How Does Forecasting Affect the Convergence of DRL Techniques in O-RAN Slicing?

Ahmad M. Nagib*‡, Hatem Abou-zeid†, Hossam S. Hassanein*
*School of Computing, Queen's University, Canada, {ahmad, hossam}@cs.queensu.ca
†Department of Electrical and Software Engineering, University of Calgary, Canada, hatem.abouzeid@ucalgary.ca
‡Faculty of Computers and Artificial Intelligence, Cairo University, Egypt

GLOBECOM 2023 - 2023 IEEE Global Communications Conference | 979-8-3503-1090-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/GLOBECOM54140.2023.10437780

***Abstract*—The success of immersive applications such as virtual reality (VR) gaming and metaverse services depends on low latency and reliable connectivity. To provide seamless user experiences, the open radio access network (O-RAN) architecture and 6G networks are expected to play a crucial role. RAN slicing, a critical component of the O-RAN paradigm, enables network resources to be allocated based on the needs of immersive services, creating multiple virtual networks on a single physical infrastructure. In the O-RAN literature, deep reinforcement learning (DRL) algorithms are commonly used to optimize resource allocation. However, the practical adoption of DRL in live deployments has been sluggish. This is primarily due to the slow convergence and performance instabilities suffered by the DRL agents both upon initial deployment and when there are significant changes in network conditions. In this paper, we investigate the impact of time series forecasting of traffic demands on the convergence of the DRL-based slicing agents. For that, we conduct an exhaustive experiment that supports multiple services including real VR gaming traffic. We then propose a novel *forecasting-aided* DRL approach and its respective O-RAN practical deployment workflow to enhance DRL convergence. Our approach shows up to 22.8%, 86.3%, and 300% improvements in the average initial reward value, convergence rate, and number of converged scenarios respectively, enhancing the generalizability of the DRL agents compared with the implemented baselines. The results also indicate that our approach is robust against forecasting errors and that forecasting models do not have to be ideal.**

***Index Terms*—Deep Reinforcement Learning, Forecasting-aided DRL, Generalizable DRL, Accelerated DRL, O-RAN, RAN Slicing**

I. Introduction

The open radio access network (O-RAN) paradigm and 6G networks are expected to play a crucial role in making immersive applications a reality [1]. Services such as virtual reality (VR) gaming and metaverse applications require low latency and reliable connectivity to provide a seamless and immersive user experience. Radio access network (RAN) slicing, a critical component of the O-RAN architecture, enables the creation of multiple virtual RANs on a single physical infrastructure. This allows for improved user experiences ensuring that users have the necessary resources to engage in immersive activities [2].

O-RAN enables mobile network operators (MNOs) to deploy their own applications (xApps) to intelligently control the various network functionalities in near-real-time (near-RT) via standard open interfaces [3]. Deep reinforcement learning (DRL) algorithms are among the promising tools used to design O-RAN-compliant data-driven xApps [4]. Although O-RAN intelligent controllers offer promising advantages [3], the practical adoption of DRL algorithms in live deployments has been sluggish [4]. This is primarily because of the slow convergence and performance instability suffered by DRL agents [5]. This becomes apparent when agents are newly deployed in a live network or experience substantial changes in network conditions [6]. DRL convergence to the optimal RAN configuration needs to be quick and stable so that the users' quality of experience (QoE) is not affected. Nonetheless, due to the stochastic nature of 6G systems and the exploratory behavior of DRL agents, it may require thousands of time steps to regain stability. This holds significant importance in O-RAN deployments, as 6G networks can only afford a few exploration iterations while optimizing near-RT O-RAN functionalities [5].

Traffic demand forecasting [7], [8] is a powerful tool that can be utilized to enhance the performance of O-RAN slicing intelligent controllers. Since 6G networks are expected to support a wide range of immersive services, accurate forecasting can help MNOs proactively optimize the allocation of network resources to the admitted slices. This ensures that each slice has sufficient capacity to meet its service level agreements (SLAs). Combining traffic demand forecasting with a flexible tool such as DRL can enhance the convergence of DRL-based slicing and its generalizability. This enables the DRL agent to make informed slicing decisions based on the current network conditions while also considering the forecasted conditions.

