POLITECNICO DI TORINO
Repository ISTITUZIONALE

CROWN: Cross-attention reinforcement learning for O-RAN wireless networks

Original
CROWN: Cross-attention reinforcement learning for O-RAN wireless networks / Monaco, D., Sacco, A., Marchetto, G.. - In: COMPUTER NETWORKS. - ISSN 1389-1286. - 282:(2026). [10.1016/j.comnet.2026.112276]

Availability:
This version is available at: 11583/3013691 since: 2026-07-29T08:33:59Z

Publisher:
Elsevier

Published
DOI:10.1016/j.comnet.2026.112276

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

20 November 2026

Computer Networks 282 (2026) 112276

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

CROWN: Cross-attention reinforcement learning for O-RAN wireless networks

Doriana Monaco *, Alessio Sacco, Guido Marchetto

Dept. of Control and Computer Engineering, Politecnico di Torino, Turin, Italy

ARTICLE INFO

Keywords:
O-RAN
Slice placement
Reinforcement learning
Energy consumption

ABSTRACT

Fifth-generation (5G) cellular networks promise unprecedented connectivity through ultra-low latency and high-speed mobile broadband, driving the need for intelligent slice placement strategies in Open Radio Access Network (O-RAN) architectures. O-RAN promotes openness and vendor interoperability, but existing Machine Learning (ML)-based embedding solutions often prioritize performance metrics such as delay and availability while neglecting energy efficiency. To address this gap, we propose Crown, a Reinforcement Learning (RL)-based service placement framework that jointly optimizes Service Level Agreement (SLA) compliance and power consumption. Crown extends a traditional Deep Q-Network (DQN) by integrating cross-attention layers to model complex dependencies between virtualized O-RAN functions and heterogeneous physical servers, enabling more informed placement decisions. We evaluate Crown in a simulated O-RAN environment and compare it against state-of-the-art RL approaches and heuristic baselines. Results demonstrate that Crown reduces power consumption by 57% compared to a fixed deployment and by 15% relative to a DQN without cross-attention, while meeting stringent latency and bandwidth requirements through action masking and achieving high slice admission rates and low deployment cost via its cost-aware reward design. Furthermore, we measure the inference time, showing that the attention-enhanced RL design remains practical for large-scale deployments.

1. Introduction

Fifth-generation (5G) cellular networks are designed to support a broader range of use cases, including ultra-low latency services, high-speed mobile broadband, and dense device connectivity. Meeting these demands requires a shift from the closed, black-box design of 4G to open, flexible architectures [1,2]. The Radio Access Network (RAN), a key component of cellular networks, has evolved over time from distributed RAN (D-RAN) to centralized RAN (C-RAN) and then to virtualized RAN (vRAN) to meet the rising demands of a growing number of users [3]. However, vRAN has struggled to fully meet current 5G requirements [3–5]. The Open Radio Access Network (Open RAN) initiative is proposed as an evolution of vRAN, serving as the next generation of RAN technology. It's a central part of the 5G transformation, addressing the limitations of previous, closed RAN architectures. By opening up interfaces, Open RAN encourages competition among vendors, which helps to reduce costs and foster innovation [6]. This open architecture also facilitates the integration of intelligent components, leading to improvements like energy savings and dynamic power management [7,8]. O-RAN disaggregates traditional RAN components into the Open-Central Unit (O-CU), Open-Distributed Unit (O-DU), and Open-Radio Unit (O-RU) to enable interoperability and innovation [9]. Additionally, the advent of 5G and Beyond 5G (B5G) networks has shifted Mobile Network Operators (MNOs) towards service-oriented models supported by network slicing. Network slicing creates multiple virtual networks over shared infrastructure, offering tailored service experiences across domains like Enhanced Mobile Broadband (eMBB), Ultra Reliable Low Latency Communications (URLLC), and Massive Machine-Type Communication (mMTC). Enabled by Software-Defined Networking (SDN) and Network Function Virtualization (NFV), this architecture improves resource management, operational efficiency, and scalability. Furthermore, emerging technologies like Deep Reinforcement Learning (DRL) optimize service provisioning and tackle complex resource allocation challenges in B5G and future 6G networks, which will integrate terrestrial, non-terrestrial, and cloud-based networks into a unified, intelligent ecosystem [10–13].

Notably, the attention mechanism is transforming the Machine Learning (ML) field by enabling agents to focus on the most relevant features within high-dimensional environments. In networking, attention is increasingly applied to various problems, such as task

* Corresponding author.
E-mail addresses: doriana.monaco@polito.it (D. Monaco), alessio_sacco@polito.it (A. Sacco), guido.marchetto@polito.it (G. Marchetto).

https://doi.org/10.1016/j.comnet.2026.112276
Received 19 September 2025; Received in revised form 4 February 2026; Accepted 29 March 2026
Available online 2 April 2026
1389-1286/© 2026 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

scheduling [14], VNF flow graph embedding [15,16], and the VNF placement and routing problem [17]. Specifically, cross-attention can learn how two objects are correlated.

Building on these advances, several ML-based approaches have been proposed for slice embedding in O-RAN. Most focus on performance metrics such as maximizing availability or minimizing delay, often overlooking energy consumption as a core optimization objective [18–21]. While energy-aware approaches do exist, they generally focus on DU activation patterns or application-level scaling, without integrating energy considerations into the placement decision [22–25]. Additionally, none of these studies have explored cross-attention for slice placement in the context of O-RAN networks, leaving a gap in the joint optimization of performance and sustainability.

To overcome these limitations, we propose Crown, a flexible CU/DU placement strategy that embeds both Service Level Agreement (SLA) satisfaction and energy efficiency directly into the placement layer. Unlike standard Deep-Q Networks, Crown employs a task-specific cross-attention mechanism that jointly embeds FU-level SLA descriptors with heterogeneous server and link features, enabling the agent to focus on the most relevant infrastructure components for each requested FU. We further provide explainability evidence showing that Crown's attention heads specialize on distinct node- and link-level attributes, and that SHAP analysis confirms these are the features actually driving the placement decisions-whereas a plain DQN fails to identify and balance such relevant inputs. In our evaluation, Crown simultaneously optimizes SLA compliance, energy consumption, and deployment cost by combining a cost- and energy-aware reward function with action masking for GOPS, latency, and bandwidth constraints. Experiments show that Crown reduces power consumption by 57% compared with a fixed-deployment baseline and by 15% relative to a standard DQN without cross-attention, while preserving high slice admission rates and low deployment cost. Moreover, inference-time measurements confirm that the cross-attention module introduces negligible computational overhead, making Crown suitable for large-scale O-RAN deployments.

The manuscript is structured in the following way: Section 2 reviews current state-of-the-art work concerning the placement of Functional Units (FUs) and methods to reduce power consumption in RANs. The adopted O-RAN deployment is introduced in Section 3, while we formally define our problem in Section 4. Section 5 contains our DRL formulation, while Section 6 shows the experimental settings and results. We draw our conclusion in Section 7.

2. Related work

In the following section, we review the state-of-the-art on O-RAN functionalities placement and energy-aware solutions in RAN scenarios, highlighting the different approaches that drive the solutions.

