A Tenant-Driven Slicing Enforcement Scheme based on Pervasive Intelligence in the Radio Access Network

Arcangela Rago[a,c], Sergio Martiradonna[a,c], Giuseppe Piro[a,c], Andrea Abrardo[b,c], Gennaro Boggia[a,c]

[a] *Dept. of Electrical and Information Engineering - Politecnico di Bari, Bari, Italy*
[b] *Dept. of Information Engineering and Applied Mathematics, University of Siena, Siena, Italy*
[c] *CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni*

Abstract

In Beyond 5G mobile networks, the network slicing paradigm offers the possibility of sharing the network infrastructure among different tenants: the tenant declares communication service requirements and the Infrastructure Provider configures (potentially on-demand) the corresponding network slice instances. The management of network slicing in the core network has been deeply investigated in the current scientific literature. On the contrary, handling network slicing in the Radio Access Network still represents an open and very challenging research topic, mostly due to the unpredictable variability of the wireless channel, network dynamics and heterogeneity, slice isolation, as well as different Quality of Service requirements of various services. In order to achieve an important step forward in this direction, this paper proposes a tenant-driven Radio Access Network slicing enforcement scheme based on Pervasive Intelligence. The proposed approach grounds its roots in the *Pay for What You Get* paradigm: it promises to avoid the radio resources over-provisioning while saving bandwidth. To achieve these goals, Artificial Intelligence mechanisms are innovatively and pervasively integrated into some key functionalities of both Infrastructure Provider and tenants. On the one hand, the Infrastructure Provider exploits a Deep Learning scheme (i.e., convolutional autoencoder) to compress the information on network resources and connectivity and share the actual (but hidden through compression) network status with the tenants. On the other hand, each tenant implements a Deep Reinforcement Learning algorithm (i.e., Deep Deterministic Policy Gradient) to dynamically adapt bandwidth requests according to its own users' requirements. The outcomes of this algorithm are then used by the Infrastructure Provider to effectively enforce network slicing at the Radio Access Network level. Computer simulations investigate the proposed approach in a realistic scenario, which jointly embraces enhanced Mobile BroadBand, Advanced Driver Assistant Systems, and best-effort applications. Obtained results demonstrate the effectiveness of the proposal against conventional resource allocation methods.

Keywords: Network Slicing, Pervasive Intelligence, Deep Learning, Deep Reinforcement Learning, Beyond 5G

1. Introduction

Beyond 5G (B5G) networks are expected to support various new use cases from vertical industries, which impose a wide range of performance and requirements. In this context, network slicing is emerging as a valid key enabler to support customized network services on-demand, permitting multiple vertical industries to execute their solutions on the top of a shared infrastructure and accommodating heterogeneous services [1–6]. At the same time, it promises to open new business models for all the interested stakeholders (that are Infrastructure Providers and tenants), while intensifying the collaboration among all the involved parties and keeping their requirements distinct [7–9]. On the one hand, in fact, the Infrastructure Provider should manage and accept resource requests issued by tenants, without having access to their most significant data. On the other hand, tenants should be able to submit their requests, without having complete comprehension of the network itself. The management of network slicing in the core network was deeply investigated in the current scientific literature. On the contrary, handling network slicing in the Radio Access Network (RAN) is still an open issue. In fact, the unpredictable variability of the wireless channel, network dynamics, slice isolation, scarcity of resources, increased inter-cell/inter-tier interference caused by spatial multiplexing of the spectrum, as well as diverse Quality of Service (QoS) requirements of different services pose significant technical challenges on the management and provisioning of RAN slicing [5, 10]. To this end, it would be essential to pervasively adopt Artificial Intelligence (AI), especially in view of the ever-increasing network complexity due to resource sharing among multiple entities [11, 12]. The advent of programmable networks and network virtualization, as well as the easier large-scale data acquisition, promoted the allocation, management, and orchestration of network resources through AI techniques [12–16]. Most of the contributions in this field, which employs AI-based methods for channel estimation and to manage network slicing in the core network and RAN, proposes central-

ized solutions based on Deep [17–19] and Reinforcement Learning (RL)/Deep Reinforcement Learning (DRL) [20–33], where the network status is fully observable (please see Section 2 for further details).

In the business vision of network slicing, however, tenants are decoupled from the Infrastructure Provider and they can only have a partial vision of the network status [4, 34, 35]. The current state-of-the-art approaches do not handle these aspects, by presenting slicing enforcement schemes driven directly by the Infrastructure Provider.

To bridge this gap, this paper proposes a tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence, perfectly aligned with the emerging business and privacy-preserving vision of network slicing. In the reference scenario, the Infrastructure Provider exposes its resources to heterogeneous tenants, willing to provide customized services to a group of end-users. Here, the proposed tenant-driven approach allows each tenant to interact with the Infrastructure Provider during the RAN slice enforcement process. Differently from baseline solutions (where the tenant always uses a fixed amount of bandwidth initially negotiated with the Infrastructure Provider), the proposed strategy permits the tenant to re-formulate its requests over time (i.e., every second), allowing it to periodically update the set of required resources.

During the RAN slice enforcement process, tenants can operate independently (by only using their own partial vision of the network status) on the underlying infrastructure to dynamically adapt bandwidth requests and guarantee their expected service performance [4, 36]. Differently from most of centrialized and non-tenant-driven approaches proposed in the current scientific literature (see the next Section for more details), this surely simplifies the Infrastructure Provider network control and management, while also ensuring higher levels of scalability [8].

Moreover, according to the Pervasive Intelligence paradigm and starting from the outcomes of the preliminary contribution presented by the same authors in [37], both Infrastructure Provider and tenants exploit AI to accomplish their tasks. This is perfectly in line with the pervasive intelligent endogenous design of future generations of mobile networks [12]. Specifically, the Infrastructure Provider exploits a Deep Learning method based on a convolutional autoencoder, which compresses the information on network resources and connectivity and shares the actual (but hidden through compression) network status with the tenants. In turn, each tenant exploits the resulting hidden knowledge of the network status in a DRL agent based on the Deep Deterministic Policy Gradient (DDPG) algorithm in order to dynamically adapt bandwidth requests according to its own users' requirements. Finally, the Infrastructure Provider employs the outcomes of the DDPG algorithm to effectively enforce the network slices in the RAN. In addition to the DRL agent, the proposed approach, while reaching similar performance with respect to the preliminary contribution by the same authors [37], innovatively adopts an autoencoder in the Infrastructure Provider to compress network status and thus it perfectly matches the privacy-preserving vision of network slicing. Thus, even if each tenant does not fully know network resources and conditions information, the bandwidth requested for offering services and respecting a given QoS constraint (i.e., target Service Availability) could be optimally allocated according to the *Pay for What You Get* paradigm: the lower the requested bandwidth, the higher the tenant savings, while avoiding the radio resources overprovisioning. In other words, each tenant is able to dynamically and autonomously formulate, without knowing all the details of the RAN, the amount of bandwidth to allocate to its RAN slice. At the same time, the Infrastructure Provider still maintains the control of its resources by achieving slice enforcement through its RAN control. This also allows to filter the requests, verify the respect of Service Level Agreement, or implement countermeasures in high loaded periods.

Independently from the number of tenants operating into a specific RAN, the main benefits provided by the proposed approach are summarized in what follows:

- tenants can dynamically and autonomously formulate the amount of bandwidth to allocate to its RAN slice, hence providing higher scalability and offloading the related computations from the Infrastructure Provider;
- each tenant can hide some valuable information, i.e., business strategies, subscriber numbers, etc., to the Infrastructure Provider;
- accordingly, the overall approach promotes a collaboration between the involved stakeholders while guaranteeing the separation and the respect of their roles.

It is important to highlight that these main benefits are important independently of the number of tenants operating at the RAN.

The efficiency of the devised tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is investigated for the enhanced Mobile BroadBand (eMBB) and Advanced Driver Assistant Systems (ADAS) use cases, by using computer simulations with real and conceivable network and QoS settings in compliance with ITU and 5^{th} Generation (5G) specifications [38–40]. In addition to the slices managed by tenants subsystems, a third Best Effort (BE) slice, that uses the bandwidth left by the eMBB and ADAS slices, is handled directly by the Infrastructure Provider. Note that the role covered by the Infrastructure Provider in the actual slice enforcement process makes the resulting approach semi-distributed: each tenant autonomously implements the DRL algorithm (distributed scheme), and the Infrastructure Provider collects bandwidth requests and enforces the slices (last mile centralized decision). The comparison with conventional resource allocation methods, corresponding to the optimal, random, and dynamic (i.e., proportional to the users' requests) allocation of bandwidth, demonstrates that the proposed

approach ensures the best trade-off between bandwidth savings and bandwidth over-provisioning, while always ensuring the target Service Availability.

The remainder of the paper is as follows. Section 2 reviews the related contributions in this area and identifies the gaps bridged in this paper. Section 3 is dedicated to the proposed approach. Section 4 presents numerical results of Deep Learning and DRL algorithms coming from computer simulations. Finally, Section 5 concludes the paper and draws future research directions.

For readers' convenience, the list of employed acronyms is reported herein.

List of acronyms

Notation	Description
3GPP	3rd Generation Partnership Project
5G	5^{th} Generation
ADAS	Advanced Driver Assistant Systems
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
B5G	Beyond 5G
BE	Best Effort
C-RAN	Cloud-Radio Access Network
CNN	Convolutional Neural Network
CQI	Channel Quality Indicator
CSI	Channel State Information
DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
ECDF	Empirical Cumulative Distribution Function
eMBB	enhanced Mobile BroadBand
ITU	International Telecommunication Union
LOS	Line-Of-Sight
LSTM	Long Short-Term Memory
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
MSE	Mean Square Error
NLOS	Non-Line-Of-Sight
O-RAN	Open Radio Access Network
QoS	Quality of Service
RAN	Radio Access Network
ReLU	Rectified Linear Unit
RIC	RAN Intelligent Controllers
RL	Reinforcement Learning
RMSE	Root Mean Square Error
RRM	Radio Resource Management
SINR	Signal-to-Interference-plus-Noise Ratio
SLA	Service Level Agreement
UE	User Equipment
URLLC	Ultra Reliable and Low Latency Communication

2. Related Work

AI-based methods represent powerful instruments for solving typical technical problems of interest for Academia and Industry working on mobile communication systems and the scientific literature already provides a preliminary qualitative investigation. In particular, Deep Learning, RL, and DRL have been widely adopted for channel estimation and resource allocation problems [12–14, 41]. In this context, even in the presence of perfect traffic estimation, evaluating the optimal Radio Resource Management (RRM) setting is a very difficult task owing to the random nature of the radio conditions, requiring the use of optimization tools with unmanageable computational complexity. Alternatively, Deep Learning and RL/DRL offer low-complexity and effective solutions for RRM in communication and computing systems [15, 16].

