

SMARTWIN: Smart Reinforcement Learning Based Digital Twin for Resource Optimization in O-RAN

Mahnoor Yaqoob
Middlesex University
my365@live.mdx.ac.uk

Ramona Trestian
Middlesex University
R.Trestian@mdx.ac.uk

Mallik Tatipamula
Ericsson Silicon Valley
mallik.tatipamula@ericsson.com

Huan X. Nguyen
Middlesex University
H.Nguyen@mdx.ac.uk

Abstract—The exponential increase of heterogeneous devices and vertical applications in 5G and Beyond 5G (B5G) networks has catalysed a paradigm shift in cellular network design, fostering a transition towards disaggregated, fully virtualized, and programmable architectures. To meet these demands, the Open Radio Access Network (O-RAN) architecture standardized by the O-RAN Alliance enables hardware independence, while the use of Digital Twins (DTs) for network emulation and validation is becoming increasingly popular. Although O-RAN introduces new technologies and opportunities, advances in Machine Learning (ML) based network automation remain limited, mainly because of insufficient large-scale datasets and experimental testing environments. To address the challenges of accurate network modelling and efficient resource management in O-RAN, this paper introduces SMARTWIN, a Smart Reinforcement Learning based Digital Twin framework. SMARTWIN enables precise network modelling and intelligent resource allocation and scheduling, with the objective of optimizing the Key Performance Indicators (KPIs) of eMBB, mMTC, and URLLC network slices. It also implements a Conservative Q-Learning (CQL) algorithm to learn from data generated by the DT, where the performance comparison with the baseline Implicit Q-Learning (IQL) algorithm shows an improvement of approximately 41% across all slices. This indicates that the proposed SMARTWIN framework can generalize well beyond the behaviour policy, enabling more efficient and intelligent resource allocation decisions within the DT.

I. INTRODUCTION

The ever growing demand of higher data rates, ultra low latency expectations, seamless connectivity and exponential rise in user applications have been driving the shift from 5G to Beyond 5G (B5G) and 6G networks. Future 6G systems are expected to support diverse services such as augmented and virtual reality (AR/VR), immersive media, autonomous systems, and telemedicine [2], each with unique performance and reliability demands. The 5G era has already introduced key innovations such as open interfaces, network disaggregation, and software-defined architectures within the Radio Access Network (RAN), paving the way for the Open RAN (O-RAN) paradigm [1]. O-RAN enables flexibility, interoperability, and programmability by decoupling hardware and software components while promoting open innovation through standardized interfaces. As we progress toward 6G, the network design architecture is expected to become highly user centric, intelligent, and autonomous. In this evolution, Digital Twins (DTs) are envisioned to play a pivotal role by enabling real time, data-driven network modelling, performance monitoring,

and intelligent control as shown in Fig. 1. A DT network creates a virtual replica of the physical network [3], capable of continuously mirroring its state, analyzing Key Performance Indicators (KPIs), and performing predictive analytics and decision making through closed-loop optimization. Furthermore, the integration of Artificial Intelligence (AI) and Machine Learning (ML) within O-RAN and DT frameworks will be essential for realizing fully autonomous network management. AI/ML models can learn from historical and real time data to optimize resource allocation, enhance energy efficiency, and adapt to dynamic traffic patterns. Leveraging AI/ML combined with DT technology [4] will enable proactive predictive maintenance, performance forecasting and network self optimization which are cornerstones of the envisioned self-evolving 6G networks.

A. Related Works and Motivation

The network modelling paradigm is progressively transitioning towards digital representations, while the integration of AI and ML techniques for network optimization and resource management is gaining significant momentum. In this context, several studies have explored the use of DTs for cellular networks, alongside the application of AI and ML algorithms to enhance network performance and efficiency. For instance authors in [5] have presented a Reinforcement Learning (RL) based resource allocation scheme in 6G IOT scenario, called as collective RL or CRL. Their approach is inspired by human collaboration and they have used Markov decision process to simulate the DT system. In contrast the authors in [6] and [7] have used Proximal Policy optimization (PPO) to optimize the KPI of individual slices. In [6] authors designed the ML agent to learn from RAN telemetry and deploy it as an xApp. Similarly authors in [7] used PPO to allocate resources in an O-RAN environment. They have designed slice specific reward functions to optimize the performance of each slice. Their results show that PPO outperforms the standard static resource allocation schemes. In [8], the authors introduce a DT framework for next-generation mobile networks that focuses on constructing digital replicas of network components and utilizing a virtualized RAN to enhance network intelligence. Similarly, [9] outlines a vision for applying DTs within the O-RAN ecosystem, addressing key challenges such as data privacy, data flow management, annotation, and regulatory compliance. The work in [10] presents a theoretical foundation for developing DTs