2.1. O-RAN functionalities placement

Many research works investigate the placement problem in the RAN context. The main differences lie in the processing chain considered, the optimization objective, and the 3GPP deployment scenario analyzed.

Some works propose Linear Programming (LP) solutions. For example, [26] presents a Binary Integer Programming (BIP) model to optimize the placement of requested FUs to maximize their availability and minimize downtime. In recent years, however, the focus has shifted toward machine learning-based solutions. In [18], the authors propose to forecast the traffic requests through an LSTM, issue scaling decisions for each FU type, and identify the servers that satisfy computing and latency constraints to host the new FU instances. The processing chain comprises O-DU, O-CU, and nrRIC, and the scaling decision is taken through three different MLPs, one for each FU type. Such RL neural networks were trained using a synthesized dataset generated through an Integer Linear Programming model. The reward punishes any requirement violation and awards if the minimum delay is reached.

Another work [19], considers CU and DU placement in a hybrid cloud architecture: several edge servers are placed near the RUs and serve DU functions. The CU functions are placed in the regional cloud, which is further connected to the core network. The placement problem is solved through a Reinforcement Learning (RL) model, which is trained with Proximal Policy Optimization (PPO) with invalid action masking. The state space includes the physical network characteristics and ongoing slice requests. The agent can choose between accepting the slice and placing the corresponding FUs, or rejecting it if it would reduce the overall profit. The reward function is defined as the difference between the revenue generated from the slice and the server operational cost, with a reward of zero if the slice is rejected.

[20] proposes a DQN-based approach to associate the correct RU to the user and also find the best placement scheme of CU and DU to minimize the end-to-end delay of UEs while minimizing the deployment cost of DU and CU FUs. The input state includes the user position and a flag for the satisfaction of the delay constraints. At the same time, the action associates the user with a specific RU and then indicates the servers to place O-CU and O-DU. The reward function promotes constraint satisfaction and penalizes higher delay and processing cost.

In contrast to the previous studies, the authors of [21] solve the placement problem considering flexible deployment scenarios. While the O-DU is fixed at the edge cloud, the O-CU can be placed at the edge or in the regional cloud. The objective is to maximize the number of admitted UEs while minimizing the deployment cost. To solve this NP-HARD problem, they develop an RNN-based model that receives the features of each user request as input and is trained to minimize the loss with respect to the ILP solution.

Similarly to them, we explore a flexible CU/DU placement, but, differently, we propose a placement strategy that aims to reduce power consumption while satisfying SLA requirements. We employ a Deep Q-Network (DQN) architecture augmented with cross-attention and feed-forward layers, enabling the model to learn efficient placement decisions based on dynamic network conditions and application requirements. Our goal is to simultaneously satisfy stringent delay and bandwidth constraints, maximize acceptance rates, and minimize both server energy consumption and infrastructure deployment costs.

2.2. Energy modeling

Recent research has increasingly focused on energy-efficient mechanisms for O-RAN. However, most solutions tackle energy optimization outside the placement layer, targeting specific components of the RAN rather than the joint placement of CU/DU functions. For example, [22] minimizes DU power consumption through RB-level allocation and DU selection using MILP. Similarly, Mollahasani et al. [23] propose an actor-critic method that dynamically relocates RB scheduling functions and activates or deactivates DUs based on traffic conditions. These strategies effectively reduce DU-related power usage but do not address CU/DU placement.

Other approaches, such as ScalO-RAN [24], focus on dynamically scaling xApps, rApps, and dApps across distributed compute resources. While energy-aware, these methods optimize application-level orchestration, not the placement of functional units along the O-RAN processing chain. Likewise, works on RU sleeping or UE-RU association [27,28] reduce radio-side power consumption but operate independently of CU/DU placement decisions.

Only a few studies consider placement from an energy perspective. Singh et al. [25] investigate the energy-efficient orchestration of vRAN by assigning DUs to Edge Processing Modules (EPMs) and CUs to Central Processing Modules (CPMs). However, this and similar works [29,30] assume a fixed deployment model in which the roles of edge and regional servers are predetermined. As a result, they cannot exploit the benefits of flexible CU/DU placement across hybrid cloud infrastructures, nor do they incorporate server-level energy consumption into the placement decision.

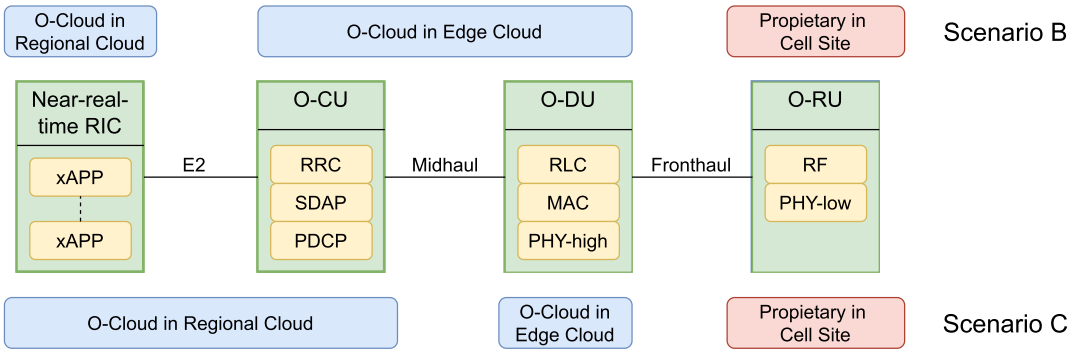


Fig. 1. Crown concurrently considers two possible deployments for O-RAN FUs: Scenario B and C.

To the best of our knowledge, no prior work jointly integrates flexible CU/DU placement, SLA satisfaction, and server-level energy awareness within a unified RL framework. Our solution, Crown, directly addresses this gap.

3. O-RAN overview and system model

The main contribution of the O-RAN [31] standard is the disaggregation of hardware and software. Traditionally, RAN components were proprietary, but thanks to virtualization, O-RAN functions can run on general-purpose hardware. O-RAN defines three components: O-RU, O-DU, and O-CU. The disaggregation creates many possible functional splits defined by 3GPP [32], which determine the complexity of the operations at O-RU and O-DU. O-RAN Alliance selected the split 7.2x, as it finds a trade-off between the complexity of operations at the radio unit and the load on the fronthaul connection [9]. Specifically, the O-RU handles the time-domain and low-PHY operations such as FFT, cyclic-prefix processing, and beamforming. The O-DU is responsible for the high-PHY procedures–including frequency-domain processing steps such as channel coding, rate matching, modulation–along with the MAC and RLC layers. The O-CU hosts the upper-layer protocols, specifically RRC, SDAP, and PDCP. The architecture also includes RAN Intelligent Controllers (RICs) that manage network operations using AI and ML algorithms. These RICs operate on two time scales: near-real-time ($10ms$ to $1s$) and non-real-time ($1s$ or more), enabling automated, data-driven network optimization. O-RAN components can be hosted on a hybrid cloud computing platform called O-Cloud [33], providing virtualization infrastructure, resource management, and hardware-software decoupling. This cloud-based environment supports automated deployment and efficient resource sharing across different tenants. Multiple scenarios are envisioned for the deployment in the cloud. In particular, we focus on a flexible deployment of scenarios between B and C, shown in Fig. 1. In the former, O-CU and O-DU are both deployed in the edge cloud. In the latter, O-CU is placed in the regional cloud. Such flexibility is required, as the latency can be reduced by deploying FUs closer to cell sites and, therefore, end-users. On the other hand, regional clouds can be leveraged for their computational capacities, lower costs, and power consumption.