Deep Learning can extract important features from data and model its high-level abstractions, avoiding manual description of a data structure [14, 15]. For this peculiarity, Deep Learning architectures have been successfully proposed in channel estimation. For example, Long Short-Term Memory (LSTM) networks perform the prediction of RAN resource usage by a network slice [17] and the prediction of future Channel Quality Indicator (CQI) values in a data-driven RAN slicing framework with Ultra Reliable and Low Latency Communication (URLLC) and eMBB slices [18]. Furthermore, an encoder-decoder structure based on Convolutional Neural Network (CNN) is presented in [19] for estimating the traffic of slices deployed at Cloud-Radio Access Network (C-RAN), Multi-access Edge Computing (MEC), and core datacenters.

The time-varying wireless channel largely impacts the optimal decision-making process for resource allocation problems. Differently from traditional solutions that require to rerun the algorithms every time the environment changes, RL and DRL methods fit for these challenges [14]. A slice admission strategy based on RL is presented in [20] for a flexible RAN. Q-learning is adopted to handle RAN slicing [25], supporting an eMBB and a Vehicle-to-Everything slice on the same RAN infrastructure. In [21], LSTM is incorporated into the actor-critic DRL algorithm for an intelligent resource management of RAN slicing. Deep Q-Network [23, 24] and its modified versions [22, 26] are exploited for slice management in RAN. Specifically, the contribution in [22] entails a Generative Adversarial Network-powered Deep distributional Q-Network for

Table 1: Comparison among this work and the other contributions adopting AI-based techniques for the management of network slicing.

Contributions	AI-based techniques		Reference Dataset		Network Slicing			
	Deep Learning	RL/ DRL	Simulated	Real	Core Network	Radio Access Network	Tenant-driven	Network status compression
[17]	✓		✓			✓		
[18]	✓		✓	✓		✓		
[19]	✓			✓	✓	✓		
[20–30]		✓	✓			✓		
[31–33]		✓	✓		✓	✓		
[8]	✓	✓	✓		✓		✓	✓
This work	✓	✓	✓			✓	✓	✓

demand-aware resource allocation, while resource block allocation to multiple slices is optimized in [26] by exploiting a method called Ape-X, that uses distributed learning in the Deep Q-Network (DQN) with multiple actors. Cooperative multi-agent deep Q-learning jointly solves the RAN slicing and computing task offloading problem in [27]. Moreover, by jointly optimizing radio and computation resources in the context of RAN network slicing, the utility maximization problem formulated as a Markov Decision Process (MDP) is solved in [28] through the DDPG algorithm, that combines the DQN and the actor-critic approach. Similarly, the contributions in [29] and [30] extend the DDPG algorithm to obtain an optimal RAN slicing policy, by minimizing the long-term system cost in the context of vehicular networks and both the long-term QoS of services and spectrum efficiency of slices, respectively. Q-learning [31], Deep Q-learning [31, 32], and a distributed DRL strategy based on the Advantage Actor Critic (A2C) algorithm [33] also assist network slicing involving both RAN and core network.

Deep Learning can also support DRL-based resource allocation methods. In particular, the compression of high-dimensional CQI information, obtained through an autoencoder, is exploited in a DQN-based framework in [41]. This valuable contribution aims at optimizing computation offloading in the large-scale MEC system, but it does not focus on the network slicing problem. Autoencoders are also adopted in the core network slicing context. In particular, the framework proposed in [8] firstly entails an autoencoder-based classifier, which is used by the Infrastructure Provider to distribute tenants' virtual network slicing requests with similar characteristics to its different agents. Then, an autoencoder-based compression module extracts the key features of the virtual network requests. The compressed representation of features is fed into a DDPG-based model for resource pricing, advertising, and motivating tenants to request resources in a load-balanced manner. Therefore, virtual network slicing is accomplished in a distributed and tenant-driven manner: after compressing the features of requests, tenants compute their own virtual network embedding schemes independently and distributedly, according to the resource information (i.e., the available resources and their prices) advertised by the DRL agent.

To conclude, Table 1 summarizes the goals, the considered datasets, and the methodologies followed by the reviewed contributions employing AI for the management of network slicing. Related works were generally validated with simulated data, excepting for the contributions presented in [19] and in [18]. Moreover, to the best of authors' knowledge, there are no scientific contributions jointly exploit Deep Learning and DRL for a tenant-driven RAN slicing enforcement scheme, able to dynamically adapt bandwidth requests according to users' requirements of tenants and without fully knowing the network status. It is important to note that the proposal presented herein is not a simple adaptation of the contribution in [8] to the RAN context. Independently from the tenant-driven nature of conceived approach (as for [8]), our work addresses a new research problem, that is the design of an enforcement scheme for the RAN, through an innovative technical approach that i) completely differs from what already presented in [8] and ii) fulfills a new set of technical challenges not considered in [8].

3. The proposed Tenant-Driven RAN Slicing Enforcement Scheme based on Pervasive Intelligence

To present the main theoretical aspects related to the conceived approach, this Section focuses on a single Infrastructure Provider, willing to lease a portion of its hardware and software resources to a number of heterogeneous tenants to provide customized services to a group of end-users. Without loss of generality, it is assumed that each service is deployed through a dedicated and independent slice (specifically, a RAN slice in this contribution) [42]. In turn, each tenant uses these resources to install applications, hold its own data, enable its preferred security and privacy policies, and provide services with different bandwidth, latency, and QoS requirements [2, 4, 34]. To this end, different logical entities and networking functions are exploited across the overall communication infrastructure, including core network, edge network, and RAN (see Fig. 1).

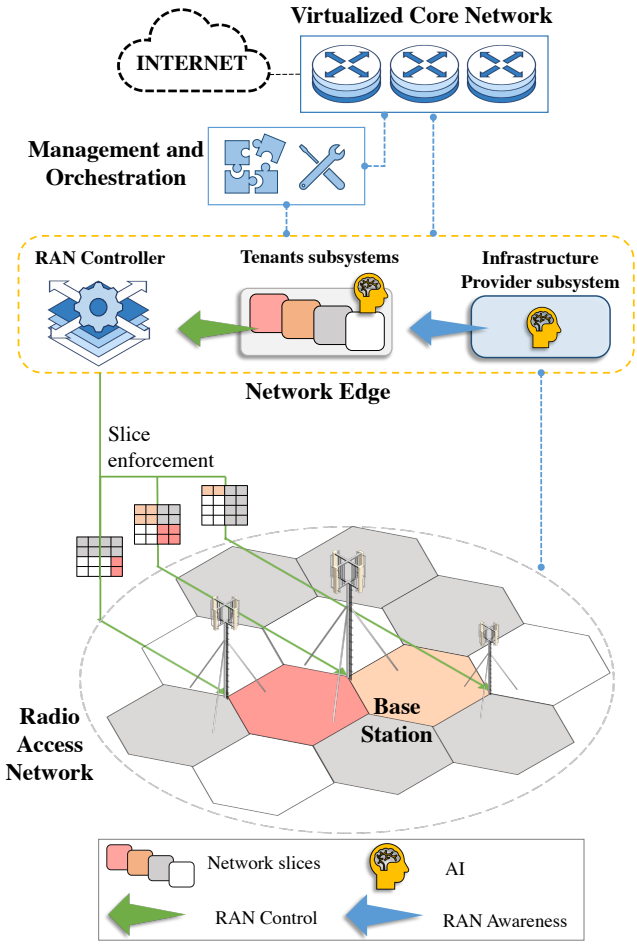

Figure 1: The reference architecture.

Specifically, the Management and Orchestration entity represents a suite for physical and virtual resources related to the life cycle of the deployed Network Services [43]. Indeed, it is in charge of configuring the entire considered system (i.e., core network and RAN). For instance, it sets traffic routing policy and flow priorities throughout the network and initiates and manages network resources. At the same time, the network edge hosts the Infrastructure Provider subsystem, a number of tenants subsystems, and the RAN controller, which continuously interact over time to ensure a dynamic RAN slicing enforcement strategy. These entities are placed on the edge of the network to leverage MEC native capabilities (i.e., intensive computing and memory capabilities in the proximity of end-users, while guaranteeing reliable and low-latency communication to the new heavy demanding and real-time services) [4]. In the proposal, the RAN controller can be considered a RAN Intelligent Controllers (RIC), i.e., a controller plus an orchestrator [44]. Thus, the RAN controller is able to monitor and orchestrate the RAN and then automatically and intelligently manage its resources to be assigned to each tenant (i.e., with management and orchestration functionalities only related to the resources of the RAN). It is important to highlight that the considered architecture is suitable to manage the entire network slices lifecycle, including preparation, instantiation, configuration, activation, run-time, and decommissioning phases [45]. During the preparation phase, the Infrastructure Provider implements an admission control algorithm, willing to define i) the number of multiple tenants to be hosted at the same time and ii) the upper bound of bandwidth assigned to each tenant. However, as already mentioned, the discussion will consider the run-time phase only, dedicated to the slicing enforcement. Therefore, the proposed approach could be integrated with any admission control technique already formulated in the current state of the art (e.g., the dynamic approach proposed in [46]).

As expected, the Infrastructure Provider has a comprehensive view of the network and it can accede to data not natively accessible to tenants (for example, the Infrastructure Provider subsystem is the only entity able to retrieve information from the RAN). Moreover, it has the control of the RAN and it is the only entity able to actually perform the RAN slice enforcement. Based on these premises, the proposed tenant-driven approach allows each tenant to interact with the Infrastructure Provider during the RAN slice enforcement process. Indeed, differently from baseline solutions where the tenant always uses a fixed amount of bandwidth initially negotiated with the Infrastructure Provider, the proposed strategy permits the tenant to periodically re-formulate its requests (i.e., every second), thus asking for a suitable set of required resources over time. The Infrastructure Provider, once received the tenant's request, implements the RAN slice enforcement through its own RAN controller.

The most important data exploited in this work is the radio channel condition experienced by end-users, shared with the Infrastructure Provider through Channel State Information (CSI) feedbacks. Being a well-founded methodology in legacy cellular networks [47], in fact, it is assumed to be used in 5G & Beyond as well. Among the other parameters carried by the CSI feedback, the CQI provides information about the current communication channel quality. Indeed, the developed methodology assumes that the tenant subsystem may formulate its bandwidth requests based on the information carried out by CQI feedbacks. At the same time, however, it is not reasonable to suppose that the Infrastructure Provider subsystem forwards all the collected CQI feedbacks to each tenant subsystem. Otherwise, privacy-preserving requirements and business roles of the Infrastructure Provider and tenants would be compromised, and the communication overhead at the network edge would be unnecessarily high [18]. To solve these issues, according to the Pervasive Intelligence paradigm, the Infrastructure Provider subsystem processes the collected CQI feedbacks through Deep Learning and exposes a compressed vision of the RAN status to the tenant subsystems. This task is performed through an autoencoder and represents one of the main novel ideas presented in this work. Specifically, by discarding irrelevant information and reducing the dimen-

sionality of data, the autoencoder is used to generate a feature learning representation of CQI feedbacks, without requiring the knowledge of data distribution nor the explicit identification of a certain structure [48–50]. As a result, by compressing the CQI information, it is possible to hide the network status (because tenants cannot reconstruct original CQI indexes) and to limit network complexity (because of reduced information exchanged with tenants subsystems). Indeed, the Infrastructure Provider subsystem sends the compressed CQI feedbacks to the tenant subsystem. The adopted autoencoder will be thoroughly described in Section 3.3. Then, the tenant subsystem further processes the received data (also in this case, through specific AI algorithms falling into DRL, as discussed hereafter) and supplies instructions for the successful handling of its RAN slice.