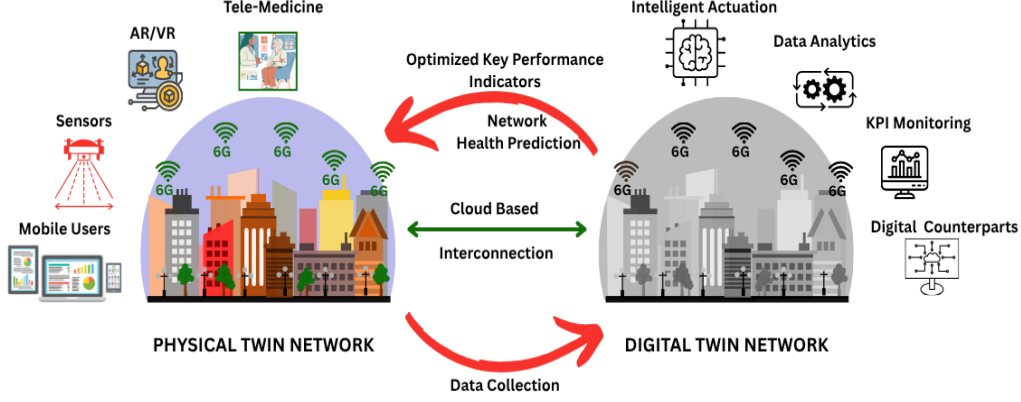


Fig. 1. An Overview of role of Digital Twin Network for Beyond 5G networks along with the applications.

in 6G networks, identifying them as essential enablers for the forthcoming communication paradigm. In another study, [11] Deep Reinforcement Learning (DRL) technique is deployed to design a cooperative DT-based framework aimed at developing a parameterized admission control policy for cellular networks. However, this work focuses primarily on comparing the DT model with simulation based systems rather than a physical network twin. Additionally, [12] proposes a dual-layer DT architecture for intelligent network orchestration, framing the traffic engineering problem in heterogeneous networks through an auto-regressive integrated moving average (ARIMA) model to characterize the DT dynamics.

Most existing studies on DTs remain largely conceptual, offering theoretical insights into their potential role within cellular networks. Similarly, research focusing on AI/ML-based optimization approaches primarily emphasizes online reinforcement learning methods such as PPO, DRL and DQN. However, these approaches face significant challenges, including the limited availability of large-scale datasets, high computational complexity, and the time-intensive nature of online training. To address these limitations, we extend our previous work, ORBIT-DT [13], by introducing an offline Conservative Q-Learning (CQL) framework. The proposed method leverages historical network data to enable efficient learning directly within the DT environment. This design minimizes the computational overhead typically associated with training xApps in the near-real-time RIC, thereby facilitating scalable and resource efficient network optimization.

B. Contributions

To deliver cost effective services, RAN slicing must enable real-time resource management that efficiently adapts to the specific requirements of each slice while dynamically optimizing the corresponding KPIs. Traditional resource allocation approaches, however, fail to meet these demands due to their limited adaptability and responsiveness. To address the gaps in literature and the challenges mentioned above, in this paper, we propose SMARTWIN (A Smart Reinforcement Learning

Digital Twin), which comprises a Physical Twin (PT) of O-RAN, a DT of O-RAN, and a smart resource allocation block. Our contributions are as follows:

- 1) We developed a DT model of the O-RAN architecture, where the PT is implemented using the POWDER testbed [14], and the DT environment is emulated using the open-source Colosseum platform [15]. All O-RAN components are based on the srsRAN framework.
- 2) For network slicing and resource allocation, we designed a dedicated container within the Smart Resource Allocation layer of the proposed SMARTWIN framework. This container is responsible for managing slice orchestration and dynamic scheduling decisions in real time.
- 3) To enable automation and intelligent decision making, we propose a CQL based approach that learns from both historical and synthetically generated traffic data. The proposed CQL algorithm effectively generalizes to unseen network conditions and captures meaningful representations of network dynamics, leading to improved resource management.
- 4) For performance evaluation, we compare the proposed CQL based approach against a baseline Implicit Q-Learning (IQL) algorithm. The experimental results demonstrate that CQL consistently outperforms IQL, confirming its superior learning efficiency and adaptability in optimizing resource allocation within the DT enabled O-RAN environment.

II. PROPOSED SMARTWIN: A SMART REINFORCEMENT LEARNING DIGITAL TWIN

In this section, we provide a comprehensive description of the fundamental components of the proposed SMARTWIN framework, as illustrated in Fig. 2. The SMARTWIN architecture is organized into three functional layers: (1) the Physical Twin, (2) Smart Resource Allocation, and (3) the Digital Twin.

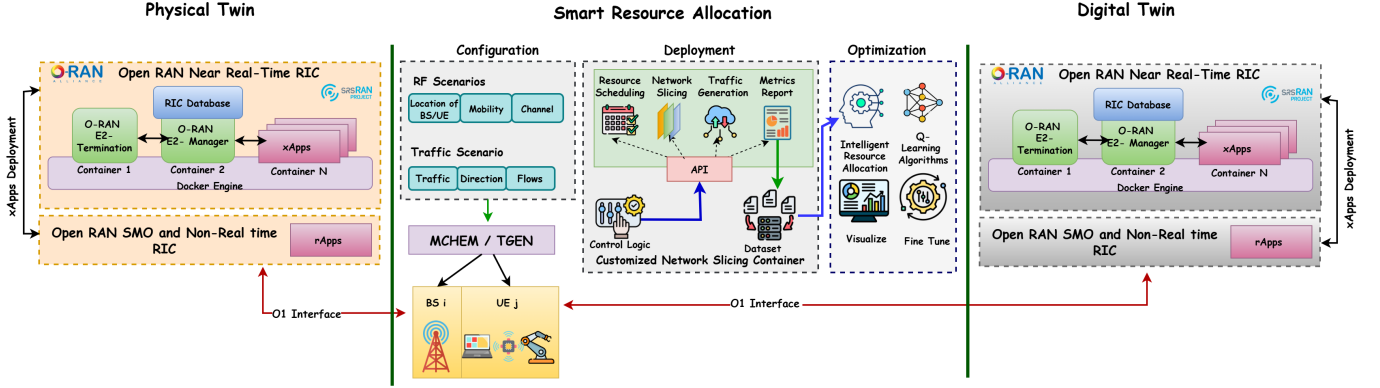


Fig. 2. Proposed Architecture of SMARTWIN for resource allocation and optimization in O-RAN.

A. Physical Twin

In the proposed SMARTWIN framework shown in Fig. 2, the PT is implemented using POWDER [14], a 5G testbed deployed in Salt Lake City, U.S. Additional details about the PT implementation are available in [13]. The POWDER platform features both indoor and outdoor over-the-air (OTA) experimental environments. The indoor OTA testbed consists of four NI X310 SDRs with UBX160 daughterboards, wideband antennas, and linked via fiber to near-edge, high performance computing servers. It also includes four Intel NUC systems, each interfaced with an NI B210 SDR, wideband antennas, and a Quectel RM500Q 5G COTS UE module. The X310 SDR units operate within a 10 MHz frequency band and ensure accurate frequency and time synchronization. The O-RAN testbed for the PT layer was implemented using the open-source srsRAN framework, where we deployed a near-real-time RIC, an E2 Termination Agent, and an E2 Manager Agent. We deployed near real-time RIC, an E2 termination agent, and an E2 Manager Agent in O-RAN setup for the PT layer using open-source srsRAN.