4. Problem formulation

In this section, we formalize the optimization problem of O-RAN FU placement in our scenario. We consider a hybrid cloud architecture with a set of edge cloud servers S_e and a set of regional cloud servers S_r; the global set of servers is $S = S_e \cup S_r$. O-DU functions can be placed only in the edge cloud, while O-CU functions can be flexibly placed in edge or regional servers. Each server is characterized by its capacity G_s in terms of Giga Operations Per Second (GOPS) and cost per GOPS C_s. The servers are fully connected, and each link is characterized by a distance $d_{ss'}$, bandwidth $b_{ss'}$, and propagation delay $l_{ss'}$.

The propagation delay is modeled in the following way:

$$l_{ss'} = \frac{d_{ss'} \cdot n}{c}, \tag{1}$$

where $n = 1.5$ is the refractive index of the fiber optic cable and c is the speed of light.

I slice requests are considered, belonging to URLLC, eMBB or mMTC. Each slice is characterized by three hard requirements: GOPS r_i^{FU}, bandwidth b_i, and latency l_i. These requirements constitute the slice's SLAs and must be strictly satisfied for the slice to be accepted and placed.

The server resources required by the slice $i \in I$ are estimated in the following way:

$$r_i^{FU} = \frac{\alpha^{FU}(3A + A^2 + M \cdot CR \cdot L/3)RB_i}{10}, \tag{2}$$

where α is a scaling factor of a specific functional unit FU, A identifies the number of antennas, M are the modulation bits, CR is the coding rate, L is the number of MIMO layers, and RB is the resource blocks. Based on [34], $\alpha^{DU} = 0.5$ and $\alpha^{CU} = 0.1$.

We model the power consumption of a server as in [25]:

$$P_s = \mathcal{E} \cdot P^{idle} + P^{act} \cdot load, \tag{3}$$

where $\mathcal{E}$ is a binary variable that indicates whether the server is on/off. P^{idle} represents the static power, which covers the fixed costs of operating a server for as long as it remains powered on. On the other hand, P^{act} denotes the dynamic power consumption, which varies linearly with the server's load. In our scenario, this is equivalent to:

$$P_s = \mathcal{E} \cdot P^{idle} + P^{act} \cdot \frac{GOPS_{alloc}}{G_s}, \tag{4}$$

where $GOPS_{alloc}$ corresponds to the server GOPS currently allocated to active slices.

Our objective is to minimize the operational cost and the energy consumption while maximizing the number of accepted slices. x_{is}^{DU}, x_{is}^{CU} are the binary decision variables that associate the functional units O-DU and O-DU belonging to the slice $i \in I$ to server $s \in S$. We introduce an additional decision variable $z_{iss'}$, which indicates the placement of the functional units in the servers $s, s' \in S$. See Table 1 for the complete list of symbols.

$$\text{minimize} \quad \sum_{i \in I} \sum_{s \in S} C_s \cdot P_s \cdot (x_{is}^{DU} + x_{is}^{CU}) - \sum_{i \in I} \sum_{s \in S} (x_{is}^{DU} + x_{is}^{CU}) \tag{5}$$

$$\text{s.t.} \quad \sum_s x_{is}^{DU} \leq 1, \sum_s x_{is}^{CU} \leq 1, \forall i \in I \tag{6}$$

$$\sum_s x_{is}^{DU} = \sum_s x_{is}^{CU}, \forall i \in I \tag{7}$$

$$\sum_{s \in S_r} x_{is}^{DU} = 0, \forall i \in I \tag{8}$$

Table 1
Summary of key notation.

Symbol	Description
S	Global set of servers
S_e	Set of servers in the edge cloud
S_r	Set of servers in the regional cloud
G_s	GOPS capacity of server s
C_s	Cost per GOPS of server s
$d_{ss'}$	Distance of link between servers s and s'
$b_{ss'}$	Bandwidth of link between servers s and s'
$l_{ss'}$	Latency of link between servers s and s'
n	Refractive index of the fiber optic cable
c	Speed of light
I	Set of slice requests
FU	Generic O-RAN functional unit
r_i^{FU}	GOPS required by FU of slice i
b_i	Bandwidth required by link between O-DU and O-CU in slice i
l_i	Latency required by link between O-DU and O-CU in slice i
α^{FU}	Scaling factor of FU
A	Number of antennas
M	Modulation bits
CR	Coding rate
L	MIMO layers
RB_i	Resource Blocks required by slice i
P_s	Power consumption of server s
x_{is}^{FU}	Decision variable describing the placement of FU of slice i in server s
$z_{iss'}$	Decision variable describing the placement of the FUs composing slice i in servers s and s'

$$\sum_{i\in I}(r_i^{DU}\cdot x_{is}^{DU}+r_i^{CU}\cdot x_{is}^{CU})<G_s,\forall s\in S \tag{9}$$

$$z_{iss'}\geq x_{is}^{DU}+x_{is'}^{CU}-1,\forall s,s'\in S,i\in I \tag{10}$$

$$z_{iss'}\leq (x_{is}^{DU}+x_{is'}^{CU})/2,\forall s,s'\in S,i\in I \tag{11}$$

$$\sum_i b_i\cdot z_{iss'}\leq b_{ss'},\forall i\in I,s,s'\in S \tag{12}$$

$$l_{ss'}\cdot z_{iss'}\leq l_i,\forall i\in I,s,s'\in S \tag{13}$$

$$x_{is}^{DU},x_{is}^{CU},z_{iss'}\in[0,1],\forall i\in I,\forall s\in S \tag{14}$$

Our optimization problem is subject to the following constraints: a functional unit cannot be placed more than once, as formulated by (6), and a slice is active only if both its O-DU and O-CU functions are placed, as in (7). Constraint (8) restricts the placement of O-DU functions to edge servers only. Server capacity is respected by constraint (9), which ensures that the total GOPS of a server assigned to slices do not exceed its capacity. The joint placement of functions on servers s and s' is captured by constraints (10) and (11). Network constraints are enforced through (12), which ensures that the allocated bandwidth does not exceed the link's capacity, and (13), which restricts the use of links to those that satisfy the latency requirements of each slice. Finally, constraint (14) enforces binary values for the decision variables.

Due to its constraints, our problem can be reduced to the knapsack problem [35,36], which is proven to be NP-HARD. For this reason, we propose a DRL-based formulation to solve it.