Fig. 2 shows the interaction between Infrastructure Provider and tenants subsystems. It is important to remark that the tenant subsystem cannot manage RAN slices directly. However, any action is controlled (first) and implemented (then) by the Infrastructure Provider. For this reason, the tenant subsystem sends the aforementioned instructions to the RAN controller, which decides to accept/deny them, allocates RAN resources to the slice, and enforces the slicing policy on the available spectrum. At the same time, it is possible to affirm that the proposed approach does not require major changes to the communication architecture, including those envisaged by both Open Radio Access Network (O-RAN) project or 3rd Generation Partnership Project (3GPP) specifications. According to [44], the O-RAN architecture includes two RICs, namely non-Real-Time RAN Intelligent Controller (referred to as "non-RT RIC", which operates at more than 1s time scale) and near-Real-Time RAN Intelligent Controller (referred to as "near-RT RIC", which operates at a lower time scale ranging from 10ms to 1s). In this context, non-RT RIC is a component of the service management and orchestration, able to provide policies and guidelines to the Infrastructure Provider, useful for the actual RAN enforcement. The near-RT RIC entity, instead, directly controls the RAN through other specific Application Programming Interfaces (APIs). The proposed tenant-driven approach easily matches such an architecture: the role of the tenant to dynamically request a different amount of bandwidth can be implemented by the non-RT RIC entity. At the same time, the functionalities of the RAN controller of the Infrastructure Provider are implemented by the near-RT RIC entity. With reference to a 3GPP-compliant deployment, the proposed approach only requires the definition of an interface, e.g., APIs, between the tenant and the RAN controller. Such an interface allows the tenant to formulate and update its requests dynamically, based on its own information and compressed data received from the Infrastructure Provider. As a consequence, RAN procedures (e.g., bearer control, scheduling, power control and scheduling) are fully in charge of the Infrastructure Provider and, hence, no major modification to the architecture is required.

3.1. Problem statement

As illustrated in Fig. 1, this work considers a cluster of cells, each divided into sectors, i.e., portions of the cell served by one of the co-located base stations in the RAN. In addition, several network slices may be deployed in each of these cells requiring a target QoS, characterized by the service requirements indicated in the Service Level Agreement (SLA). The goal of the slice enforcement is to reserve dynamically, and for each network slice, the minimum amount of bandwidth in these cells to satisfy the agreed QoS. At the same time, it is requested that *i*) intra-slice interference should be minimized to avoid system performance degradation and *ii*) bandwidth allocation decisions should be executed almost in real time to reduce communication latency [35, 51].

3.2. The proposed Tenant-Driven Solution

The conceived slice enforcement strategy allows the tenant subsystem to estimate, in real-time and slot by slot, the amount of radio resources to allocate to the controlled slice. Thanks to the *Pay for What You Get* paradigm, only the required amount of bandwidth is allocated, so that the radio resources over-provisioning is avoided. At the same time, however, it is requested that allocation decisions must be executed instantaneously to reduce communication latency and avoidable expenses [35]. In this context, the scenario appears as a classical MDP:

- the *environment* is the presented cellular network architecture;
- the changing *state* $\mathbf{s} \in \mathcal{S}$ of the environment is represented by information coming from the Infrastructure Provider subsystem as well as data already available at the tenant subsystem;
- the *action* $a \in \mathcal{A}$ is executed by the tenant subsystem to modify the environment, i.e., to control the RAN;
- the *reward* R is the efficiency of the chosen action a subject to tenant QoS constraints.

Accordingly, the developed solution definitively employs DRL to dynamically adjust the amount of bandwidth to request to the Infrastructure Provider and support the slice enforcement strategy. Note that the interaction between the Infrastructure Provider and tenant permits each tenant to formulate RAN-aware requests, while satisfying the upper bounds negotiated during the preparation phase. Tenants subsystems act as DRL agents that process the features extracted by the autoencoder (and provided by the Infrastructure Provider subsystem, as illustrated before), and optimize their actions. Since DRL provides autonomous decision-making, the resulting system also ensures a high scalability level. Indeed, tenants subsystems can make observations and obtain the best policy

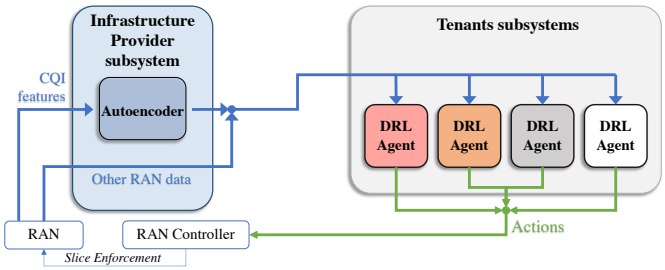

Figure 2: The interaction between Infrastructure Provider and tenants subsystems.

locally without exchanging information among each other. This reduces communication overheads and also improves the security and robustness of the networks [14]. The agent–environment interaction breaks into episodes, that consist of a certain number of time steps, during which the agent selects the action in $\mathcal{A}$. Then, as a consequence of its action, the agent receives the reward R and finds itself in a new state $\mathbf{s}$ [52]. Section 3.4 will provide more details about the implemented DRL framework.

3.3. Design of the Autoencoder used by the Infrastructure Provider subsystem

The autoencoder is a particular Artificial Neural Network (ANN) implementing two key functionalities: the encoder generates the corresponding feature learning representation of input data, while the decoder provides a reconstruction of the input data, starting from the aforementioned feature learning representation.

The input data are spatial snapshots (i.e., matrices) related to the CQI indexes of mobile users, namely $\mathbf{Y} \in \mathbb{R}^{K \times L}$, where K and L are the chosen numbers of rows and columns of the snapshot, respectively. Note that K and L are design parameters. Since the scientific literature clearly states that spatial correlations can be effectively learned by CNNs [15, 53], the investigated encoder is made of three chained 2-dimensional convolutional layers to extract spatial correlations of CQI indexes snapshots [54, 55], as depicted in Fig. 3. Each convolutional layer comprises a set of kernels (or filters) which are convolved with the CQI indexes snapshot for extracting the features of a certain input region and with the Rectified Linear Unit (ReLU) activation function. Each kernel is composed of a number of weights and acts as a filter whose weights are re-used across the entire input. This makes the network connectivity structure sparse: a small set of parameters (i.e., the kernel weights) suffices to map the input into the output leading to a considerably reduced computational complexity with respect to fully connected feed-forward neural networks (without convolutional layers) [54, 55]. Then, two pooling layers follow each convolutional layer to perform down-sampling (i.e., max-pooling picks the maximum value) of intermediate representations, for complexity reduction and overfitting mitigation [15, 56]. The first and the second convolutional layers use N_1 filters $(r_1 \times c_1)$ and N_2 filters $(r_2 \times c_2)$, respectively. Note that N_1 and N_2 represent the number of filters, while $(r_1 \times c_1)$ and $(r_2 \times c_2)$ describe the dimensions of filters, where r_1, c_1 and r_2, c_2 represent the number of rows and columns of the first and the second convolutional layers. In addition, the filters can have diverse strides $[v_1\ h_1]$ and $[v_2\ h_2]$ (where v_1, v_2 and h_1, h_2 represent the vertical and the horizontal step size for the first and the second convolutional layers). Then, a channel-wise normalization with ch_1 channels and ch_2 channels per element is performed for the first and the second convolutional layers, respectively. A typical operation in CNN is indeed the channel normalization for rescaling each channel (whose number determines the depth of the snapshot) into the range of [0,1], thus avoiding vanishing gradients [57]. By receiving as input the snapshot of CQI indexes $\mathbf{Y} \in \mathbb{R}^{K \times L}$, the encoder generates the corresponding feature learning representation vector, namely $\mathbf{f} \in \mathbb{R}^{F}$, with F depending on $(K, L, r_1, c_1, r_2, c_2)$.

The output of the encoder, i.e., the features $\mathbf{f}$ extracted for each input snapshot, is obtained by the third convolutional layer with 1 filter (1×1) and then it is given to tenant subsystems as input for the DRL agents.

Finally, the decoder provides the reconstruction of the CQI indexes, namely $\hat{\mathbf{Y}} \in \mathbb{R}^{K \times L}$, starting from the aforementioned feature learning representation $\mathbf{f}$. By going backwards to input reconstruction, the decoder makes use of two up-sampling layers, corresponding to the two max-pooling layers in the encoder [15, 56], and three convolutional layers, with the ReLU activation function, except for the output layer, which uses the sigmoid activation function [15]. The convolutional layers employ N_2 filters (1×1), N_1 filters $(r_2 \times c_2)$, and 1 filter $(r_1 \times c_1)$, respectively.

All the CQI indexes stored in $\mathbf{Y}$ are normalized within the range [0,1] to accelerate the training convergence [58]. The autoencoder uses weights that are properly configured during the training phase and iteratively updated for each mini-batch of the dataset in order to minimize the Mean Square Error (MSE) loss function. Formally, the MSE loss function is defined as [54, 59]:

$$MSE = \frac{1}{\beta} \sum_{b=1}^{\beta} \sum_{k=1}^{K} \sum_{l=1}^{L} \left(\hat{y}_{b,k,l} - y_{b,k,l} \right)^2, \qquad (1)$$

where β represents the mini-batch size, $y_{b,k,l} \in \mathbf{Y_b}$, and $\hat{y}_{b,k,l} \in \hat{\mathbf{Y}}_\mathbf{b}$.

The encoder is the key building block of the presented Deep Learning architecture because it generates the compressed CQI indexes (i.e., the CQI features $\mathbf{f}$) [48] to be shared with the DRL framework. The decoder, instead, is used to train the autoencoder to return the reconstructed CQI indexes and evaluate the performance of the designed encoder. Other than this analysis, it will not be employed by the DRL framework because at runtime, once the autoencoder is trained, only the encoder part is utilized.

