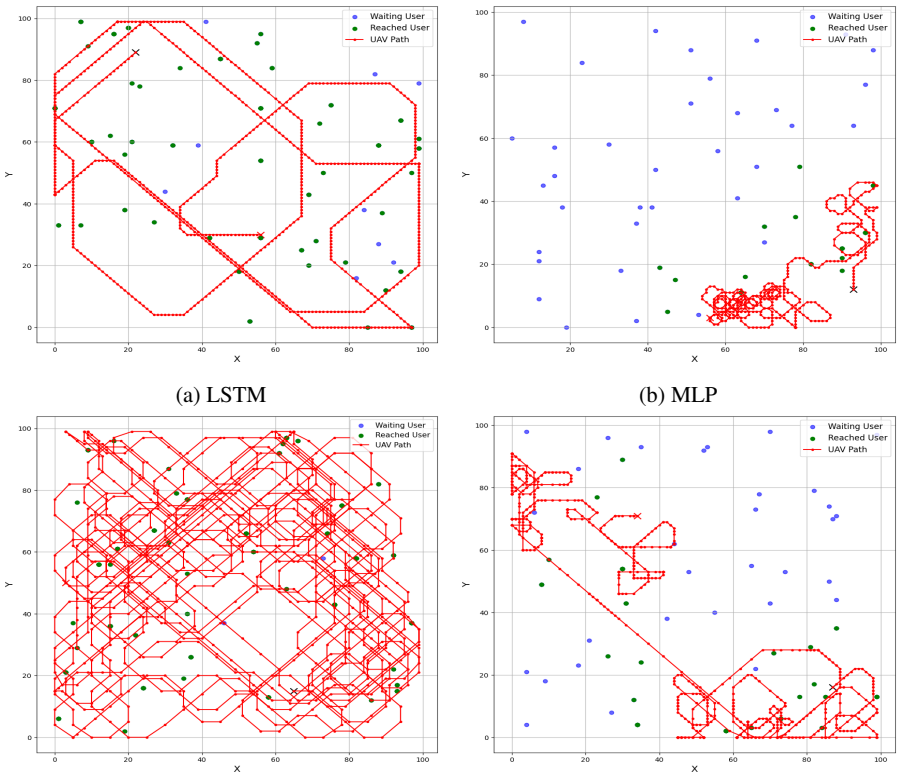


Fig. 3: Energy cost for MEC-UAV movement (J).

Fig. 4 shows the UAV trajectory executed in a simulation instance, guided by each model and its future perspective. The figure illustrates the position of all users with tasks to be remotely processed as blue points, while green ones represent users covered by the MEC-UAV. The MEC-UAV's movements are depicted by a red line. The LSTM model showcases a more coherent movement style, signaling a more assertive learning relative to the objective, achieving a well-balanced performance. Its movement behavior aligns more closely with the real-world UAV operation, striving to serve users across all regions more equitably and avoiding over-concentration in specific areas. The MLP and random models restricted their movements to certain limited areas. Although the K-means model traversed the entire environment (see Fig. 4c), it resulted in excessive energy consumption due to consistently making long and high-speed movements.

5 Conclusion

This work presented a reinforcement learning-based solution for the MEC-UAV trajectory optimization problem, focusing on computational offloading services in 5G mobile networks. The proposed solution, which integrates CNN and LSTM, was evaluated against alternative approaches based on neural networks, clustering, and random movement strategies. Our results demonstrated that the proposed solution achieved a better balance across key metrics, including low energy consumption, a high percentage of admitted and completed tasks, and a more consistent MEC-UAV trajectory, effectively avoiding unnecessary movements. We also observed that traditional movement strategies, such as moving towards the nearest user, as employed by the K-means approach, can yield reasonable performance in specific scenarios. This finding suggests that hybrid movement policies, combining insights from machine learning models with rule-based strategies for straightforward decision-making, may offer a promising alternative for trajectory optimization.