5. Crown algorithmic solution

Deep Reinforcement Learning (DRL) has demonstrated strong capabilities in addressing complex problems without requiring prior domain knowledge. By leveraging neural networks as function approximators, DRL algorithms can learn policies that maximize expected long-term return through interaction with the environment. Solving such problems with reinforcement learning involves formulating them as Markov Decision Processes (MDPs). In the following sections, we define the key components of our MDP.

- State: this variable describes the environment in a certain timestep. It must contain all the necessary information to take an appropriate action. Our state space includes information about the physical network (server interconnections) and the FUs to place. The physical network is represented through a matrix whose rows contain nodes and edge features. Specifically, servers are characterized by their location (edge or regional), computing capacity, current GOPS allocated, idle power consumption, dynamic power consumption, and cost per GOPS. Links between servers are characterized by their bandwidth, delay, and connected endpoints. The two FUs (O-DU and O-CU) are characterized by the GOPS required, the type of the slice they belong to (URLLC,eMBBor mMTC), bandwidth, and delay required for their connection.
- Action: it defines the server on which we run the functional units O-DU and O-CU. We take two sequential actions with the same neural network: first, we place the O-DU and, after the state is updated, the O-CU. We enforce the capacity SLA through action masking and discard all the servers that do not have enough capacity to host the FUs by setting their q-values to $-\infty$. This masking highly reduces the convergence time and improves the performance [37–40].
- Reward: the training procedure aims at maximizing the long-term reward. For this reason, the reward function corresponds to our optimization objective. Since we want to maximize the accepted slices while achieving lower costs and power consumption, we define it in the following way:

$$R=\begin{cases}\alpha(1-C_s/C_{max})+\beta(1-E/E_{max})\\ -\gamma\end{cases} \tag{15}$$

where $\alpha,\beta,\gamma\in\{0,1\}$ are some coefficients that balance the objectives. If the placement is unsuccessful, e.g., the bandwidth and latency SLAs are violated, the agent is penalized.

We employ a Deep-Q neural network (DQN) to perform the placement decision for the two FUs enhanced with cross-attention as described in Fig. 2.

Once the neural network receives the state as input, it computes the cross attention between the physical server network and the processing chain in the following way:

$$\text{CrossAttention(Q,K,V)}=softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{16}$$

where d_k = hidden_features//n_heads. In cross-attention, Q comes from a projection of the FU matrix's hidden states, and K and V come from projections of the physical network's hidden states. The similarity between Q and K determines matching scores, which control how much

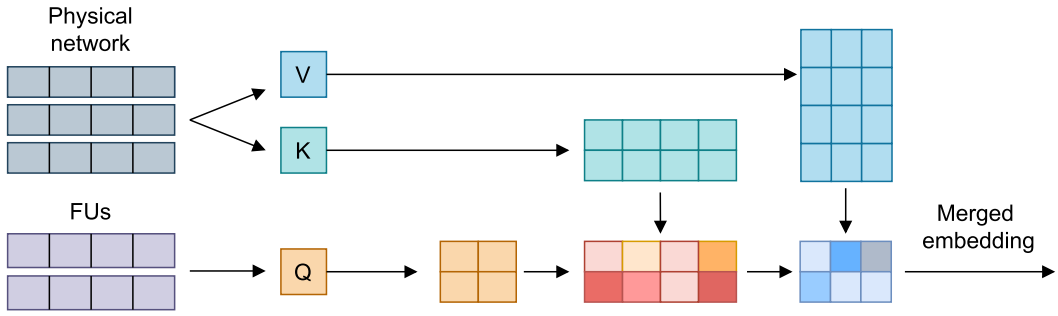


Fig. 2. Cross attention for O-RAN FU embedding.

Table 2
Servers settings for the edge cloud deployment, as from [41].

	Type 1	Type 2
Idle Consumption (W)	200	280
Max. Consumption (W)	657	1566

Table 3
Radio channel parameters for simulation tests.

Parameter	Value
MCS index	[17, 28]
Modulation Scheme	64-QAM
Symbols per sub-frame	14
Subcarrier	12
MIMO layers	2
Bits per symbol	$log_2(64)$

each V contributes to the final output. This operation corresponds to interrogating the physical network for the most relevant servers to place an FU, as represented in the final embedding used for the downstream task of server selection. The obtained merged embedding is then fed to multiple Linear layers to obtain Q-values. Such Q-values are estimates of the expected discounted cumulative long-term reward. At each iteration, the highest Q-value index determines the action taken.

6. Evaluation

In this section, we present the results obtained from a simulated infrastructure using a Python-based discrete-event simulator. We first focus on the slice acceptance ratio, the deployment cost, and power consumption, demonstrating the advantages of a flexible deployment. Later, we compare against alternative approaches for a more complete evaluation.

6.1. Settings

The DQN was trained for 100000 steps using a learning rate of 0.0001. The discount factor γ was set to 0.99 to balance the importance of immediate and future rewards. Training was performed with a batch size of 256 samples per iteration. Cross-attention was computed with an embedding size of 64 and 2 heads. To prevent overfitting, we set the dropout rate to 0.1.

To test the trained model, we built a simulation framework in Python. We simulate the arrival of 80 slice requests through a negative exponential distribution with a coefficient of 0.25 [18]. Each slice request has a random duration from 200 to 300 time units. Based on the slice type, the requirements can vary. For midhaul latency bounds, we consider random values in the following ranges, as outlined in [34]: for URLLC users, the latency is between $100 - 300\mu$ sec, for eMBB, it is set at 500μ sec, and for mMTC users, the latency bound is 1000μ sec. According to [42], we assign 50 RBs to eMBB users, 25 to URLLCand 5 to mMTC. Additionally, the bandwidth requirement for the midhaul links is of $10Mbps$ [20].

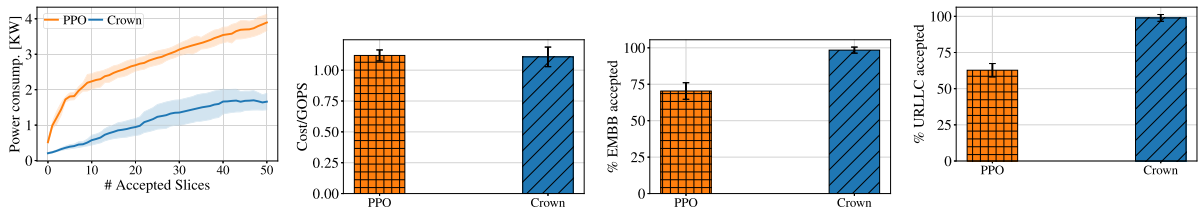
To simulate a generic infrastructure, we assume a varying number of antennas and servers across different runs: 2 to 4 antennas, 3 to 5 edge servers, and 3 to 4 regional servers. Each edge server is situated at a distance of 10 km from the O-RUs and may be co-located with other edge servers or positioned at a maximum distance of 5 km from one another. Meanwhile, each regional server is located between 40 km and 80 km from the O-RUs [21]. To create a heterogeneous infrastructure, we assign random capacities to each server within predefined ranges. Similar to previous studies [21,42], edge servers have a capacity range of 500 to 1000 GOPS, while regional servers fall within the range of 1000 to 2000.