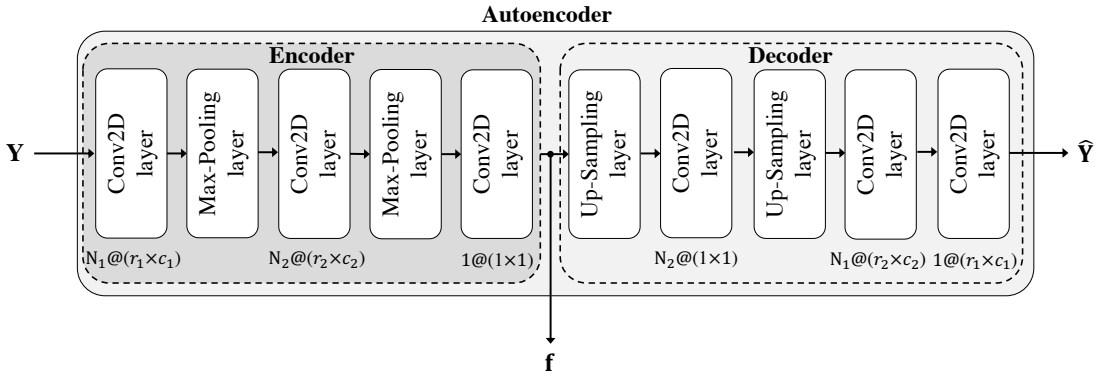

Figure 3: Architecture of the adopted convolutional autoencoder.

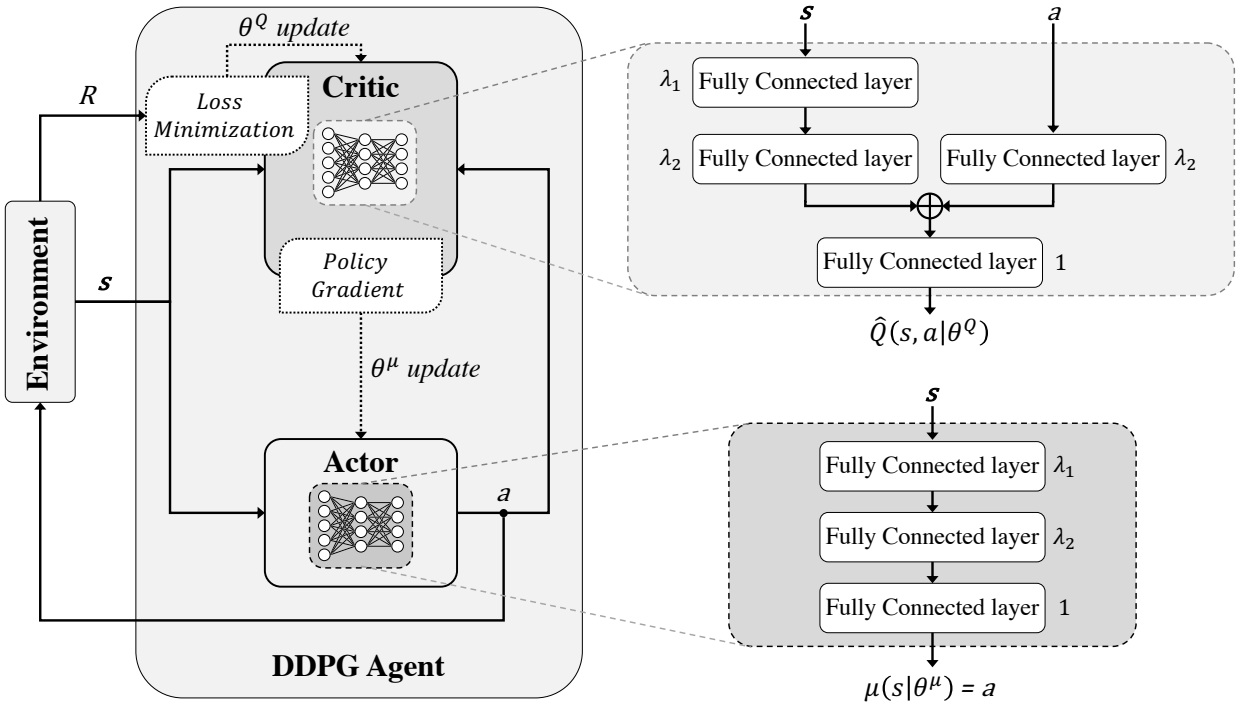

Figure 4: Architecture of the adopted DDPG algorithm.

3.4. Design of the Deep Reinforcement Learning Agents used by the tenant subsystems

As already mentioned, policies based on *Pay for What You Get* paradigm are used by the Infrastructure Provider to prevent the over-provisioning of a tenant. In other words, the Infrastructure Provider associates a unitary cost to each bandwidth resource and determines a maximum amount of bandwidth to use in each cell. The role of the DRL agent of each tenant subsystem is to reserve the minimum amount of bandwidth in each cell to satisfy its QoS requirements, so as to avoid resource over-provisioning. Therefore, the tenant subsystem places its bandwidth allocation requests expressed as a fraction of the maximum available bandwidth within a fixed allocation period.

The *action* $a \in \mathcal{A}$ is the ratio between the amount of bandwidth the tenant requests to the Infrastructure Provider every allocation period and the maximum bandwidth allowable by the Infrastructure Provider. It is a continuous value between 0 and 1 (i.e., 0% and 100% of the bandwidth made available to the tenant). Thus, the Infrastructure Provider and the tenants of different slices negotiate and sign the SLA [60]. Then, the tenant will pay according to the amount of bandwidth: the more the tenant spends the bandwidth, the more the tenant has to pay [61].

The *state* $\mathbf{s} \in \mathcal{S}$ is a vector defined as in the following:

$$\mathbf{s} = (\mathbf{u}, \mathbf{f}, \sigma) \tag{2}$$

where $\mathbf{u} \in \mathbb{N}^W$ is the number of users per sector (W is the number of cell sectors in the system, i.e., portions of the cell served by one of the W co-located base stations),

$\mathbf{f} \in \mathbb{R}^F$ represents the feature learning representation, and $\sigma \in \mathbb{R}$ is the communication Service Availability throughout each episode. In more detail, the component of the feature learning representation $\mathbf{f}_i, \forall i = 1, 2, ..., F$ are the features on radio channel conditions extracted by the autoencoder. The communication Service Availability σ is defined as the percentage value of the amount of time the tenant service is delivered according to the agreed QoS levels, divided by the amount of time the tenant is expected to deliver that service [39]. As anticipated, the agreed QoS refers to the service requirements stipulated between the Infrastructure Provider and tenants through the signed SLA. In line with 3GPP specifications [39], the communication Service Availability can be much lower with respect to reliability.

It is also important to highlight that the details of the radio interface, e.g., the adopted numerology, the scheduling policy, the packet fragmentation rules, and so on, are fully in charge of the Infrastructure Provider and are not known by the tenant agents, which rely only on the compressed version of the radio link conditions of their users (i.e., compressed network status) [47, 62].

Finally, the choice of the *reward* R in a DRL problem is subject to empirical considerations: a good reward function should capture the essence of the problem. In this study, the reward should take into account the amount of bandwidth the tenant subsystem saves with respect to the maximum bandwidth B, provided that the QoS constraints can be satisfied. To elaborate, the *reward* R is computed as:

$$R = \begin{cases} 1 - a, & \text{if the target Service Availability } \sigma \text{ is} \\ & \text{guaranteed;} \\ -1, & \text{otherwise.} \end{cases} \quad (3)$$

Thus, the less the bandwidth requested by the tenant, the higher the reward. The strategy is adopted to terminate the training episode when $R = -1$, i.e., as soon as the tenant subsystem is not providing the service with the target Service Availability.

Among the possible DRL techniques, a deterministic policy gradient algorithm, i.e., the DDPG algorithm, is considered since it is known to be suitable for dealing with continuous states and actions [52, 63]. It is an actor-critic, model-free, off-policy DRL method which computes an optimal policy that maximizes the long-term reward [14, 64]. DDPG primarily uses two neural networks, one for the actor and one for the critic, as illustrated in Fig. 4. The critic evaluates the Q function through an approximation $\hat{Q}(s, a|\theta^Q)$ and the actor models the policy through $\mu(s|\theta^\mu)$. θ^Q and θ^μ are the parameters of the critic and actor networks, respectively. Besides, two duplicates of the actor and critic networks, which are called target networks, are employed to improve the stability during learning, being the target values forced to change slowly. The target critic is identified by $\hat{Q}'(s, a)$ and $\theta^{Q'}$, while $\mu'(s)$ and $\theta^{\mu'}$ are related to the target actor.

The update of the actor and critic networks occurs with the gradient descent method. Specifically, the critic's θ^Q is updated by minimizing the loss L:

$$L = \frac{1}{M} \sum_{i=1}^{M} \left(y_i - \hat{Q}(s_i, a_i|\theta^Q) \right)^2, \quad (4)$$

where M is the number of experiences sampled from the experience replay (i.e., where the agent stores each of its experiences during training) [65], $y_i = R_i + \gamma \hat{Q}'(s_i, \mu'(s'_i|\theta^\mu)|\theta^{Q'})$ is the Q function target approximated through bootstrapping [52], γ is the future reward discount factor [52], and s'_i represents the next observation.

In turn, the actor's θ^μ is updated by following the sampled policy gradient to maximize the expected discounted reward:

$$\nabla_{\theta^\mu} J \approx \frac{1}{M} \sum_{i=1}^{M} \nabla_{\mu(s_i)} \hat{Q} \left(s_i, \mu(s_i)|\theta^Q \right) \nabla_{\theta^\mu} \mu(s_i|\theta^\mu), \quad (5)$$

where J is the environment start distribution as defined in the *policy gradient theorem* [52].

To further elaborate, the *state* $\mathbf{s}$ passes through the first and second fully-connected layers of critic and actor neural networks with λ_1 and λ_2 neurons, respectively, and ReLU activation function. Then, as shown in the right part of Fig. 4, the actor network provides the *action* $\mu(s|\theta^\mu) = a$ as output by using a fully-connected layer with 1 neuron and hyperbolic tangent (i.e., tanh) activation function. The *action* a is also received as input by the critic network and it passes through a fully-connected layer with λ_2 neurons. After adding the processed state, the *expected cumulative long-term reward* $\hat{Q}(s, a|\theta^Q)$ is obtained by the critic network through a fully-connected layer with 1 neuron and ReLU activation function.

