Politecnico di Bari

Department of Electrical and Information Engineering
ELECTRICAL AND INFORMATION ENGINEERING
Ph.D. Program
SSD: ING-INF/03–TELECOMMUNICATIONS

Final Dissertation

Analysis and Optimal Management of Mobile Networks Towards Pervasive Intelligence

---

by

Arcangela Rago:

Supervisor:

Prof. Gennaro Boggia

Coordinator of Ph.D. Program:

Prof. Mario Carpentieri

Course n°34, 01/11/2018-31/10/2021

Politecnico di Bari

LIBERATORIA PER L'ARCHIVIAZIONE DELLA TESI DI DOTTORATO

Al Magnifico Rettore
del Politecnico di Bari

La sottoscritta RAGO ARCANGELA nata a TERLIZZI il 30/08/1994

residente a GIOVINAZZO in via SEBENICO 2 e-mail arcangelarago94@gmail.com

iscritto al 3° anno di Corso di Dottorato di Ricerca in INGEGNERIA ELETTRICA E DELL'INFORMAZIONE ciclo XXXIV

ed essendo stato ammesso a sostenere l'esame finale con la prevista discussione della tesi dal titolo:

Analysis and Optimal Management of Mobile Networks Towards Pervasive Intelligence

DICHIARA

1) di essere consapevole che, ai sensi del D.P.R. n. 445 del 28.12.2000, le dichiarazioni mendaci, la falsità negli atti e l'uso di atti falsi sono puniti ai sensi del codice penale e delle Leggi speciali in materia, e che nel caso ricorressero dette ipotesi, decade fin dall'inizio e senza necessità di nessuna formalità dai benefici conseguenti al provvedimento emanato sulla base di tali dichiarazioni;
2) di essere iscritto al Corso di Dottorato di ricerca in Ingegneria Elettrica e dell'informazione ciclo XXXIV, corso attivato ai sensi del "*Regolamento dei Corsi di Dottorato di ricerca del Politecnico di Bari"*, emanato con D.R. n.286 del 01.07.2013;
3) di essere pienamente a conoscenza delle disposizioni contenute nel predetto Regolamento in merito alla procedura di deposito, pubblicazione e autoarchiviazione della tesi di dottorato nell'Archivio Istituzionale ad accesso aperto alla letteratura scientifica;
4) di essere consapevole che attraverso l'autoarchiviazione delle tesi nell'Archivio Istituzionale ad accesso aperto alla letteratura scientifica del Politecnico di Bari (IRIS-POLIBA), l'Ateneo archivierà e renderà consultabile in rete (nel rispetto della Policy di Ateneo di cui al D.R. 642 del 13.11.2015) il testo completo della tesi di dottorato, fatta salva la possibilità di sottoscrizione di apposite licenze per le relative condizioni di utilizzo (di cui al sito http://www.creativecommons.it/Licenze), e fatte salve, altresì, le eventuali esigenze di "embargo", legate a strette considerazioni sulla tutelabilità e sfruttamento industriale/commerciale dei contenuti della tesi, da rappresentarsi mediante compilazione e sottoscrizione del modulo in calce (Richiesta di embargo);
5) che la tesi da depositare in IRIS-POLIBA, in formato digitale (PDF/A) sarà del tutto identica a quelle **consegnate**/inviate/da inviarsi ai componenti della commissione per l'esame finale e a qualsiasi altra copia depositata presso gli Uffici del Politecnico di Bari in forma cartacea o digitale, ovvero a quella da discutere in sede di esame finale, a quella da depositare, a cura dell'Ateneo, presso le Biblioteche Nazionali Centrali di Roma e Firenze e presso tutti gli Uffici competenti per legge al momento del deposito stesso, e che di conseguenza va esclusa qualsiasi responsabilità del Politecnico di Bari per quanto riguarda eventuali errori, imprecisioni o omissioni nei contenuti della tesi;
6) che il contenuto e l'organizzazione della tesi è opera originale realizzata dal sottoscritto e non compromette in alcun modo i diritti di terzi, ivi compresi quelli relativi alla sicurezza dei dati personali; che pertanto il Politecnico di Bari ed i suoi funzionari sono in ogni caso esenti da responsabilità di qualsivoglia natura: civile, amministrativa e penale e saranno dal sottoscritto tenuti indenni da qualsiasi richiesta o rivendicazione da parte di terzi;
7) che il contenuto della tesi non infrange in alcun modo il diritto d'Autore né gli obblighi connessi alla salvaguardia di diritti morali od economici di altri autori o di altri aventi diritto, sia per testi, immagini, foto, tabelle, o altre parti di cui la tesi è composta.

Luogo e data Bari, 21/12/2021 Firma Arcangela Rago

Il/La sottoscritto, con l'autoarchiviazione della propria tesi di dottorato nell'Archivio Istituzionale ad accesso aperto del Politecnico di Bari (POLIBA-IRIS), pur mantenendo su di essa tutti i diritti d'autore, morali ed economici, ai sensi della normativa vigente (Legge 633/1941 e ss.mm.ii.),

CONCEDE

- al Politecnico di Bari il permesso di trasferire l'opera su qualsiasi supporto e di convertirla in qualsiasi formato al fine di una corretta conservazione nel tempo. Il Politecnico di Bari garantisce che non verrà effettuata alcuna modifica al contenuto e alla struttura dell'opera.
- al Politecnico di Bari la possibilità di riprodurre l'opera in più di una copia per fini di sicurezza, back-up e conservazione.

Luogo e data Bari, 21/12/2021 Firma Arcangela Rago

Politecnico di Bari

Department of Electrical and Information Engineering
ELECTRICAL AND INFORMATION ENGINEERING
Ph.D. Program
SSD: ING-INF/03–TELECOMMUNICATIONS

Final Dissertation

Analysis and Optimal Management of Mobile Networks Towards Pervasive Intelligence

---

by

Arcangela Rago:

Arcangela Rago

Referees:

Prof. Michele Rossi

Dr. Marco Miozzo

Supervisor:

Prof. Gennaro Boggia

Gennaro Boggia

Coordinator of Ph.D Program:

Prof. Mario Carpentieri

Mario Carpentieri

Course n°34, 01/11/2018-31/10/2021

Declaration of Authorship

I, Arcangela RAGO, declare that this thesis titled, "Analysis and Optimal Management of Mobile Networks Towards Pervasive Intelligence" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Arcangela Rago

Date: 21/12/2021

"What is research but a blind date with knowledge?"

W. Harvey

POLITECNICO DI BARI

Abstract

Doctoral School of Politecnico di Bari
Department of Electrical and Information Engineering

Doctor of Philosophy

Analysis and Optimal Management of Mobile Networks Towards Pervasive Intelligence

by Arcangela RAGO

This thesis investigates machine learning techniques as powerful tools for the analysis and the optimal management of mobile networks towards Pervasive Intelligence. In this context, it describes recent solutions conceived for addressing three different, but related, issues. Firstly, after tailoring an unsupervised learning methodology to characterize radio resource utilization patterns, a Multi-Task Learning model, running directly at the edge of the network, is conceived to perform data mining from the control channel of an operative mobile network. Two configurations of neural networks, based on Undercomplete or Sequence to Sequence autoencoders, are exploited to obtain common feature representations of traffic profiles. Then, softmax and fully-connected layers are used to anticipate information on the type of traffic to be served and the radio resource utilization pattern requested by each service during its execution, respectively. Moreover, a Software-Defined Networking approach is exploited to monitor users' mobility. Consequently, the prediction of both the distribution of users among cells and the communication and computational resources they request over different look-ahead horizons is performed through a Convolutional Long Short-Term Memory architecture. This information is used to perform anticipatory allocation and distribution of users' requests via Dynamic Programming. Hence, a Tenant-driven Radio Access Network slicing enforcement scheme based on Pervasive Intelligence is proposed to avoid the radio resources over-provisioning while saving bandwidth (i.e., the Pay for What You Get paradigm). The Infrastructure Provider exploits a convolutional autoencoder to compress the information on network resources and connectivity and share it with the Tenants. In turn, each Tenant implements a Deep Deterministic Policy Gradient algorithm 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. The investigation in real scenarios and the comparison against conventional approaches adopted for the analysis and the optimal management of mobile networks demonstrate the effectiveness of the proposed machine learning-based solutions. In terms of applicability, the conceived methodologies are also in line with the evolution of mobile networks, where Artificial Intelligence will be natively and pervasively integrated for enabling full network automation.

Contents

List of Figures

List of Tables

List of Acronyms

3GPP	3rd Generation Partnership Project
5G	5th Generation
6G	6th Generation
AI	Artificial Intelligence
ANN	Artificial Neural Network
B5G	Beyond 5G
BBU	Base Band Unit
CAPEX	CAPital EXpenditure
CDF	Cumulative Distribution Function
CDR	Call Detail Record
Cloud-RAN	Cloud-Radio Access Network
CNN	Convolutional Neural Network
ConvLSTM	Convolutional Long Short-Term Memory
CQI	Channel Quality Indicator
CSI	Channel State Information
DCI	Downlink Control Information
DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
eMBB	enhanced Mobile BroadBand
HARQ	Hybrid Automatic Repeat reQuest
LOS	Line-Of-Sight
LSTM	Long Short-Term Memory
LTE	Long Term Evolution
MCS	Modulation and Coding Scheme
MDP	Markov Decision Process
MEC	Multi-access Edge Computing

ML	Machine Learning
MLP	Multi-Layer Perceptron
mMTC	massive Machine Type Communication
MSE	Mean Square Error
MTL	Multi-Task Learning
NFV	Network Function Virtualization
NLOS	Non-Line-Of-Sight
NR	New Radio
OFDM	Orthogonal Frequency Division Multiplexing
OPEX	OPerational EXpenses
OWL	Online Watcher for LTE
PDCCH	Physical Downlink Control CHannel
PDSCH	Physical Downlink Shared CHannel
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RB	Resource Block
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
RRH	Remote Radio Head
RRM	Radio Resource Management
SDN	Software-Defined Networking
SDR	Software Defined Radio
Seq2Seq	Sequence to Sequence
SINR	Signal-to-Interference-plus-Noise Ratio
TBS	Transport Block Size
TTI	Transmission Time Interval
UE	User Equipment
URLLC	Ultra-Reliable and Low Latency Communication

VNF Virtual Network Function

A mia nonna Arcangela

Personal Scientific Contributions

The most significant scientific contributions generated during the PhD are listed in what follows. Those works have been accepted for publication in international journals and conferences or they are still waiting for revision.

International Journals:

- A. Rago, G. Piro, G. Boggia and P. Dini, "Multi-Task Learning at the Mobile Edge: An Effective Way to Combine Traffic Classification and Prediction," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 10362-10374, Sept. 2020, doi: 10.1109/TVT.2020.3005724.
- A. Rago, G. Piro, G. Boggia and P. Dini, "Anticipatory Allocation of Communication and Computational Resources at the Edge using Spatio-Temporal Dynamics of Mobile Users," in *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4548-4562, Dec. 2021, doi: 10.1109/TNSM.2021.309947.

International Conferences:

- A. Rago, G. Piro, H. D. Trinh, G. Boggia and P. Dini, "Unveiling Radio Resource Utilization Dynamics of Mobile Traffic through Unsupervised Learning," in *Proc. of IEEE Network Traffic Measurement and Analysis Conference (TMA)*, 2019, pp. 209-214, doi: 10.23919/TMA.2019.8784692.
- A. Rago, P. Ventrella, G. Piro, G. Boggia and P. Dini, "Towards an Optimal Management of the 5G Cloud-RAN through a Spatio-Temporal Prediction of Users' Dynamics," in *Proc. of IEEE Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2020, pp. 1-4, doi: 10.1109/MedComNet49392.2020.9191492.
- A. Rago, G. Piro, G. Boggia and P. Dini, "A Softwarized Service Infrastructure for the Dynamic Orchestration of IT Resources in 5G Deployments," in *Proc. of IEEE 22nd International Conference on Transparent Optical Networks (ICTON)*, 2020, pp. 1-4, doi: 10.1109/ICTON51198.2020.9203757.

In the process of sumbission:

- A. Rago, S. Martiradonna, G. Piro, A. Abrardo and G. Boggia, "A Tenant-Driven Slicing Enforcement Scheme based on Pervasive Intelligence in the Radio Access Network".

Project Involvement

This thesis is framed in the Apulia Region (Italy) Research project INTENTO (36A49H6). This project aims to provide a framework, i.e., INTElligent NeTwork Orchestrator framework (INTENTO), for the control and the management of an optical transport network, with Software-Defined Networking (SDN) paradigm, including Transport SDN (T-SDN) that embraces the transport network, and Network Function Virtualization (NFV) technology. It is funded by the Apulia Region (Italy) and is run by *Politecnico di Bari* in collaboration with *SM-Optics S.r.l*, a leading company in the software development and integration, and the optical transport sector, and *Experis S.r.l*, a leading company in the IT consulting and Engineering solutions.

The Research project INTENTO stems from the need to create a platform that could manage the complexity of network infrastructures, by including hardware and software IT components, optical nodes, and so on. In fact, the telecommunications infrastructure is evolving towards a progressive but fast integration with the IT infrastructure. In the near future, such integration will allow delivering new services and applications with specific and very high Quality of Transmission and Quality of Service. A suitable composition of open reference frameworks drives the implementation of the management system of such complex infrastructure and vendors should provide enhanced tools for its optimization.

In this context, the research activities carried out during the PhD and detailed in this thesis contributes to the development of innovative methodologies based on machine learning for the analysis and optimal management of network resources (e.g., radio resources, IT resources, virtual network functions at the edge of the network) in the 5G infrastructure through a high-level orchestrator. In fact, the outcomes of the classification and prediction of traffic, mobility, and users' requests based on machine learning can efficiently and anticipatorily aid the optimal and dynamic orchestration of 5G infrastructure while respecting service requirements and avoiding resource over-provisioning.

Award:

- "When Machine Learning and programmable 5G networks meet autonomous driving services," Best Poster Award at *International Workshop on Smart Mobility in Future Cities: The Apulia Industry Summit (SMFC 2019)*, in conjunction with *IEEE International Conference on Systems, Man and Cybernetics (IEEE SMC 2019)*, Bari, Italy, October 2019.

Introduction

Machine Learning (ML) is the branch of Artificial Intelligence (AI) that investigates algorithms able to learn and improve their experience and performance over time directly from data examples, without being explicitly programmed. Algorithms can extract knowledge from complete data: hidden patterns in the training data are identified and used to unveil useful information at runtime and to drive the execution of a given task (typically classification, prediction, or clustering). To improve these capabilities, the sub-branch of ML called deep learning further enables the mining of valuable information of data coming from heterogeneous sources and finding hidden correlations automatically, which would have been too complex to extract by human experts. Deep learning also supports Reinforcement Learning (i.e., Deep Reinforcement Learning), providing autonomous decision-making. Therefore, AI and ML are powerful tools that are increasingly spreading in the mobile networking domain, where the growing diversity and complexity of the mobile network architectures have made the monitoring and the instantaneous managing of the multitude of network elements intractable. In fact, moving toward the 5th Generation (5G) and Beyond networks, since a large amount of network resources needs to be intelligently configured for offering extremely high data transmission rate and stringent low-latency requirements, AI will be pervasively integrated into the mobile network design.

In line with this emerging research trend and differently from most of the literature in the field, this thesis addresses ML techniques for the analysis and the joint mobile traffic classification and prediction, for the anticipatory allocation of communication and computational resources at the network edge, and for the management of network slicing in the Radio Access Network (RAN). A brief description of the chapters that make up the thesis is provided below.

Chapter 1 introduces 5G and Beyond networks with their enabling technologies, that are Software-Defined Networking (SDN), Network Function Virtualization (NFV), network slicing, and Multi-access Edge Computing (MEC) paradigms. After highlighting AI advantages in the context of mobile networks, this Chapter provides an overview of ML techniques and introduces the related work on the analysis and optimal management of mobile systems, by identifying the gaps that this thesis intends to fill.