To simulate the energy profile, we consider two types of servers. Type 1 is a *Think System SR630 V4* equipped with an *Intel Xeon 6780E* processor running at 2.2 GHz, while Type 2 is a *ThinkSystem D3 Chassis* where each ThinkSystem SD535 V3 has an *AMD EPYC 9845* processor operating at 2.1 GHz. Their characteristics are detailed in Table 2. Based on these features, the following power parameters are specified: $P^e_{idle} \in [150, 200]$, $P^e_{\max} \in [600, 657]$, $P^r_{idle} \in [250, 280]$, $P^r_{\max} \in [1500, 1566]$. As the computational characteristics of the servers vary, the cost per GOPS also varies, falling within these ranges: $C_{edge} \in [1, 1.59]$, $C_{reg} \in [0.2, 0.55]$. We assume a mesh connectivity of fiber links among all servers in the system. The fronthaul links have an available bandwidth of 1 Gbps, while edge interconnections have a random capacity ranging from 1 to 10 Gbps, and midhaul links have a random capacity between 10 and 20 Gbps.

Additional radio parameters used in the experiments are outlined in Table 3. We perform 45 simulations with different seeds and report the confidence intervals for each graph.

6.2. Benchmarks

- PPO: we compare against [19] that proposes a DRL solution based on Proximal Policy Optimization (PPO) that aims at maximizing the profit. It assumes that O-DU functions can be placed in the edge cloud, while O-CU in the regional cloud only. The reward function is formulated to maximize the revenue of the slice accepted and reduce the cost of idle servers. The state space includes the remaining capacity of each server, the remaining bandwidth of each transport link, the next sleep time of servers, and the slice request features. The slice request is characterized by data rate, holding time, type, and the RU to which the user is connected. The type of slice considered is URLLC and eMBB. The action space includes the possibility to reject slices to prefer more convenient ones.
- Transformer: we compare against a decoder-only Transformer that handles sequential data similarly to [21]. They propose a flexible



(a) Power consumption in KW (b) Cost per GOPS (c) eMBB acceptance ratio (d) URLLC acceptance ratio

Fig. 3. Crown achieves comparable placement cost to PPO, while reducing power consumption and accommodating more slices.

O-CU and O-DU placement such that O-DU can be placed in both regional and edge clouds based on constraints and optimization objectives. The state space includes per-user features, such as its distance to the RU, number of RBs, MCS index, requested latency, and GOPS. The reward function aims at maximizing the admitted slices while reducing their cost. Different from [21], which trains a supervised RNN with a dataset constructed by solving an ILP problem, we train a Transformer through RL with the same state, action, and reward formulation as they propose.

- FF: we also compare against a non-learning-based algorithm. In particular, we implement a first-fit algorithm that sorts the servers based on their available resources. For the O-CU, it chooses regional if it finds one that satisfies the constraints; otherwise, it will choose an edge server. It is cost and energy-agnostic.
- DQN: we train and test a simple feed-forward DQN that is not equipped with a cross-attention module. Since the RL formulation is identical to Crown, this comparison is useful to highlight the impact of our contribution to the performance achieved by the model.

6.3. Comparison against fixed deployment

We first study the advantages of a flexible placement with respect to a fixed placement, such as the one proposed by PPO. Since [19] is designed to accommodate only eMBB and URLLC slices, we limit the evaluation to these types of slices for this subsection.

Fig. 3a shows the power consumption in KW. The visualization is capped to 50 accepted slices, which is the maximum for PPO. While the power consumption of PPO explodes over time, Crown keeps it under control and converges to a reduction of 57% on average. We also evaluate and report the total incurred cost divided by the number of allocated GOPS in Fig. 3b. PPO does not model the deployment cost; however, the fixed placement of O-CU in the regional cloud automatically achieves a low cost, as regional servers have more resources and, therefore, lower prices [20]. Despite this, Crown achieves comparable cost, indicating that it learns to place the O-CU effectively at the regional or edge level to reduce deployment cost. Finally, we compare the acceptance rate of the slices. PPO favors slices with higher revenue; therefore, many eMBB slices are rejected. However, since it can only place O-CU FUs in the regional cloud, which fails to satisfy the latency constraints most of the time, this translates to a reduced number of URLLC slices correctly placed. On the other hand, the flexibility of Crown translates into an acceptance ratio of 98.4% for eMBB 98.9% for URLLC on average.

6.4. Flexible placement solutions

In the next set of experiments, incoming slices will also include mMTC. Here, we compare against another learning solution, inspired by [21], a simple DQN without cross-attention and a heuristic algorithm.

Fig. 4 shows the power consumption, the cost per GOPS incurred, and the percentage of accommodated slices.

The First-Fit heuristic prioritizes servers with elevated available resources first, by definition. This translates into a policy that places O-CUs in the regional when possible. This automatically results in lower costs (Fig. 4b) but also high power consumption (Fig. 4a), despite being agnostic to these metrics. On the contrary, since we explicitly design our RL model to prioritize lower-priced and less consuming servers, Crown learns a placement strategy that further reduces the incurred cost, while simultaneously reducing power consumption, sacrificing only a small percentage of accepted slices.

The Transformer-based policy selects edge servers most of the time, as they easily satisfy the placement constraints. However, while this automatically guarantees a low energy consumption (Fig. 4a), it also results in higher deployment costs (Fig. 4b). In fact, it does not effectively learn to choose the regional cloud when it satisfies the constraints. This further translates into a placement strategy that cannot satisfy all the requests (Fig. 4c).

The DQN trained without a cross-attention model is able to satisfy the constraints and accommodate most of the slice requests (Fig. 4c). While it achieves comparable deployment costs to Crown (Fig. 4b), it fails to integrate the energy metric, resulting in higher power consumption than the other RL solutions (Fig. 4a).

On the other hand, Crown achieves the lowest power consumption (Fig. 4a), converging to a 15% reduction with respect to a DQN without cross-attention. It reduces power consumption while satisfying 98% of requests on average (Fig. 4c), indicating that it meets capacity, bandwidth, and latency constraints and incurs the least deployment cost among the DQN and other baselines (Fig. 4b). Once again, Crown demonstrates that it can balance multiple objectives.

Fig. 5 offers a deeper understanding of the different policies learned by the two RL formulations. The Transformer model has the lowest acceptance rate among the models. In particular, it struggles with the placement of URLLC slices, whose latency requirements are very strict, and eMBB as they require most resources. On the contrary, Crown and DQN, which deploy the RL formulation described in Section 5, achieve very high acceptance ratios across all slice types, comparable to the heuristic algorithm.

We now evaluate the overhead of adding the cross-attention module and its impact on inference time. As shown in Fig. 4d, the Transformer model is the slowest solution, due to its complex and deep neural network made of 4 decoder layers. Instead, adding only a cross-attention module maintains a low inference time, slightly more than 1 ms, making Crown more scalable than the first-fit heuristic.