In this paper, we investigate ways to leverage the power of time series forecasting for more robust and generalizable DRL-based O-RAN slicing. Moreover, we propose an O-RAN intelligent deployment workflow that incorporates a forecasting module to enhance convergence. The contribution of this research study can be summarized as follows:

- We propose a novel *forecasting-aided* algorithm to enhance the convergence and generalizability of O-RAN slicing DRL agents. A forecasting model is employed to predict the future contribution of slices to the overall traffic demand and a resource allocation configuration is suggested accordingly. This acts as a guide for the DRL agent when allocating resources to slices. Hence, the agent considers both its policy and the forecasted demand levels when taking allocation action given a certain situation.

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant RGPIN-2019-05667 and Grant RGPIN-2021-04050.

979-8-3503-1090-0/23/$31.00 ©2023 IEEE

- We propose an O-RAN deployment workflow that incorporates our *forecasting-aided* approach in the O-RAN architecture to guide the convergence of DRL-based xApps.
- We conduct an exhaustive performance study that supports multiple services including live VR gaming data to examine the impact of forecasting and its errors on the convergence of DRL-based O-RAN slicing. We then compare our approach against three implemented baselines. Our approach shows up to 22.8%, 86.3%, and 300% improvements in the average initial reward value, convergence rate, and number of converged scenarios respectively. The results also demonstrate our approach's robustness against forecast errors that follow a Gaussian distribution with a standard deviation up to 0.25, given that the range of the forecasted values is 1. The implementations of the proposed approach and baselines are publicly available[1] to facilitate research on trustworthy DRL in O-RAN.

To the best of our knowledge, this is the first study to 1) identify the need, and investigate the effect of time series forecasting on the convergence of DRL-based O-RAN slicing, especially for immersive 6G applications, and 2) propose an algorithm to improve DRL convergence by using a novel form of *forecasting-aided* DRL.

The paper's remaining sections are structured as follows: Section II presents a discussion on related work. Section III details the *forecasting-aided* DRL approach, O-RAN workflow, and the baselines proposed in this study. In Section IV, a description of the experimental setup, and an analysis of the results are provided. Finally, Section V concludes our work and presents potential future directions.

II. Related Work

The challenge of slow DRL convergence has been recently addressed using approaches such as transfer learning, meta-learning, structure awareness, and heuristics [5]. However, the focus of this paper is exploring the effect of time series forecasting on the convergence of DRL-based O-RAN slicing. Several research studies have explored the use of forecasting in resource allocation and network slicing. Nonetheless, most of these studies use statistical and machine learning (ML)-based forecasting approaches to optimize slicing directly [9]. For instance, the work in [7] extends long short-term memory (LSTM) neural networks to forecast the physical resource block (PRB) utilization. Consequently, the PRB allocation to slices is made based on such a forecast.

Only a few studies make use of forecasting models to enhance DRL. The authors of [10] propose a traffic offloading scheme that combines deep Q-network and traffic demand forecasting. A forecasting model uses the raw data collected from the DRL environment to predict traffic load statistics as a representation of the DRL state. Then, the DRL agent makes offloading decisions according to such a state. The results show that this approach outperforms tabular Q-learning. Similarly, the authors of [11] use a forecasting model to predict the mobile traffic volume. Hence, such a forecasted value is used as part of the state of a base station (BS) sleep control DRL agent.

Both studies, however, do not investigate the impact of forecasting on the convergence performance of the used DRL algorithms. Furthermore, the surveyed studies only utilize the forecasting model as part of a pre-processing step for state representation. Finally, the live network deployment, especially in the context of O-RAN, has not been addressed.

III. Forecasting-aided DRL-based O-RAN Slicing

We propose to utilize a forecasting module to guide the DRL agent when newly deployed in a live network or when the network conditions change significantly as described in Section III-C. This allows the agent to consider future traffic demand when allocating resources to the available slices with the goal of meeting slices' SLAs. For that, we also propose a deployment workflow to enhance the DRL convergence and generalizability in the context of the O-RAN architecture.

A. System Model

In this paper, we are concerned with the downlink case of the radio access part of O-RAN slicing. Radio resource allocation in slicing aims at assigning the limited available PRBs to the admitted slices while satisfying the slices' various requirements. The problem can be formulated as follows [6]:

A BS supports a range of services realized through a set of virtual slices, $\mathcal{S} = \{1, 2, \dots, S\}$. Such slices share the available bandwidth, B. Each BS has a set of user equipments (UEs), $\mathcal{U} = \{1, 2, \dots, U\}$, connected to it. A UE, denoted as u, is capable of requesting a single service type for downlink transmission at any given moment. Users associated with a particular slice, s, generate a set of requests, $\mathcal{R}_s = \{1, 2, \dots, R_s\}$. The overall demand, D_s, of these users can be denoted as follows:

$$D_s = \sum_{r_s \in R_s} d_{r_s}, \tag{1}$$

where d_{r_s} is the demand of a request, r_s, made by a user associated with slice s. Furthermore, the contribution of a slice, s, to the total BS's traffic demand at a slicing step, t, is:

$$\kappa_s(t) = \frac{D_s(t)}{\sum_{i=1}^{\|S\|} D_i(t)} \tag{2}$$

PRB allocation among the available slices, S, can be represented by the vector, $a \in \mathrm{I\!R}^{\mathrm{S}}$. At the start of a slicing window, a RAN slicing controller selects a slicing PRB allocation configuration, a, out of the A feasible configurations, where $\mathcal{A} = \{1, 2, \dots, A\}$. Consequently, the system performance, represented in terms of the latency of the admitted slices within the context of this paper, is impacted. This is primarily influenced by a queue maintained at the BS.

B. Reinforcement Learning Mapping

For the traditional DRL approach, we follow the mapping in Table I. The system state is defined as the slices' contribution to the total BS's traffic within the past slicing window, that is,

$$\kappa = (\kappa_1(t-1), ..., \kappa_s(t-1), ..., \kappa_S(t-1)) \tag{3}$$

[1] Available at http://www.github.com/ahmadnagib/forecasting-aided-DRL

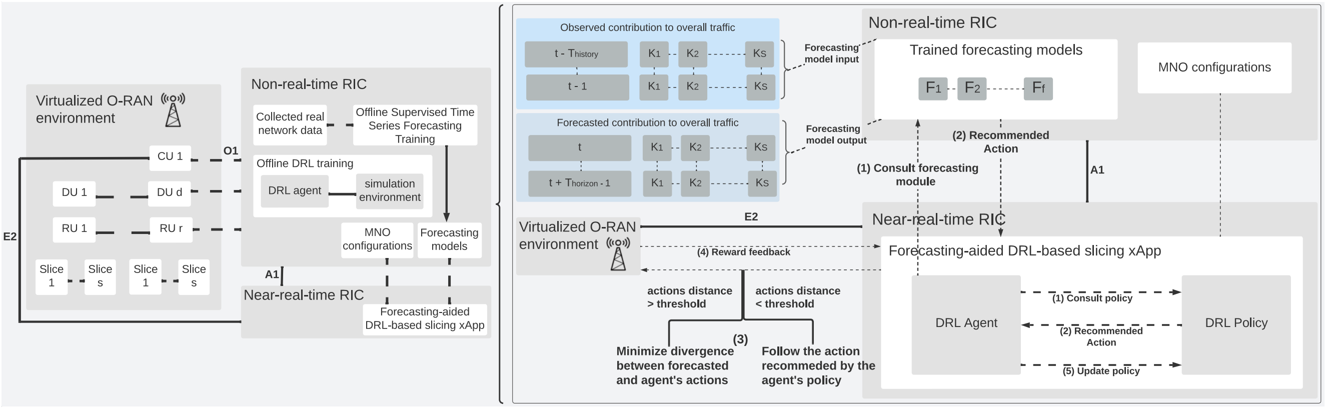

Virtualized O-RAN environment
CU 1
DU 1
DU d
RU 1
RU r
Slice 1
Slice s
Slice 1
Slice s
E2
O1
Non-real-time RIC
Collected real network data
Offline Supervised Time Series Forecasting Training
Offline DRL training
DRL agent
simulation environment
MNO configurations
Forecasting models
A1
Near-real-time RIC
Forecasting-aided DRL-based slicing xApp
Observed contribution to overall traffic
t - Thistory
t - 1
K1
K2
Ks
Forecasted contribution to overall traffic
t
t + Thorizon - 1
Forecasting model input
Forecasting model output
Non-real-time RIC
Trained forecasting models
F1
F2
Ff
MNO configurations
(2) Recommended Action
(1) Consult forecasting module
A1
E2
Virtualized O-RAN environment
Near-real-time RIC
Forecasting-aided DRL-based slicing xApp
(4) Reward feedback
actions distance > threshold
actions distance < threshold
(3)
Minimize divergence between forecasted and agent's actions
Follow the action recommeded by the agent's policy
DRL Agent
DRL Policy
(1) Consult policy
(2) Recommended Action
(5) Update policy

Fig. 1: Proposed forecasting-aided DRL-based O-RAN slicing system.