In Chapter 2, after tailoring an unsupervised learning methodology to characterize radio resource utilization patterns, a Multi-Task Learning model, running directly at the network edge, is conceived to perform data mining from the control

channel of an operative mobile network. It guarantees reduced storage requirements, fast data processing, and limited monitoring complexity. Autoencoders are used as key building blocks of the proposed Multi-Task Learning methodology for common feature representations, shared by both classification task (anticipation on the type of traffic to be served) and prediction task (anticipation on the resource allocation pattern requested by each service throughout its duration).

In Chapter 3, an ETSI-MEC compliant architecture adopts SDN facilities to monitor users' mobility during the time. Then, it exploits a deep learning architecture based on Convolutional Long Short-Term Memory for predicting the distribution of users among cells and their related service demands over a look-ahead temporal horizon. A centralized orchestrator uses this information to distribute users' demands among available MEC servers in an anticipatory manner via Dynamic Programming, while satisfying communication and computational constraints at the network edge and the upper bound for latency expected by mobile users.

In Chapter 4, a Tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is proposed. The proposed approach grounds its roots in the Pay for What You Get paradigm, which promises to avoid the radio resources overprovisioning while saving bandwidth. To achieve these goals, the Infrastructure Provider exploits a convolutional autoencoder to compress the information on network resources and connectivity and share it with the Tenants. In turn, each Tenant implements a Deep Deterministic Policy Gradient algorithm 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 RAN level.

To demonstrate the effectiveness of the conceived methodologies for the analysis and the optimal management of mobile networks, they are investigated in real scenarios and compared against conventional approaches.

Finally, the thesis closes with the Conclusions, summing up the main findings while drawing future research directions towards Pervasive Intelligence.

Chapter 1

Introduction to 5G and Beyond Networks: Enabling Technologies Towards Pervasive Intelligence

In this Chapter, current and future generations of mobile networks, with the enabling technologies, are introduced. The description carried out throughout this part of the thesis describes their main functionalities. The dissertation continues providing an overview of the potential of Machine Learning (ML) in the analysis, management, and optimization of mobile networks for pursuing full network automation in pervasive intelligent endogenous future mobile systems. After this general overview, the Chapter presents the related work on mobile traffic analysis, anticipatory resource allocation, and the management of network slicing in which ML plays a key role. In particular, the gaps bridged in this thesis are identified and detailed.

1.1 5G and Beyond Networks

Driven by the increasing demand of mobile data traffic and the tremendous growth in connectivity, the 5th Generation (5G) of mobile technology has begun to revolutionize the existing wireless networks [1], [2].

High-level performance targets of 5G are classified into three different categories of use cases [3]:

- *enhanced Mobile BroadBand (eMBB)*, which provides the service of mobile broadband in order to ensure consistent Quality of Service (QoS)/Quality of Experience (QoE). Typical examples of eMBB services are 4K/8K video and virtual reality.
- *Ultra-Reliable and Low Latency Communication (URLLC)*, with extends network capabilities in terms of reliability and latency (e.g., autonomous driving).
- *massive Machine Type Communication (mMTC)*, enabling massive connectivity (i.e., massive access by a large number of devices). For example, smart cities fall into this category.

In order to extend the performance of mobile networks in terms of throughput, latency, reliability, density, and mobility, 5G is characterized by embedded flexibility to optimize the network usage. Specifically, for accommodating a wide range of advanced use cases in an agile and cost efficient manner, the 5G architecture includes modular network functions that could be deployed and scaled on demand. To this end, a new radio air interface, namely 5G New Radio (NR) or 5G NR air interface, has been standardized by 3rd Generation Partnership Project (3GPP) and developed for adding novel features like flexible spectrum, scalable numerology, forward compatibility, and ultra-lean design, as compared to the prior Long Term Evolution (LTE) systems.

1.1.1 5G New Radio

5G NR is a totally new air interface that can operate alongside 4G LTE. In addition to the ability of handling much faster data rates and of providing higher capacity to users with respect to previous generations, the key NR pillar ensuring the implementation of numerous 5G applications in different scenarios is the flexibility [3].

Regarding the spectrum, different frequency bands are flexibly used for different types of communication so that several functionalities and features can be enabled. In fact, 5G NR operates on any frequency band between 450 MHz and 52.6 GHz. The lower bands are needed for coverage, while the higher bands will provide high data rate services and high traffic capacity. Specifically, 3GPP defines two frequency ranges: the first one (Frequency Range 1) covers the frequencies between 450 MHz and 6 GHz range, whereas the second one (Frequency Range 2) refers to the frequencies within the 24.250–52.600 GHz interval. These frequency ranges are commonly referred to as sub-6 GHz and millimeter-wave, respectively [3].

Moreover, 5G NR supports flexible numerology. In LTE, Orthogonal Frequency Division Multiplexing (OFDM) is used with a fixed subcarrier spacing of 15 kHz and 12 subcarriers are combined in the frequency domain to define the basic radio resource, namely the Resource Block (RB) [3]. Also in 5G a RB has 12 subcarriers. However, 3GPP introduces in the NR standard the idea of flexible and scalable numerology, which is characterized by a set of supported subcarrier spacings and cyclic prefixes [4]. Specifically, the 5G NR supported carrier spacing is scaled by multiplying the factor 2^n to the LTE supported carrier spacing (i.e., 15 kHz), with the integer n ranging from 0 to 5. Therefore, 5G NR supports spacing of 15, 30, 60, 120, and 240 kHz, i.e., RBs of 180, 360, 720, 1440, and 2880 kHz width, respectively. In the dime domain, to maintain a certain backward compatibility with LTE, the NR frame is 10 ms long, and it is composed of ten subframes of 1 ms each. Nonetheless, according to the chosen numerology, each subframe is split into a variable number of slots, which increases with the subcarrier spacing: the smaller the slot length, the higher the subcarrier spacing. Then, each slot contains a fixed number of OFDM symbols, that are 14 and 12 for the normal and the extended cyclic prefix length, respectively.

Thus, this architecture enables a flexible NR frame structure, allowing a different number of slots per subframe, as well as varying OFDM symbol and slot lengths [5].

In addition to the flexibility, the NR system is characterized by an ultra-lean design, so as to decrease the power utilization through the reduction of 'always on' signals (i.e., synchronization signals, broadcast signals, and the reference signal). It helps increasing the energy efficiency of the system and minimizing the interferences, especially in high traffic load conditions. It also enhances forward compatibility, that is another feature of the NR system. In fact, besides guaranteeing backward compatibility with LTE, 5G NR allows the network to support future applications which will characterize 5G and Beyond 5G (B5G) networks.

1.1.2 Enabling Technologies

Concrete opportunities to develop advanced 5G services are offered by the integration of Software-Defined Networking (SDN), Network Function Virtualization (NFV), and network slicing paradigms. They ensure high performance in terms of scalability and rapid time to market, while increasing the network programmability and reducing CAPital EXpenditure (CAPEX) and OPerational EXpenses (OPEX) [6], [7]. Specifically, SDN decouples the data and control plane, enabling the control and orchestration of switches and routers from a central entity, which is configured in a way that is agnostic to the underlying hardware infrastructure [6], [8]. NFV is the virtualization of network functions, i.e., Virtual Network Functions (VNFs), which are decoupled from dedicated hardware devices [8]. Network slicing consists of executing multiple logical network instances on a shared infrastructure [6]. It can be considered as a convergence of SDN with NFV because SDN provides a global view of network infrastructure and the capability of programmable network control by creating the control plane, while, through NFV, network functions and resources are not restricted to dedicated network infrastructures anymore [8].

However, since end-users mostly have limited storage and processing capacities, running compute-intensive applications on resource-constrained users is still an important issue [9]. For this purpose, in both 5G and B5G networks, Multi-access Edge Computing (MEC) is emerging as a fundamental enabling technology for the rapid diffusion of advanced services, such as autonomous driving, virtual/augmented reality, e-Health, robotics, and tactile Internet [8]–[11]. According to ETSI-MEC specifications [12], MEC servers are deployed at the network edge to offer intensive computing and memory capabilities in the proximity of end-users, while guaranteeing low communication latencies to new heavy demanding and real-time services [9]. They are also able to limit network congestions by processing data directly at the edge, instead of forwarding a big amount of data to the cloud. This particularly applies to MEC servers co-located with gNBs (base station of 5G networks), that can provide computational capabilities as close as possible to end-users and capture information for further purposes (data analytics and big data processing) [9].

1.2 Towards Pervasive Intelligence

In order to fulfill the expanding diversity and complexity of the mobile network architectures and the even growing amount of users' requests, mobile traffic and communication and computational resources available in the network infrastructure, including core network, edge network, and Radio Access Network (RAN), should be properly managed [13], [14]. Networking researchers have been recently recognizing the importance of ML and its ability to solve the hard monitoring and managing of the multitude of network elements in current and future generations of mobile systems [15]–[19].

ML is the branch of Artificial Intelligence (AI) that investigates algorithms able to learn and improve their experience and performance over time directly from data examples, without being explicitly programmed. With these algorithms, a system can deduce knowledge from data: hidden patterns in the training data are identified and used to analyze unknown information and drive the execution of a given task (typically classification, prediction, or clustering) [20]. AI algorithms can be classified into three main categories, the choice of which depends on the type of available data and on the problem goals:

- *Unsupervised Learning*, when the training phase, whose objective is to learn relationships and structure from the input data, is based on unlabelled data or an undefined and unspecific output;
- *Supervised Learning*, when algorithms learn to build a statistical model for predicting or estimating an output based on one or more inputs by using labelled data;
- *Reinforcement Learning (RL)*, that is a growing field related to sequential decision making, where optimal policies (the decisions or actions) are learnt by an agent in a given environment by interacting with the environment itself, i.e., by iteratively implementing actions and receiving rewards (the feedback).

To improve ML capabilities, its sub-branch called deep learning further enables the mining of valuable information of data coming from heterogeneous sources and unveils hidden correlations automatically, which would have been too complex to extract by human experts [18]. Deep learning techniques employ Artificial Neural Networks (ANNs), that mimic biological neural networks from which they get their name. In fact, as the biological neurons are interconnected to form the brain networks, which allow the individuals, for example, to reason, recognize sounds and images, and learn to act, ANNs are interconnected groups of artificial neurons for information processing. Artificial neurons are grouped into sets called layers, so that the real numbers, which replace the electrical signals of brain synapses, go from one layer to another one. There are three types of layers: input layer, where the input data are received; hidden layers, through which the real numbers travel, undergoing modifications based on the weights of the connections and neurons; and

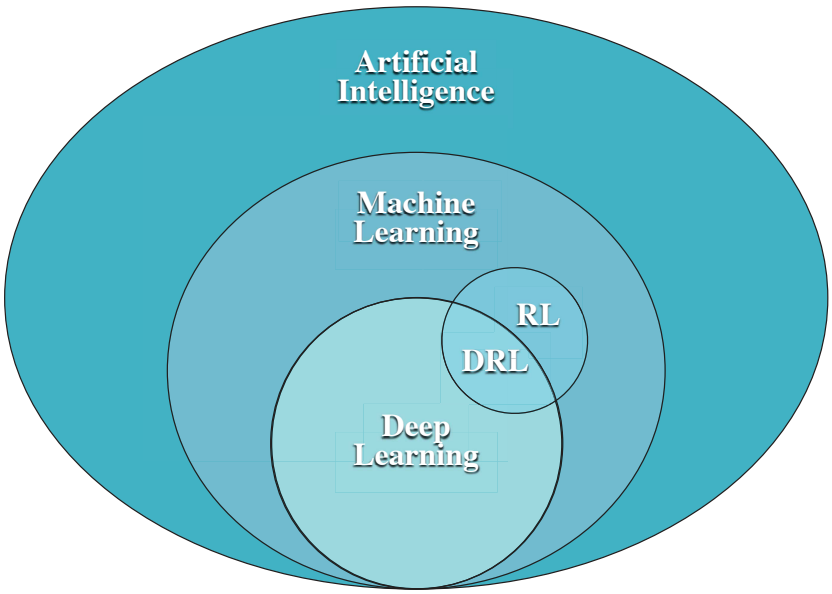


FIGURE 1.1: Venn diagram of the relation among AI techniques.

output layer, which returns the result of the processing performed by the network. In this sense, only ANNs that have a lot of hidden layers (i.e., even if one hidden layer can be enough by definition) can be regarded as deep models, i.e., deep neural networks [17], [21]. By combining deep learning with RL techniques, there are also Deep Reinforcement Learning (DRL) algorithms, where the agents exploit neural networks to make decisions from unstructured input data without manual engineering of the state space (i.e., the set of the possible situations in which the agent finds itself) for obtaining the optimal policy [17], [22]. Figure 1.1 illustrates the relation among AI techniques through a Venn diagram.

Nowadays, ML and AI in general are taking an increasingly central role in the context of mobile networks. Moving toward 6th Generation (6G) and Beyond, a huge amount of network resources needs to be intelligently configured for offering extremely high data transmission rate and very stringent low-latency requirements. This motivates the utilization of AI to fundamentally rethink the 6G communication system design [23]. Differently from 5G networks, which did not take AI into account at the beginning of the architectural plan, the design of 6G architecture should natively and pervasively integrate AI in various layers of the network for enabling full network automation [3]. Thus, the traditional and obsolete approaches in network planning, analysis, and optimization will be replaced with automated methods that use AI/ML to guide planning decisions and dynamically manage physical and virtual network resources [21].

In line with these emerging research trends, this thesis describes innovative solutions that exploit the potential of AI/ML towards Pervasive Intelligence for addressing three different, but related, issues, with the common goal of analyzing and

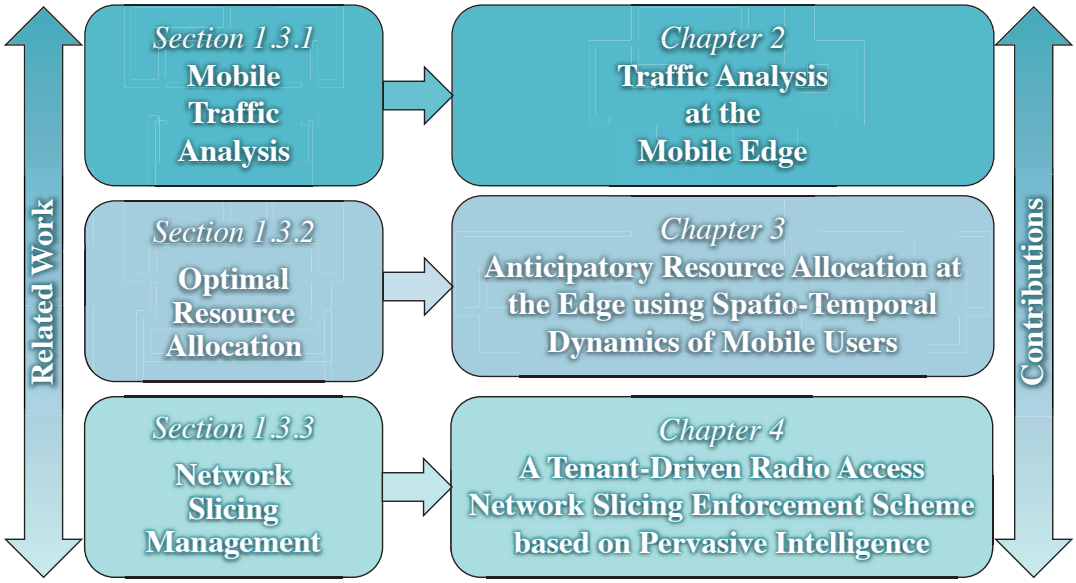


FIGURE 1.2: Thesis organization.

optimally managing mobile networks. In particular, the dissertation targets the following objectives: i) to unveil radio resource utilization dynamics through ML and to jointly classify and predict mobile traffic through deep learning; ii) to understand spatio-temporal dynamics of mobile users in order to optimally and anticipatorily allocate communication and computational resources at the network edge through deep learning; and iii) to manage the network slicing at the RAN level by using deep learning and DRL. In order to achieve the first target, an unsupervised learning methodology is put forward to analyze mobile traffic at the radio link level and identify similar radio resource utilization patterns of mobile traffic. Then, the potential of deep learning is investigated in order to jointly perform mobile traffic classification and prediction, which are key tasks for network optimization. Section 1.3.1 reviews the related work on this area and identifies the gaps bridged by Chapter 2, where the conceived approach is detailed. Regarding the user mobility, this PhD work anticipatorily allocates communication and computational resources at the network edge, based on the prediction of spatio-temporal dynamics of mobile users and the related requests performed through deep learning. The novelty of the proposed solution is pointed out in Section 1.3.2, while more details can be found in Chapter 3. Finally, to effectively enforce the network slices in the RAN, this work goes beyond the state-of-the-art literature on network slicing management, as outlined in Section 1.3.3. In fact, a Tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is conceived. It exploits deep learning and DRL to compress the information on network resources and connectivity and to dynamically adapt bandwidth requests, respectively. For further details, see Chapter 4. Figure 1.2 summarizes the thesis organization: each following subsection introduces the scientific work used as background for the main contributions of this thesis explained in the next chapters.