3.5. Elastic Resource Scaling

Let B and B_i^{max} the total bandwidth available for the gNB and the maximum bandwidth assignable to the $i-th$ tenant (i.e., the one configured during the RAN slice creation procedure). Furthermore, the bandwidth request formulated by the i-th tenant in a given time instant is simply denoted with B_i. Of course, it holds that $B_i \leq B_i^{max}$. As widely explained before, at every run, the Infrastructure Provider collects tenants' requests, enforces the related RAN slices, and allocates the remaining bandwidth to BE applications. In this regard, although the presence of upper bound values could implicitly prevent diverging decisions, it is possible that in the presence of high loads the total requested bandwidth exceeds the maximum aggregate available bandwidth.
Indeed, in the case $\sum_i B_i > B$, even if $B_i \leq B_i^{max}\ \forall i$, the RAN controller of the Infrastructure Provider will exploit the *elastic resource scaling algorithm*, as reported in [66]. In particular, it will proportionally reduce the amount of

bandwidth to assign to each tenant, by jointly considering each request (that is B_i) and the computed surplus (that is $\sum_i B_i - B$). Note that, in the considered scheme, this is the only mechanism that allows the Infrastructure Provider to accept or scale the requests of tenants. It should be also noted that, in the proposed approach, a per-slice maximum bandwidth that can be used by each tenant is set and it implicitly helps to mitigate the possibility of diverging decisions. Hence, elastic scaling rarely comes into action. On the other hand, elastic scaling is completely transparent to the DRL agents, which place requests and see assigned resources. The fact that sometimes (rarely) the formulated request is not accepted is part of the learning mechanism of the DRL. Indeed, the scaling mechanism may yield a reduction of the actual bandwidth allocation which, in turn, translates in an increased probability of not respecting the target Service Availability σ in equation (3), or, ultimately, in a reduced average Reward. Specifically, the reduced amount of bandwidth to assign to the i-th tenant, namely $\hat{B}_i$, is computed as:

$$\hat{B}_i = B_i - \frac{B_i}{\sum_i B_i} \max\left(\sum_i B_i - B, 0 \right). \tag{6}$$

Note that the elastic resource scaling mechanism can be implemented also in the case the Infrastructure Provider would assign a minimum amount of resources to BE applications. The performances reported in the next Section are always evaluated in the presence of elastic scaling.

4. Performance Evaluation

The performance of the conceived tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is evaluated through computer simulations. To this aim, a system-level simulator of a mobile system is developed in MATLAB, based on the ITU's methodology recommendation [38]. The tool specifically models the downlink transmission. However, similar considerations and results can be obtained for the uplink case. A given number of base stations is placed in a regular grid, following a hexagonal layout. All cell sites consist of 3 sectors, where a configurable number of mobile terminals, or User Equipments (UEs), are dropped independently with a uniform distribution. The UEs, which have a fixed and identical speed with a randomly distributed direction, are attached to the base station able to ensure the highest Signal-to-Interference-plus-Noise Ratio (SINR).

All the links between base stations and UEs in the system are simulated with dynamic channel properties, taking into account the wrap-around configuration in the network layout. The implemented channel modeling considers inter-site interference and large-scale parameters, i.e., pathloss, shadow fading, and Line-Of-Sight (LOS)/Non-Line-Of-Sight (NLOS) propagation condition, according to the International Telecommunication Union (ITU) guidelines [38]. Indeed, the propagation condition is determined by comparing a realization of a random variable with the distance-dependent LOS probability. If the value of the random variable is less than the LOS probability, the simulation considers the UE in a LOS propagation condition. Otherwise, a NLOS propagation condition is taken into account.

Let d_{2D} be the distance between the base station and the UE in km, d_{3D} the 3D distance (including heights in the computation), ω the center frequency in Hz, c the speed of light, H_{gNB} the height of the base station, and H_{UE} the height of the UE. The LOS probability is given by the following:

$$P_{LOS} = \begin{cases} 1, & d_{2D} \leq 18m; \\ P'_{LOS}, & d_{2D} > 18m; \end{cases} \tag{7}$$

where

$$\begin{aligned} P'_{LOS} = &\left[\frac{18}{d_{2D}} + e^{-d_{2D}/63} \left(1 - \frac{18}{d_{2D}} \right) \right] \cdot \\ &\cdot \left(1 + C'(H_{UE}) \frac{5}{4} \left(\frac{d_{2D}}{100} \right)^3 e^{-d_{2D}/150} \right), \end{aligned} \tag{8}$$

with

$$C'(H_{UE}) = \begin{cases} 0, & H_{UE} \leq 13m; \\ \left(\frac{H_{UE}-13}{10} \right)^{1.5}, & 13m < H_{UE} \leq 23m. \end{cases} \tag{9}$$

According to the selected propagation condition (that is LOS or NLOS), a specific pathloss model is applied. In the case of LOS propagation condition, the pathloss model is:

$$PL_{LOS} = \begin{cases} PL_1, & d_{2D} < d_{bp1} \\ PL_2, & d_{2D} > d_{bp1} \end{cases} \tag{10}$$

where $d_{bp1} = 4(H_{gNB} - 1)(H_{UE} - 1)(\omega/c)$ and

$$PL_1 = 28.0 + 22 \log_{10}(d_{3D}) + 20 \log_{10}(\omega), \tag{11}$$

$$\begin{aligned} PL_2 = &\, 40 \log_{10}(d_{3D}) + 28.0 + 20 \log_{10}(\omega) + \\ &- 9 \log_{10} \left({d_{bp1}}^2 + (H_{gNB} - H_{UE})^2 \right). \end{aligned} \tag{12}$$

Otherwise, in the case of NLOS propagation condition, the pathloss is modeled as:

$$PL_{NLOS} = max(PL_{LOS}, PL'_{NLOS}), \tag{13}$$

where

$$\begin{aligned} PL'_{NLOS} = &161.69 + (43.42 - 3.1 \log_{10}(H_{gNB}))(\log_{10}(d_{3D}) - 3) + \\ &+ 20 \log_{10}(\omega) - \left(24.37 - \frac{1480 \log_{10}(H_{gNB})}{H_{gNB}^2} \right) + \\ &- 0.6(H_{UE} - 1.5) - \left((3.2 \log_{10}(17.625)^2 - 4.97 \right). \end{aligned} \tag{14}$$

The shadow fading is modeled as a log-normal random variable, with standard deviation set to 4 dB and 6 dB for LOS and NLOS propagation conditions, respectively.

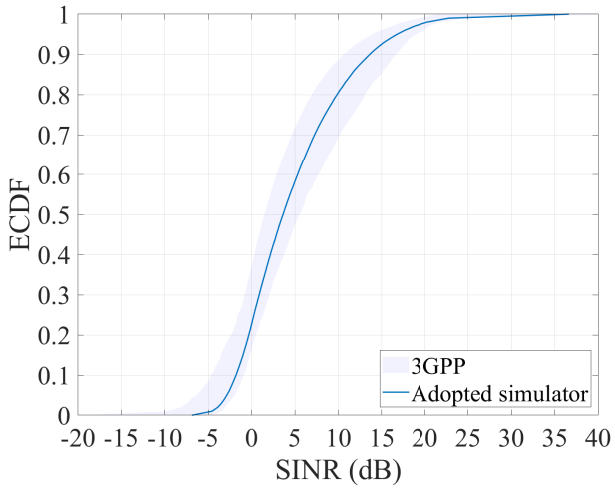

Figure 5: ECDF of the wideband SINR of the developed simulator with respect to 3GPP Phase 1 dense-urban (macro-layer) system-level calibration for multi-antenna systems.

At the application level, the full-buffer traffic model (where the queue depths are assumed to be infinite) is implemented. The user-experienced data rate is derived through the Shannon theorem. Finally, the MAC scheduling strategy enforced by Infrastructure Provider's base stations is Round Robin.

To evaluate the compliance of the developed simulator with 3GPP specifications, Fig. 5 shows the Empirical Cumulative Distribution Function (ECDF) of the wideband SINR experienced by the UEs adopting the developed MATLAB simulator. The reported curve demonstrates that the simulator is well calibrated according to the 3GPP Phase 1 NR MIMO system level calibration for multi-antenna systems [67].

Without loss of generality, the tenant subsystems are assumed to operate two types of network slices, i.e., eMBB and ADAS slices, with real and conceivable network and service settings. In addition to these slices managed by tenants subsystems, there is a third BE slice handled directly by the Infrastructure Provider that use the bandwidth left by the eMBB and ADAS slices. Thus, the performance evaluation is carried out by considering the concurrent presence of 3 different and independent application services. Without loss of generality and validity, the proposed approach can be surely and effectively applied also in more complex scenarios embracing a higher number of tenants.

The speed and density of eMBB slice UEs are set to 3 km/h [38] and 2000 UE/km^2 [39], respectively. As for the QoS requirements for the eMBB slice, the downlink target rate (i.e., metric used as example of the agreed QoS) and the Service Availability are set to $\mathcal{T} = 50$ Mbps [39] and $\sigma = 90\%$, respectively, according to the ITU Dense Urban eMBB deployment [38]. Instead, for the ADAS slice, in line with the ITU Urban Macro deployment [38], the speed and density of UEs are set to 30 km/h [38] and 1200 UE/km^2 [40], respectively, and the downlink target rate and the Service Availability are set to $\mathcal{T} = 400$ kbps [40] and $\sigma = 99\%$, respectively. Note that in the ADAS scenario more than 99% (e.g., 99.999%) availability can only be achieved through specific additional mechanisms (e.g., multi-connectivity with packet duplication across multiple links), which however are out of scope of this work. Then, the full-buffer traffic model is implemented for both the scenarios [38]. Table 2 summarizes the parameter settings.

Table 2: Types of network slices operated by tenant subsystems.

	eMBB	**ADAS**
Network deployment	ITU Dense Urban [38]	ITU Urban Macro [38]
UE speed	3 km/h [38]	30 km/h [38]
UE density	2000 UE/km^2 [39]	1200 UE/km^2 [40]
Downlink target rate $\mathcal{T}$	50 Mbps [39]	400 kbps [40]
Service Availability σ	90%	99%
Traffic model	Full-buffer [38]	

4.1. Performance of the Autoencoder used by the Infrastructure Provider subsystem

The autoencoder, used by the Infrastructure Provider subsystem to compress the network status, leverages data related to the radio channel conditions. Specifically, a dataset generated by the implemented MATLAB simulator in compliance with 3GPP specifications is used. For both the scenarios reported in Table 2, the adopted dataset consists of 10000 realizations reporting the CQI indexes. Each realization is a snapshot with $K \times L = 3 \times 30 = 90$ CQI values for each base station: if the number of attached UEs is greater than 90, only the worst 90 values are included; if the number of attached UEs is less than 90, an appropriate padding is performed. As anticipated in Section 3.3, a convolutional autoencoder is adopted because of spatial snapshots as input data. In fact, the scientific literature already demonstrates that in the analyzed context this type of autoencoder has greater performance than a simple flat autoencoder. The latter is not specialized in modeling spatial data (e.g., images) [15] because it cannot extract spatial correlations of the analyzed CQI indexes snapshots. Moreover, any increase in the number of trainable parameters of CNNs is not significant [55], especially compared to the advantageous performance in terms of loss.

Different configurations of convolutional autoencoders, characterized by different values of parameters listed in Section 3.3, are investigated for identifying the suitable configuration to be used in the DRL framework. In particular, different numbers N_1 and N_2, dimensions $r_1 \times c_1$ and $r_2 \times c_2$, and strides $[v_1\ h_1]$ and $[v_2\ h_2]$ of filters for the first and the second convolutional layer are analyzed (please see Table 3 for further details). Also the channel-wise normalization is performed with diverse ch_1 channels for the first convolutional layer, while ch_2 for the second

convolutional layer is omitted because it is always set equal to 1. Note that the dimension of the feature learning representation vector $\mathbf{f}$ is the same for all the configurations in Table 3, that is $F = 6$.