The DRL agent observes such a state and takes action accordingly at the start of each slicing step to decide the PRB allocation for each slice, that is,

$$a = (b_1, ..., b_s, ..., b_S), \text{ subject to } b_1 + ... + b_S = B \quad (4)$$

The reward function is used to optimize the allocation process. We use sigmoid function-based rewards similar to the one proposed in [6]. This enables controlling the effect of approaching the performance threshold defined by each slice's SLAs. In this study, we prioritize immersive services, and hence, their latency requirements. Therefore, the reward function is a weighted sum of an inverse form of latency and can be defined as follows:

$$R = \sum_{s=1}^{\|S\|} w_s \ * \ \frac{1}{1 + e^{\ c1_s \ * \ (\ l_s \ - \ c2_s \ )}} \quad (5)$$

where l_s is the average latency underwent by slice s's UEs during the preceding slicing window, at $t-1$. The weight, w_s, defines the priority of satisfying the delay requirement of slice s. $c1$ configures the slope of the sigmoid function, and hence, controls the starting point for penalizing the agent's actions. Moreover, $c2$ defines the inflection point that reflects the latency performance threshold for each slice based on its SLAs.

C. Proposed Forecasting-aided O-RAN Architecture

We propose a deployment workflow that incorporates a forecasting module as part of the O-RAN architecture to guide DRL-based xApps. This is primarily needed upon the initial xApp deployment and when there are significant changes in network conditions. This module includes forecasting models that predict relevant network conditions in the future such as the traffic demand of various slices. Consequently, the module interacts with the DRL agents via O-RAN's A1 interface to guide and enhance their convergence performance as described in the next subsections. Fig. 1 shows the overall system architecture (left) and the interaction steps between the DRL agent, the forecasting module, and the virtualized O-RAN environment in the proposed approach (right).

As seen in the figure, the traffic demand forecasting is based on observations made during a time window, $T_{\text{history}}, [t - T_{\text{history}}, t-1]$. The contribution of slice s to the overall traffic demand in such a period can be described by the following vector:

$$\kappa^{(\mathbf{s})}(t) = \Big(\kappa^{(\mathbf{s})}(t - T_{\text{history}}), \kappa^{(\mathbf{s})}(t - (T_{\text{history}} - 1)), \\ \cdots, \kappa^{(\mathbf{s})}(t-1)\Big) \quad (6)$$

Based on such an observed history of traffic demand, the forecasting model, $\mathbb{F}$, provides the predicted traffic demand for the slicing window that is about to begin or a longer time window, T_{horizon}, for the time period $[t, \ t + T_{\text{horizon}} - 1]$. The demand in this period can be denoted as follows:

$$\hat{\kappa}^{(\mathbf{s})} = \Big(\hat{\kappa}^{(s)}(t), \hat{\kappa}^{(s)}(t+1), \ \cdots, \hat{\kappa}^{(s)}(t + T_{\text{horizon}} - 1)\Big) \quad (7)$$

Accordingly, the forecasting model depicted in Fig. 1 predicts the slices' future contribution to the overall BS traffic demand as follows:

$$\begin{aligned} &\mathbb{F} : \mathbb{R}^{T_{\text{history}} * S} \to \mathbb{R}^{T_{\text{horizon}} * S} \\ &\kappa[t - T_{\text{history}}, \ t-1] \to \hat{\kappa}[t, \ t + T_{\text{horizon}} - 1] \end{aligned} \quad (8)$$

where $\hat{\kappa}$ denotes the slices' forecasted contribution to the overall traffic demand. For the purpose of this paper, we imitate the behavior of forecasting models with errors that follow Gaussian distributions as detailed in Section IV. Hence, forecasting model training is not addressed.

D. Proposed Forecasting-aided DRL-based O-RAN Slicing

As detailed in Fig. 1, a forecasting model is incorporated to predict future traffic demand. Accordingly, the forecasting module suggests a PRB allocation action purely based on its forecasted contribution to the total demand, $\hat{\kappa}$, as defined in Equation 7. The DRL agent continuously monitors such a suggested guiding action via O-RAN's A1 interface. The agent follows its current policy unless the action is significantly different from that suggested by the forecasting module. The forecasting module overwrites the agent's policy in such a case to prevent potentially damaging actions. A distilled action that minimizes the divergence between the agent's policy and the forecasting module's action is taken. The difference between two actions is measured in terms of the Euclidean distance between the action vectors as follows:

TABLE I: Experiment Setup: RAN Slicing DRL Design

State	Slices' contribution to the overall BS's traffic within a specific time window as defined in (2)
Action	PRBs allocated to each slice as defined in (4)
Reward function	A weighted sum of a sigmoid function of the average latency experienced in a slicing window by the various slices as defined in (5)
Reward function weights	VoNR: 0.1, VR gaming: 0.7, Video: 0.2
DRL algorithm	Proximal Policy Optimization (PPO)
Learning steps per run	10,000
Exploration rate	0.5
Exploration decay rate	0.5 (every 200 steps)
Action distance threshold	7%
Learning rate	0.01
Batch size	4

Algorithm 1 Proposed Forecasting-aided DRL Approach

Input: trained forecasting model, $\mathbb{F}$, traffic demand historical observations of size T_{history}, forecast horizon, T_{horizon}, current state, κ, current DRL policy, π, set of possible actions, A, actions distance threshold, $\gamma_{\text{threshold}}$

Output: distilled action, $a_{\text{distilled}}$ as defined in Section III-D

```
while $t < T$ do:
    Forecast $\hat{\kappa}$ for the next $T_{\text{horizon}}$ time steps, using $\mathbb{F}$
    Generate an action, $a_{\text{forecast}}$, purely based on $\hat{\kappa}$
    Consult $\pi$ given $\kappa$, and get the recommended action, $a_\pi$
    if $\gamma(a_\pi, a_{\text{forecast}}) > \gamma_{\text{threshold}}$ do:
        Find the midpoint between the vectors, $a_\pi$ and $a_{\text{forecast}}$
        Select an action, $a_{\text{distilled}}$, closest to the midpoint to
            minimize the divergence between $a_\pi$ and $a_{\text{forecast}}$
        Take the distilled action, $a_{\text{distilled}}$, to allocate PRBs for
            each admitted slice
        Update the value function, $V$, based on the received
            reward, $R$
    end if
end while
```

$$\gamma\left(a_\pi, a_{\text{forecast}}\right) = \sqrt{\sum_{s=1}^{S}\left(a_{\pi s} - a_{\text{forecast}s}\right)^2} \tag{9}$$

where a_π and a_{forecast} are vectors of actions recommended by the DRL agent's policy and the forecasting module respectively. The agent does not follow the exact action recommended by the forecasting module as it may not explicitly consider the slices' SLAs, i.e., latency in our case. A distilled action that represents the midpoint between the two actions' vectors is taken instead. This prevents the agent from taking actions that contradict the forecasted demand. This additionally accommodates potential forecast errors. The distilled action is integrated into the DRL agent's learning process to speed up its convergence to the optimal slicing configuration. Algorithm 1 defines how a DRL-based slicing xApp seeks guidance from the forecasting module and updates its policy accordingly while it is active (i.e., $t < T$).

E. Baselines

a) Forecasting-based DRL state representation: We first implement a forecasting-based approach that embeds the forecasted traffic demand as part of the DRL state. This approach utilizes the forecasting module in a preprocessing step similar to [10] and [11]. We integrate this step in the O-RAN flow proposed in Section III-C and use the forecasting model's output as an extra input feature to the DRL-based agent via O-RAN's A1 standardized interface. Hence, the forecasted traffic demand is embedded as part of the DRL state representation in addition to the traffic demand observed in the preceding slicing window as defined in (3), where $\kappa \in \mathbb{R}^{(\text{T}_{\text{horizon}}+1)*\text{S}}$. The first baseline approach's steps are described in Algorithm 2.

Algorithm 2 Forecasting-based DRL State Representation

Input: trained forecasting model, $\mathbb{F}$, traffic demand historical observations of size T_{history}, forecast horizon, T_{horizon}, current DRL policy, π, set of possible actions, A

Output: action, a as defined in (4)

```
while $t < T$ do:
    Forecast $\hat{\kappa}$ for the next $T_{\text{horizon}}$ time steps, using $\mathbb{F}$
    Embed the future traffic demand in the system state,
        i.e., $\kappa \in \mathbb{R}^{(\text{T}_{\text{horizon}}+1)*\text{S}}$
    Consult $\pi$ given $\kappa$, and get the recommended action, $a_\pi$
    Update the value function, $V$, based on the received
        reward, $R$
end while
```

b) Non-forecasting-aided DRL: We also implement a traditional DRL approach that follows the same DRL mapping defined in Section III-B. This approach is not guided by the proposed forecasting module.

c) Non-DRL forecasting approach: Finally, we implement an approach that purely relies on forecasting to allocate resources. PRBs are allocated to each slice solely based on such a slice's forecasted contribution to the total demand, $\hat{\kappa}$.