1.3 State of the Art

This section reviews the related work on mobile traffic analysis, resource allocation, and network slicing management and identifies the gaps bridged in this thesis by using AI/ML techniques in the context of the analysis and the optimal management of mobile networks.

1.3.1 Mobile Traffic Analysis

Understanding the dynamics of mobile traffic demands is of utmost importance for proper and secure management of network resources (e.g., spectrum, energy, computation) [24], [25]. Specifically, traffic classification at the radio link level and forecast may enable advanced QoS and QoE enforcement policies based on a priori knowledge of application behaviors.

The scientific literature addressed traffic classification in mobile networks through machine or deep learning methodologies [26].

Several approaches, based on Support Vector Machine and Random Forest algorithms, have been conceived to identify applications or smartphone types starting from the observation of encrypted communication flows [27]–[30]. Nevertheless, mobile data are usually generated by heterogeneous sources, exhibit non-trivial spatio/temporal patterns, and often embrace high volumes of different information [31]. Flows' characteristics are also rapidly prone to be out of date and need to be frequently updated [32]. In these complex and dynamics conditions, ML algorithms generally fail to automatically extract and use the key features describing the investigated flows [17]. On the contrary, deep learning methods demonstrated to be able to overcome the traditional ML approaches because of their native ability to successfully support traffic analysis and accurately characterize traffic dynamics [17], [32]–[38]. Unfortunately, mobile networking and deep learning problems have been explored mostly independently and only recently crossovers between the two research areas have emerged.

Reference deep learning solutions for traffic classification leverage Convolutional Neural Networks with one-dimensional [39]–[41] or two-dimensional [41], [42] convolutional layers, Stacked Autoencoder with five stacked layers[40]–[42], Multi-Layer Perceptron (MLP) with one [41] or two hidden layers [41], [42], and standard or hybrid Long Short-Term Memory (LSTM) combined with two-dimensional convolutional layers [41]. However, only the contribution in [41] focuses on mobile networks. Among the other important investigations it provides, [41] also demonstrates how deep neural networks guarantee greater accuracy levels than conventional ML approaches in mobile networks. On the other hand, deep learning also outperforms baseline approaches for traffic prediction, including the conventional Auto Regressive Integrated Moving Average scheme [17], by accurately predicting traffic volume/load [17], [34]. Here, reference methodologies are based on MLPs [43], densely connected Convolutional Neural Networks (CNNs)

with two-dimensional convolutional layers [44], LSTMs [45]–[49], and Multivariate LSTMs [50], as well as on the combination of CNNs and LSTMs [51]–[53], that can extract spatial and temporal correlations of data through the convolutional operation and LSTM memory cells, respectively. In that case, all of the reviewed contributions focus on mobile networks.

The analysis of the state of the art on deep learning strategies reveals that traffic classification and prediction are generally treated separately. In other words, classification and prediction are achieved as two isolated tasks. This represents a major drawback because their parallel execution involves the training of different learning architectures, as well as an inevitable increase of computational requirements [54].

The Multi-Task Learning (MTL) approach solves the aforementioned issue, while often reaching greater performance levels when compared with single-tasks approaches [54], [55]. Differently from the single-task scheme, MTL basically embraces a learning architecture that extracts common feature representations from the training data and jointly executes multiple, but related, tasks. Therefore, MTL, especially in view of Pervasive Intelligence in 6G and Beyond, emerges as a suitable solution for meeting the computational and memory constraints that increasingly affect mobile networks [17]. Valuable contributions in this direction are presented in [56], [57], where an MTL architecture is designed to implement multiple tasks related to the traffic classification only. Unfortunately, they do not address traffic prediction and do not focus on mobile networks.

Another important consideration emerging from the scientific literature is that all the investigated contributions in this field perform data mining from the messages exchanged over the data plane (i.e., traffic volume/load collected at the network or application layers, equipped with application labels for classification task). Therefore, by considering the huge amount of data handled by mobile systems, the reviewed methodologies cannot be applied to the control plane and require high computational and memory capabilities, thus rendering their use impractical at the mobile edge.

Recently, researchers are given access to Call Detail Records (CDRs) of mobile operators and the analysis is more oriented on extrapolating spatio-temporal characteristics of the mobile user traffic [58]–[60]. However, datasets on CDRs are rarely made available by operators and they are difficult to achieve. Although they were used by researchers, the considered datasets miss characterizing fine-grained details such as access level dynamics, which, instead, are considered crucial for mobile network optimization. In fact, CDRs would help identify user service requests and throughput, but do not offer any detail on wireless link level information, channel conditions, retransmissions, packet fragmentation, and so on. Thus, CDRs are incomplete and do not suffice to get appropriate representations of mobile traffic and train neural networks.

To bridge this gap, Chapter 2 exploits data mining from the unencrypted control channel of an operative mobile network to properly characterize the mobile traffic

TABLE 1.1: Comparison among this work and the other contributions focusing on traffic analysis through deep learning.

Contributions	**Task**			**Mobile scenario**	**Processed messages**		**Dataset type**		
	Classification	Prediction	*Joint*		Data plane	*Control plane*	Network/ application level data	Traffic volume/ load	*Radio link-level data*
[39], [40], [42]	✓				✓		✓	✓	
[41]	✓			✓	✓		✓	✓	
[43]–[53]		✓		✓	✓		✓	✓	
[56], [57]	✓				✓		✓	✓	
[61]	✓			✓		✓		✓	✓
[62]	✓			✓		✓	✓	✓	✓
[63]		✓		✓		✓		✓	✓
Chapter 2 of this thesis			✓	✓		✓	✓	✓	✓

at the radio interface, in addition to getting out data plane information (i.e., traffic volume/load and application labels) and reducing storage and monitoring processing. Therefore, even if the data mining is performed on the control plane, the accuracy of the classification and prediction tasks is still evaluated on the derived data plane information. Interesting contributions in this direction address traffic pattern analysis and classification [61], [62] and traffic prediction [63] through data mining performed on the Physical Downlink Control CHannel (PDCCH). The proposed solutions, however, are not based on an MTL approach.

Differently from the current state of the art, the goal of this PhD work is to mainly adopt an MTL architecture at the edge of the network to jointly classify mobile services and forecast future traffic demands, enabling advanced QoS and QoE enforcement policies based on a priori knowledge of application behaviors. Thus, network operators can configure and manage network resources in a more intelligent and prolific mode thanks to the knowledge extracted by deep learning algorithms.

To conclude, Table 1.1 summarizes the goals and the methodologies followed by the scientific contributions reviewed in this section, while highlighting the main differences with respect to the innovative MTL model proposed in this thesis.

1.3.2 Optimal Resource Allocation

In the context of network optimization, network resource management, computational resource allocation, task offloading, and VNF placement represent typical technical problems of interest for industry and academia working on mobile communication systems [64]–[66]. Very frequently, they are addressed with optimization algorithms willing to minimize energy consumption [66]–[70], delay [71]–[73], or both [74], [75]. Sometimes, a constraint on the maximum allowed delay is taken into account as well [66]–[69], [72].

Emerging methodologies exploit AI technologies, like ML, deep learning, and DRL, for network optimization [76]. While most of the contributions in this context focus on the optimal management of computational resources only [77]–[79], some

other works consider at the same time the goal of managing and allocating communication and computational resources [80], [81]. Available approaches intend to maximize the overall resource capacity [79], to minimize energy consumption [78] and delay [77], [80], [81], as well as to fulfill the expected upper bound for the overall delay [77], [81].

The contributions presented in [64], [65], [82] highlight that the knowledge (i.e., prediction) of users' mobility and/or the set of requests that they may formulate in a given geographical area over time introduce further key information for network optimization tasks.

The prediction of users' trajectory and location can be achieved with mathematical models [83]–[85]. The mobility forecasting obtained in [83] is used to offload computing tasks (requested by mobile users) to a single remote MEC server. To this end, an optimization problem that jointly minimizes energy consumption and latency, satisfying the expected maximum delay, is formulated in [83]. The knowledge of trajectories during the next look-ahead window is considered in [84] for planning the migration of virtual machines at the network edge. This goal is reached by employing an optimization problem that minimizes communication latencies, ensuring at the same time expected upper bounds. Finally, the work in [85] leverages a Markov Decision Process (MDP) to predict user mobility and formulates an iterative approach for jointly allocating communication resources among available users and placing virtual machines at the network edge. Similarly to [83], the presented solution minimizes energy consumption and delay.

Differently from the above-discussed methodologies, solutions based on ML promise to better anticipate network behaviors and dynamics, also in heterogeneous and large scale scenarios [86], [87]. For example, the prediction of trajectory and location is performed through deep learning architectures, as LSTMs [88]–[91], LSTMs with attention mechanism [92], CNNs [93], and a combination of recurrent and CNNs with Markov Chains [94]. Furthermore, the number of users in a given geographical area is predicted through ML-based Regressors in [95] and a combination of deep learning and Bayesian networks in [96]. Mobility forecasting in [89] supports an optimization problem willing to distribute computing caching capabilities among mobile users, maximizing the overall resource capacity and satisfying the expected maximum delay. The knowledge of locations into the future, forecasting one [93] or multiple steps ahead [91], [94], is also adopted to drive the migration of virtual machines at the network edge. In more detail, the contribution in [93] describes an iterative procedure for minimizing the communication latencies and satisfying the expected maximum delays. Optimization problems willing to minimize delay [91] and energy consumption [94] are formulated in [91], [94]. Finally, the work discussed in [90] adopts DRL to manage computation offloading tasks among different remote MEC servers in order to minimize the delay.

Instead, traffic volume/load can be accurately predicted through deep learning methods [17], [34], as anticipated and detailed in Section 1.3.1. Traffic forecasting

TABLE 1.2: Comparison among this work and the other contributions performing mobility/requests prediction and network optimization.

Contributions	Prediction							Network optimization							
	Mobility			Requests	Characterization		Look-ahead horizon	Communication	Computation			RRH-BBU map	Delay constraint	Solution	
	Trajectory and location		Number of users per cell		Spatial	Temporal			Offloading/ Execution		Migration				
	Mathematical model	Deep learning	Deep learning	Deep learning					Edge servers	Users				Optimization algorithm	iterative procedure
[83]	✓					✓			1	✓			✓	✓	
[84]	✓					✓	✓				✓		✓	✓	
[85]	✓					✓		✓			✓				✓
[89]		✓				✓				✓			✓	✓	
[93]		✓			✓						✓		✓		✓
[91], [94]		✓				✓	✓				✓			✓	
[90]		✓				✓			>1						✓
[48], [49]				✓		✓	✓		1					✓	
[43]				✓		✓		✓	>1				✓	✓	
[50], [97]				✓	✓	✓						✓		✓	
Chapter 3 of this thesis			✓	✓	✓	✓	✓	✓	>1				✓	✓	

during the next look-ahead horizon through LSTMs assists network optimization in terms of computation offloading and resource allocation with one MEC server in [48], [49], minimizing energy consumption. Traffic prediction performed through MLPs also aids the joint communication and computational resource allocation for user association and Service Function Chain placement among MEC servers in [43]. In the last paper, an optimization algorithm is adopted for minimizing delay, while meeting delay guarantees (a worst case service latency). Moreover, the knowledge of traffic requests obtained by means of Multivariate LSTMs in Cloud-Radio Access Network (Cloud-RAN) context supports the Remote Radio Head (RRH)-Base Band Unit (BBU) mapping in [50], where an optimization problem minimizes deployment cost and energy consumption. The traffic volume of RRHs with the number of users moving between a pair of two RRHs is predicted in [97] through Multivariate LSTM. This information is exploited to optimally perform RRH-BBU mapping, minimizing energy consumption and delay.

To conclude, Table 1.2 summarizes the goals and methodologies followed by the reviewed scientific contributions performing mobility/requests prediction and network optimization, highlighting the main differences with respect to the approach proposed in Chapter 3 of this thesis. It emerges that no contributions in the current state of the art jointly predict, through deep learning, the geographical distribution of users over time (i.e., the number of users available within each cell in a given moment) and the related requests for a look-ahead horizon, as proposed in this work in order to better manage task offloading also in a 5G slicing context. Thus, they do not take advantage of mobility and requests prediction to dynamically and anticipatorily optimize communication and computational resource management among

available MEC servers, satisfying the upper bound of communication latencies.

1.3.3 Network Slicing Management

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 [17], [22], [98]–[101].

Since deep learning can extract important features from data and model its high-level abstractions, avoiding manual description of a data structure [17], [22], deep learning architectures have been successfully proposed in channel estimation in recent years. For example, LSTM networks perform the prediction of RAN resource usage by a network slice [102] and the prediction of future Channel Quality Indicator (CQI) values in a data-driven RAN slicing framework with URLLC and eMBB slices [103]. Furthermore, an encoder-decoder structure based on CNN is presented in [104] for estimating the traffic of slices deployed at Cloud-RAN, MEC, and core datacenters.

The time-varying wireless channel and its unpredictable variability, network dynamics and heterogeneity, slice isolation, as well as different QoS of various services largely impact the optimal decision-making process for the management of network slicing in the RAN. Differently from traditional solutions that require to rerun the algorithms every time the environment changes, RL and DRL methods are fit for these challenges [22]. A slice admission strategy based on RL is presented in [105] for a flexible RAN. Q-learning is adopted to handle RAN slicing [106], supporting an eMBB and a Vehicle-to-Everything slice on the same RAN infrastructure. In [107], LSTM is incorporated into the actor-critic DRL algorithm for an intelligent resource management of RAN slicing. Deep Q-Network [108], [109] and its modified versions [110], [111] are exploited for slice management in RAN. Specifically, the contribution in [110] entails a Generative Adversarial Network-powered Deep distributional Q-Network for demand-aware resource allocation, while resource block allocation to multiple slices is optimized in [111] 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 [112]. Moreover, by jointly optimizing radio and computation resources in the context of RAN network slicing, the utility maximization problem formulated as a MDP is solved in [113] through the Deep Deterministic Policy Gradient (DDPG) algorithm, that combines DQN and the actor-critic approach. Similarly, the contributions in [114] and [115] 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 [116], Deep Q-learning [116], [117], and a distributed

TABLE 1.3: Comparison among this work and the other contributions adopting AI-based techniques for the management of network slicing.

Contributions	AI-based techniques		Network Slicing			
	Deep Learning	RL/ DRL	Core Network	Radio Access Network	Tenant-driven	Network status compression
[102], [103]	✓			✓		
[104]	✓		✓	✓		
[105]–[115]		✓		✓		
[116]–[118]		✓	✓	✓		
[119]	✓	✓	✓		✓	✓
Chapter 4 of this thesis	✓	✓		✓	✓	✓

DRL strategy based on the Advantage Actor Critic (A2C) algorithm [118] also assist network slicing involving both RAN and core network.

