

An Intelligent eUPF for Time-Sensitive Path Selection in B5G Edge Networks

Rodrigo Moreira[1], Larissa F. Rodrigues Moreira[1], Tereza C. Carvalho[2], and Flávio de Oliveira Silva[3(✉)]

[1] Federal University of Viçosa, Minas Gerais, Brazil
{rodrigo,larissa.f.rodrigues}@ufv.br
[2] University of São Paulo, São Paulo, Brazil
terezacarvalho@usp.br
[3] University of Minho, Braga, Portugal
flavio@di.uminho.pt

Abstract. In Beyond 5G (B5G) networks, intelligent, flexible traffic management is essential to meet the stringent speed and reliability requirements of new applications. This paper presents an improved User Plane Function (eUPF) design that uses a Deep Q-Network (DQN) agent for real-time path selection between Multi-access Edge Computing (MEC) and cloud endpoints. The path selection problem is formulated as a Partially Observable Markov Decision Process (POMDP). We propose a novel passive delay measurement method that uses eBPF programs to link TEID-based timestamps in GTP-U traffic, allowing for low-cost delay estimation without active testing. Experiments show that the DQN agent substantially outperforms a random baseline, with lower average latency, more stable rewards, and more reliable low-delay path choices. These results demonstrate the effectiveness of AI-driven control in B5G core networks and the promise of reinforcement learning for modern network management.

Keywords: Beyond 5G Networks · User Plane Function · Reinforcement Learning · eBPF/XDP · Multi-access Edge Computing · Time-Sensitive Path Selection

1 Introduction

The increasing prevalence of compute-intensive applications, coupled with the diversity of mobile devices, results in substantial data volumes in 5th Generation Mobile Network (5G). To accommodate this demand, especially in the Beyond Fifth Generation (B5G) context, it is crucial to minimize service access delay by bringing services closer to users through approaches such as Multi-access Edge Computing (MEC). Industries such as automation, autonomous driving, and cloud gaming, for instance, impose stringent ultralow-latency requirements [18], and in response, placing network functions near the edge not only helps meet these demands but also offers a promising way to alleviate backbone traffic

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16768, pp. 432–446, 2027.
https://doi.org/10.1007/978-3-032-30488-9_27

surges, particularly for Network Function Virtualization (NFV)-based components [2]. This drives ongoing research on efficient path-selection strategies in the data plane of mobile networks.

Building on these advancements, the state of the art presents various approaches for service steering and resource allocation within the broader context of path selection [13], particularly across two key segments of mobile networks. The first research stream focuses on the Radio Access Network (RAN), where strategies range from traffic-aware RAN slicing to open and disaggregated solutions, such as Open RAN [1]. The second stream focuses on traffic steering in the mobile network data plane, with particular emphasis on solutions to enhance the performance of the User Plane Function (UPF) [19]. Despite this progress, the literature identifies further opportunities to instrument the UPF to enable greater programmability and intelligent behavior [3].

Recent efforts to enhance programmability and intelligent behavior in the mobile core have led to the introduction of analytics capabilities in the 3rd Generation Partnership Project (3GPP) Release 16, facilitated by the Network Data Analytics Function (NWDAF) entity [7]. However, the mechanisms for data collection and component instrumentation have posed challenges across various deployments [12]. This study explores the potential of transforming the UPF into a more intelligent entity, capable of considering latency by examining GPRS Tunnelling Protocol (GTP) Tunnel Endpoint Identifiers (TEIDs) and employing a Reinforcement Learning (RL) mechanism to optimize its real-time performance.

In this work, we introduce and evaluate the UPF instrumentation for estimating the end-user latency of the User Equipment (UE) by analyzing the time shifts of TEIDs arriving at the UPF. This approach empowers the UPF to autonomously select the optimal interface for traffic forwarding and apply intelligent traffic steering based on latency awareness, thereby reducing end-to-end delay. The efficacy of our methodology is demonstrated through an MEC and cloud-based application, validating how the UPF, in conjunction with an RL agent, can non-intrusively direct traffic along various paths by monitoring UE latency and using eXpress Data Path (XDP). This design positions the eUPF as an intelligent user-plane element for time-sensitive path selection across MEC and cloud environments.

The contributions of this study are as follows: (i) the development of a novel UPF instrumentation for UE latency estimation; (ii) a comparative evaluation of a naive UPF path selection versus an RL-based approach; (iii) the design and implementation of a non-intrusive telemetry mechanism using XDP to enable real-time TEID monitoring and latency estimation at the UPF; and (iv) an end-to-end experimental validation showcasing how the proposed architecture improves latency-sensitive service delivery in MEC and cloud scenarios.