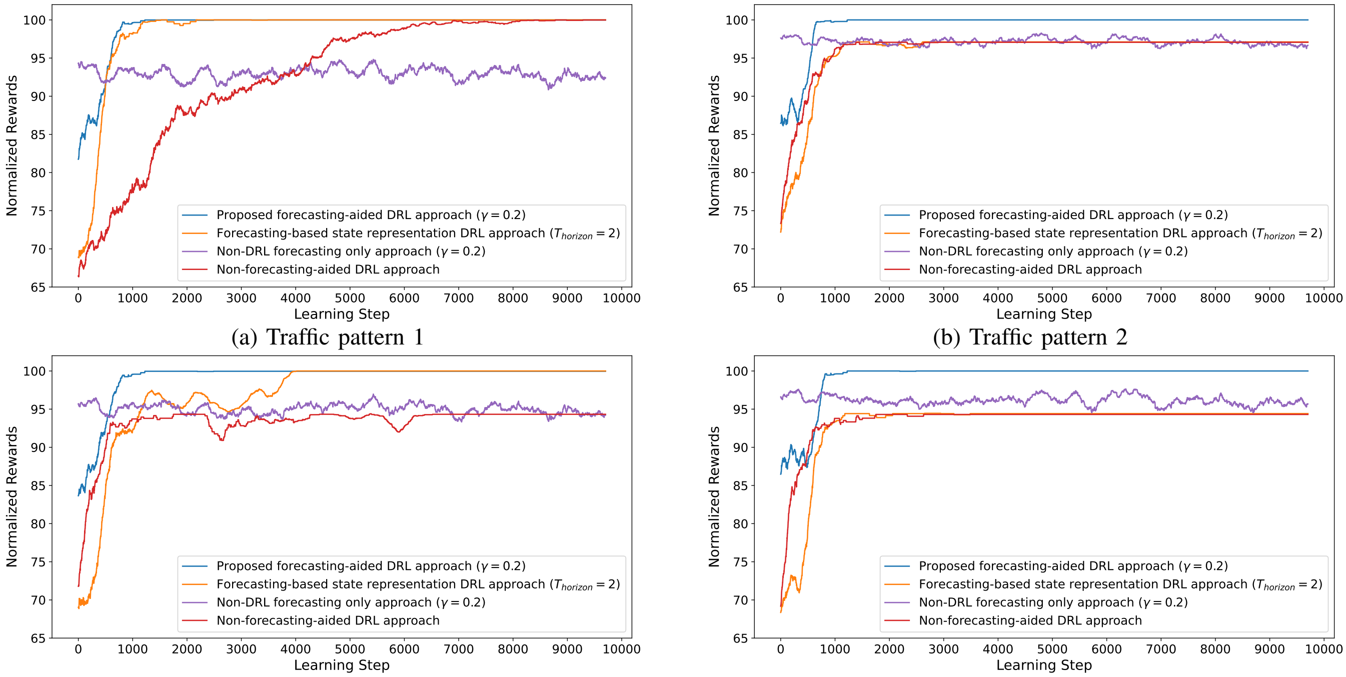
IV. Experiment Setup and Numerical Results

A. Experiment Setup

We investigate a deployment scenario using the O-RAN workflow proposed in Section III-C. Hence, we restrict DRL agents' exploration as reflected by the exploration rate and its decay specified in Table I. We conduct an exhaustive experiment that follows the mapping defined in Section III-B and implements the proposed approach. The simulation is designed to reflect extreme situations in which the available PRBs are configured to be less than the actual demand. We then compare the convergence performance of the proposed approach against

TABLE II: Experiment Setup: Simulation Parameters Settings

	Video	**VoNR**	**VR gaming**
Scheduling algorithm	Round-robin per 1 ms slot		
Slicing window size	PRB allocation among slices every 100 scheduling time slots		
Forecasting error	Gaussian distribution, mean = 0, standard deviation = 0, 0.1, 0.2, 0.23, 025, 0.3, 0.4		
Forecasting horizons	$T_{\text{horizon}} = \{1, 2, \ldots, 10\}$		
Packet interarrival time	Truncated Pareto (mean = 6 ms, max = 12.5 ms)	Uniform (min = 0 ms, max = 160 ms)	Real VR gaming dataset [12]
Packet size	Truncated Pareto (mean = 100 B, max = 250 B)	Constant (40 B)	Real VR gaming dataset [12]
Number of users	Poisson (max = 43, mean = 20)	Poisson (max = 104, mean = 70)	Poisson (max = 7, mean = 1)



(a) Traffic pattern 1

(b) Traffic pattern 2

(c) Traffic pattern 3

(d) Traffic pattern 4

Fig. 2: Convergence performance of the proposed forecasting-aided approach under 4 different traffic patterns.

the baselines defined in Section III-E. Moreover, we test the agents that follow Algorithm 1, and the non-DRL forecasting approach against various forecasting errors as defined in Table II. This allowed us to examine the effect of forecasting errors on the performance of the two approaches.