Deep learning can also support DRL-based resource allocation methods, especially in pervasive intelligent endogenous future mobile systems [100]. For example, the compression of high-dimensional CQI information, obtained through an autoencoder, is exploited in a DQN-based framework in [99]. 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 [119] 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.

Table 1.2 summarizes the goals and methodologies followed by the reviewed contributions adopting AI-based techniques for the management of network slicing. To the best of the authors' knowledge, it emerges that no contributions in the current scientific literature jointly exploit deep learning and DRL for a Tenant-driven RAN slicing enforcement scheme, as proposed in Chapter 4 of this thesis in order to dynamically adapt bandwidth requests according to users' requirements of Tenants, without fully knowing the network status. CQI information (i.e., network status) is compressed by the Infrastructure Provider through a deep learning architecture (i.e., convolutional autoencoder). Then, the compressed network status is used as the input of DRL frameworks based on the DDPG algorithm, that are implemented by Tenants.

Chapter 2

Traffic Analysis at the Mobile Edge

This Chapter concerns the mobile traffic analysis (i.e., characterization, classification, and prediction) at the radio link level. Since the widespread and growing usage of smartphones and machine-type communications are deeply changing the type of traffic that traverses the mobile networks, mobile traffic analysis at the edge can be very useful for proper and optimal management of network resources.

To this end, in this Chapter the mobile traffic analysis is firstly treated as an unsupervised learning problem, which aims at identifying and characterizing spatio-temporal radio resource utilization patterns of mobile sessions. Note that any active session can be described, at the radio link level, through a traffic profile reporting the amount of data exchanged between the base station and a mobile terminal as time evolves, referred to as *radio resource utilization pattern*. The Online Watcher for LTE (OWL) tool [120], [121] is used for monitoring the unencrypted PDCCH of an operating LTE network deployed in Spain. The advantage of this tool is the richness of the gathered information (i.e., link level data) and the temporal granularity of the data (i.e., 1 ms). Obtained datasets, referring to residential and campus areas, are processed to group monitored sessions according to the achieved data rate, the adopted transmission settings, the radio resource usage, and the duration. The outcomes of the conducted study highlight the properties of groups of sessions with similar characteristics, expressed in terms of bandwidth demands and application level requirements.

Then, the potential of deep learning for mobile traffic classification and prediction, which are key tasks for network optimization, is investigated. In fact, the envisaged architecture of the 5G and B5G mobile broadband systems will integrate new technology components (e.g., massive MIMO, mm-Wave communication, network slicing, vehicular networks, more and broader frequency bands), a higher variety of devices (e.g., smartphone, sensors, and different types of machines), a larger number of services (e.g., augmented/virtual reality and autonomous driving) with tighter latency requirements, so that resource allocation is expected to reach unprecedented complexity [33], [122], [123]. In this context, network optimization frameworks may be supported by deep learning algorithms, which, when properly tailored, may anticipate information on: i) the type of traffic to be served, e.g., its main characteristics in terms of bandwidth and latency requirements (i.e., *traffic classification*) and ii) the

resource allocation pattern requested by each service throughout its duration (i.e., *traffic prediction*). Differently from most of the literature in this field, which treats traffic classification and prediction separately (please see Section 1.3.1 for further details), an MTL approach [124] is proposed herein. It reduces the number of training samples to be learnt by the two tasks and leads to performance improvement compared with learning them individually [54]. At the same time, it is important to remark that offloading the huge amount of data generated from edge to cloud is intractable in 5G scenarios since it causes network congestion. Therefore, it is highly preferable that deep learning algorithms run at the edge of the network and give online support to optimization frameworks to promptly make decisions and trigger the proper management actions (e.g., radio resource scheduling, cell selection, and sleep mode enabling, to name a few) [80], [87], [125], [126]. Almost all the approaches presented in the current state of the art implement data mining on the huge amount of information collected at the network or application layers of the data plane. Instead, the proposed MTL model considers data belonging to the control plane, as recently investigated, and it is trained with information extracted from the PDCCH of an operative mobile network. The rationale behind the choice of using the control channel in the conceived MTL approach is twofold. First, the volume of control messages from the control plane is much smaller than the user traffic from the data plane (which may also be encrypted, i.e., it does not necessarily require to be decrypted), leading to fast and efficient classification and prediction, which are still evaluated on the derived data plane information. Second, the algorithm runs at the radio interface, which allows fast execution of the two tasks directly at the edge. Note that the lack of need to decrypt traffic for classification may be good as the network operator does not have to go deep into the user traffic to classify and make decisions. However, it could be a disadvantage as even a malicious user may infer what type of messages and applications are exploited without decrypting the mobile traffic. The comparison with conventional single-task learning approaches for traffic classification and prediction, that do not use autoencoders and tackle classification and prediction tasks separately, clearly demonstrates the effectiveness of the proposed MTL approach under different system configurations, investigating the impact of the observation window of traffic profiles on the classification accuracy, prediction loss, complexity, and convergence.

2.1 Unveiling Radio Resource Utilization Dynamics through Unsupervised Learning

In this section, a multivariate analysis on mobile traffic sessions (i.e., monitored mobile traffic associated with a certain service during its execution) collected at the radio link level is carried out to unveil and characterize radio resource utilization dynamics of mobile traffic. To this end, an unsupervised learning methodology (i.e.,

K-means) is used to identify similar radio resource utilization patterns of mobile traffic gathered from the *PDCCH* of an operating mobile network.

2.1.1 The proposed Clustering approach

With reference to the radio interface, LTE embraces two main communicating entities: the base station and the mobile terminal. At the physical layer, radio resources are distributed among mobile terminals in a time-frequency domain and the RB represents the smallest assignable radio resource unit. It lasts 1 ms in the time domain, namely Transmission Time Interval (TTI), and 180 kHz in the frequency domain. Every TTI, the base station allocates RBs to mobile terminals according to a specific scheduling algorithm. Transmission settings, expressed in terms of Modulation and Coding Scheme (MCS), are dynamically defined through a link adaptation mechanism. Moreover, the resulting amount of data to send during one TTI is fixed and depends on the selected MCS and the number of RBs assigned to a given mobile terminal, as reported by the Transport Block Size (TBS) table [127]. The scheduling decisions are shared with mobile terminals through control messages exchanged, at the beginning of the TTI, by using the PDCCH. Immediately after, data packets are exchanged through the Physical Downlink Shared CHannel (PDSCH).

The possibility of accessing to mobile traffic data represents a challenging task to accomplish. For security reasons, in fact, mobile operators avoid sharing their logs and data packets sent across the radio interface are encrypted. Nevertheless, control messages exchanged through the PDCCH are transmitted as clear text. This represents a valid opportunity to extract key information related to mobile traffic data, as well as generating reference datasets to be used for traffic analysis. In this context, data collection from the PDCCH has been already presented in [120], [121]. In those papers, an online decoder based on Software Defined Radio (SDR), namely OWL, is used to decode PDCCH messages sent by the base station within a given coverage area, thus collecting scheduling decisions every TTI. OWL generally produces a raw file. Then, a script can be used to generate a usable dataset that summarizes the main data of interest, associated with each captured traffic session.

The methodology proposed in this contribution evaluates mobile traffic sessions by investigating radio resource utilization dynamics in the downlink. Specifically, a multivariate analysis has been conceived to characterize mobile traffic sessions at the radio link level, according to their properties (the average data rate, the average MCS index, the average number of allocated RBs, and the duration of sessions). To this end, K-means [128], [129], which is a well-known unsupervised ML scheme, is used to map sessions with similar properties into K clusters. The number of clusters, i.e., K, is selected with the silhouette analysis, as detailed below. The variables of interest are firstly normalized within the range]0,1]. Then, each session is represented as a point in a hyperplane, whose dimensions refer to the variables of interest of the conducted analysis. At this point, the dissimilarity associated with two sessions is defined as the Euclidean distance between the two related points in the

aforementioned hyperplane. Indeed, the optimal value of K is calculated in order to ensure that the intra-cluster distances are minimized and the inter-cluster distances are maximized [129] (note that, according to K-means terminology, this means that the silhouette [130], [131] is maximized). Finally, the clustering process provides in output the sessions of each cluster and a special point of the hyperplane, namely *centroid*, that identifies the cluster itself. Their coordinates are obtained by averaging the value of variables associated with the sessions belonging to the considered cluster. By studying the obtained groups of sessions, it is then possible to extract statistical details associated with each cluster and characterize the traffic.

2.1.2 Mobile Traffic Analysis and Discussion

Two LTE base stations in a residential and campus area of Barcelona (Spain), operating in a bandwidth of 20 MHz, are monitored to collect mobile data. The residential area has been monitored from 6 February 2018 to 5 March 2018 and the resulting dataset contains 521 sessions. Instead, the campus area has been monitored from 22 March 2017 to 26 April 2017 and the resulting dataset contains 4946 sessions. The analysis proposed next discusses the properties of the gathered mobile data for each base station considering (i) the dataset as a whole and (ii) the dataset divided into 4 time-slots (i.e., parts of the day) that are morning, afternoon, evening, and night.

Study of the datasets as a whole

The two monitored base stations show different behavior in terms of radio resource usage patterns, as detailed below. The first difference refers to the output of the silhouette analysis, which is used to study the separation distance between the resulting clusters. For each session, the silhouette value, ranging from –1 to 1, is a measure of how similar that session is to sessions in its own cluster, when compared to the ones in other clusters: a high silhouette value indicates that the session is well matched to its own cluster, and poorly matched to other clusters. Since the silhouette analysis groups the residential and campus traffic in four and two clusters, respectively, a deeper study of sessions should be appropriate for characterization purposes (as carried out later). Figures 2.1 and 2.3 show the outcome of the K-means clustering process, carried out for the residential area and the campus area, respectively. For each variable of interest, the figures highlight the identified clusters, their centroids (i.e., the red dots), the 25^{th} and the 75^{th} percentile (i.e., the bottom line and the top line of the blue rectangle), as well as the minimum and the maximum measured value (i.e., the edges of the vertical red line) of the variables of interest. The number of sessions belonging to every single cluster and the related percentage, instead, are reported in Table 2.1.

Comments for the residential area. It is important to note that there is a strict relation between the number of sessions belonging to the cluster and the average data rate experienced by its traffic sessions. About 45% of the sessions report an

TABLE 2.1: Distribution of sessions among clusters.

Dataset	Cluster id	Number of sessions per cluster [#]	Percentage of sessions per cluster [%]
Residential area	1	235	45.11
	2	187	35.89
	3	82	15.74
	4	17	3.26
Campus area	1	4942	99.92
	2	4	0.08

average data rate equal to 0.46 Mbps. Instead, only 3.26% of them register an average data rate of 5.74 Mbps. Intermediate average data rates refer to intermediate groups of sessions (i.e., 35.89% and 15.74% of sessions present an average data rate of 1.33 Mbps and 2.60 Mbps, respectively).

Similar behavior is observed for MCS indexes and the allocated RBs. Figure 2.1(b) shows that the selected average MCS index is lower than 4 for about 45% of sessions. Only 3.26% of sessions use an average MCS index close to 10. Moreover, 35.89% and 15.74% of sessions use an average MCS index approximately equal to 6 and 8, respectively. Only intermediate clusters register peaks of MCS up to nearly 26. Considering that LTE allows a maximum MCS index of 31, these findings highlight that the channel quality experienced by mobile terminals during the monitored sessions is relatively scarce.

Very interesting details related to the distribution of radio resources among mobile terminals are depicted in Figure 2.1(c). About 45% of sessions, with an average data rate of 0.46 Mbps, consume the lowest amount of physical resources. Considering that 100 RBs per TTI are available in a slice of 20 MHz of bandwidth, an average number of RBs per TTI approximately equal to 23 means that sessions belonging to the first cluster occupy less than 1/4 of the overall amount of resources available within a cell. On the other hand, only 17 sessions consistently use a larger amount of resources per TTI, thus obtaining higher data rates.

A quite different behavior emerges from the analysis of the average session duration. Sessions that register the average data rate equal to 5.74 Mbps remain active for about 40 s, which is the lowest amount of time among the four clusters. For other clusters, instead, the duration increases with the number of sessions belonging to the cluster. It is also important to note that the session duration always presents an extremely high variability: the actual duration of 75% of sessions in each cluster is lower than the one associated with the related centroid.

To conclude, Figure 2.2 provides a summary of the obtained results. The proposed clustering methodology brings to important remarks concerning the usage of radio resources and the QoS requirements of analyzed traffic sessions. On average, 45% of sessions use about 1/4 of bandwidth per TTI for about 156 s; 35.89% of sessions use about 1/3 of bandwidth per TTI for about 190 s; 15.74% of sessions use almost 1/2 of bandwidth per TTI for about 634 s; 3.26% of sessions use 60% of bandwidth per TTI for about 40 s. Therefore, 235 out of 521 sessions consume

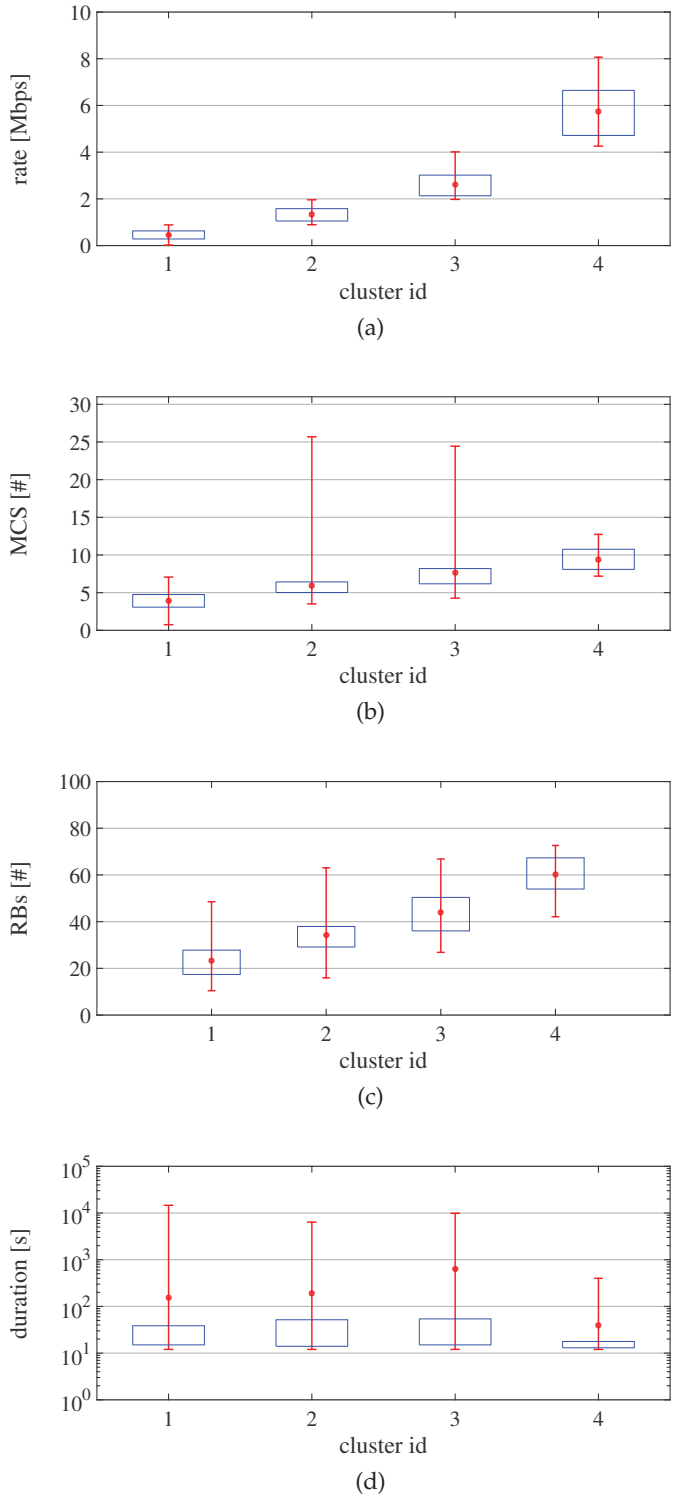


FIGURE 2.1: Study of dataset related to the residential area, as a whole: (a) rate, (b) MCS, (c) RBs, and (d) duration.

a small amount of resources for less than 3 minutes each. At the same time, there exists a little percentage of sessions (i.e., 3.26%), that register peaks of bandwidth consumptions, while lasting for a very short period.