The remainder of this paper is organized as follows: Sect. 2 presents representative approaches for MEC-aware decision-making in 5G/6G. In Sect. 3, we present our method for intelligent intervention in the dataplane. In Sect. 4, we describe the technologies used to perform the proof of concept. In Sect. 5,

we present the discussions and insights gained from our approach. Finally, in Sect. 6, we provide the conclusions and lessons learned.

2 Related Work

Automated management systems are essential for emerging eXtended Reality (XR) and the metaverse [4,5]. Recent research focus includes instrumenting different segments of 5G, including the RAN, core, and transport, to improve adaptability and performance monitoring. This section reviews studies that use the UPF to measure UE connectivity parameters, adapt to dynamic traffic conditions, and enable Internet Service Providers (ISPs) to track and enforce ServiceLevel Agreement (SLA) guarantees effectively.

Table 1. Summary of representative approaches for MEC aware decision making in Fifth Generation and Sixth Generation networks.

Paper	Learning	Optimization	User Plane Function	Tunnel Endpoint Identifier	Real Testbed
Shokrnezhad et al. [17]	●	●	○	○	○
Tran et al. [19]	○	●	●	○	●
Gan et al. [6]	●	●	○	○	○
Maleki et al. [11]	●	●	○	○	○
Sasithong et al. [16]	●	●	●	○	○
Hisyam et al. [8]	●	●	○	○	○
Nguyen et al. [14]	●	●	●	○	○
Pimpalkar et al. [15]	○	●	○	○	○
Kibalya et al. [10]	●	●	●	○	○
Ibrahimi et al. [9]	●	●	○	○	○
Our Approach	●	●	●	●	●

Deep Reinforcement Learning is widely applied for intelligent resource management. Shokrnezhad et al. [17] address joint Virtual Network Function (VNF) placement, traffic prioritization, and path selection in B5G networks under latency and capacity constraints using Double Deep Q-learning (DDQL) Double Deep Q learning. Gran et al. [6] use Multi-Agent Deep Reinforcement Learning (MADRL) with collaborative Deep Q-Network (DQN) models for computation offloading in 5G MEC systems to minimize total system cost.

Similarly, Sasithong et al. [16] apply DQN and Actor-Critic (AC) algorithms for dynamic UPF allocation in Cellular Vehicle to-Everything (C-V2X) [16]. Nguyen et al. [14] utilize Proximal Policy Optimization (PPO) and Support Vector Machine (SVM) for scaling UPF instances [14], while Kibalya et al. [10] employ policy networks for joint UPF and application placement in multi-slice MEC [10]. For Ultra-Reliable and Low Latency Communications (URLLC), Ibrahimi et al. [9] implement a DQN strategy to balance congestion and throughput [9].

Other research employs heuristics, optimization, or supervised learning. Tran et al. [19] introduce a rule-based computing-aware traffic steering architecture for 5G to route traffic to optimal MEC instances using Segment Routing over

IPv6 (SRv6) [19]. Maleki et al. [11] utilize Multi-Armed Bandit (MAB) and Deep Neural Network (DNN) models for Quality of Service (QoS) aware content delivery under mobility [11]. In 6th Generation Mobile Network (6G) contexts, Pimpalkar et al. [15] formulate a multi-constraint convex problem for Service-Aware Optimal Path Selection (SOPS) [15]. Hisyam Ng and Mahmoodi [8] propose a supervised learning scheme using Random Forests and Mixed Integer Linear Programming (MILP) for dynamic steering [8].

Table 1 summarizes representative approaches for MEC aware decision making in Fifth Generation and Sixth Generation networks using a binary comparison of key capabilities, namely whether each work employs learning, performs explicit optimization, targets the User Plane Function, incorporates Tunnel Endpoint Identifier awareness, and validates results on a real testbed. The table shows that prior studies largely rely on simulation and do not exploit Tunnel Endpoint Identifier signals for data-plane decisions, even when learning or optimization is applied. In contrast, our approach integrates a DQN-based reinforcement learning agent with User Plane Function-level control and explicit Tunnel Endpoint Identifier awareness, and it is validated on a real testbed, enabling passive latency-driven N6 interface selection for Beyond Fifth Generation scenarios spanning MEC and cloud paths.

3 Proposed Approach

Building on the insights and limitations identified in prior work, we propose an enhancement to the traditional UPF, hereafter referred to as eUPF (enhanced UPF), which integrates an RL agent to enable intelligent, real-time path selection for individual network slices. As shown in Fig. 1, the DQN-based agent observes latency and network conditions to dynamically steer traffic across multiple N6 paths (e.g., Path A and Path B), targeting either MEC or Cloud endpoints. This architecture enables the UPF to surpass static policies by continuously adapting to network dynamics, user mobility, and service-level requirements. By leveraging AI control at the data plane, the proposed eUPF architecture enables smarter, slice-aware forwarding decisions aligned with the vision of agile and autonomous 5G/6G core networks.