We use live VR gaming data from [12] as an example of realistic patterns of immersive services in 6G networks. We specifically incorporate 4 different trace files reflecting traffic patterns from multiple games and distinct configurations per game. Moreover, we combine such patterns with video and voice over new radio (VoNR) traffic requests to reflect 3 different slice types in our experiment. Such requests are generated following the models defined in Table II as in [6]. In such models, VoNR users produce requests of small and static sizes, while VR gaming users generate the largest requests. Besides, video users experience more frequent requests than the other two services. Different constant values of $c1$ and $c2$ parameters are utilized for the slices in the reward function based on their respective latency requirements.

B. Numerical Results

a) Convergence Performance: Despite the aforementioned restricted exploration settings, the proposed approach converges to the optimal allocation configurations given the scenarios shown in Fig. 2. On the other hand, due to such restrictions, the non-forecasting-aided DRL approach fails to converge in almost all the scenarios. Moreover, the number of steps needed by our approach to converge to the optimal configuration is significantly less than the non-forecasting-aided DRL approach when both converge as in Fig. 2a. This is primarily due to guidance from the forecasting module that prevents the agent from exploring potentially damaging actions. Algorithm 2 only includes the forecasted demand in its state so it does not directly overwrite potentially damaging actions or accommodate forecasting errors. Hence, its performance is also inferior to Algorithm 1.

The figure also shows a remarkable improvement in the initial reward values of our approach compared to the other two DRL-based baselines. In our approach, a distilled action is only triggered when there is a big gap between the agent's action and that recommended by the forecasting model. Hence, only potentially damaging actions are replaced by an action closer to the forecasted conditions allowing for a safer exploration. Eventually, based on the configured action distance threshold of our approach, the forecasting model was only consulted 8.9% of the time on average. Consequently, the DRL agent recovers quickly to a near-optimal slicing configuration and the received

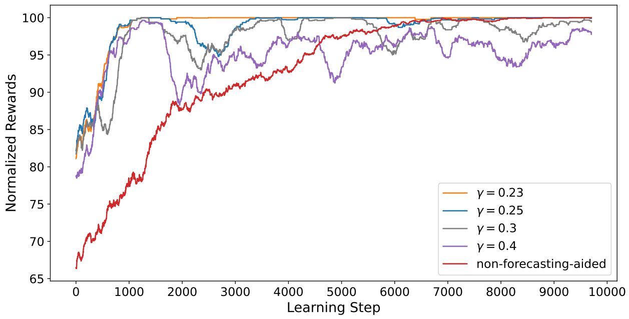


Fig. 3: Convergence performance of the proposed approach under different forecasting error models (traffic pattern 1).

reward becomes relatively higher.

Relying on forecasting solely never leads to convergence given imperfect predictors as Fig. 2 suggests. This confirms our hypothesis in Section III-D. The non-DRL forecasting approach does not explicitly consider the various slices' SLA fulfillment. Hence, our approach outperforms the non-DRL forecasting approach as it aims at satisfying latency requirements reflected in the reward function.

b) Forecast Error Effect: We also examine the effect of the forecast error on the convergence performance. The proposed approach is robust against forecast errors that follow a Gaussian distribution with a standard deviation up to 0.25 in the case of traffic pattern 1 as shown in Fig. 3. This is a significant error given that the range of the forecasted values is 1. Such robustness is attributed to accommodating potential forecasting errors in Algorithm 1 through divergence minimization instead of solely relying on the forecasted action. Nevertheless, when the standard deviation is higher than 0.25, the exploration becomes relatively unstable. DRL agents fail to converge in such scenarios. However, the overall reward is still kept in a relatively good range. Furthermore, the proposed algorithm still has relatively high initial reward values compared with the traditional DRL approach as seen in Fig. 3.