6.5. Cross attention impact

In this subsection, we focus on how cross-attention enhances the neural network. For this aim, we plot the attention probabilities of our model during inference time for two FUs of the same processing chain (Fig. 6). Such probabilities do not explain the reason for a given action [43,44], mainly because multiple linear layers act after the cross-attention to determine the neural network's output. However, they still reflect how each FU attends to the physical network input. This information is later embedded and passed to the subsequent layers. To

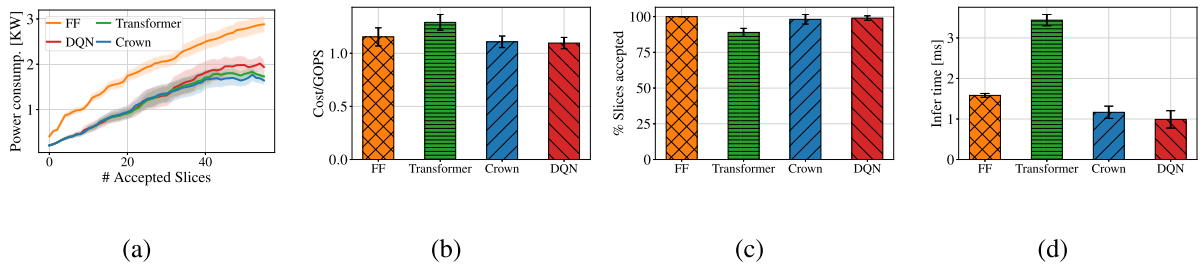


Fig. 4. Comparison of placement solutions in terms of (a) power consumption, (b) cost per GOPS allocated, (c) percentage of slices accepted and (d) inference time. Crown balances deployment cost, energy consumption, and number of allocated slices. The cross attention adds a negligible overhead in terms of processing time during inference.

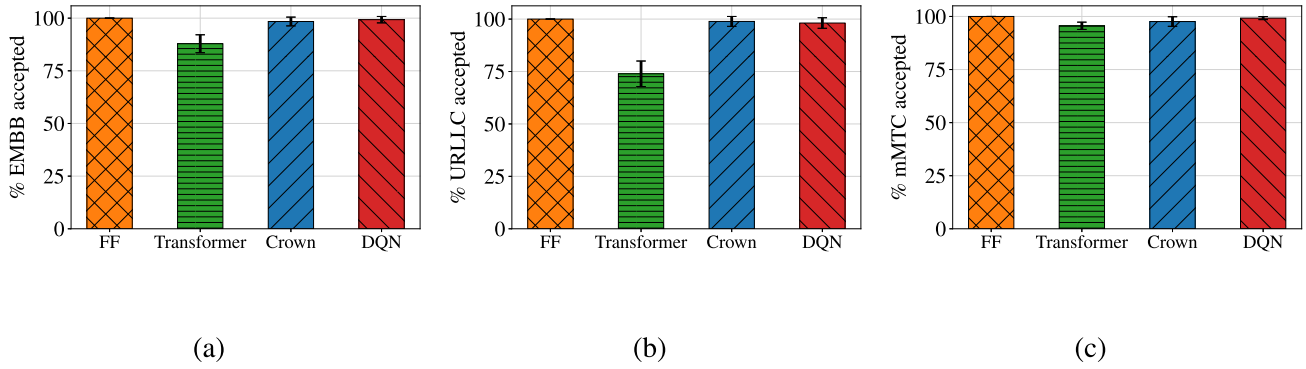


Fig. 5. Comparison of placement solutions in terms of acceptance ratio per slice type: (a) EMBB, (b) URLLC, and (c) mMTC. Crown has almost 100% acceptance ratio for all types of slice.

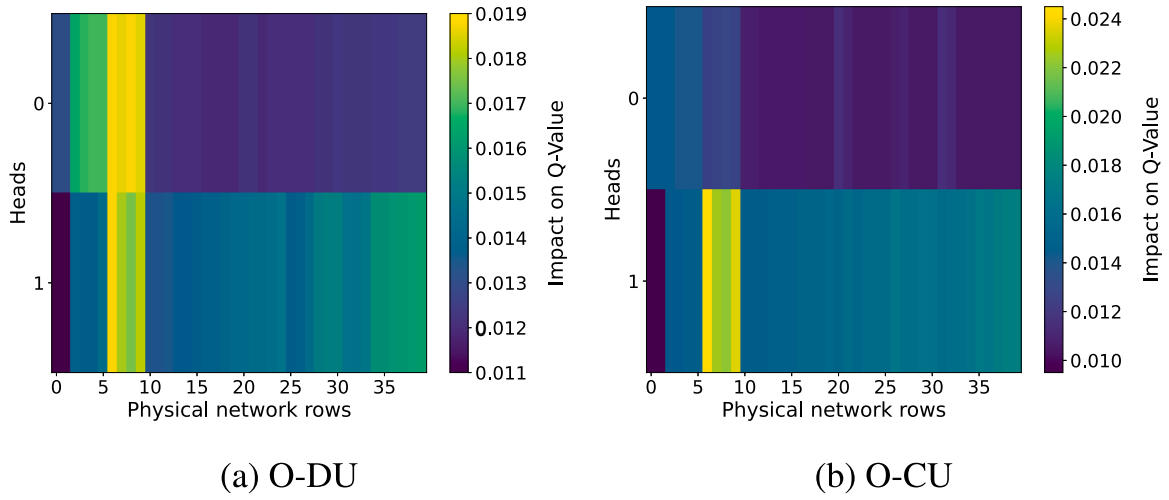
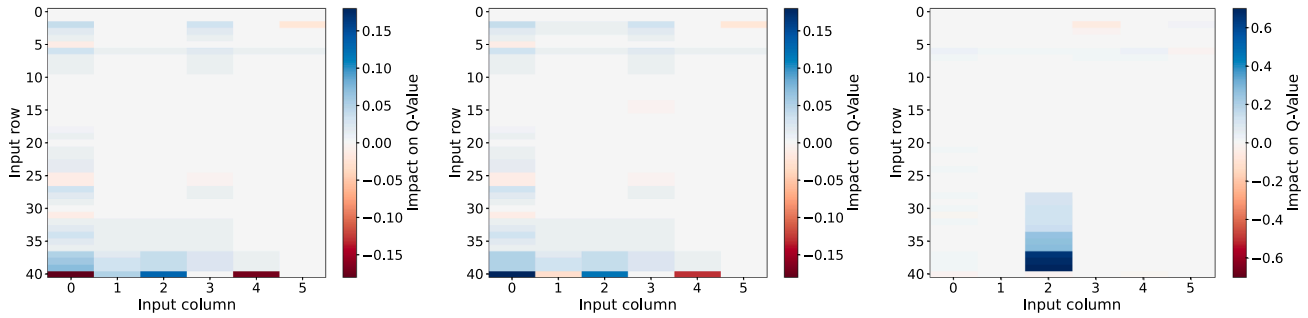


Fig. 6. Heatmap representing the cross-attention probabilities. It describes how impactful physical network features are in determining FU placement, as indicated by color intensity.

determine if the neural network effectively uses such information, we apply the SHAP Explainable AI (XAI) method to Crown and the simple DQN, revealing which input features are used to make a decision (Fig. 7).