3.1 Problem Formulation

We formulate adaptive path selection as a Partially Observable Markov Decision Process, where the enhanced UPF applies a slice-specific forwarding policy at run time, and the reinforcement learning agent makes per-flow decisions using the Tunnel Endpoint Identifier (TEID) as the in-kernel key that identifies the active Packet Data Unit (PDU) session associated with that slice context.

States and Transitions. Let $S_t = (s_{n6a}(t), s_{n6b}(t))$ denote the latent environment state at time t, where $s_i(t) \in \{\text{GOOD}, \text{BAD}\}$ for $i \in \{\texttt{n6a}, \texttt{n6b}\}$. Transitions from GOOD to BAD occur independently with fixed probabilities p_i per

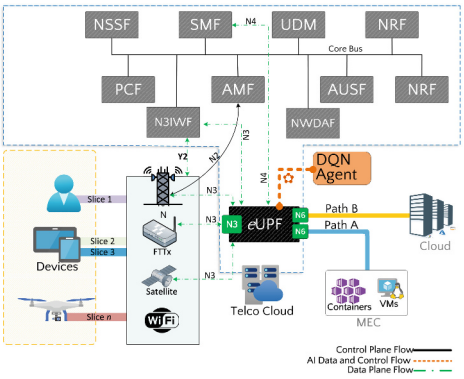


Fig. 1. Proposed Approach.

packet. Once degraded, each interface remains in the BAD state for a deterministic duration D_i before reverting to the GOOD state. These semi-Markov dynamics introduce memory that must be inferred from observations.

Observations. The agent does not observe S_t directly. Instead, after selecting an action $A_t \in \{\texttt{n6a}, \texttt{n6b}\}$, it receives a noisy delay measurement. Let i denote the interface selected by A_t (that is, $i = A_t$). The observation process (Eq. 1) is:

$$\begin{aligned} RTT(A_t) &= d_i \cdot \mathbb{I}[s_i(t) = \text{BAD}] + \xi_t \\ \xi_t &\sim \mathcal{U}(-J_{\max}, +J_{\max}) \end{aligned} \tag{1}$$

where d_i is the additional delay incurred when interface i is in the BAD state, $\mathbb{I}[\cdot]$ is the indicator function, equal to 1 if the condition holds and 0 otherwise, and ξ_t models bounded measurement noise (jitter) with maximum amplitude $J_{\max}$.

Because the agent operates under partial observability, we define a scalar observed state s_t derived from the latest measurement. In our implementation, s_t corresponds to the most recent per TEID delay proxy exposed by the data plane. This s_t is the input used by the learning agent at decision time t.

Actions and Rewards. At each time step t, the agent selects an interface $A_t \in \{\texttt{n6a}, \texttt{n6b}\}$ via a policy $\pi(a_t|s_t)$. We define the reward as the negative observed delay (Eq. 2):

$$R_t = -RTT(A_t). \tag{2}$$

We normalize R_t via min-max scaling to stabilize training, mapping rewards to a bounded range while preserving ordering, so that lower delays correspond to higher normalized rewards.

Learning Objective. The agent aims to maximize expected discounted return (Eq. 3):

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R_t\right], \tag{3}$$

where $\gamma \in (0, 1]$ is the discount factor, T is the episode horizon in decision steps, and the expectation is taken over trajectories induced by policy π and the environment dynamics. Because S_t is latent and only s_t is observed, deep reinforcement learning, particularly DQN, is suitable to learn effective decisions from sequences of noisy measurements.

3.2 TEID-Correlated Delay Measurement

To avoid injecting additional active probes, we leverage a passive data-plane measurement approach based on TEID- correlated timestamps within the UPF. The system attaches eBPF programs directly to the GTP-U traffic path to extract the TEID from each packet. For each TEID i, a stateful map stores a request timestamp t_i^{req} when a request packet is observed.

When the corresponding response packet for the same TEID i is observed at the UPF, the program computes an in network round trip proxy (Eq. 4):

$$\Delta t_i = t_i^{\text{resp}} - t_i^{\text{req}}. \tag{4}$$

Here, t_i^{req} is the timestamp captured for the request, and t_i^{resp} is the timestamp captured for the matched response, both observed at the UPF for the same TEID i. In our proof of concept, the request-response pattern is instantiated with Internet Control Message Protocol (ICMP) echo traffic, and the response is identified by matching the echo reply to the previously observed echo request within the same TEID context, thereby providing a consistent pairing signal for timestamp correlation.