Comments for the campus area. The campus area presents a number of sessions

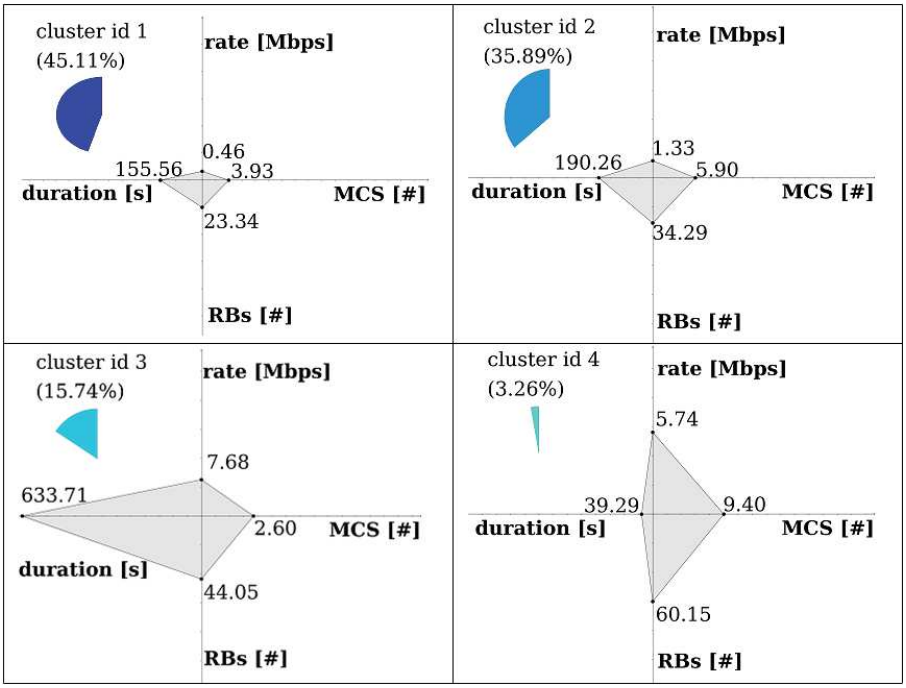


FIGURE 2.2: Clusters related to the residential area, as a whole.

extremely higher than the residential case, but the reported bandwidth requirements are extremely lower. As expected, there is a strict relation between the number of sessions per cluster and the average data rate. Nevertheless, almost all the sessions monitored in the campus area (i.e., 99.92%) fall within the same cluster and register a very low average data rate equal to 0.14 Mbps. Only 0.08% of sessions register an average data rate of 5.47 Mbps.

The study of MCS indexes provides a reverse relation, as shown in Figure 2.3(b). The former group of sessions experiences variable channel conditions, translating into the usage of all the admitted transmission settings. While the average MCS index is 13, the maximum value is equal to 31. The second group of sessions (4 out of 4946) registers worse channel conditions. In this case, the average and the maximum MCS indexes are about 7 and 10, respectively.

Figure 2.3(c) confirms what observed for the residential area: the higher the average number of RBs used per TTI, the higher the achieved data rate. Reported results still show that 4942 sessions use about 1/4 of the bandwidth per TTI. On the contrary, only 4 sessions use a larger amount of resources per TTI (i.e., more than 52).

As depicted in Figure 2.3(d), the campus area hosts sessions with very short durations. Apart from one exception (e.g., the graph reports one session duration equal to 1465 s), the former group of sessions registers an average session duration of 5 s. The duration of sessions belonging to the second cluster, instead, is lower than 2 s.

To conclude, Figure 2.4 provides a summary of obtained results. More than 99% of sessions use, on average, about 1/4 of bandwidth per TTI for about 5 s, and 0.08% of sessions use about 1/2 of bandwidth per TTI for about a second and a half. Thus,

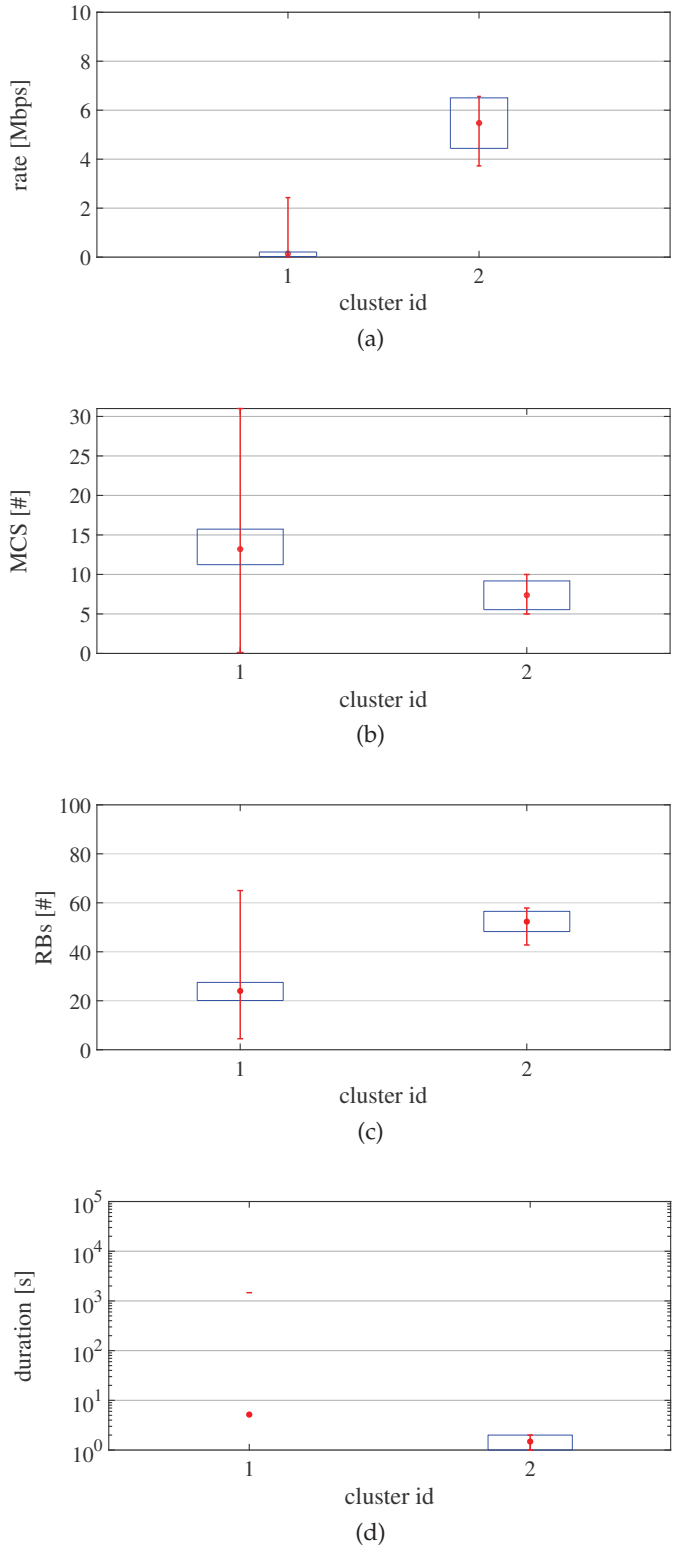


FIGURE 2.3: Study of dataset related to the campus area, as a whole: (a) rate, (b) MCS, (c) RBs, and (d) duration.

the campus area emerges as a place where a mobile network is called to manage many short sessions with limited bandwidth requirements.

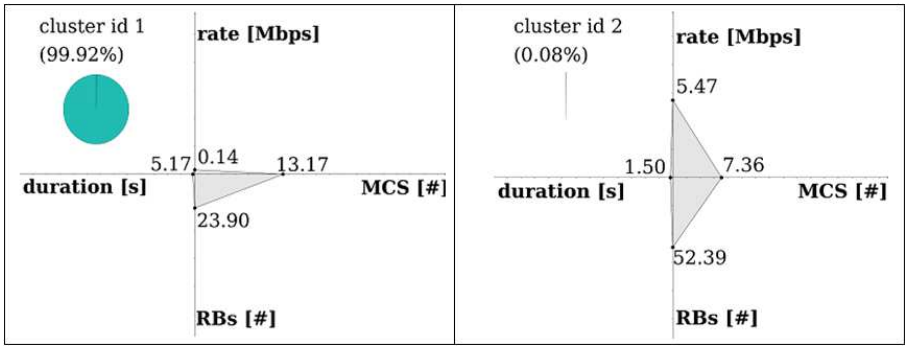


FIGURE 2.4: Clusters related to the campus area, as a whole.

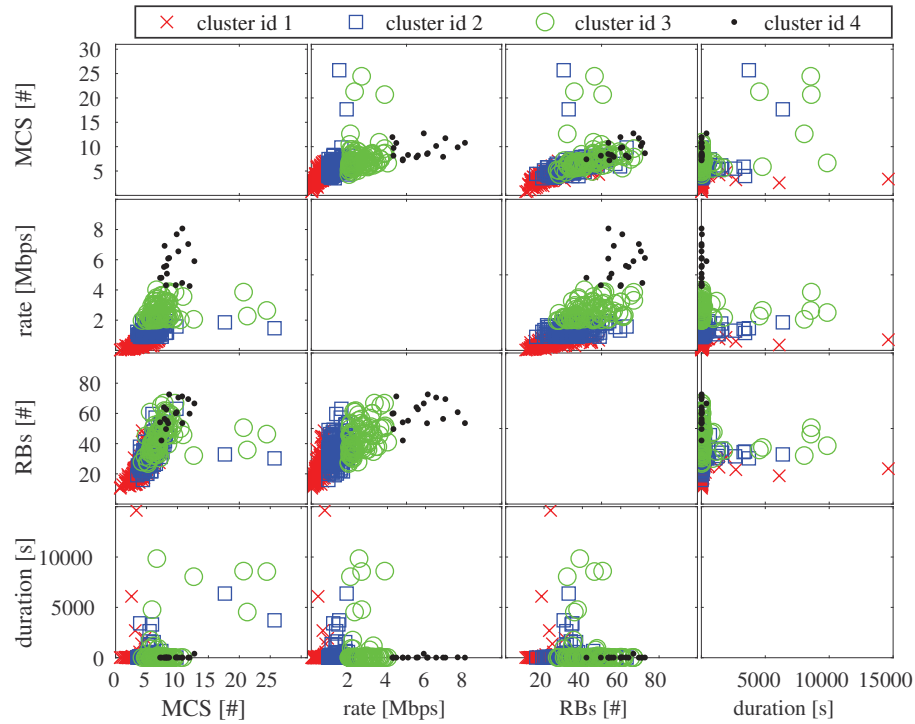
Scatter-plots analysis

The scatter-plot is a well-known cluster visualization technique [131], which allows to improve the study presented in Section 2.1.2, by highlighting the importance that every single variable of interest has in the multivariate analysis. The scatter-plot for both datasets is shown in Figure 2.5. Here, markers represent sessions and colors help distinguish clusters.

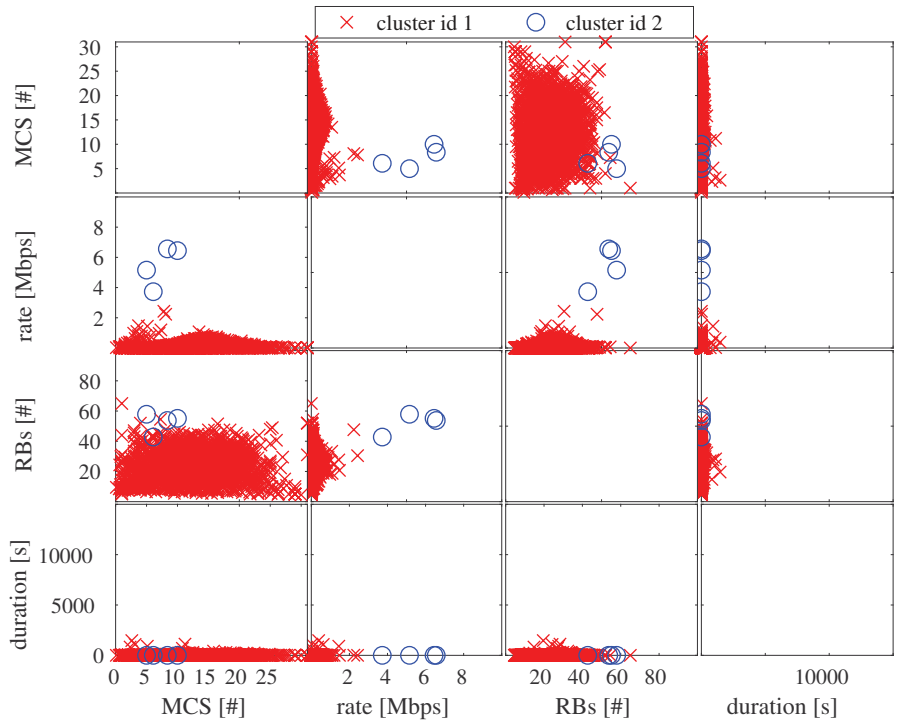
In general, the duration of sessions frequently emerges as a key parameter for classifying traffic according to the related application [31]. Nevertheless, for the evaluation of the radio resource utilization dynamics, it plays a minor role. In highly loaded scenarios, as the analyzed residential area, average MCS indexes, average data rate and average number of RBs allocated per TTI present a heavy weight in the clustering procedure: the scatter-plot reported in Figure 2.5(a) demonstrates that the joint study of these parameters allows the division of sessions among their reference clusters. Thus, as expected, it validates the suitability of the conceived multivariate feature design for clustering mobile traffic sessions. With reference to the campus area, reporting a lower traffic load, the data rate emerges as the main discriminatory parameter for traffic clusterization, as can be seen from the second column of Figure 2.5(b). In fact, the separation of clusters could be achieved by only considering the data rate with any other measured variable (that is the average MCS, the average RBs or the duration). However, a multivariate analysis, which considers more than two variables, is still useful to further characterize the properties of clusters from a different perspective.

Time-slot analysis

The analysis of mobile traffic on time-slots (i.e., parts of the day) basis leads to a detailed characterization of sessions, with a consequent more accurate identification of the resource usage and QoS requirements that a mobile network has to address during different parts of the day. The outcomes of the proposed clustering methodology on time-slots basis, applied to both residential and campus areas, are summarized in Tables 2.2 and 2.3, respectively.



(a) Dataset related to the residential area



(b) Dataset related to the campus area

FIGURE 2.5: Scatter-plot matrix.