The training set, whose performance is listed and evaluated in Table 2, the validation set, and the test set consist of 70%, 15%, and 15% of the adopted dataset, respectively. The training phase, during which weights are iteratively updated in order to minimize the MSE loss function, is done with 100 epochs (i.e., complete passes through the training data [68] such that each example has been seen once) for all the evaluated configurations of convolutional autoencoders. The Adam optimization [69], with a learning rate equal to 0.01, is used to iteratively update the network weights. The performance is investigated in terms of training Root Mean Square Error (RMSE) and the number of trainable parameters (i.e., quantitative insights on the expected/achieved computational complexity), that refer to the training phase. The RMSE represents the root of the MSE, as defined in (1), and allows a better understanding of resulting values. The number of trainable parameters measures the complexity of selected learning architectures: the higher the number of parameters, the higher the complexity level. Note that the RMSE gives the reconstruction performance of the whole autoencoder, even if the CQI reconstruction is not the focus of this work. However, if an autoencoder is able to well reconstruct the input, it means that its single blocks (i.e., encoder and decoder) have high performance.

Obtained results are reported in Table 3 for all the evaluated configurations of convolutional autoencoders. The second-last configuration, which is highlighted in Table 3, represents a good trade-off of performed loss and complexity. As a consequence, the rest of the presented work considers this configuration as the best one of the proposed autoencoder used by the Infrastructure Provider subsystem. Then, its compressed CQI feature learning representation is passed to the DRL agents employed by the tenant subsystems.

Once the best autoencoder configuration is selected, the convergence analysis evaluates the performance of the Deep Learning architecture as a function of the number of epochs considered during the training phase. Fig. 6 shows the autoencoder loss as a function of the number of epochs for the training set and the validation set. The reported curves confirm that the selected convolutional autoencoder fastly converges to stable values, without underfitting nor overfitting after the training phase, and does not need a long training period.

Finally, Fig. 7 reports the reconstruction errors on the test set with the relative frequency. It is evident that the selected configuration of the convolutional autoencoder reconstructs data with very high accuracy during the test phase. In fact, the reconstruction with an overestimation/underestimation of more than 2 CQI indexes occurs with a relative frequency always lower than 0.01.

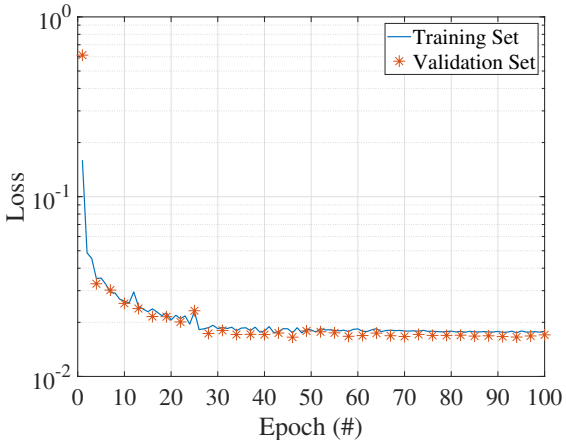

Figure 6: Autoencoder loss (i.e., MSE) vs number of epochs.

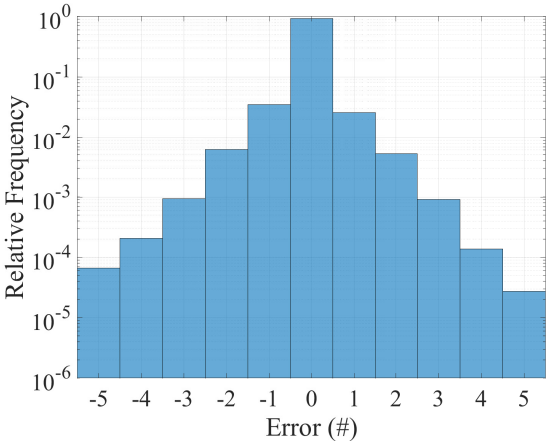

Figure 7: Relative frequency of the reconstruction errors on the test set.

4.2. Performance of the Deep Reinforcement Learning Agents used by the Tenant subsystems

The performance of the DRL agents used by the tenant subsystems is evaluated through the calibrated simulator, by employing its dynamic capabilities. Specifically, a DDPG algorithm is implemented by tenant subsystems. Agents perform their actions every second (i.e., with an allocation period of 1s), but all the time slots included therein are still simulated. Indeed, during this period users move slot by slot and the radio channel quality they experience changes accordingly, hence modifying the user-experienced data rate, which is derived through the Shannon theorem. Two different tenant subsystems are taken into account. They are assumed to provide eMBB services and ADAS services, whose downlink target rate $\mathcal{T}$ and Service Availability σ are reported in Table 2). As anticipated in Section 3.4, in line with 3GPP specifications [39], the values of the Service Availability are not the values of reliability and specifically they are lower. Each episode lasts 1 s, i.e., the DRL agent performs its actions every second. This means that the tenant subsystems can update their

Table 3: Performance of the different configurations of convolutional autoencoders.

Configuration											**Performance**	
Filters										**Channel-wise normalization**	**Training RMSE**	**Number of trainable parameters**
Number		Dimension				Stride						
N_1	N_2	r_1	c_1	r_2	c_2	v_1	h_1	v_2	h_2	ch_1		
200	100	3	3	1	3	4	4	1	1	2	1.2012	123202
200	100	3	2	1	6	4	4	1	1	2	1.2032	243202
200	100	3	2	1	5	4	2	3	2	2	0.1277	203202
200	100	3	2	1	5	4	2	3	2	3	0.1384	203202
300	150	3	2	1	5	4	2	3	2	2	0.1295	454802
400	200	3	3	1	5	3	3	1	1	2	0.1237	808802
200	**100**	**3**	**3**	**1**	**5**	**3**	**3**	**1**	**1**	**2**	**0.1188**	**204402**
200	100	3	3	1	5	3	3	1	1	3	0.1195	204402

bandwidth allocation requests every second (i.e., with an allocation period of 1 s). Each tenant subsystem may request a maximum bandwidth B_i^{max} of 100 MHz for every considered sector. Moreover, the total bandwidth is 200 MHz and the maximum bandwidth that can be assigned to tenants following the elastic resource scaling described in Section 3.5 is $B = 150$ MHz, i.e., the minimum bandwidth allocated to BE traffics is 50 MHz.

As for the actor and critic neural networks, the learning rate is set equal to 0.001 and 0.0001, respectively; the number of neurons is set to $\lambda_1 = 2000$ and $\lambda_2 = 1500$. The number of training epochs, each corresponding to 100 training episodes, is set equal to 50.

Fig. 8 shows the achieved reward as a function of the number of training epochs for the two analyzed scenarios. Each point is the average reward obtained during the related epoch (i.e., 1 epoch = 100 episodes). It is possible to observe that both the DRL agents fastly learn the policy from the state: the average reward in eMBB and ADAS scenarios converges to stable values after 20 and 15 training epochs, respectively. Thus, 50 training epochs are sufficient for convergence.

Fig. 9 shows the ECDF of the execution time of the trained DDPG agents over 250000 samples (i.e., 250000 different actions). First of all, it is important to note that the test has been conducted on a general-purpose computer (Quad-Core Intel Core i5-8259U CPU @ 2.30GHz with 2x4GB LPDDR3 @2133 MHz RAM). Nonetheless, the plot shows that the execution time is less than 1 ms in 90% of the samples, hence demonstrating an extremely reduced computational complexity. Obviously, even better performance is expected on more powerful machines.

To deeply analyze the performance of the proposed framework, the proposed approach based on the DDPG algorithm is compared with different conventional resource allocation methods, which are implemented with the same parameter settings:

- *Genie*, that corresponds to the optimal allocation of bandwidth for each slice, i.e., the minimum amount of bandwidth that guarantees $\sigma = 100\%$ as Service Availability. It represents the upper-bound performance of any non-tenant-driven approach and therefore it is exploited as the main benchmark for the proposal. It is important to note that the bandwidth, in this case, is determined through iterative adjustments

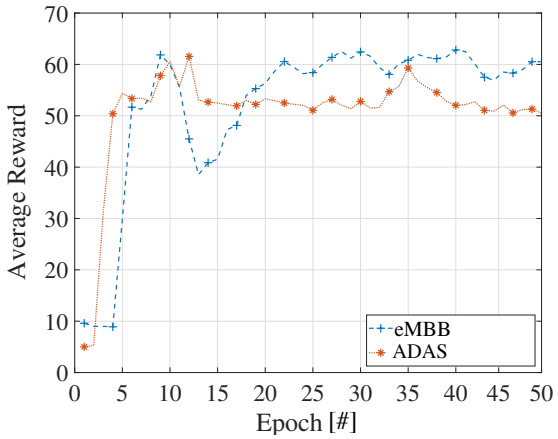

Figure 8: Average episode reward vs number of epoch (with 1 epoch corresponding to 100 training episodes) for eMBB and ADAS scenarios.

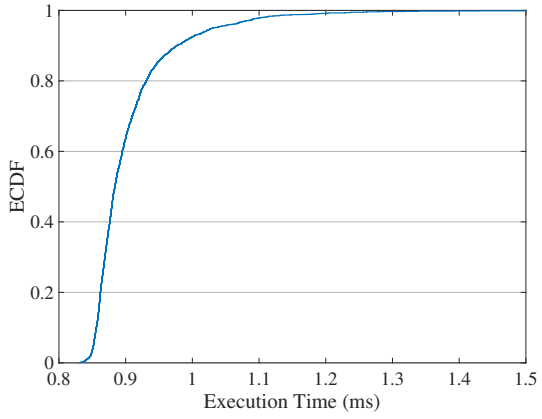

Figure 9: ECDF of the execution time of the trained DDPG agents.

during simulations. Therefore, the Genie approach is unfeasible in actual deployments;

- *Random*, i.e., the bandwidth allocated to each slice is randomly chosen between 10% and 90% of the maximum bandwidth B_s.

- *Heuristic*, which represents the dynamic allocation of bandwidth. Specifically, the bandwidth is proportional to the highest number of UEs in a sector (that is an information available in the state $\mathbf{s}$). In particular, for each scenario (i.e., eMBB and ADAS use cases), the action a is computed at each step according to the following:

$$a = \min\left\{\left(\max_{1\leq i\leq W} u_i \cdot \frac{\mathcal{T}}{B\log_2(1+SINR_p)}\right), 1\right\} \quad (15)$$

where u_i is the number of UEs in the $i-th$ sector, $\mathcal{T}$ is the downlink target rate, and $SINR_p$ is a specific percentile p of the SINR distribution. In the following, $p = 5\%$, namely the cell-edge SINR [70], and $p = 50\%$, namely the median SINR [70], are considered. Thus, the Heuristic approach allocates a bandwidth which is proportional to the maximum number of UEs per sector with two different choices for the proportionality factor: *Heuristic ($p = 5\%$)* represents a worst-case situation calibrated for mobile users at the cell edge, whereas *Heuristic ($p = 50\%$)* is calibrated for median users.