This measurement is a UPF observed delay proxy, not a fully instrumented end-to-end RTT across the entire Data Network (DN) and RAN. Its purpose is to provide a low-overhead, online signal that reflects the experienced user-plane delay under the current forwarding choice, enabling the agent to adapt its decisions even when only partial visibility is available.

To implement passive latency measurement, we developed an in-kernel routine that associates TEID- identified flows with timestamped events on the UPF data plane. Algorithm 1 presents the logic executed by the eBPF program attached to the GTP-U traffic path. Upon receiving a packet, the program extracts the corresponding TEID and records a request timestamp if none already exists. When a subsequent packet for the same TEID arrives, it computes the delay proxy and resets tracking state, enabling continuous measurement without additional probe injection.

4 Evaluation Setup

Our evaluation was conducted on the FABRIC testbed, using a high-performance virtual machine provisioned with 32 GB of RAM and 32 vCPUs. The envi-

Algorithm 1: TEID-Correlated Round-Trip Time (RTT) Measurement.

```
Initialize rtt_map as a hash map with key: TEID and value: round_trip structure;
Function measure_rtt(skb):
    Extract TEID from incoming GTP-U packet;
    now ← current kernel timestamp (bpf_ktime_get_ns());
    Lookup entry in rtt_map by TEID;
    if entry does not exist then
        Insert new entry with ts_request ← now;
    else
        if ts_request is zero then
            Set ts_request ← now;
        else
            Compute RTT: last_rtt ← now − ts_request;
            Reset ts_request ← 0;
            Increment count;
```

ronment was orchestrated with Kubernetes v1.28.15 running on Ubuntu, and Free5GC was used to emulate the 5G core network. Figure 2 illustrates the deployment, where the DQN agent interacts with the user plane through Extended Berkeley Packet Filter (eBPF)/XDP hooks and shared maps. We leveraged a custom BPF toolset to implement programmable datapath control, enabling dynamic action updates and observation retrieval directly at the UPF. This setup enables reproducible experimentation for low-level latency estimation and adaptive path selection across the `n6a` and `n6b` interfaces.

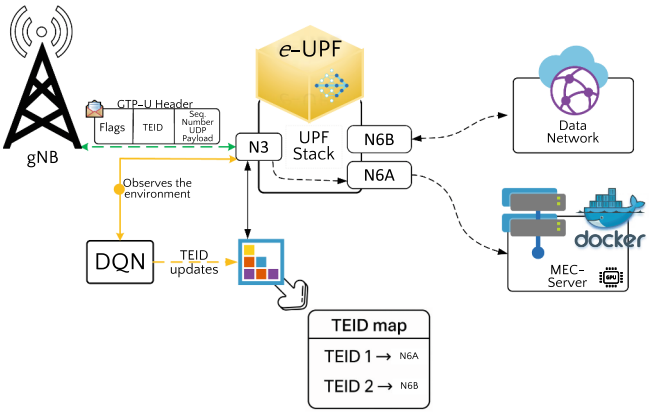


Fig. 2. Experimental Setup.

The stochastic degradation parameters used in our experiments are summarized in Table 2. For each interface $i \in \{\texttt{n6a}, \texttt{n6b}\}$, a degradation event is triggered independently at each packet arrival according to a failure probability p_i. Once triggered, the interface remains degraded for a fixed failure duration D_i and then automatically recovers. When an interface is in the degraded state,

an additional delay d_i is induced, matching the observation model in Eq. 1. A bounded jitter term with maximum amplitude $J_{\max}$ is applied globally across both paths, as defined in Eq. 1.

Table 2. Network Path Parameters for Delay Induction Scenario.

Path	Failure Probability	Failure Duration (s)	Bad State Delay (ms)
Path A (MEC)	0.01	10	800
Path B (Cloud)	0.10	20	800
Global Parameter: Maximum Jitter $J_{\max} = 3$ ms			

4.1 DQN Based Path Selection Architecture

The learning component is implemented as an DQN, which approximates the action-value function $Q(s_t, a_t; \theta)$ using a deep neural network parameterized by θ. The agent receives as input a scalar state s_t derived from the most recent per TEID delay proxy, and outputs Q values for the two discrete actions corresponding to the available egress interfaces, namely `n6a` and `n6b`. Action selection follows an ϵ greedy exploration strategy, where ϵ decays from an initial exploration probability to a smaller final value over training.