B. Smart Resource Allocation

The Smart Resource Allocation layer in the proposed framework operates in three main stages: **configuration**, **deployment**, and **optimization**. In the **configuration stage**, the DT is initialized to emulate network scenarios that reflect real-world traffic conditions. This stage simulates both radio frequency (RF) and traffic scenarios, as described in [13] [16]. The **deployment stage** is responsible for decisions related to network slicing and scheduling. Three network slices; eMBB, mMTC, and URLLC are deployed, while the scheduling policies considered for traffic generation include Round Robin (RR), Proportional Fairness (PF), and Waterfilling (WF) [13]. After the slicing and scheduling configuration, traffic is generated to produce large-scale datasets derived from RAN metric reports. Finally, in the **optimization stage**, the collected data is used to train the network using the CQL algorithm. This stage enables the automation of resource allocation and fine-tuning of learned policies to improve network performance. The detailed

implementation of the CQL-based resource allocation strategy is provided in Section III.

C. Digital Twin

To emulate the O-RAN near-RT RIC and non-RT RIC functionalities, we utilize Colosseum [15], the world's largest open source wireless network emulator with hardware-in-the-loop support [15]. The Colosseum testbed consists of 128 Standard Radio Nodes (SRNs), a Massive Channel Emulator (MChEM), a Traffic Generator (TGEN), and a set of GPU-enabled compute nodes. Each SRN is implemented using a high-performance Dell PowerEdge R730 server integrated with a USRP X310 Software-Defined Radio (SDR), capable of operating across 10 MHz and 6 GHz frequency bands. The MChEM, composed of 16 NI ATCA 3671 FPGA boards, enables the emulation of diverse radio frequency (RF) propagation environments, referred to as RF scenarios. Complementing this, the TGEN module generates IP traffic flows among multiple nodes, facilitating realistic network interactions and stress testing under various traffic conditions. To ensure modular and scalable deployment, we containerize the E2 Termination and E2 Manager components of the near-RT RIC using Docker, enabling flexible integration and experimentation within the O-RAN architecture. The deployment of DT is similar to the one proposed in ORBIT-DT [13].

III. CQL BASED OPTIMIZATION

Conservative Q-Learning (CQL) is an offline RL algorithm designed to learn reliable value functions from fixed datasets without requiring online environment interaction. Unlike traditional off-policy RL methods that often overestimate Q-values for unseen actions, CQL introduces a conservative regularization term to explicitly penalize overconfident predictions on out-of-distribution actions. This penalty forces the learned Q-function to assign higher values only to actions that are well-supported by the data, thus mitigating extrapolation error, a common challenge in offline learning. In the proposed SMARTWIN framework, CQL leverages historical interaction data collected from distributed units (DUs) to learn optimal resource allocation and scheduling policies across

Algorithm 1 CQL for Resource Allocation in SMARTWIN

Require: Offline dataset $\mathcal{D} = \{(s, a, r, s', \text{slice})\}$ from DUs, slice encoders $\{E_{\text{eMBB}}, E_{\text{mMTC}}, E_{\text{URLLC}}\}$, action space $\mathcal{A}$, γ, η, α

- 1: Initialize Q-networks $Q_\theta, Q_{\bar{\theta}}$; optimizer Adam(η)
- 2: Encode states $z = E_{\text{slice}}(s)$, $z' = E_{\text{slice}}(s')$ and store (z, a, r, z') in buffer $\mathcal{B}$
- 3: **for** each epoch **do**
- 4: Sample batch $(z, a, r, z') \sim \mathcal{B}$
- 5: $y = r + \gamma \max_{a'} Q_{\bar{\theta}}(z', a')$
- 6: $\mathcal{L}_{\text{TD}} = \mathbb{E}[(Q_\theta(z, a) - y)^2]$
- 7: $\mathcal{L}_{\text{CQL}} = \mathbb{E}[\log \sum_a e^{Q_\theta(z, a)} - Q_\theta(z, a_{\text{data}})]$
- 8: Update $\theta \leftarrow \theta - \eta \nabla_\theta (\mathcal{L}_{\text{TD}} + \alpha \mathcal{L}_{\text{CQL}})$
- 9: $\bar{\theta} \leftarrow \tau \bar{\theta} + (1 - \tau) \theta$ ▷ Polyak averaging
- 10: **end for**
- 11: Deploy greedy policy: $\pi^*(s) = \arg \max_a Q_\theta(E_{\text{slice}}(s), a)$