Comments for the residential area. The relation between the number of sessions belonging to the cluster and the average variable registered by related traffic

TABLE 2.2: Study of the dataset related to the residential area, on time-slots (i.e., parts of the day) basis.

cluster id	**# of sessions**	**rate** [Mbps]			**MCS** [#]			**RBs** [#]			**duration** [s]		
		min	max	avg	min	max	avg	min	max	avg	min	max	avg
morning (from 6:00 am to 12:59 pm, 58 sessions)													
1	17	0.08	0.37	0.22	1.59	3.56	2.68	11.85	24.29	17.90	12	69	19.88
2	23	0.46	0.97	0.65	2.96	7.07	4.59	19.11	37.75	26.14	12	2696	324.39
3	18	0.99	2.17	1.31	4.28	6.83	5.75	26.75	47.93	34.64	12	3304	367.11
afternoon (from 1:00 pm to 7:59 pm, 297 sessions)													
1	120	0.06	0.86	0.48	1.15	6.95	4.11	12.08	48.55	23.72	12	478	51.67
2	116	0.87	1.80	1.26	3.51	25.69	5.80	19.09	63.05	33.79	12	3716	158.72
3	53	1.84	3.39	2.39	4.27	21.28	6.99	27.78	61.86	40.99	12	9845	598.09
4	8	3.56	6.12	4.47	7.39	11.96	9.49	45.81	72.62	61.34	12	20	15.38
evening (from 8:00 pm to 12:59 am, 134 sessions)													
1	78	0.09	1.58	0.79	1.98	7.45	4.67	12.98	57.24	27.12	12	135	26.14
2	46	1.64	4.31	2.48	5.14	24.44	8.07	26.86	66.82	44.36	12	8600	571.87
3	10	4.80	8.07	6.26	7.19	11.74	9	42.10	70.52	57.83	12	41	17.60
night (from 1:00 am to 5:59 am, 31 sessions)													
1	27	0.02	0.71	0.30	0.75	6.72	3.38	10.45	44.89	23.19	12	14630	793.81
2	2	1.15	1.60	1.38	3.97	6.87	5.42	34.77	48.56	41.67	20	3416	1718
3	1	2.85	2.85	2.85	7.83	7.83	7.83	46.86	46.86	46.86	126	126	126
4	1	5.91	5.91	5.91	12.73	12.73	12.73	66.62	66.62	66.62	399	399	399

TABLE 2.3: Study of the dataset related to the campus area, on time-slots (i.e., parts of the day) basis.

cluster id	**# of sessions**	**rate** [Mbps]			**MCS** [#]			**RBs** [#]			**duration** [s]		
		min	max	avg	min	max	avg	min	max	avg	min	max	avg
morning (from 6:00 am to 12:59 pm, 1225 sessions)													
1	681	<0.01	0.16	0.06	0.79	29	12.54	5	47.33	24.18	<1	292	8.45
2	538	0.16	0.79	0.25	1.55	20.93	14.44	10.83	38.25	24.99	<1	11	<1
3	6	0.88	2.43	1.42	3.70	14.19	8.60	20.73	33.34	27.73	<1	908	151.33
afternoon (from 1:00 pm to 7:59 pm, 1134 sessions)													
1	425	<0.01	0.07	0.02	1	29	11.35	5	49	24.18	<1	1114	12.04
2	298	0.07	0.17	0.12	6.27	27	13.80	11.65	54.50	22.66	<1	<1	<1
3	278	0.17	0.27	0.21	2.65	21.12	14.32	11.66	40.08	24.60	<1	3	<1
4	99	0.27	0.44	0.32	2.34	20	14.38	13.62	36.95	25.43	<1	1465	23.10
5	32	0.45	1.10	0.57	3.44	17.65	13.72	18.73	31.36	24.86	<1	<1	<1
6	1	2.23	2.23	2.23	8.09	8.09	8.09	47.64	47.64	47.64	16	16	16
7	1	3.73	3.73	3.73	6.09	6.09	6.09	42.78	42.78	42.78	1	1	1
evening (from 8:00 pm to 12:59 am, 848 sessions)													
1	846	<0.01	0.82	0.13	0.58	31	13.09	4.50	52	23.41	<1	234	3.04
2	2	5.16	6.45	5.80	5	9.99	7.49	55.16	57.86	56.51	2	2	2
night (from 1:00 am to 5:59 am, 1738 sessions)													
1	627	<0.01	0.09	0.03	0.12	26.67	11.22	4.75	52	22.37	<1	695	12.39
2	690	0.09	0.22	0.16	0.72	23.11	14.08	8.77	45.25	23.62	<1	1	<1
3	363	0.22	0.42	0.28	1.90	21.29	14.66	11.77	36.63	25.31	<1	11	<1
4	57	0.42	0.98	0.56	2.92	16.82	14.07	16.14	31	25.01	<1	1080	19.14
5	1	6.56	6.57	6.56	8.36	8.36	8.36	53.74	53.74	53.74	1	1	1

sessions still exists. About 30% of the morning sessions report an average data rate of 0.22 Mbps, while about 40% have an average data rate of 0.65 Mbps. During the afternoon, which is the part of the day with the highest number of active residential sessions, the data rate starts growing. In fact, about 40% of the afternoon sessions have an average data rate equal to 0.48 Mbps and the other 40% of sessions have an average data rate of 1.26 Mbps. The data rate still grows during the evening. The average data rate is 0.79 Mbps and 2.48 Mbps for about 60% and 35% of the evening sessions, respectively. Considering night sessions, whose number is limited because people tend to sleep, the average data rate goes down: about 87% of the

night sessions report an average value of 0.30 Mbps.

The average MCS index is lower than 5 for about 70% of the morning sessions. In particular, around 40% use an average MCS index close to 5 and around 30% even use an average MCS index approximately equal to 3. During the afternoon, average MCS indexes increase. In fact, about 40% of the afternoon sessions have an average MCS index close to 4 and a further 40% have an average value close to 6. The MCS indexes still grow during the evening, as the data rate does. The average MCS index is approximately 5 and 8 for about 60% and 35% of the evening sessions, respectively. As for the night sessions, MCS indexes tend to reduce. In fact, about 87% of the night sessions have an average value close to 3.

The distribution of radio resources follows a similar pattern. About 30% of the morning sessions report an average number of RBs per TTI equal to 1/6 of the overall amount of resources available within a cell, while about 40% have an average value equal to 1/4. During the afternoon, about 40% of sessions use an average number of RBs per TTI close to 1/4 of bandwidth per TTI and a further 40% stabilize to 1/3 for this aspect. During the evening, the average amount of resources per TTI is more than 1/4 and about 1/2 of the overall bandwidth for about 60% and 35% of sessions, respectively. Then, bandwidth consumptions decrease during the night: about 87% of the night sessions consume less than 1/4 of bandwidth per TTI.

The average duration, which varies greatly, has different behavior. Sessions generally show a short duration (i.e., 600 s), except during the night. In fact, about 87% of the night sessions last about 800 s. Moreover, around 6.5% have an average duration of 1718 s.

Comments for the campus area. The campus area reports a more balanced distribution of sessions among the whole-day slots. As expected, traffic characterization on time-slots basis offers a better characterization of sessions. For example, up to 7 clusters are identified for the afternoon time-slot, against the only two clusters reported for the analysis of the dataset as a whole.

Regarding the data rate in the campus area, more than 55% of the morning sessions report an average data rate of 0.06 Mbps, while about 40% have an average data rate of 0.25 Mbps. During the afternoon, the data rate tends to decrease. In fact, about 40% of the afternoon sessions report an average data rate equal to 0.02 Mbps and about 26% register an average data rate equal to 0.12 Mbps. The data rate is still low during the evening. In fact, the average value of 0.13 Mbps is measured for more than 99% of such sessions. During the night, that is the time-slot with the highest number of sessions, the average data rate tends to increase. From Table 2.3, it is 0.03 and 0.16 Mbps for about 36% and 40% of the night sessions, respectively.

The average MCS index is similar among the time-slots. In particular, about 55% of the morning sessions have an average value close to 13. About 40% of the afternoon sessions register an average MCS index close to 11, while about 26% and 25% use an average MCS index approximately equal to 14 and more than 14, respectively.

During the evening, the average MCS index is approximately 13 for 99.76% of sessions. Lastly, it is close to 11 and 14 for about 36% and 40% of the night sessions, respectively.

Also the allocated RBs per TTI have similar behavior. In particular, they slightly increase and decrease during the morning and the afternoon and during the evening and the night, respectively. About 56% of the morning sessions and 38% of the afternoon sessions report an average number of RBs per TTI close to 1/4 of the overall amount of resources available within a cell. Instead, the average amount of resources per TTI is more than 1/4 of the overall bandwidth for 99.76% of the evening sessions and about 76% of the night ones.

The average duration is extremely low during all the considered time-slots. In particular, about 56% of the morning sessions last about 9 s. Furthermore, about 38% of the afternoon sessions last longer than 10 s (i.e., about 12 s), while more than 50% (the clusters 2 and 3 in the afternoon) last less than 1 s. The average duration is approximately equal to 3 s for about 99% of the evening sessions. As the last report, about 36% of the night sessions last longer than 10 s (i.e., about 12 s), while around 60% last less than 1 s.

2.1.3 Lessons learned and network optimization opportunities

The proposed study shows that the analysis of mobile traffic across the different parts of the day (i.e., on time-slots basis) gives a deep insight into radio resource utilization dynamics. Interesting outcomes are summarized in what follows. As far as the residential area is concerned, a high number of sessions are measured in the afternoon and peaks of bandwidth requirements are measured both in the afternoon and in the evening. By observing data related to the night time-slot, it is possible to understand that a residential area significantly reduces its traffic load when people usually go to sleep. Nonetheless, differently from daily time-slots, the few sessions that remain active during the night present very high durations. As far as the campus area is concerned, sessions use a higher MCS index than residential sessions, but a very low rate: analyzed campus sessions do not transmit a lot of data, even if the quality of channel could be good, because the traffic load is not significant.

By knowing the radio resource utilization patterns of mobile traffic, possible challenges that can be addressed for optimizing mobile networks include:

- Advanced QoE/QoS management through dynamic radio resource scheduling algorithms exploiting the patterns and dynamics of mobile sessions at the radio link level;
- Dynamic and fine-grained management of slices and virtual functionalities offered through the radio access networks in 5G and B5G architectures;

- Optimal energy saving mechanisms (e.g., sleep mode of base stations and discontinuous reception in mobile terminals) and opportunistic handover management procedures that leverage the predicted behavior of classified traffic sessions;
- Planning for new base stations deployments in geographical regions where higher traffic load is expected;
- Massive usage of mobile base stations (i.e., deployed as drones), chasing the actual radio resource utilization dynamics.

2.2 Multi-Task Learning at the Mobile Edge

After tailoring an unsupervised learning methodology to characterize radio resource utilization patterns, an MTL model, running directly at the edge of the network, is conceived to perform data mining from the *PDCCH* control channel of an operative mobile network and *jointly* classify and predict mobile traffic. To this end, *autoencoders*, i.e., Undercomplete and Sequence to Sequence (Seq2Seq) architectures, are used as key building blocks of the proposed MTL methodology for common feature representations, shared by both classification and prediction tasks.

2.2.1 The proposed Multi-Task Learning approach

The developed methodology originates from the consideration that by observing such a traffic profile reporting the amount of data exchanged between the base station and the mobile terminal during a time interval T (i.e., radio resource utilization pattern), it could be possible to classify the application type the investigated session belongs to (task 1) and predict the radio resource utilization pattern that the session will experience in the upcoming time instants (task 2). This goal is successfully achieved through an MTL architecture running directly at the edge of a mobile network (Figure 2.6). Without loss of generality, the contribution directly focuses on the downlink communication, but it should be remarked that the contribution can be applied to the uplink as well. Moreover, note that Figure 2.6 depicts one base station, but it is precisely in view of the Pervasive Intelligence paradigm that the MTL approach can be pervasively applied to all the base stations in the network.

To facilitate the understanding of the notations adopted in what follows, a summary of symbols is reported in Table 2.4.

Following these initial considerations, the proposed MTL approach grounds its roots into the *feature learning representation* concept [54], according to which the features for a common representation of the input (i.e., traffic profiles) are extracted and jointly used to execute the two tasks (i.e., classification and prediction). In particular, the conceived methodology uses an autoencoder to obtain the common feature representations of input data for the two tasks because it can directly accomplish

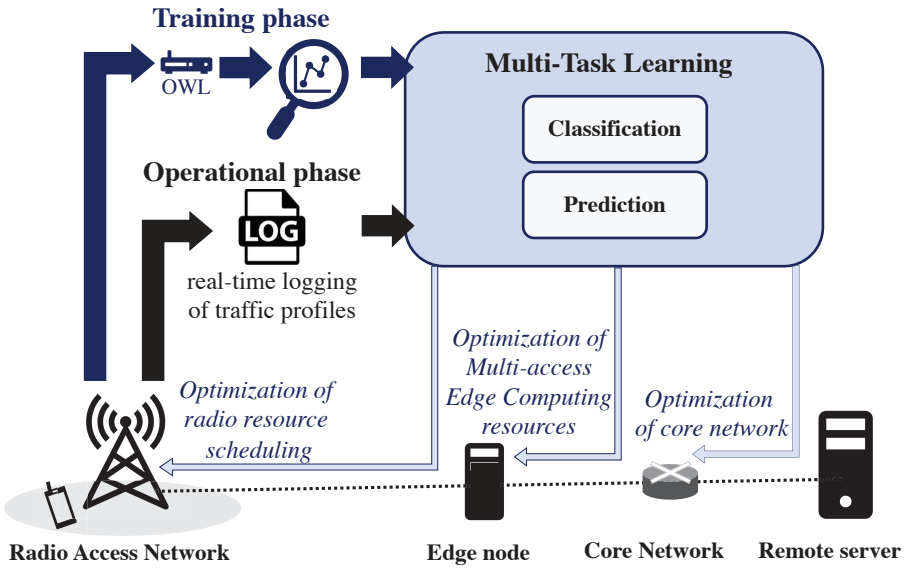


FIGURE 2.6: Input and output of the proposed MTL approach in a mobile network.

this operation without requiring the knowledge of data distribution nor the explicit identification of a certain structure [37]. Classification and prediction tasks are later executed through softmax and fully-connected layers, respectively. Accordingly, the autoencoder is a key building and enabling block of the proposed MTL methodology, that effectively allows the joint execution of classification and prediction tasks.

As depicted in Figure 2.6, the outcomes of the proposed scheme can be exploited to implement advanced methodologies for the management and the optimization of mobile networks. This approach is conceived to process data directly at the edge, so that the right actions may be triggered faster and locally. Possible strategies that may benefit from the implementation of the proposed architecture range from radio resource scheduling and admission control, mobility management and energy saving mechanisms, to network slicing and dynamic placement of virtualized functions, as well as to the optimization of computing resources at both edge and core network (see Figure 2.6). Nevertheless, note that the rest of this Chapter focuses on the MTL approach and the reference dataset taken into account for training purposes.

The training dataset

Being the proposed approach intended to work at the mobile edge, data exchanged through the radio interface are needed for training the model. An operator owing the mobile infrastructure can simply retrieve this information and use it for both the training and operating phases. However, the adopted dataset [62] consists of traffic traces containing the Downlink Control Information (DCI) messages carried within the PDCCH with a time granularity of 1 ms. This information is used by the eNodeB to communicate scheduling information to the connected mobile terminals. DCI messages are unencrypted and decoded by the hardware/software tool called OWL [120], already presented in Section 2.1.1. A key characteristic of the training dataset is

TABLE 2.4: List of mathematical symbols adopted in Chapter 2.