Fig. 6 highlights that the two heads of the attention focus on different parts. Head 0 gives significant attention to the server features (rows $2-9$ of input) while head 1 also considers link features (starting at row 10). It is also evident that the two FUs, O-DU (a) and O-CU (b), handle the server infrastructure differently. While it is not clear how attention probabilities affect the output, it is evident that they do, as the heatmap is correlated with the SHAP output. Fig. 7a and b represent the entire input, comprising both the physical network (up to row 39) and the FU request (row 40), and highlight the importance of a cell on the final decision with more intense colors. Besides the specific FU characteristics, it is clear that the neural network considers the servers' features (represented in rows $2-9$) and some links' metrics, in line with the cross-attention heatmaps. In contrast, the SHAP heatmap for a simple DQN (Fig. 7c) shows a strong focus on a link feature, in particular the vertex of the link, and demonstrates a lack of ability to focus on important metrics. This result suggests that the cross-attention effectively guides the neural network towards the most meaningful input characteristics to determine the action placement for a specific FU request.



(a) O-DU placement decision (b) O-CU placement decision (c) O-DU placement decision

Fig. 7. SHAP output for (a-b) Crown model and (c) simple DQN. While a simple DQN cannot capture the most important metrics, Crown is guided by the cross-attention module, whose attention probabilities align with the SHAP results.

7. Conclusion

In this work, we introduced Crown, a cross-attention-enhanced DRL framework for flexible CU/DU placement in O-RAN. Crown operates under a hybrid deployment model that combines Scenarios B and C, allowing O-CUs to be instantiated either at the edge or in the regional cloud. This flexibility enables the agent to exploit the intrinsic trade-offs among latency, cost, and energy consumption, leading to more efficient placement decisions than the fixed-deployment strategies commonly assumed in prior studies. At the core of Crown is a task-specific cross-attention module that aligns two heterogeneous information sources: (i) the requested functional unit (O-DU/O-CU) and its SLA requirements (GOPS, bandwidth, latency), and (ii) the characteristics of the available physical infrastructure, including server cost-per-GOPS, idle and dynamic power, residual capacity, and link bandwidth/latency. By integrating this cross-attention mechanism into a DQN, Crown learns joint embeddings that allow the agent to selectively focus on the server- and link-level features most relevant to each FU and SLA. This leads to significantly improved placement decisions compared to a conventional feed-forward DQN. Importantly, the features emphasized by the attention mechanism are the same ones empirically shown to steer the agent's decision-making process, confirming the functional relevance of the architecture. These benefits translate into substantial performance gains: Crown reduces power consumption by 57% compared to fixed deployment and by 15% relative to a non-attention DQN, all while maintaining comparable slice admission rates and deployment costs. This demonstrates that integrating cross-attention with DQN is both effective and impactful for achieving flexible and energy-aware CU/DU placement in O-RAN. Moreover, our measurements of inference time indicate that the attention-based RL framework introduces negligible overhead.

Regarding dynamic conditions, Crown is designed to operate under variable network states during training, as each episode includes a newly generated infrastructure and sequential slice arrivals with heterogeneous demands. As future work, we plan to extend the state representation with real-time indicators (e.g., traffic trends, slice expiration) and explore policies for functional-unit migration under changing load conditions. Overall, Crown shows that a flexible-deployment, attention-guided RL strategy can significantly reduce energy consumption and operational cost while still meeting stringent O-RAN SLA requirements.

CRediT authorship contribution statement

Doriana Monaco: Writing - original draft, Methodology, Conceptualization; **Alessio Sacco:** Writing - review & editing, Validation, Supervision; **Guido Marchetto:** Writing - review & editing, Supervision.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] L. Bonati, M. Polese, S. D'Oro, S. Basagni, T. Melodia, Open, programmable, and virtualized 5G networks: state-of-the-art and the road ahead, Comput. Netw. 182 (2020) 107516.
[2] A.S. Abdalla, P.S. Upadhyaya, V.K. Shah, V. Marojevic, Toward next generation open radio access networks: what O-RAN can and cannot do!, IEEE Netw. 36 (6) (2022) 206–213.
[3] W. Azariah, F.A. Bimo, C.-W. Lin, R.-G. Cheng, N. Nikaein, R. Jana, A survey on open radio access networks: challenges, research directions, and open source approaches, Sensors 24 (3) (2024) 1038.
[4] B. Agarwal, R. Irmer, D. Lister, G.-M. Muntean, Open RAN for 6G networks: architecture, use cases and open issues, IEEE Commun. Surv. Tut. (2025).
[5] M. Kassi, S. Hamouda, RAN virtualization: how hard is it to fully achieve?, IEEE Access 12 (2024) 38030–38047.
[6] D. Wypiór, M. Klinkowski, I. Michalski, Open ran-radio access network evolution, benefits and market trends, Appl. Sci. 12 (1) (2022) 408.
[7] X. Liang, Q. Wang, A. Al-Tahmeesschi, S.B. Chetty, D. Grace, H. Ahmadi, Energy consumption of machine learning enhanced open RAN: a comprehensive review, IEEE Access 12 (2024) 81889–81910.
[8] X. Liang, A. Al-Tahmeesschi, Q. Wang, S. Chetty, C. Sun, H. Ahmadi, Enhancing energy efficiency in O-RAN through intelligent xapps deployment, in: 2024 11Th International Conference on Wireless Networks and Mobile Communications (WINCOM), IEEE, 2024, pp. 1–6.
[9] M. Polese, L. Bonati, S. D'oro, S. Basagni, T. Melodia, Understanding O-RAN: architecture, interfaces, algorithms, security, and research challenges, IEEE Commun. Surv. Tut. 25 (2) (2023) 1376–1411.
[10] F. Rezazadeh, L. Zanzi, F. Devoti, S. Barrachina-Muñoz, E. Zeydan, X. Costa-Pérez, J. Mangues-Bafalluy, A multi-agent deep reinforcement learning approach for RAN resource allocation in O-RAN, in: IEEE INFOCOM 2023-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2023, pp. 1–2.
[11] O.T. Ajayi, S. Zhang, Y. Cheng, Machine learning assisted capacity optimization for B5G/6G integrated access and backhaul networks, in: IEEE INFOCOM 2023-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2023, pp. 1–6.
[12] A.A. Khan, A.A. Laghari, A.M. Baqasah, R. Alroobaea, T.R. Gadekallu, G.A. Sampedro, Y. Zhu, ORAN-B5G: A next-generation open radio access network architecture with machine learning for beyond 5G in industrial 5.0, IEEE Trans. Green Commun. Netw. 8 (3) (2024) 1026–1036.
[13] M.Q. Hamdan, H. Lee, D. Triantafyllopoulou, R. Borralho, A. Kose, E. Amiri, D. Mulvey, W. Yu, R. Zitouni, R. Pozza, et al., Recent advances in machine learning for network automation in the o-ran, Sensors 23 (21) (2023) 8792.
[14] Z. Chen, L. Zhang, Y. Laili, X. Wang, F. Wang, Online simulation task scheduling in cloud manufacturing with cross attention and deep reinforcement learning, J. Intell. Manuf. (2024) 1–22.
[15] L. Slim, F. Bannour, Hierarchical multi-agent deep reinforcement learning with an attention-based graph matching approach for multi-domain VNF-FG embedding, in: GLOBECOM 2023-2023 IEEE Global Communications Conference, IEEE, 2023, pp. 2105–2110.
[16] R. Sahraoui, O. Houidi, F. Bannour, Energy-Aware VNF-FG placement with transformer-based deep reinforcement learning, in: 2024 IEEE/IFIP Network Operations and Management Symposium (NOMS 2024), 2024.
[17] N. He, S. Yang, F. Li, S. Trajanovski, L. Zhu, Y. Wang, X. Fu, Leveraging deep reinforcement learning with attention mechanism for virtual network function placement and routing, IEEE Trans. Parallel Distrib. Syst. 34 (4) (2023) 1186–1201.