eMBB, mMTC, and URLLC slices. The algorithm jointly optimizes two losses: a standard temporal-difference (TD) loss for value fitting, and a conservative loss that lowers Q-values for rarely observed or unsupported actions. This results in a value function that is both accurate and robust to dataset bias. Once trained, the greedy policy derived from the CQL critic, $\pi^*(s) = \arg \max_a Q_\theta(E_{\text{slice}}(s), a)$, is used for deterministic action selection within the DT, ensuring stable and generalizable performance across diverse traffic and channel conditions. Proposed CQL enables the DT to safely deploy policies learned purely from offline datasets while maintaining high reliability and adaptability across network slices. The procedural steps of proposed methodology are listed in Algorithm 1.

A. State and Action Space

The proposed SMARTWIN includes states and actions space where each state s_t is comprised of current network state which includes UE type, buffer occupancy, PRBs allocated to each UE and downlink throughput. The input to CQL and IQL agents is the features which are passed through a slice specific pre-trained autoencoder E_{slice} to obtain compact latent representation z_t . The action space $\mathcal{A}$ is defined as the product of two discrete control dimensions, the scheduling policy (sched0 for RR, sched1 for WF, sched2 for PF) and the set of 28 possible PRB allocations. This product produce a total of 84 possible actions. This formulation enables the agent to jointly determine both the scheduler type and resource allocation pattern per slice. This allows adaptive optimization of throughput, latency, and reliability across dedicated slices.

B. Slice Specific Reward Function Design

Let b_t denote the downlink buffer size (bytes), x_t denote the downlink throughput (Mbps) and ρ_t denote the granted/requested PRB ratio at time t . For each network slice (eMBB, mMTC, URLLC) we define a slice-specific reward $r_t^{(s)}$ to reflect its service objectives.

TABLE I
KEY NOTATIONS USED IN SECTION III

Symbol	Description
s	Network state
z	Encoded latent state
a	Action (scheduler + PRB allocation)
r	Reward
γ	Discount factor
Q_θ	Q-network with parameters θ
Q_{θ^-}	Target Q-network
α_{CQL}	CQL conservatism weight
τ	Polyak averaging coefficient
$\mathcal{B}$	Replay buffer
E_{slice}	Slice-specific encoder
π^*	Greedy policy derived from Q-function
ρ_t	Granted/requested PRB ratio at time t
b_t	Buffer size at time t
x_t	Throughput at time t
ω_s	Slice reward weights

1) *Per-slice Reward Functions:* For eMBB, the objective is to maintain high throughput while avoiding congestion:

$$r_t^e = \tanh\left(\frac{x_t - T_e}{T_e + \varepsilon}\right), \quad T_e \approx 3.0 \text{ Mbps.} \quad (1)$$

The objective of mMTC is to sustain low but stable throughput:

$$r_t^m = \tanh\left(\frac{x_t - T_m}{T_m + \varepsilon}\right), \quad T_m \approx 0.03 \text{ Mbps.} \quad (2)$$

Whereas the goal of URLLC is to optimize both reliability and delay simultaneously:

$$r_t^u = 0.8 \tanh\left(\frac{\rho_t - T_u}{\sigma_\rho}\right) - 0.2 \tanh\left(\frac{b_t}{\sigma_b}\right), \quad (3)$$

$$T_u \approx 0.95, \quad \sigma_\rho \in [0.05, 0.08], \quad \sigma_b \approx 300.$$

2) *Shared Multi-slice Reward:* For multi-slice joint training, the individual rewards are aggregated as a weighted sum:

$$r_t^{\text{shared}} = \omega_e r_t^e + \omega_m r_t^m + \omega_u r_t^u \quad (4)$$

where

$$\omega_e=0.3, \quad \omega_m=0.2, \quad \omega_u=0.5, \quad \sum_s \omega_s = 1.$$

3) *CQL Bellman Target:* The target value for Conservative Q-Learning (CQL) is defined as:

$$y_t = r_t^{\text{shared}} + \gamma \max_{a'} Q_{\theta^-}(z_{t+1}, a'), \quad (5)$$

where z_t is the latent state obtained from the slice encoder, $\gamma = 0.99$ is the discount factor, and θ^- represents the parameters of the target Q-network.