The performance is investigated in terms of Episode Availability Indicator $\mathcal{E}$ and bandwidth saved with respect to the *Genie* (that represents the optimal bandwidth for 100% of communication Service Availability σ). The Episode Availability Indicator $\mathcal{E}$ is defined as:

$$\mathcal{E} = \frac{1}{N_E}\sum_{i=1}^{N_E}\epsilon_i \cdot 100 \quad (16)$$

where N_E is the number of test episodes and ϵ_i is related to the target Service Availability σ, that is:

$$\epsilon_i = \begin{cases} 1, & \text{if the Service Availability } \sigma \text{ is guaranteed} \\ & \text{in the } i-th \text{ test episode, } \forall i; \\ 0, & \text{otherwise.} \end{cases} \quad (17)$$

Therefore, the Episode Availability Indicator $\mathcal{E}$ is the percentage value of the number of test episodes the service of the tenant subsystem is delivered according to the agreed Service Availability σ, divided by the total number of test episodes N_E. In other words, the Episode Availability Indicator shows how the different bandwidth allocation strategies perform during the test phase, indicating the number of test episodes the service is correctly delivered according to the agreed availability. Note that the total number of test episodes N_E is set to 500, that is 500 different simulations.

Table 4: Episode Availability Indicators $\mathcal{E}$ for the analyzed approaches.

	$\mathcal{E}$ (%)	
	eMBB	**ADAS**
Random	74	0
Heuristic ($p = 5\%$)	100	32
Heuristic ($p = 50\%$)	100	0
DDPG	100	100

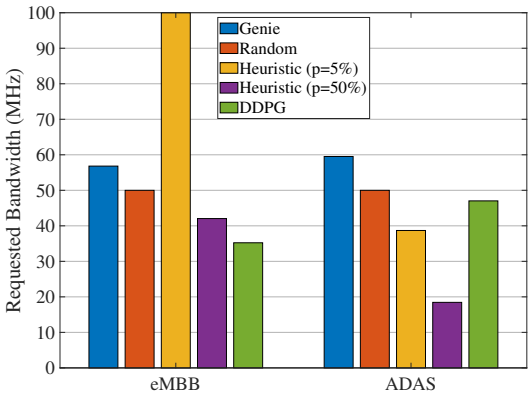

Figure 10: Average bandwidth requested by eMBB and ADAS tenant subsystems.

Table 4 reports the Episode Availability Indicators $\mathcal{E}$ performed by the analyzed approaches for eMBB and ADAS scenarios, respectively. As a first observation, it is worth noting that the proposed approach based on the DDPG algorithm always guarantees the 100% of Episode Availability Indicator $\mathcal{E}$ (being the only one that never fails for all the 500 test episodes), i.e., it allows to always provide the service with 90% and 99% Service Availability σ for eMBB and ADAS cases, respectively. Specifically, in the eMBB scenario, the Episode Availability Indicator $\mathcal{E}$ equal to 100% can also be obtained by both the *Heuristic* approaches (with $p = 5\%$ and $p = 50\%$), while it is never achieved by the *Random* approach. In the ADAS scenario, only the proposed solution based on the DDPG algorithm has the Episode Availability Indicator $\mathcal{E}$ equal to 100%, which far exceeds those of the other approaches.

Fig. 10 shows the percentage of bandwidth savings performed by the analyzed approaches for both scenarios. In the eMBB scenario, the proposed approach based on the DDPG algorithm saves the highest amount of bandwidth (i.e., around 40%) with respect to the other approaches. Note that, in this case, the *Heuristic ($p = 5\%$)* approach performs worse than the *Genie*, by requiring a greater amount (i.e., almost 80%) of bandwidth.

In the case of ADAS, the proposed approach based on the DDPG algorithm does not ensure the highest bandwidth saving: the bandwidth saving of the DDPG-based approach is the lowest one (i.e., 20%), except for the *Random* approach. However, as anticipated, only the proposed solution based on the DDPG algorithm has the Episode

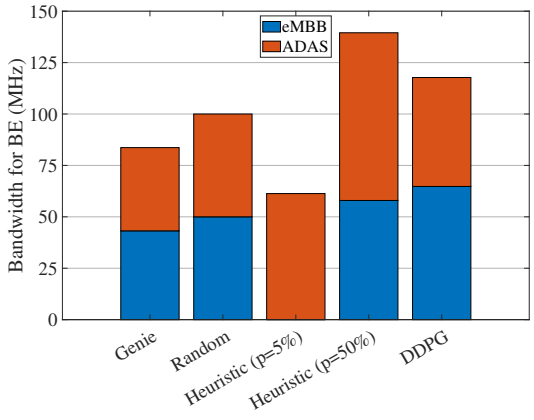

Figure 11: Average bandwidth left free by eMBB and ADAS tenant subsystems and allocated to the BE slice.

Availability Indicator $\mathcal{E}$ equal to 100%. Thus, it can be considered as the winning approach also for this scenario.

Fig. 11 shows the average bandwidth left free by eMBB and ADAS tenant subsystems and allocated to the BE slice. It further confirms the greater effectiveness of the DDPG algorithm, that saves a high amount of bandwidth to be allocated to BE applications. It is not the higher amount of bandwidth, which refers to *Heuristic* ($p = 50\%$), but the latter has the Episode Availability Indicator $\mathcal{E}$ equal to 0% for ADAS.

To sum up, the DRL agents used by the tenant subsystems, which implement the DDPG algorithm, actually learn to save bandwidth (to be allocated to BE applications), while always ensuring the Service Availability and avoiding the bandwidth over-provisioning in contrast to the *Genie*. Overall, the proposed approach outperforms other conventional strategies also because it can be intelligently and flexibly tuned on the required Service Availability of the tenant subsystem during training, as demonstrated by the results of the both scenarios. Thus, the bandwidth requested for offering services and respecting the target Service Availability could be optimally allocated according to the *Pay for What You Get* paradigm.

5. Conclusions and Future works

This work presented a novel tenant-driven Radio Access Network slicing enforcement scheme based on Pervasive Intelligence. At the basis of the proposed solution, there is the idea that the tenant dynamically decides the amount of bandwidth to assign to its slice, based on the *Pay for What You Get* paradigm at the Radio Access Network. The Infrastructure Provider supports this activity by exploiting a Deep Learning scheme (i.e., convolutional autoencoder) to compress network status and share it with tenants. In turn, each tenant implements a Deep Reinforcement Learning algorithm (i.e., Deep Deterministic Policy Gradient) to dynamically adapt bandwidth requests. Finally, the resulting outcomes are employed by the Infrastructure Provider to effectively enforce the Radio Access Network slicing. The comparison with conventional resource allocation methods demonstrated that the proposed approach ensures the best trade-off between bandwidth savings and bandwidth over-provisioning, while always guaranteeing the target Service Availability. Future research activities will further extend the presented study by predicting the network status, in addition to its compression, to boost the Pervasive Intelligence presence in the network. Moreover, solutions based on distributed learning will further enhance the privacy preservation, as well as the independence of tenant actions in multiple slices scenarios. Further investigations on the interactions between the decisions taken by the Infrastructure Provider (i.e., elastic scaling) and the actions coming from tenants' Deep Reinforcement Learning agents will be conducted, as well as the investigations and evaluations in more complex scenarios embracing a higher number of tenants (since the benefits provided by the proposed approach are always valid and important, independently of the number of tenants operating). In addition, other relevant concerns that affect Radio Access Network slicing (especially for emerging latency-sensitive services) will be addressed in the future. A comprehensive architecture enabling Radio Access Network slicing should be designed by considering different time scales for slice generation in the B5G heterogeneous context. At the same time, granularity constraints in spectrum- and radio-level resource sharing are primary issues. In order to enhance the performance and maximize the flexibility, appropriate APIs between the Infrastructure Provider and tenants will be investigated as well.

Acknowledgment

This work was supported by the Italian MIUR PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges", by the Italian MISE project entitled "5G experimentation Program in Milan, Italy" (also supported by Vodafone Italia S.p.A.), and by the Apulia Region (Italy) Research project INTENTO (36A49H6). It has been also partially supported by the Italian MIUR PON projects Pico&Pro (ARS01_01061), AGREED (ARS01_00254), FURTHER (ARS01_01283), and RAFAEL (ARS01_00305).

References

[1] S. Zhang, An overview of network slicing for 5G, IEEE Wireless Communications 26 (2019) 111–117.

[2] R. Khan, P. Kumar, D. N. K. Jayakody, M. Liyanage, A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions, IEEE Communications Surveys & Tutorials 22 (2020) 196–248.

[3] M. Maule, J. Vardakas, C. Verikoukis, 5G RAN Slicing: Dynamic Single Tenant Radio Resource Orchestration for eMBB Traffic within a Multi-Slice Scenario, IEEE Communications Magazine 59 (2021) 110–116.

[4] A. A. Barakabitze, A. Ahmad, R. Mijumbi, A. Hines, 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges, Computer Networks 167 (2020) 106984.

[5] J. Mei, X. Wang, K. Zheng, An intelligent self-sustained RAN slicing framework for diverse service provisioning in 5G-beyond and 6G networks, Intelligent and Converged Networks 1 (2020) 281–294.

[6] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, Y.-D. Lin, Security and privacy for 6G: A survey on prospective technologies and challenges, IEEE Communications Surveys & Tutorials 23 (2021) 2384–2428.

[7] S. Martiradonna, A. Abrardo, M. Moretti, G. Piro, G. Boggia, Architecting RAN slicing for URLLC: Design decisions and open issues, in: 2019 IEEE/ACM 23rd International Symposium on Distributed Simulation and Real Time Applications (DS-RT), IEEE, 2019, pp. 1–4.

[8] X. Zhang, B. Li, J. Peng, X. Pan, Z. Zhu, You Calculate and I Provision: A DRL-Assisted Service Framework to Realize Distributed and Tenant-Driven Virtual Network Slicing, Journal of Lightwave Technology 39 (2021) 4–16.

[9] X. Zhang, W. Lu, B. Li, Z. Zhu, Drl-based network orchestration to realize cooperative, distributed and tenant-driven virtual network slicing, in: 2019 Asia Communications and Photonics Conference (ACP), 2019, pp. 1–3.

[10] S. E. Elayoubi, S. B. Jemaa, Z. Altman, A. Galindo-Serrano, 5G RAN slicing for verticals: Enablers and challenges, IEEE Communications Magazine 57 (2019) 28–34.

[11] M. E. Morocho-Cayamcela, H. Lee, W. Lim, Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions, IEEE Access 7 (2019) 137184–137206.

[12] Y. Chen, W. Liu, Z. Niu, Z. Feng, Q. Hu, T. Jiang, Pervasive intelligent endogenous 6g wireless systems: Prospects, theories and key technologies, Digital Communications and Networks 6 (2020) 312–320.

[13] S. Zhang, D. Zhu, Towards artificial intelligence enabled 6g: State of the art, challenges, and opportunities, Computer Networks (2020) 107556.