Symbol	Description
i	Traffic session index
j	Time instant index
$\boldsymbol{\mathcal{D}}$	Original input matrix with traffic profiles
$\mathbf{d_i}$	Row vector in $\boldsymbol{\mathcal{D}}$ that represents traffic session
$r_{\boldsymbol{\mathcal{D}}}$	Number of rows (traffic sessions) in $\boldsymbol{\mathcal{D}}$
Δ	Number of columns (time instants) in $\boldsymbol{\mathcal{D}}$
$\mathbf{c}$	Column vector of labels associated with $\boldsymbol{\mathcal{D}}$
c_i	Label (i.e., component of $\mathbf{c}$) associated with $\mathbf{d_i}$
T	Observation window
$\boldsymbol{\mathcal{M}}$	Pre-processed input matrix with traffic profiles
$\mathbf{m_i}$	Row vector in $\boldsymbol{\mathcal{M}}$ that represents traffic session lasting T
$m_{i,j}$	Component of $\mathbf{m_i}$ during the j-th time instant
$r_{\boldsymbol{\mathcal{M}},tr}$	Number of rows of $\boldsymbol{\mathcal{M}}$ selected as training set
$\boldsymbol{\mathcal{H}}$	Codeword matrix with feature learning representations of $\boldsymbol{\mathcal{M}}$
$\mathbf{h_i}$	Feature learning representation (i.e., component of $\boldsymbol{\mathcal{H}}$) of $\mathbf{m_i}$
$\hat{\boldsymbol{\mathcal{M}}}$	Reconstructed input matrix
$\hat{\mathbf{m}}_\mathbf{i}$	Reconstructed traffic session in $\hat{\boldsymbol{\mathcal{M}}}$
$\hat{m}_{i,j}$	Reconstructed component of $\hat{\mathbf{m}}_\mathbf{i}$ during the j-th time instant
$\mathbf{l}$	Column vector of labels associated with $\boldsymbol{\mathcal{M}}$
l_i	Label (i.e., component of $\mathbf{l}$) associated with $\mathbf{m_i}$
$\hat{\mathbf{l}}$	Column vector of learned labels associated with $\boldsymbol{\mathcal{M}}$
$\hat{l}_i$	Learned label (i.e., component of $\hat{\mathbf{l}}$) associated with $\mathbf{m_i}$
$\mathbf{m_{T+1}}$	Column vector with data exchanged at $T+1$
$m_{i,T+1}$	Component subsequent to $\mathbf{m_i}$ with data exchanged at $T+1$
$\hat{\mathbf{m}}_{\mathbf{T+1}}$	Predicted column vector with data exchanged at $T+1$
$\hat{m}_{i,T+1}$	Predicted component at $T+1$ related to $\mathbf{m_i}$
$\mathcal{L}_\mathcal{A}$	Mean Square Error (loss) of the Autoencoder
$\mathcal{L}_\mathcal{C}$	Mean Square Error (loss) of the Classifier
$\mathcal{A}_\mathcal{C}$	Classifier accuracy
$\mathcal{L}_\mathcal{P}$	Mean Square Error (loss) of the Predictor
$\mathcal{P}_{\mathcal{MTL}}$	Multi-objective performance metric for the MTL model

that it is gathered from the control channel, which simplifies the monitoring system complexity, assures fast data processing, and reduces the storage capacity due to the limited volume of data.

The captured traces are generated by different applications running in a mobile terminal under control and attached to an operative mobile network in Spain. Six different applications grouped in three categories have been tested: YouTube and Vimeo for *video-streaming*, Spotify and Google Music for *audio-streaming*, and Skype and WhatsApp Messanger for *video-call*. Those applications have been selected because they generally produce, according to recent Ericsson [132] and Cisco [133] reports, more than 80% of the mobile data traffic and require optimal resource management due to their strict quality requirements. The proposed approach, however, can be applied to other mobile network scenarios with a different set of applications

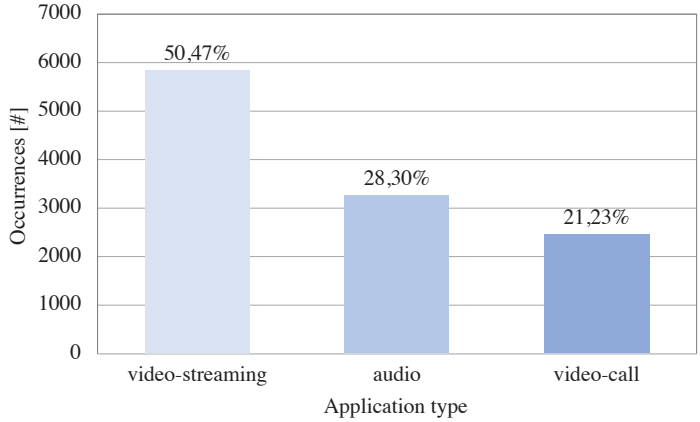


FIGURE 2.7: Number of sessions vs application types in the considered dataset.

and services, only requiring a new training procedure. Also, after an effective training, the adopted methodology can be extended to any number of classes because it is general and not restricted to a specific use-case (see Section 2.2.2 for more details).

Among the several parameters extracted from the DCI messages, the TBS, which specifies the length of the packet burst to be sent to/from the considered mobile terminal in the current time slot [127], has been used. Then, TBS values are processed to generate the radio resource utilization patterns describing the amount of data exchanged between the base station and mobile terminal, with a time granularity of 1 s.

Formally, let $r_{\mathcal{D}}$ be the number of traffic sessions collected in a period of time equal to Δ. In this work, $r_{\mathcal{D}} = 11574$ and $\Delta = 60$s. The distribution of the sessions among the considered application categories is reported in Figure 2.7. The original training dataset contains a matrix $\boldsymbol{\mathcal{D}}$ and a vector $\mathbf{c}$ of labels. In particular, the original input matrix $\boldsymbol{\mathcal{D}}$ describes the captured traffic profiles (also referred to as the radio resource utilization patterns) of $r_{\mathcal{D}}$ different sessions for an amount of time equal to Δ. Thus, the matrix $\boldsymbol{\mathcal{D}}$ has a dimension of $r_{\mathcal{D}} \times \Delta$, where $r_{\mathcal{D}}$ and Δ are the number of rows (traffic sessions) and the number of columns (time instants) in $\boldsymbol{\mathcal{D}}$. The vector $\mathbf{c}$ of labels contains the application type of the controlled sessions, with a dimension of $r_{\mathcal{D}} \times 1$. For example, given the i-th investigated session, it holds that $d_{i,j} \in \boldsymbol{\mathcal{D}}$ and $c_i \in \mathbf{c}$ are the amount of data delivered across the radio interface during the j-th time slot and the label describing the application type of the i-th session, respectively. All the values stored in $\boldsymbol{\mathcal{D}}$ are normalized within the range [0,1] to accelerate the training convergence [134].

The training dataset has been conveniently pre-processed to be managed by the proposed deep learning models. For the sake of clarity, the pre-processing procedure has been depicted in Figure 2.8. A new matrix $\boldsymbol{\mathcal{M}}$ is generated from $\boldsymbol{\mathcal{D}}$, whose rows represent the observation windows of duration T. The resulting matrix $\boldsymbol{\mathcal{M}}$ has a dimension of $r_{\mathcal{D}}(\Delta - T + 1) \times T$. The vector $\mathbf{c}$ is used to generate a new set of labels, namely $\mathbf{l}$, describing the application type associated with each portion of the investigated session stored in $\boldsymbol{\mathcal{M}}$. The vector $\mathbf{l}$ has a dimension of $r_{\mathcal{D}}(\Delta - T + 1) \times$

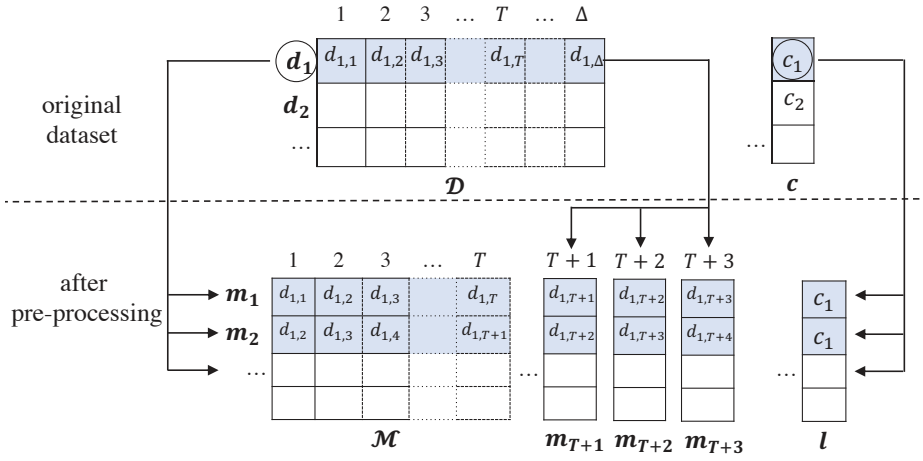


FIGURE 2.8: Pre-processing of the training dataset.

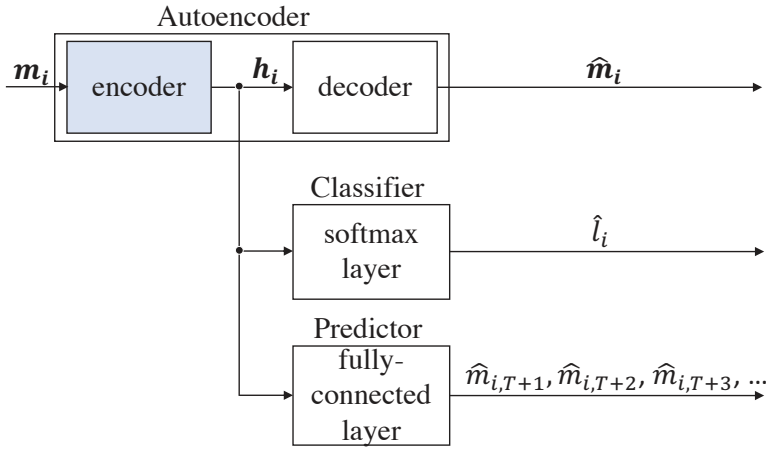


FIGURE 2.9: The proposed MTL model.

1. A set of new column vectors, namely $\mathbf{m_{T+1}}$, $\mathbf{m_{T+2}}$, and $\mathbf{m_{T+3}}$, with dimension $r_{\mathcal{D}}(\Delta - T + 1) \times 1$, are generated from $\mathcal{D}$ to store the amount of data exchanged between base station and the mobile terminal after the observation window T.

Finally, 80% of $\mathcal{M}$ is used as training set, while the remaining 20% is used as validation set. The number of rows of the matrix $\mathcal{M}$ selected as training set, whose performance will be listed and evaluated, is simply denoted by $r_{\mathcal{M},tr}$.

Components of the developed MTL model

Figure 2.9 shows the proposed MTL model, embracing three main components: autoencoder, classifier, and predictor. Each component presents specific input and output parameters. The training of the developed MTL model is divided into two stages. The first stage consists of the training of the autoencoder. The second stage refers to the training of both classifier and predictor, known the set of feature learning representations provided by the encoder.

The autoencoder: It represents a particular ANN implementing two key functionalities. Given an input data $\mathbf{m_i} = \{m_{i,1}, ..., m_{i,T}\}$, that is a row of the matrix $\boldsymbol{\mathcal{M}}$, the encoder generates the corresponding feature representation, namely $\mathbf{h_i}$, which then allows the joint execution of the two tasks. Specifically, $\mathbf{h_i} \in \boldsymbol{\mathcal{H}}$ appears like a compression of input data [37] and it is referred to as codeword in the next sections. On the other hand, the decoder provides a reconstruction of the input data, namely $\hat{\mathbf{m}}_\mathbf{i} = \{\hat{m}_{i,1}, ..., \hat{m}_{i,T}\}$, starting from the aforementioned feature learning representation. The autoencoder uses the sigmoid activation function for the output layer and Rectified Linear Unit (ReLU) for other layers [17]. In addition, it also uses weights, that are properly configured during the training phase.

This work investigates two different autoencoder schemes:

- the *Undercomplete Autoencoder*, leveraging regular densely-connected neural network layers, based on MLP [135]. In particular, MLP is a fully-connected and feed-forward neural network, that has low computational complexity.

- the *Seq2Seq Autoencoder*, that manages encoder and decoder functionalities through LSTM [136]. The LSTM is a popular variant of Recurrent Neural Networks (RNNs) that can extract long range temporal dependencies through input, forget, and output gates and mitigate gradient vanishing and exploding problems. This type of neural network is suitable for processing time series because the output of each memory cell may depend on the entire sequence of previous cell states [17], [41], [137]. Due to the intrinsic temporal relations in mobile traffic data, LSTM-based architecture appears as the logical choice, at the cost of higher computational complexity.

To train the two types of autoencoder, weights are iteratively updated in order to minimize the Mean Square Error (MSE) loss function $\mathcal{L}_\mathcal{A}$, formally defined as [135], [138]:

$$\mathcal{L}_\mathcal{A} = \frac{1}{r_{\boldsymbol{\mathcal{M}},tr}} \sum_{i=1}^{r_{\boldsymbol{\mathcal{M}},tr}} \sum_{j=1}^{T} (m_{i,j} - \hat{m}_{i,j})^2 \tag{2.1}$$

As shown in Figure 2.9, the common feature representation $\mathbf{h_i}$ generated by the autoencoder is provided to both classifier and predictor for driving classification and prediction tasks.

The classifier: It maps the feature learning representation $\mathbf{h_i}$ to a learned label $\hat{l}_i$ describing the application type of the investigated session. To this end, it uses the softmax layer, based on the softmax activation function [17], working with a number of classes (i.e., the considered application types) equal to 3, even if the conceived methodology is extendable to any number of classes.

The softmax layer of the classifier is configured by penalizing the MSE loss function $\mathcal{L}_\mathcal{C}$ between the true label l_i associated with the input data $\mathbf{m_i}$ and the learned

label $\hat{l}_i$ associated with the feature learning representation $\mathbf{h_i}$:

$$\mathcal{L}_{\mathcal{C}} = \frac{1}{r_{\mathcal{M},tr}} \sum_{i=1}^{r_{\mathcal{M},tr}} (l_i - \hat{l}_i)^2. \tag{2.2}$$

Once configured, the classifier accuracy $\mathcal{A}_{\mathcal{C}}$ quantifies the percentage of correct classifications with respect to the total number of classifications [139]:

$$\mathcal{A}_{\mathcal{C}} = \frac{\text{number of correct classifications}}{r_{\mathcal{M},tr}} \cdot 100. \tag{2.3}$$

The predictor: It predicts the amount of data that a given session is expected to exchange with the base station after the observation window T, that are: $\hat{m}_{i,T+1}$ stored in $\hat{\mathbf{m}}_{\mathbf{T+1}}$, $\hat{m}_{i,T+2}$ stored in $\hat{\mathbf{m}}_{\mathbf{T+2}}$, $\hat{m}_{i,T+3}$ stored in $\hat{\mathbf{m}}_{\mathbf{T+3}}$, and so on. It makes use of a fully-connected layer with the ReLU activation function [17].

The predictor is configured in order to minimize the MSE loss function $\mathcal{L}_{\mathcal{P}}$, formulated for $T + 1s$ as [140]:

$$\mathcal{L}_{\mathcal{P}} = \frac{1}{r_{\mathcal{M},tr}} \sum_{i=1}^{r_{\mathcal{M},tr}} \left(m_{i,T+1} - \hat{m}_{i,T+1} \right)^2. \tag{2.4}$$

Of course, it is expected that the prediction loss function, which minimizes the difference between the true and the predicted amount of exchanged data, will increase with the time distance between the latest value of the investigated traffic profile and the predicted one.

2.2.2 Performance Evaluation

The conceived MTL architecture has been implemented in Keras, a high-level neural networks API written in Python, running on top of TensorFlow [141], and simulations have been executed on an Intel Core i7 CPU with 16 GB of RAM. Moreover, different configurations of neural networks are investigated to quantify the impact of the observation window, T, on the classifier accuracy, $\mathcal{A}_{\mathcal{C}}$, and the prediction loss, $\mathcal{L}_{\mathcal{P}}$. Once the best solutions are selected, a complete analysis on the classification and prediction performance together with a discussion on the complexity and convergence of the selected architectures is presented.

To simplify the understanding of the analysis presented in this section, the proposed MTL architectures are named as follows: MTL-U refers to the MTL architecture based on the Undercomplete Autoencoder; MTL-S2S refers to the MTL architecture based on the Seq2Seq Autoencoder.

Assuming to describe the ratio between the size of the input layer and the size of hidden layers in the form *X:Y* for the neural networks with only one hidden layer and *X:Y:Z* for the neural networks with two hidden layers, the investigated configurations include *8:5*, *8:6*, *8:8*, and *8:5:3*. The observation window T is chosen in the

TABLE 2.5: Performance of MTL-U. For each T, the Best Configuration is highlighted.