The network architecture is a feedforward multilayer perceptron with two hidden layers of 64 units each and ReLU activations. Learning is stabilized using experience replay and a separate target network with parameters θ^- that are periodically synchronized from the online network. At each decision step, the agent stores a transition (s_t, a_t, r_t, s_{t+1}) in a replay buffer and samples mini-batches to update θ by minimizing the temporal difference objective defined in Eq. (5). Table 3 summarizes the hyperparameters and model configuration used in all experiments.

4.2 Agent Environment Interaction via XDP Maps

The interaction between the DQN agent and the forwarding plane is implemented using two XDP maps that serve as the interface between learning logic and packet processing. The first map exposes observations to the agent by providing per TEID delay values computed as the UPF observed proxy Δt_i in Eq. 4, obtained by correlating TEID keyed timestamps for request and response packets observed at the same UPF datapath. In the proof of concept, ICMP traffic provides the request-response pattern required for pairing, while the measurement itself remains passive with respect to additional monitoring probes.

The second map enables actions by storing the selected egress interface per TEID, so that packet forwarding for that PDU session follows the agent's chosen action. Specifically, at each decision time t, the agent selects $a_t \in \{\texttt{n6a}, \texttt{n6b}\}$ and

Table 3. Hyperparameters and Network Architecture of the DQN Agent.

Parameter	Value
Discount factor (γ)	0.99
Learning rate (Adam) (α)	5×10^{-4}
Batch size	32
Replay memory size	2000 transitions
Exploration start (ϵ_{start})	0.9
Exploration end (ϵ_{end})	0.01
Exploration decay rate	0.990
Target network update frequency	Every 5 episodes
Training episodes	400
Network Architecture	
Input dimension	1 (scalar state)
Hidden layers	2 × 64 units (Linear + ReLU)
Output dimension	2 (Q values for `n6a`, `n6b`)

writes the corresponding value into the action map, which the XDP program reads to enforce forwarding on the chosen N6 path.

Learning follows the standard DQN update rule, minimizing the temporal difference loss, as Eq. (5):

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1})}\Big[\big(r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-) - Q(s_t, a_t; \theta)\big)^2\Big] \tag{5}$$

where $Q(s_t, a_t; \theta)$ is the online action value network, θ^- denotes the target network parameters, γ is the discount factor (Table 3), and (s_t, a_t, r_t, s_{t+1}) are transitions sampled from the replay buffer. The reward r_t is computed from the observed delay using Eq. 2 and then normalized by min-max scaling, ensuring stable gradients during training. Episodes last 60 s, during which the agent repeatedly reads the latest per TEID observation state s_t, selects an action, updates the action map, and stores the resulting transition for learning.

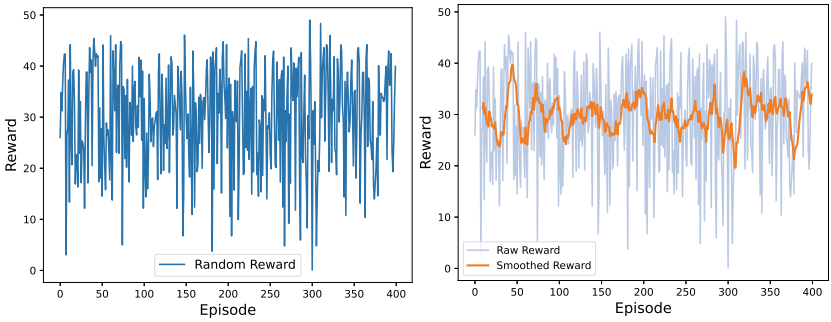
This design enables the DQN agent to translate passive latency observations into fine-grained forwarding control, achieving adaptive N6 interface selection under stochastic path degradations.

5 Results and Discussion

Building on the experimental setup and the agent-environment interaction described above, this section analyzes learning dynamics, policy behavior, and

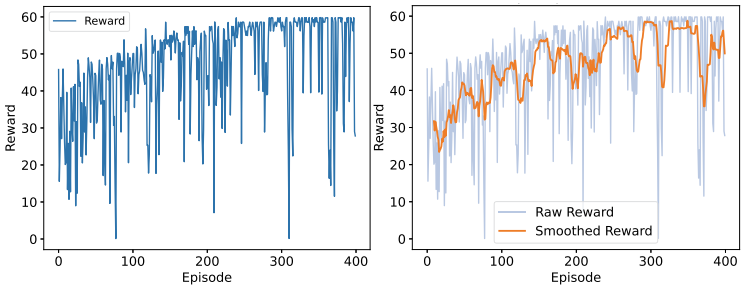
latency outcomes achieved by the proposed DQN- based path selection strategy. We compare it against a random baseline across multiple metrics, including reward evolution, interface selection patterns, and measured round-trip time.