(a) LSTM (b) MLP

(c) K-means (d) Random

Fig. 4: The MEC-UAV trajectory considering different approaches

As future work, we aim to refine personalized aspects in determining the UAV's next movement by integrating regular movement evaluations with the decisions provided by our RL-based solution. Further exploration includes evaluating alternative neural network models and architectures, as well as employing more sophisticated training hyperparameters to improve learning efficiency and consequently, the solution performance. Finally, we plan to incorporate practical features into the problem formulation, such as potential failures during task transmission and processing, to better understand their impact on service quality under realistic conditions.

Acknowledgements This study was financed by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Code 001 and by the UFPE/ Propesqi via Edital Nº 06/2024

References

1. N. Al-Falahy and O. Y. Alani. Technologies for 5g networks: Challenges and opportunities. *IT Professional*, 19(1):12–20, 2017. doi: 10.1109/MITP.2017.9.
2. S. Aljubayrin. Computational energy efficient trajectory planning for uav-enabled 6g mec communication network. *Physical Communication*, 57:102000, 2023. ISSN 1874-4907. doi: https://doi.org/10.1016/j.phycom.2023.102000.
3. Y. Gan and Y. He. Trajectory optimization and computing offloading strategy in uav-assisted mec system. In *2021 Computing, Communications and IoT Applications (ComComAp)*, pages 132–137, 2021. doi: 10.1109/ComComAp53641.2021.9652887.
4. B. Li, Y. Liu, L. Tan, H. Pan, and Y. Zhang. Digital twin assisted task offloading for aerial edge computing and networks. *IEEE Transactions on Vehicular Technology*, 71(10):10863–10877, 2022. doi: 10.1109/TVT.2022.3182647.
5. B. Li, R. Yang, L. Liu, J. Wang, N. Zhang, and M. Dong. Robust computation offloading and trajectory optimization for multi-uav-assisted mec: A multiagent drl approach. *IEEE Internet of Things Journal*, 11(3):4775–4786, 2024. doi: 10.1109/JIOT.2023.3300718.
6. L. Lin, X. Liao, H. Jin, and P. Li. Computation offloading toward edge computing. *Proceedings of the IEEE*, 107(8):1584–1607, 2019. doi: 10.1109/JPROC.2019.2922285.
7. A. Nehra, P. Consul, I. Budhiraja, G. Kaur, N. Nasser, and M. Imran. Federated learning based trajectory optimization for uav enabled mec. In *ICC 2023 - IEEE International Conference on Communications*, pages 1640–1645, 2023. doi: 10.1109/ICC45041.2023.10278857.
8. X. Qin, Z. Song, Y. Hao, and X. Sun. Joint resource allocation and trajectory optimization for multi-uav-assisted multi-access mobile edge computing. *IEEE Wireless Communications Letters*, 10(7):1400–1404, 2021. doi: 10.1109/LWC.2021.3068793.
9. A. Ranjha, D. Naboulsi, M. E. Emary, and F. Gagnon. Facilitating urllc vis-á-vis uav-enabled relaying for mec systems in 6-g networks. *IEEE Transactions on Reliability*, pages 1–15, 2024. doi: 10.1109/TR.2024.3357356.
10. Q. Wu, Y. Zeng, and R. Zhang. Joint trajectory and communication design for uav-enabled multiple access. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6, 2017. doi: 10.1109/GLOCOM.2017.8254949.
11. Y. Zeng and R. Zhang. Energy-efficient uav communication with trajectory optimization. *IEEE Transactions on Wireless Communications*, 16(6):3747–3760, 2017. doi: 10.1109/TWC.2017.2688328.
12. C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato. Completion time and energy optimization in the uav-enabled mobile-edge computing system. *IEEE Internet of Things Journal*, 7(8): 7808–7822, 2020. doi: 10.1109/JIOT.2020.2993260.
13. L. Zhang, Z.-Y. Zhang, L. Min, C. Tang, H.-Y. Zhang, Y.-H. Wang, and P. Cai. Task offloading and trajectory control for uav-assisted mobile edge computing using deep reinforcement learning. *IEEE Access*, 9:53708–53719, 2021. doi: 10.1109/ACCESS.2021.3070908.