[18] K. Ali, M. Jammal, Proactive VNF scaling and placement in 5G O-RAN using ML, IEEE Trans. Netw. Serv. Manage. 21 (1) (2023) 174–186.
[19] S. Nabhasmita, et al., Intelligent admission and placement of O-RAN slices using deep reinforcement learning, in: 2022 IEEE 8Th International Conference on Network Softwarization (NetSoft), IEEE, 2022, pp. 307–311.
[20] R. Joda, T. Pamuklu, P.E. Iturria-Rivera, M. Erol-Kantarci, Deep reinforcement learning-based joint user association and CU–DU placement in O-RAN, IEEE Trans. Netw. Serv. Manage. 19 (4) (2022) 4097–4110.
[21] H. Hojeij, M. Sharara, S. Hoteit, V. Vèque, On flexible placement of O-CU and O-DU functionalities in open-RAN architecture, IEEE Trans. Netw. Serv. Manage. (2024).
[22] T. Pamuklu, S. Mollahasani, M. Erol-Kantarci, Energy-efficient and delay-guaranteed joint resource allocation and du selection in o-ran, in: 2021 IEEE 4Th 5G World Forum (5GWF), IEEE, 2021, pp. 99–104.
[23] S. Mollahasani, T. Pamuklu, R. Wilson, M. Erol-Kantarci, Energy-aware dynamic DU selection and NF relocation in O-RAN using actor–critic learning, Sensors 22 (13) (2022) 5029.
[24] S. Maxenti, S. D'Oro, L. Bonati, M. Polese, A. Capone, T. Melodia, ScalO-RAN: energy-aware network intelligence scaling in open RAN, in: IEEE INFOCOM 2024-IEEE Conference on Computer Communications, IEEE, 2024, pp. 891–900.
[25] R. Singh, C. Hasan, X. Foukas, M. Fiore, M.K. Marina, Y. Wang, Energy-efficient orchestration of metro-scale 5G radio access networks, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, IEEE, 2021, pp. 1–10.
[26] I. Tamim, A. Saci, M. Jammal, A. Shami, Downtime-aware o-ran vnf deployment strategy for optimized self-healing in the o-cloud, in: 2021 IEEE Global Communications Conference (GLOBECOM), IEEE, 2021, pp. 1–6.
[27] S.O. Yese, S. Berri, A. Chorti, Energy efficient and delay-Guaranteed user association algorithm for O-RAN, in: International Conference on Communications (ICC), 2025.
[28] M. Bordin, A. Lacava, M. Polese, S. Satish, M.A. Nittoor, R. Sivaraj, F. Cuomo, T. Melodia, Design and evaluation of deep reinforcement learning for energy saving in open ran, in: 2025 IEEE 22nd Consumer Communications & Networking Conference (CCNC), IEEE, 2025, pp. 1–6.
[29] C. Leonelli, D. Kefalas, S. Fdida, P. Bellavista, T. Korakis, Dynamic resource allocation and energy optimization in 5G O-RAN: real-World insights and testbed evaluations, in: 2025 IEEE International Conference on Communications Workshops (ICC Workshops), IEEE, 2025, pp. 1019–1024.
[30] S. Nabhasmita, et al., Towards energy efficient functional split and baseband function placement for 5g ran, in: 2023 IEEE 9Th International Conference on Network Softwarization (NetSoft), IEEE, 2023, pp. 237–241.
[31] ORAN Alliance, 2018, (https://www.o-ran.org/). Accessed: November 5, 2024.
[32] r.G.P. Project, Technical Specification Group Radio Access Network; Study on New Radio Access Technology: Radio Access Architecture and Interfaces, Technical Report TR 38.801, 3GPP, 2017.
[33] O.-R. Alliance, O-RAN Use Cases and Deployment Scenarios WhitePaper, Technical Report, O-RAN Alliance, 2020.
[34] S. Mondal, M. Ruffini, Optical front/mid-haul with open access-edge server deployment framework for sliced O-RAN, IEEE Trans. Netw. Serv. Manage. 19 (3) (2022) 3202–3219.
[35] A. Filali, Z. Mlika, S. Cherkaoui, A. Kobbane, Dynamic SDN-based radio access network slicing with deep reinforcement learning for URLLC and eMBB services, IEEE Trans. Network Sci. Eng. 9 (4) (2022) 2174–2187.
[36] H. Hojeij, G.I. Ricardo, M. Sharara, S. Hoteit, V. Vèque, S. Secci, Flexible association and placement for open-RAN, in: IEEE INFOCOM 2024-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2024, pp. 1–6.
[37] L. Vu, T. Vu, T.-L. Vu, A. Srivastava, Safe exploration reinforcement learning for load restoration using invalid action masking, in: 2023 IEEE Power & Energy Society General Meeting (PESGM), IEEE, 2023, pp. 1–5.
[38] A. Sacco, M. Flocco, F. Esposito, G. Marchetto, Owl: congestion control with partially invisible networks via reinforcement learning, in: IEEE INFOCOM 2021-IEEE Conference on Computer Communications, IEEE, 2021, pp. 1–10.
[39] D. Monaco, A. Sacco, C. Casetti, G. Marchetto, Scheduling latency-sensitive tasks in the cloud continuum with hierarchical reinforcement learning, in: NOMS 2025-2025 IEEE Network Operations and Management Symposium, IEEE, 2025, pp. 01–09.
[40] L. Pappone, A. Sacco, F. Esposito, Mutant: learning congestion control from existing protocols via online reinforcement learning, in: 22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25), 2025, pp. 1507–1522.
[41] Standard Performance Evaluation Corporation, SPECpower results, 2025, (https://www.spec.org/power_ssj2008/results/res2025q1/). Accessed: February 17, 2025.
[42] E. Sarikaya, E. Onur, Placement of 5G RAN slices in multi-tier O-RAN 5G networks with flexible functional splits, in: 2021 17Th International Conference on Network and Service Management (CNSM), IEEE, 2021, pp. 274–282.
[43] S. Jain, B.C. Wallace, Attention is not explanation, arXiv:1902.10186 (2019).
[44] S. Wiegreffe, Y. Pinter, Attention is not not explanation, arXiv:1908.04626 (2019).