Figure 3 reports the reward dynamics of the random baseline across 400 training episodes. In Fig. 3 (a), the per-episode reward shows high variability, which is expected under stochastic action selection and stochastic path degradations. Importantly, the reward does not exhibit a consistent improving trend, indicating that the baseline does not adapt its decisions over time. Figure 3 (b) reinforces this conclusion, since the rolling mean with window size 10 remains relatively flat, providing a stable reference for assessing whether learning yields consistent performance gains.



(a) Raw reward per episode. (b) Smoothed reward.

Fig. 3. Comparison of raw and smoothed cumulative rewards (rolling mean, window size 10) over the training episodes of the baseline policy.

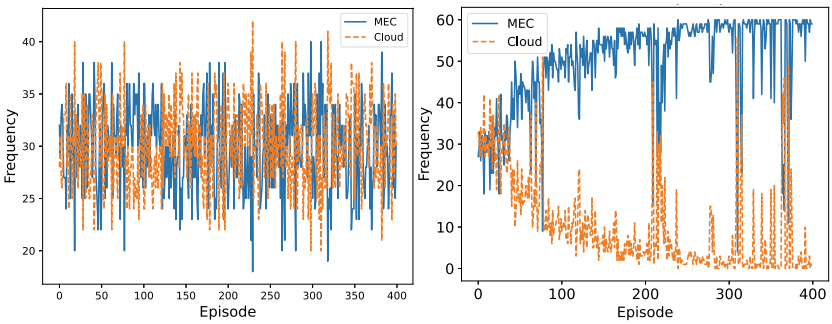


(a) Raw reward per episode. (b) Smoothed reward.

Fig. 4. Comparison of raw and smoothed cumulative rewards (rolling mean, window size 10) over the training episodes of the DQN agent.

DQN Learns an Effective and Stable Low-latency Policy under Stochastic Degradations. Figure 4 shows the reward evolution of the DQN across the same 400 training episodes. The raw reward signal in Fig. 4 (a) is initially noisy due to exploration and the inherent variability of the environment. When applying a rolling mean with a window size of 10 in Fig. 4 (b), a clear upward trend emerges, with rewards improving steadily until convergence around episode 250. This behavior indicates that the agent learns a policy that consistently achieves higher reward, corresponding to lower observed delay, while occasional drops are plausible due to exploration and transient adverse conditions.

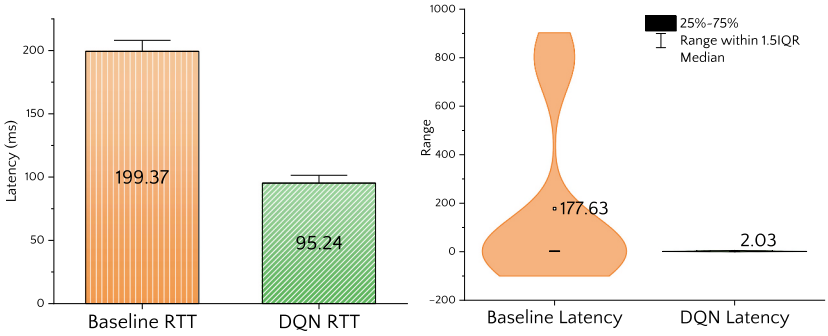
Figure 5 connects learning dynamics to policy behavior by showing action distributions over the 400 episodes. The baseline in Fig. 5 (a) alternates between the MEC path and the cloud path without a structured pattern, consistent with random selection. In contrast, Fig. 5 (b) shows that the DQN increasingly favors the MEC interface, which was configured with lower latency and jitter, while progressively reducing reliance on the cloud path. This shift indicates that the agent identifies the more reliable interface and sustains this preference as learning progresses, rather than oscillating between choices.



(a) Baseline policy. (b) DQN policy.

Fig. 5. Action distribution across all training episodes for both the baseline and the DQN policies.

Figure 6 quantifies the latency impact of these decisions. Overall episodes in Fig. 6 (a), the DQN attains a substantially lower mean round trip time, 95.24 ms, compared to 199.37 ms for the baseline, demonstrating its ability to exploit the lower latency interface. Focusing on the last 50 episodes in Fig. 6 (b), the contrast becomes stronger in terms of stability; the DQN maintains a mean variation of 2.03 ms, whereas the baseline exhibits a volatile range of 177.63 ms. Together, these results show that the learned policy improves both average latency and temporal consistency, while the baseline remains erratic under the same stochastic conditions. These results show that the proposed eUPF effectively supports time-sensitive path selection by consistently favoring the most suitable interface under dynamic conditions.



(a) Full episodes. (b) Last 50 episodes.

Fig. 6. Comparison of average round-trip time.