This observation is confirmed by the statistics compiled in Fig 4. The proposed approach noticeably outperforms all the other DRL-based baselines and maintains the highest initial reward value on average, even when the error is high. Our approach also has the fastest convergence rate. Furthermore, 100% of the conducted scenarios converge to the optimal resource allocation configuration given that the forecasting error's standard deviation is 0.25 or lower. Since forecasting models are imperfect, especially with new immersive services, this gives insights into the accepted error ranges. It also shows that forecasting models used in our approach do not have to be ideal. Finally, the forecasting-based state representation approach shows an inferior performance to the proposed approach in almost all the cases even when using perfect predictors.

V. Conclusion and Future Work

In this paper, we conduct an exhaustive experiment to study the effect of forecasting on the convergence performance of DRL-based O-RAN slicing. We propose a *forecasting-aided* DRL algorithm and an O-RAN deployment workflow that prove

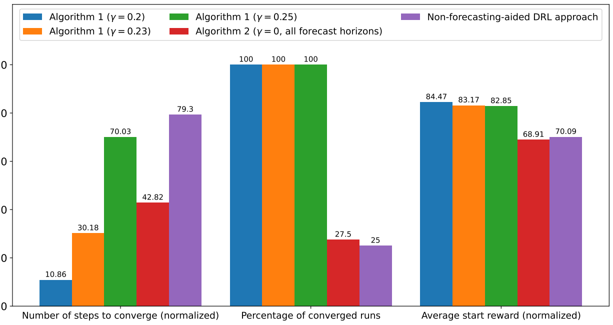


Fig. 4: Convergence performance averaged over multiple runs (the higher the better except for number of steps to converge).

to remarkably enhance the convergence performance and to be robust against forecasting errors. We plan to investigate the possibility of building forecasting models that achieve the observed acceptable error ranges using the VR gaming data. We will also explore combining such models with other approaches such as constrained DRL and transfer learning as a promising step toward trustworthy DRL in O-RAN slicing.

References

[1] C.-X. Wang, X. You, X. Gao, X. Zhu, Z. Li, C. Zhang, H. Wang, Y. Huang, Y. Chen, H. Haas, J. S. Thompson, E. G. Larsson, M. D. Renzo, W. Tong, P. Zhu, X. Shen, H. V. Poor, and L. Hanzo, "On the road to 6g: Visions, requirements, key technologies and testbeds," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2023.

[2] S. Karunarathna, S. Wijethilaka, P. Ranaweera, K. T. Hemachandra, T. Samarasinghe, and M. Liyanage, "The role of network slicing and edge computing in the metaverse realization," *IEEE Access*, vol. 11, pp. 25 502–25 530, 2023.

[3] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in o-ran for data-driven nextg cellular networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, 2021.

[4] P. H. Masur, J. H. Reed, and N. K. Tripathi, "Artificial intelligence in open-radio access network," *IEEE Aerospace and Electronic Systems Magazine*, vol. 37, no. 9, pp. 6–15, 2022.

[5] A. M. Nagib, H. Abou-zeid, and H. S. Hassanein, "Toward safe and accelerated deep reinforcement learning for next-generation wireless networks," *IEEE Network*, pp. 1–8, 2022.

[6] A. M. Nagib, H. Abou-Zeid, and H. S. Hassanein, "Accelerating reinforcement learning via predictive policy transfer in 6g ran slicing," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1170–1183, 2023.

[7] C. Gutterman, E. Grinshpun, S. Sharma, and G. Zussman, "Ran resource usage prediction for a 5g slice broker," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. Mobihoc '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 231–240.

[8] A. M. Nagib, H. Abou-Zeid, H. S. Hassanein, A. Bin Sediq, and G. Boudreau, "Deep learning-based forecasting of cellular network utilization at millisecond resolutions," in *IEEE International Conference on Communications (ICC)*, 2021, pp. 1–6.

[9] C. Ssengonzi, O. P. Kogeda, and T. O. Olwal, "A survey of deep reinforcement learning application in 5g and beyond network slicing and virtualization," *Array*, vol. 14, p. 100142, 2022.

[10] C.-W. Huang and P.-C. Chen, "Joint demand forecasting and dqn-based control for energy-aware mobile traffic offloading," *IEEE Access*, vol. 8, pp. 66 588–66 597, 2020.

[11] Q. Wu, X. Chen, Z. Zhou, L. Chen, and J. Zhang, "Deep reinforcement learning with spatio-temporal traffic forecasting for data-driven base station sleep control," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 935–948, 2021.

[12] S. Zhao, H. Abou-zeid, R. Atawia, Y. S. K. Manjunath, A. B. Sediq, and X.-P. Zhang, "Virtual reality gaming on the cloud: A reality check," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 1–6.