IV. PERFORMANCE EVALUATION OF PROPOSED SMARTWIN

In order to evaluate the performance of CQL in proposed SMARTWIN, we consider an FDD based system configuration which consists of 50 PRBs, corresponding to a total bandwidth of 10 MHz. The spectrum of each base station is divided into three network slices; eMBB, mMTC, and URLLC, where user equipments (UEs) are statically assigned to their respective slices. UDP traffic is generated using the iPerf3 tool, and the performance of the slices is evaluated based on the applied resource allocation and scheduling policies. We have generated constant bit rate (CBR) traffic of 4 Mbps per UE for eMBB, a poisson traffic of 30 packets per second for mMTC and 10 packets per second for URLLC where each packet has 125 bytes. The traffic scenario has 8 BSs and 42 UEs. The hyperparameters we used for CQL and IQL are listed in Table II.

TABLE II
CQL AGENT HYPERPARAMETERS AND DIGITAL TWIN SIMULATION SETTINGS

Parameter	Symbol	Value
Discount factor	γ	0.99
Learning rate	α	1×10^{-4} (Adam optimizer)
Target update coefficient	τ	0.999 (Polyak averaging)
Expectile coefficient	τ_{exp}	0.7
Advantage temperature	β	3.0
Batch size	B	1024
Training epochs	E	25
Conservatism weight	α_{CQL}	1.5
Behavior cloning weight	α_{BC}	0.05
Q-network L2 regularization	λ_Q	1×10^{-4}
Latent dimension (encoder output)	d_z	5
Replay buffer size	N	3×10^6 samples
Action space dimension	$ \mathcal{A} $	84 (3 schedulers $\times$ 28 PRB indices)
State space dimension	$ \mathcal{S} $	3 (buffer, throughput, ratio)

In the following results, we compare the performance of baseline IQL and proposed CQL algorithms using the same traffic traces for all slices. The results highlight that CQL is better in value calibration, training stability, and scheduling efficiency across all network slices.

A. Value Calibration and Training Stability

In Fig. 3 we compare the evolution of $\mathbb{E}[Q(s, a)]$ and TD loss during training. The results show that CQL maintains a monotonic increase in value estimates reaching approximately 8.7, whereas Q-values of IQL decay towards negative values, indicating critic instability. Similarly, TD loss of CQL stabilizes around 1.5, remaining consistently below 2.0 in comparison to IQL. These finding confirm that CQL enforces better value

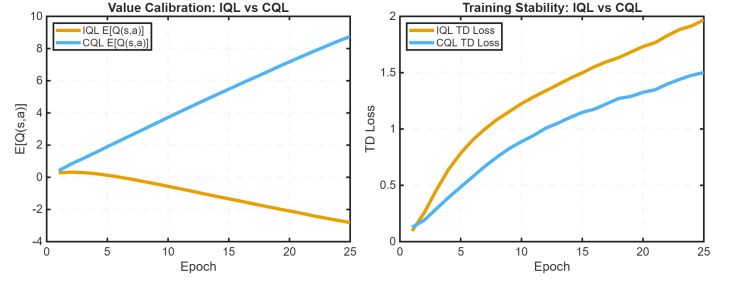


Fig. 3. Comparison of value calibration and training stability for IQL and CQL. (Left) Expected Q-values ($\mathbb{E}[Q(s, a)]$) over epochs. (Right) TD loss during training.

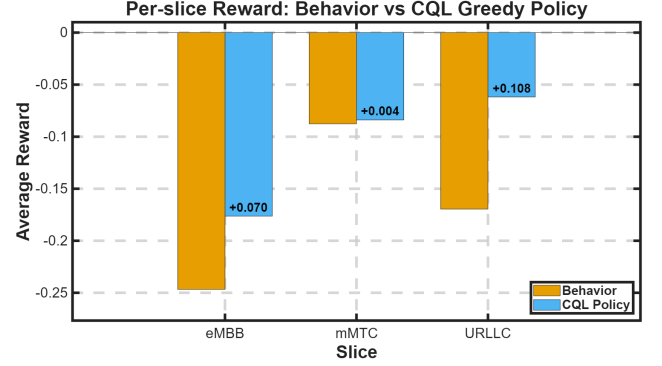


Fig. 4. Per-slice average reward under the behavior policy vs. the learned CQL greedy policy.

calibration and reduces overestimation bias by regularizing unseen actions.