Learned Steering Decisions Materialize as Consistent Data Plane Forwarding Toward the Best Path. Figure 7 presents the distribution of packets forwarded through each interface under both policies. During the experiment, the User Equipment generated Internet Control Message Protocol traffic at one packet per second, and the plot aggregates transmissions over 10-second intervals, averaged across the entire duration. Under the DQN policy, more packets are forwarded to MEC, with an average of 3.29 per interval, compared to 0.57 to the cloud, reflecting a sustained preference for the lower-latency interface. In contrast, the baseline remains nearly balanced, with 1.77 and 1.73 packets per interval toward MEC and cloud, respectively, consistent with uniform random decisions. This forwarding behavior aligns with the latency outcomes, confirming that the agent translates observed delays into actionable interface selection decisions in the data plane.

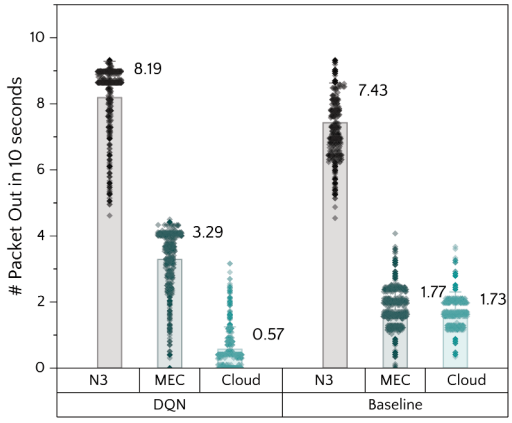


Fig. 7. Packet out distribution.

Table 4. Descriptive statistics for DQN and baseline in the last 50 episodes.

Metric	DQN (± SD)	Baseline (± SD)
Average Reward	49.40 ± 14.92	30.61 ± 9.31
Average Latency [ms]	298.46 ± 443.81	486.57 ± 398.46
Reward Median	58.59	33.25
Latency Median [ms]	6.95	453.86
Reward Range	[11.53, 59.84]	[10.41, 44.18]
Latency Range [ms]	[4.82, 1439.40]	[5.01, 1290.59]

Table 4 summarizes the last 50 episodes, where the reinforcement learning policy is already stabilized. The DQN achieves higher rewards (49.40 ± 14.92 vs. 30.61 ± 9.31) and lower average latency (298.46 ± 443.81 ms vs. 486.57 ± 398.46 ms). The median latency further highlights stability gains, 6.95 ms for the DQN versus 453.86 ms for the baseline. The ranges indicate that both policies can still experience occasional high-delay events due to induced degradations, whereas the DQN makes more decisions on the better-performing interface, thereby reducing overall delay and variability relative to the baseline.

6 Concluding Remarks

While B5G SLA management often relies on simulation-based admission control, it lacks real-time intelligent intervention in the data plane. Our evaluation on the FABRIC testbed demonstrates that the DQN based agent achieves stable convergence and significantly reduces latency compared to random baselines. These results indicate that transforming the UPF into an intelligent *e*UPF enables autonomous slice-aware, time-sensitive path selection in B5G edge networks.

Although this study focuses on uplink steering within a specific testbed scenario, it validates the feasibility of non-intrusive telemetry for RL observations. Future research will investigate alternative algorithms and packet-queue prioritization to enhance efficiency for both uplink and downlink traffic. This research provides a critical foundation for programmable architectures and intelligent network management in the era of low-latency mobile services.