[14] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, D. I. Kim, Applications of Deep Reinforcement Learning in Communications and Networking: A Survey, IEEE Communications Surveys & Tutorials 21 (2019) 3133–3174.

[15] C. Zhang, P. Patras, H. Haddadi, Deep Learning in Mobile and Wireless Networking: A Survey, IEEE Communications Surveys & Tutorials 21 (2019) 2224–2287.

[16] F. B. Mismar, B. L. Evans, A. Alkhateeb, Deep reinforcement learning for 5G networks: Joint beamforming, power control, and interference coordination, IEEE Transactions on Communications 68 (2019) 1581–1592.

[17] C. Gutterman, E. Grinshpun, S. Sharma, 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, 2019, pp. 231–240.

[18] S. Bakri, P. A. Frangoudis, A. Ksentini, M. Bouaziz, Data-Driven RAN Slicing Mechanisms for 5G and Beyond, IEEE Transactions on Network and Service Management 18 (2021) 4654–4668.

[19] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, X. Costa-Perez, DeepCog: Optimizing resource provisioning in network slicing with AI-based capacity forecasting, IEEE Journal on Selected Areas in Communications 38 (2019) 361–376.

[20] M. R. Raza, C. Natalino, P. Öhlen, L. Wosinska, P. Monti, Reinforcement learning for slicing in a 5G flexible RAN, Journal of Lightwave Technology 37 (2019) 5161–5169.

[21] R. Li, C. Wang, Z. Zhao, R. Guo, H. Zhang, The LSTM-Based Advantage Actor-Critic Learning for Resource Management in Network Slicing With User Mobility, IEEE Communications Letters 24 (2020) 2005–2009.

[22] Y. Hua, R. Li, Z. Zhao, X. Chen, H. Zhang, GAN-Powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing, IEEE Journal on Selected Areas in Communications 38 (2019) 334–349.

[23] B. Khodapanah, A. Awada, I. Viering, A. N. Barreto, M. Simsek, G. Fettweis, Slice management in radio access network via deep reinforcement learning, in: 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), IEEE, 2020, pp. 1–6.

[24] X. Chen, Z. Zhao, C. Wu, M. Bennis, H. Liu, Y. Ji, H. Zhang, Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach, IEEE Journal on Selected Areas in Communications 37 (2019) 2377–2392.

[25] H. D. R. Albonda, J. Pérez-Romero, An efficient RAN slicing strategy for a heterogeneous network with eMBB and V2X services, IEEE access 7 (2019) 44771–44782.

[26] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, H. Mineno, Flexible Resource Block Allocation to Multiple Slices for Radio Access Network Slicing Using Deep Reinforcement Learning, IEEE Access 8 (2020) 68183–68198.

[27] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, X. Shen, Joint ran slicing and computation offloading for autonomous vehicular networks: A learning-assisted hierarchical approach, IEEE Open Journal of Vehicular Technology (2021).

[28] Z. Wang, Y. Wei, F. R. Yu, Z. Han, Utility Optimization for Resource Allocation in Edge Network Slicing Using DRL, in: GLOBECOM 2020-2020 IEEE Global Communications Conference, IEEE, 2020, pp. 1–6.

[29] W. Wu, N. Chen, C. Zhou, M. Li, X. Shen, W. Zhuang, X. Li, Dynamic RAN Slicing for Service-Oriented Vehicular Networks via Constrained Learning, IEEE Journal on Selected Areas in Communications (2020).

[30] J. Mei, X. Wang, K. Zheng, G. Boudreau, A. B. Sediq, H. Abouzeid, Intelligent radio access network slicing for service provisioning in 6g: A hierarchical deep reinforcement learning approach, IEEE Transactions on Communications (2021).

[31] W. Guan, H. Zhang, V. C. Leung, Slice Reconfiguration Based on Demand Prediction with Dueling Deep Reinforcement Learning, in: GLOBECOM 2020-2020 IEEE Global Communications Conference, IEEE, 2020, pp. 1–6.

[32] R. Li, Z. Zhao, Q. Sun, I. Chih-Lin, C. Yang, X. Chen, M. Zhao, H. Zhang, Deep reinforcement learning for resource management in network slicing, IEEE Access 6 (2018) 74429–74441.

[33] F. Mason, G. Nencioni, A. Zanella, Using Distributed Reinforcement Learning for Resource Orchestration in a Network Slicing Scenario, IEEE/ACM Transactions on Networking (2022) 1–15.

[34] Q.-V. Pham, F. Fang, V. N. Ha, M. J. Piran, M. Le, L. B. Le, W.-J. Hwang, Z. Ding, A surveou of multi-access edge computing in 5G and Beyond: Fundamentals, technology integration, and state-of-the-art, IEEE Access 8 (2020) 116974–117017.

[35] S. D'Oro, F. Restuccia, T. Melodia, Toward operator-to-waveform 5g radio access network slicing, IEEE Communications Magazine 58 (2020) 18–23.

[36] X. Zhou, R. Li, T. Chen, H. Zhang, Network slicing as a service: enabling enterprises' own software-defined cellular networks, IEEE Communications Magazine 54 (2016) 146–153.

[37] S. Martiradonna, A. Abrardo, M. Moretti, G. Piro, G. Boggia, Deep reinforcement learning-aided RAN slicing enforcement supporting latency sensitive services in B5G networks, Internet Technology Letters 4 (2021) e328.

[38] ITU-R, Guidelines for evaluation of radio interface technologies for IMT-2020, Report M.2412, ITU Radiocommunication Sector, 2017.

[39] 3GPP, Service requirements for the 5G system; Stage 1 (Release 17), Technical Specification (TS) 22.261, 3rd Generation Partnership Project, 2019. V17.1.0.

[40] 5GAA Automotive Association, C-V2X Use Cases Volume II: Examples and Service Level Requirements), White Paper (2020).

[41] F. Jiang, K. Wang, L. Dong, C. Pan, K. Yang, Stacked Autoencoder-Based Deep Reinforcement Learning for Online Resource Scheduling in Large-Scale MEC Networks, IEEE Internet of Things Journal 7 (2020) 9278–9290.

[42] O. U. Akgul, I. Malanchini, A. Capone, Dynamic resource trading in sliced mobile networks, IEEE Transactions on Network and Service Management 16 (2019) 220–233.
[43] 3GPP, 5G; Management and orchestration; Architecture framework (Release 16), Technical Specification (TS) 28.533, 3rd Generation Partnership Project, 2021. V16.7.0.
[44] M. Polese, L. Bonati, S. D'Oro, S. Basagni, T. Melodia, Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges, arXiv preprint arXiv:2202.01032 (2022).
[45] 3GPP, Study on management and orchestration of network slicing for next generation network (Release 15), Technical Report (TR) 22.261, 3rd Generation Partnership Project, 2018. V15.1.0.
[46] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, X. Costa-Perez, Overbooking network slices through yield-driven end-to-end orchestration, in: Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies, CoNEXT '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 353–365.
[47] 3GPP, 5G; NR; Physical layer procedures for data (Release 15), Technical Specification (TS) 38.214, 3rd Generation Partnership Project, 2018. V15.3.0.
[48] D. Gündüz, P. de Kerret, N. D. Sidiropoulos, D. Gesbert, C. R. Murthy, M. van der Schaar, Machine Learning in the Air, IEEE Journal on Selected Areas in Communications 37 (2019) 2184–2199.
[49] A. Rago, G. Piro, G. Boggia, P. Dini, Multi-Task Learning at the Mobile Edge: an Effective Way to Combine Traffic Classification and Prediction, IEEE Transactions on Vehicular Technology 9 (2020) 10362–10374.
[50] M. Abbasi, A. Shahraki, A. Taherkordi, Deep Learning for Network Traffic Monitoring and Analysis (NTMA): ASurvey, Computer Communications (2021).
[51] F. Debbabi, R. Jmal, L. Chaari, R. L. Aguiar, An overview of inter-slice and intra-slice resource allocation in b5g telecommunication networks, IEEE Transactions on Network and Service Management (2022) 1–1.
[52] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, Cambridge, MA: MIT Press, 2018.
[53] A. Rago, G. Piro, G. Boggia, P. Dini, Anticipatory Allocation of Communication and Computational Resources at the Edge Using Spatio-Temporal Dynamics of Mobile Users, IEEE Transactions on Network and Service Managementt 18 (2021) 4548–4562.
[54] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Cambridge, MA, USA, 2016.
[55] H. D. Trinh, A. F. Gambin, L. Giupponi, M. Rossi, P. Dini, Mobile Traffic Classification through Physical Channel Fingerprinting: a Deep Learning Approach, IEEE Transactions on Network and Service Management 18 (2020) 1946–1961.
[56] G. Aceto, D. Ciuonzo, A. Montieri, A. Pescapé, Mobile Encrypted Traffic Classification Using Deep Learning: Experimental Evaluation, Lessons Learned, and Challenges, IEEE Transactions on Network and Service Management 16 (2019) 445–458.
[57] Z. Dai, R. Heckel, Channel normalization in convolutional neural network avoids vanishing gradients, arXiv preprint arXiv:1907.09539 (2019).
[58] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436.
[59] S. Hashem, B. Schmeiser, Improving model accuracy using optimal linear combinations of trained neural networks, IEEE Transactions on neural networks 6 (1995) 792–794.
[60] 3GPP, 5G; Management and orchestration; Concepts, use cases and requirements (Release 15), Technical Specification (TS) 28.530, 3rd Generation Partnership Project, 2020. V15.3.0.
[61] M. A. Habibi, B. Han, M. Nasimi, H. Schotten, The Structure of Service Level Agreement of Slice-based 5G Network, 2018.
[62] 5G PPP, 5G PPP architecture working group: View on 5G architecture, White Paper (2020). V3.0.
[63] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971 (2015).
[64] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, D. Meger, Deep reinforcement learning that matters, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.
[65] D. Zha, K.-H. Lai, K. Zhou, X. Hu, Experience replay optimization, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 4243–4249.
[66] S. Martiradonna, G. Cisotto, G. Boggia, G. Piro, L. Vangelista, S. Tomasin, Cascaded WLAN-FWA Networking and Computing Architecture for Pervasive In-Home Healthcare, IEEE Wireless Communications 28 (2021) 92–99.
[67] 3GPP, Evaluation assumptions for Phase 1 NR MIMO system level calibration, TSG RAN WG1 Meeting R1-1701824, 3rd Generation Partnership Project, 2017.
[68] J. G. Carney, P. Cunningham, The epoch interpretation of learning, IEEE Transaction on Neural Networks 8 (1998) 111–116.
[69] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the International Conference on Learning Representations (ICLR), 2015, pp. 1–15.
[70] M. Shi, K. Yang, Z. Han, D. Niyato, Coverage analysis of integrated sub-6GHz-mmWave cellular networks with hotspots, IEEE Transactions on Communications 67 (2019) 8151–8164.