B. Per-Slice Reward Improvement

In Fig. 4 we compare the average per-slice rewards of the behaviour policy of dataset and the learned CQL policy. CQL demonstrates significant improvement in the reliability sensitive URLLC slice (+0.108) and the high throughput eMBB slice (+0.07). The mMTC slice remains stable with a marginal gain (+0.004), confirming that CQL does not degrade performance on low rate devices.

C. Scheduler Usage Adaptation

In Fig. 5 and Fig. 6 we analyze how IQL and CQL differ in scheduler utilization. CQL redistributes scheduling shares across all slices, correcting IQL's strong bias toward sched1. For mMTC, the share of sched0 increases by +0.66, reflecting improved energy and PRB efficiency. In eMBB and URLLC, CQL balances decisions between sched0 and sched1, reducing overuse of latency-intensive schedulers.

Across all metrics, CQL demonstrates improved critic stability, higher expected Q-values, and more efficient scheduler usage compared to IQL. This indicates that conservative value regularization helps prevent overfitting to limited offline data, leading to better generalization and more balanced resource allocation in proposed SMARTWIN.

V. CONCLUSION

In this paper, we propose SMARTWIN, a Smart RL based Digital Twin framework for O-RAN, designed to accurately

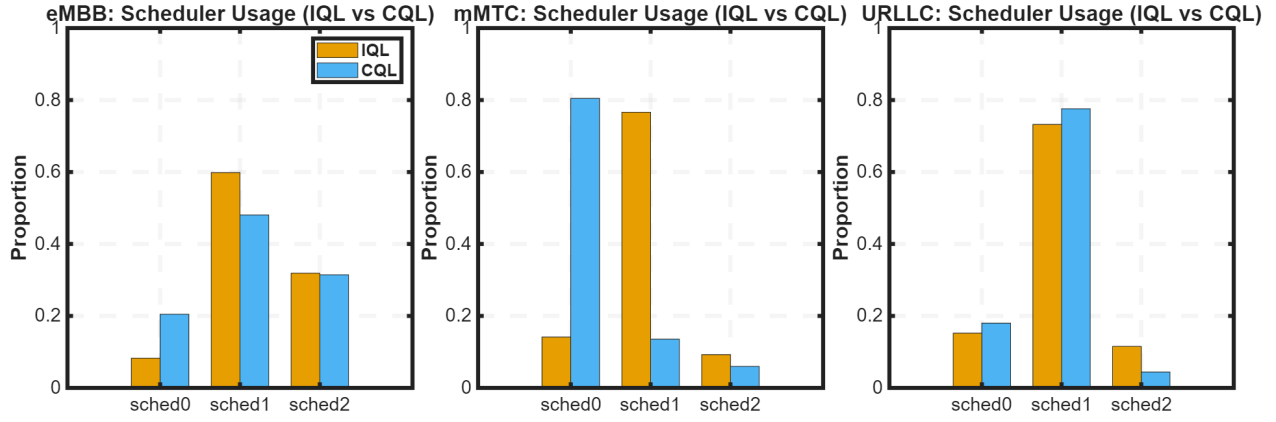


Fig. 5. Per-slice scheduler usage distribution under IQL and CQL policies.