Acknowledgments. The authors thank the National Council for Scientific and Technological Development (CNPq) under grant number 421944/2021-8 (call CNPq/ MCTI/ FNDCT 18/2021), FAPEMIG (Grant #APQ00923-24), FAPESP MCTIC/CGI Research project 2018/23097-3 - SFI2 - Slicing Future Internet Infrastructures. FCT has also supported this work – Fundação para a Ciência e Tecnologia within the R&D Unit Project Scope UID/00319/Centro ALGORITMI (ALGORITMI/UM).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Agarwal, B., Irmer, R., Lister, D., Muntean, G.M.: Open ran for 6G networks: architecture, use cases and open issues. IEEE Commun. Surv. Tutorials, 1–1 (2025). https://doi.org/10.1109/COMST.2025.3562429
2. Bellin, A., Di Cicco, N., Munaretto, D., Granelli, F.: Power consumption-aware 5G edge UPF selection using deep reinforcement learning. In: 2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 1–6 (2024). https://doi.org/10.1109/NFV-SDN61811.2024.10807472
3. Chen, C.C., Chang, C.Y., Nikaein, N.: IUP: integrated and programmable user plane for next-generation mobile networks. IEEE Netw. **39**(3), 91–98 (2025). https://doi.org/10.1109/MNET.2025.3551245
4. Christopoulou, M., Koufos, I., Xilouris, G., Dimitriou, N.: 5G/6G architecture evolution for XR and metaverse: feasibility study, security, and privacy challenges for smart culture applications. IEEE Access **13**, 103077–103094 (2025). https://doi.org/10.1109/ACCESS.2025.3578595
5. Donatti, A., et al.: Survey on machine learning-enabled network slicing: covering the entire life cycle. IEEE Trans. Netw. Serv. Manage. **21**(1), 994–1011 (2024). https://doi.org/10.1109/TNSM.2023.3287651
6. Gan, Z., Lin, R., Zou, H.: A multi-agent deep reinforcement learning approach for computation offloading in 5G mobile edge computing. In: 2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid), pp. 645–648 (2022). https://doi.org/10.1109/CCGrid54584.2022.00074
7. Garcia-Martin, M.A., Gramaglia, M., Serrano, P.: Network automation and data analytics in 3G PP 5G systems. IEEE Netw. **38**(4), 182–189 (2024). https://doi.org/10.1109/MNET.2023.3321524
8. Hisyam Ng, H.A., Mahmoodi, T.: Machine learning-driven dynamic traffic steering in 6G: a novel path selection scheme. Big Data Cogn. Comput. **8**(12) (2024). https://doi.org/10.3390/bdcc8120172
9. Ibrahimi, K., Jouhari, M., Sow, S., Ayoub, F., Kamili, M.E., Chougdali, K.: Reinforcement learning for optimized resource allocation in 5G URLLC. In: 2024 7th International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–6 (2024). https://doi.org/10.1109/CommNet63022.2024.10793310
10. Kibalya, G., et al.: Joint UPF and application placement in multi-slice edge networks: a reinforcement learning strategy. In: 2025 IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6 (2025). https://doi.org/10.1109/WCNC61545.2025.10978828
11. Maleki, E.F., Ma, W., Mashayekhy, L., La Roche, H.J.: QoS-aware content delivery in 5G-enabled edge computing: learning-based approaches. IEEE Trans. Mob. Comput. **23**(10), 9324–9336 (2024). https://doi.org/10.1109/TMC.2024.3363143
12. Moreira, L., Moreira, R., Silva, F., Backes, A.: Towards cognitive service delivery on B5G through AIaaS architecture. In: Anais do IV Workshop de Redes 6G, pp. 1–8. SBC, Porto Alegre (2024). https://doi.org/10.5753/w6g.2024.3304
13. Moreira, R., Rosa, P.F., Aguiar, R.L.A., de Oliveira Silva, F.: NASOR: a network slicing approach for multiple autonomous systems. Comput. Commun. **179**, 131–144 (2021). https://doi.org/10.1016/j.comcom.2021.07.028
14. Nguyen, H.T., Van Do, T., Rotter, C.: Scaling UPF instances in 5G/6G core with deep reinforcement learning. IEEE Access **9**, 165892–165906 (2021). https://doi.org/10.1109/ACCESS.2021.3135315

15. Pimpalkar, Y., Ravindran, S., Bapat, J., Das, D.: A novel e2e path selection algorithm for superior QoS and QoE for 6g services. IEEE Trans. Netw. Serv. Manage. **22**(2), 1174–1187 (2025). https://doi.org/10.1109/TNSM.2024.3519707
16. Sasithong, P., Sanguanpuak, T., Vanichchanunt, P., Wuttisittikulkij, L.: User plane function (UPF) allocation for C-V2X network using deep reinforcement learning. IEEE Access **13**, 4547–4561 (2025). https://doi.org/10.1109/ACCESS.2024.3524886
17. Shokrnezhad, M., Taleb, T., Dazzi, P.: Double deep Q-learning-based path selection and service placement for latency-sensitive beyond 5G applications. IEEE Trans. Mob. Comput. **23**(5), 5097–5110 (2024). https://doi.org/10.1109/TMC.2023.3301506
18. Silva, B., Moreira, L.R., de Oliveira Silva, F., Moreira, R.: Optimizing edge gaming slices through an enhanced user plane function and analytics in beyond-5G networks. In: Anais do XVI Workshop de Pesquisa Experimental da Internet do Futuro, pp. 1–8. SBC, Porto Alegre (2025).https://doi.org/10.5753/wpeif.2025.8714
19. Tran, M.N., Duong, V.B., Kim, Y.: Design of computing-aware traffic steering architecture for 5G mobile user plane. IEEE Access **12**, 88370–88382 (2024). https://doi.org/10.1109/ACCESS.2024.3418960