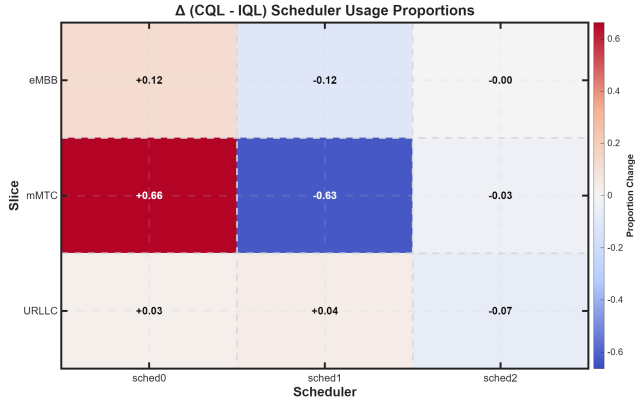


Fig. 6. Heatmap of scheduler usage change $\Delta = \text{CQL} - \text{IQL}$ across slices. Red cells indicate increased scheduler utilization under CQL, while blue cells indicate reduction. CQL shifts mMTC load heavily toward sched0 (+0.66) while balancing eMBB and URLLC usage across schedulers.

model B5G networks. Within the twin O-RAN environment, we deployed multiple scheduling algorithms and resource allocation policies to generate a large-scale dataset representing diverse network conditions. This historical data was then utilized in the reinforcement learning module, where a CQL algorithm was implemented to learn network behavior and traffic dynamics. The proposed framework effectively captured and predicted network patterns, achieving an overall performance improvement of 41% compared to the baseline approach. As a future work we intend to extend the capabilities of proposed SMARTWIN and fine tune the KPIs by implementing offline and online RL techniques.

ACKNOWLEDGMENT

This work was supported by the UK International Science Partnerships Fund (ISPF) under the British Council - International Research Collaborations programme with Japan [grant ID: 1583-Nguyen-Japan].

REFERENCES

[1] Polese, Michele, et al. "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges." *IEEE Communications Surveys & Tutorials* (2023).

[2] Yaqoob, Mahnoor, et al. "Digital-twin-driven end-to-end network slicing toward 6G." *IEEE Internet Computing* (2023).

[3] Mihai, Stefan, et al. "Digital twins: A survey on enabling technologies, challenges, trends and future prospects." *IEEE Communications Surveys & Tutorials* (2022).

[4] Manalastas, Marvin, et al. "From Simulators to Digital Twins for Enabling Emerging Cellular Networks: A Tutorial and Survey." *IEEE Communications Surveys & Tutorials* (2024).

[5] Huang, Zhongwei, et al. "Collective reinforcement learning based resource allocation for digital twin service in 6G networks." *Journal of network and computer applications* 217 (2023): 103697.

[6] Polese, Michele, et al. "CoO-RAN: Developing machine learning-based xApps for open RAN closed-loop control on programmable experimental platforms." *IEEE Transactions on Mobile Computing*, (2022).

[7] Barker, Ryan, et al. "Real: Reinforcement learning-enabled xapps for experimental closed-loop optimization in o-ran with osc ric and srsran." *arXiv preprint arXiv:2502.00715* (2025).

[8] Apostolakis, Nikolaos, et al. "Digital twins for next-generation mobile networks: Applications and solutions." *IEEE Communications Magazine* (2023).

[9] Masaracchia, Antonino, et al. "Digital twin for open ran: Towards intelligent and resilient 6g radio access networks." *IEEE Communications Magazine* (2023).

[10] Nguyen, Huan X., et al. "Digital Twin for O-RAN Toward 6G." *IEEE Communications Magazine* (2024).

[11] Jia et al. "Digital Twin Enabled Intelligent Network Orchestration for 6G: A Dual-Layered Approach." *IEEE INFOCOM*, (2023).

[12] Zhenyu et al. "Digital Twin-Accelerated Online Deep Reinforcement Learning for Admission Control in Sliced Communication Networks." *IEEE Transactions on Communications* (2024).

[13] Yaqoob, Mahnoor, et al. "ORBIT-DT: Bridging Simulation and Reality with Digital Twin Design for O-RAN in Beyond 5G Networks." *IEEE Wireless Communications and Networking Conference (WCNC)*, (2025).

[14] Breen, Joe, et al. "POWDER: Platform for open wireless data-driven experimental research." *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*. (2020).

[15] Melodia, Tommaso, et al. "Colosseum, the world's largest wireless network emulator." *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. (2021).

[16] Villa, Davide, et al. "Colosseum as a digital twin: Bridging real-world experimentation and wireless network emulation." *IEEE Transactions on Mobile Computing* (2024).