T [s]	Codeword	MTL-U											
		1 hidden layer									2 hidden layers		
		8:5			8:6			8:8			8:5:3		
		$\mathcal{L}_{\mathcal{A}}$ [$\cdot 10^{-3}$]	$\mathcal{A}_{\mathcal{C}}$ [%]	$\mathcal{L}_{\mathcal{P}}$ [$\cdot 10^{-3}$]	$\mathcal{L}_{\mathcal{A}}$ [$\cdot 10^{-3}$]	$\mathcal{A}_{\mathcal{C}}$ [%]	$\mathcal{L}_{\mathcal{P}}$ [$\cdot 10^{-3}$]	$\mathcal{L}_{\mathcal{A}}$ [$\cdot 10^{-3}$]	$\mathcal{A}_{\mathcal{C}}$ [%]	$\mathcal{L}_{\mathcal{P}}$ [$\cdot 10^{-3}$]	$\mathcal{L}_{\mathcal{A}}$ [$\cdot 10^{-3}$]	$\mathcal{A}_{\mathcal{C}}$ [%]	$\mathcal{L}_{\mathcal{P}}$ [$\cdot 10^{-3}$]
5	3	0.83	89.20	2.56	6.69	**93.02**	**2.54**	5.68	91.72	2.50	23.10	86.86	2.60
	4	0.83	88.39	2.56	4.25	91.22	2.50	2.12	92.37	2.53	23.09	87.51	3.49
10	3	6.64	95.02	1.77	6.04	95.41	1.70	2.26	91.93	1.63	8.20	93.21	1.67
	4	2.93	93.47	1.73	4.98	**96.99**	**1.66**	3.94	94.61	1.61	6.73	92.99	1.68
	5	2.06	91.02	1.73	4.85	95.45	1.72	2.12	92.90	1.61	4.49	93.33	1.69
15	3	4.32	90.56	1.27	5.12	94.75	1.18	2.28	90.51	1.15	4.25	94.95	1.21
	4	3.70	90.16	1.24	5.15	93.90	1.17	0.57	94.92	1.15	2.86	94.19	1.19
	5	1.97	93.82	1.18	3.62	98.08	1.20	0.25	94.37	1.14	4.80	97.53	1.16
	10	3.61	96.21	1.18	2.69	96.31	1.11	2.49	**98.75**	**1.03**	1.82	93.51	1.22
20	3	3.84	88.97	0.96	4.56	97.64	0.91	2.50	90.50	0.83	1.43	95.41	0.88
	4	0.52	94.57	0.95	3.32	99.43	0.88	0.30	94.94	0.81	3.38	95.22	0.85
	5	0.43	94.91	0.91	3.44	98.26	0.90	0.34	91.16	0.79	3.04	94.10	0.91
	10	2.29	96.60	0.90	2.48	**99.36**	**0.81**	1.80	97.23	0.73	1.56	94.95	0.87

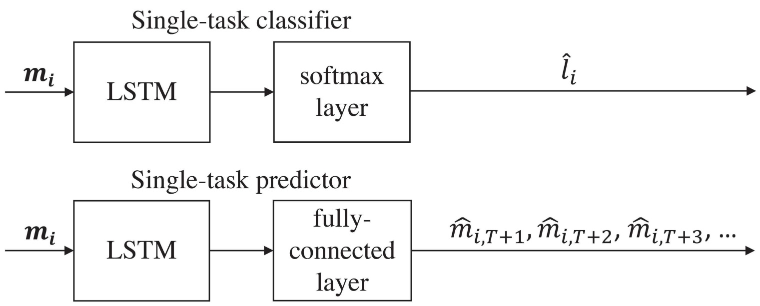


FIGURE 2.10: Baseline single-task learning architectures for classifier and predictor.

range from 5 to 20. Regarding the autoencoder, the size of the codeword is also set to different values (please see Tables 2.5 and 2.6 for further details).

The training phase for all the components belonging to the designed MTL architectures lasts 200 epochs (i.e., complete passes through the training data [142]). The Adam optimization is used to iteratively update the network weights based on the training data [143].

To provide further insight, the comparison with baseline single-task learning architectures, that do not use the autoencoder and that tackle traffic classification and prediction in isolation, is presented as well. In particular, the reference single-task architectures selected for the cross-comparison are based on LSTM because, as stated in Section 2.2.1, this type of neural network is suitable for processing time series. Furthermore, due to the wide adoption of LSTM in the state-of-the-art deep learning models (e.g., [41], [45]–[47], [50]), LSTM-based architecture appears as the logical choice for the comparison (single-task learning) schemes, as well as for the MTL approach. Assuming to work with the same training dataset and to adopt the same set of symbols, the single-task classifier and the single-task predictor are depicted in Figure 2.10.

TABLE 2.6: Performance of MTL-S2S. For each T, the Best Configuration is highlighted.

T [s]	Codeword	MTL-S2S											
		1 hidden layer									2 hidden layers		
		8:5			8:6			8:8			8:5:3		
		$\mathcal{L}_\mathcal{A}$ $[\cdot 10^{-3}]$	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{A}$ $[\cdot 10^{-3}]$	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{A}$ $[\cdot 10^{-3}]$	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{A}$ $[\cdot 10^{-3}]$	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$
5	3	0.014	92.72	2.41	0.024	92.40	2.48	0.058	92.25	2.33	16.68	90.09	2.44
	4	16.72	92.56	2.50	0.011	92.35	2.39	16.70	92.10	2.39	16.73	90.59	2.45
	5	0.027	94.55	2.46	0.048	94.26	2.42	0.017	**94.59**	**2.33**	16.70	90.36	2.51
10	3	0.0081	96.52	1.55	0.016	97.19	1.51	0.029	93.53	1.50	16.69	92.22	1.54
	4	0.0045	96.71	1.51	0.0099	96.69	1.53	0.0045	95.67	1.40	15.50	96.44	1.61
	5	0.019	96.17	1.51	0.021	**97.33**	**1.46**	0.0032	94.45	1.38	0.060	97.52	1.54
15	3	0.0066	96.21	1.17	0.023	97.15	1.07	0.011	98.03	0.87	16.68	91.99	0.97
	4	11.16	98.54	1.01	0.018	95.48	1.03	0.011	97.75	1.02	11.13	96.60	1.22
	5	0.0083	98.16	1.03	0.014	95.22	0.98	0.0034	98.26	0.95	0.0076	95.48	1.01
	10	0.0029	97.96	0.91	0.0048	98.06	0.86	0.0075	**99.33**	**0.81**	0.019	95.41	1.01
20	3	0.014	97.85	0.78	0.0035	92.52	0.66	0.012	93.68	0.61	0.0047	98.47	0.70
	4	0.0087	98.18	0.75	0.0035	94.07	0.74	0.0060	98.50	0.67	0.016	97.56	0.77
	5	0.0039	98.00	0.81	0.0045	94.92	0.75	0.0032	97.84	0.61	0.019	98.23	0.75
	10	0.0032	98.95	0.77	0.0034	97.92	0.67	0.0072	**99.64**	**0.62**	0.0055	95.02	0.68

TABLE 2.7: Performance of the single-task approach. For each T, the Best Configuration is highlighted.

T [s]	Single-task classifier			
	1 hidden layer			2 hidden layers
	8:5	8:6	8:8	8:5:3
	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{A}_\mathcal{C}$ [%]	$\mathcal{A}_\mathcal{C}$ [%]
5	88.02	**92.52**	91.44	90.81
10	95.56	94.69	**95.97**	93.22
15	96.76	96.60	**97.18**	96.07
20	95.36	**97.73**	97.52	97.04

T [s]	Single-task predictor			
	1 hidden layer			2 hidden layers
	8:5	8:6	8:8	8:5:3
	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$	$\mathcal{L}_\mathcal{P}$ $[\cdot 10^{-3}]$
5	2.64	2.56	2.47	**2.43**
10	1.77	1.67	**1.48**	1.55
15	1.22	1.12	**0.97**	1.11
20	0.91	0.84	**0.77**	0.79

Selection of suitable MTL architectures

Autoencoder loss, $\mathcal{L}_\mathcal{A}$, classification accuracy, $\mathcal{A}_\mathcal{C}$, and prediction loss, $\mathcal{L}_\mathcal{P}$, achieved for all the configurations of the designed MTL architectures are reported in Tables 2.5 and 2.6. The same performance indexes obtained with single-task approaches are reported in Table 2.7. For both MTL and single-task architectures and for each observation window T, these results are used to select the configurations that ensure the best performance.

Regarding the conceived MTL architectures, the analysis concerns multiple objectives, that refer to the maximization of $\mathcal{A}_\mathcal{C}$ and the minimization of $\mathcal{L}_\mathcal{P}$. To this end, a performance metric, $\mathcal{P}_{\mathcal{MTL}}$, is defined in (2.5) as a weighted linear sum of obtained results for each task:

$$\mathcal{P}_{\mathcal{MTL}} = \alpha \mathcal{A}_\mathcal{C} + (1-\alpha)\left[\frac{\mathcal{L}_\mathcal{P} - \mathcal{L}_{\mathcal{P}min}}{\mathcal{L}_{\mathcal{P}max} - \mathcal{L}_{\mathcal{P}min}}(\mathcal{L}'_{\mathcal{P}max} - \mathcal{L}'_{\mathcal{P}min}) + \mathcal{L}'_{\mathcal{P}min}\right] \tag{2.5}$$

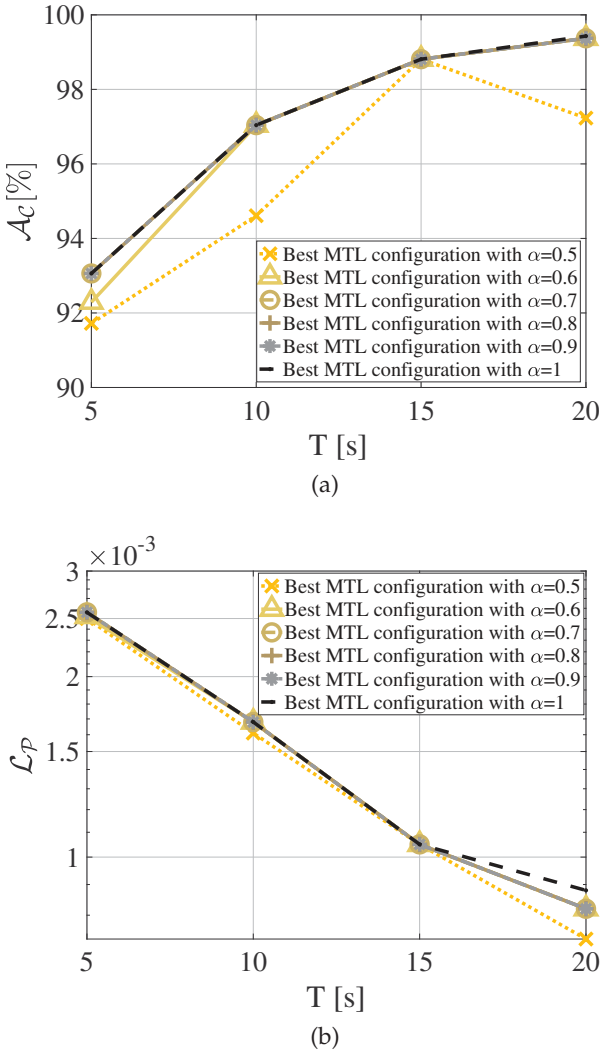


FIGURE 2.11: (a) Classification accuracy and (b) prediction loss at $T + 1s$ registered by the best configurations of MTL-U, as a function of α.

where the weight α may assume an arbitrary value from 0 to 1 [144], [145]. Since the higher the loss, the lower the performance, the min-max normalization is performed for $\mathcal{L}_\mathcal{P}$ to properly combine the two metrics [139], considering the minimum prediction loss reported in Tables 2.5, 2.6, and 2.7 (i.e., $\mathcal{L}_{\mathcal{P}min}$), the maximum prediction loss reported in Tables 2.5, 2.6, and 2.7 (i.e., $\mathcal{L}_{\mathcal{P}max}$), the value of the normalized metric describing the worst performance (i.e., $\mathcal{L}'_{\mathcal{P}max} = 0$), and the value of the normalized metric describing the best performance (i.e., $\mathcal{L}'_{\mathcal{P}min} = 100$).

Figures 2.11 and 2.12 show the performance of the MTL configurations achieving the highest $\mathcal{P}_{\mathcal{MTL}}$ metric as a function of α, for MTL-U and MTL-S2S, respectively. These figures help to identify the suitable values of α to be used for the selection of the best MTL configurations. Reported curves demonstrate that $\alpha = 0.5$ and $\alpha = 1$ cannot be used for this purpose. In fact, if $\alpha \leq 0.5$, the multi-objective metric $\mathcal{P}_{\mathcal{MTL}}$ suggests to select configurations having low classification accuracy. On the contrary, when $\alpha = 1$, the multi-objective metric $\mathcal{P}_{\mathcal{MTL}}$ suggests to select configurations that

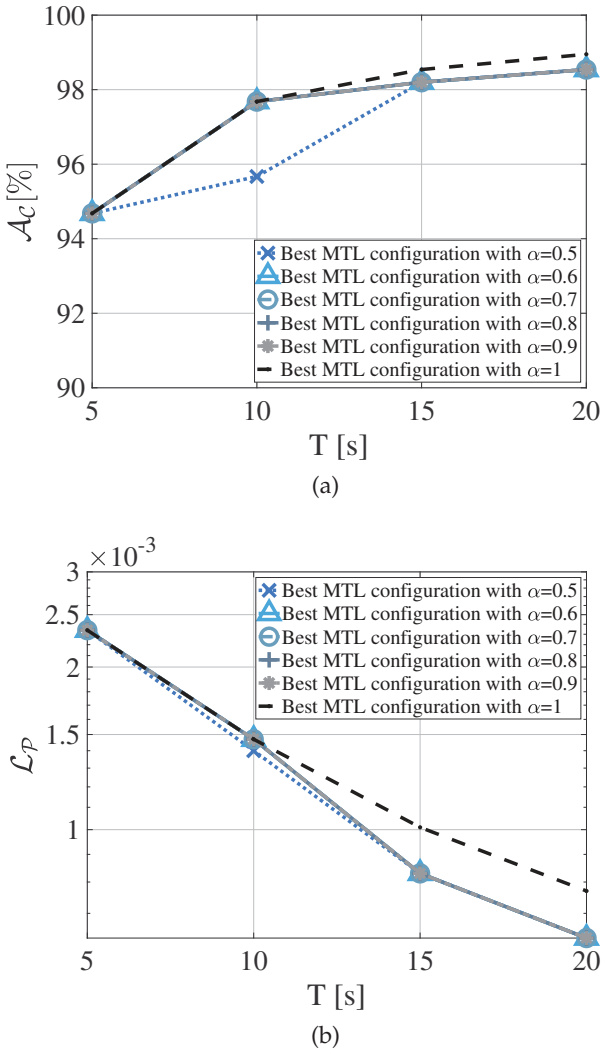


FIGURE 2.12: (a) Classification accuracy and (b) prediction loss at $T + 1s$ registered by the best configurations of MTL-S2S, as a function of α.

register higher prediction losses, especially when T increases. Other values of α provide similar outcomes. Thus, the rest of this study considers the best configurations of the proposed MTL architectures selected with $\alpha = 0.8$. They are highlighted in Tables 2.5 and 2.6.

Regarding the single-task approaches, the best configurations are simply selected by considering those that offer better performance for each T. Also in this case, they are highlighted in Table 2.7.

In general, it can be noted that the performance of both MTL and single-task approaches improve for an increasing T because more data are used to make decisions. Focusing the attention on the proposed MTL model, there is not a precise relationship between the MTL performance and the codeword size: while MTL-S2S always achieves the best performance with the biggest codeword size, the same consideration cannot be made for MTL-U.

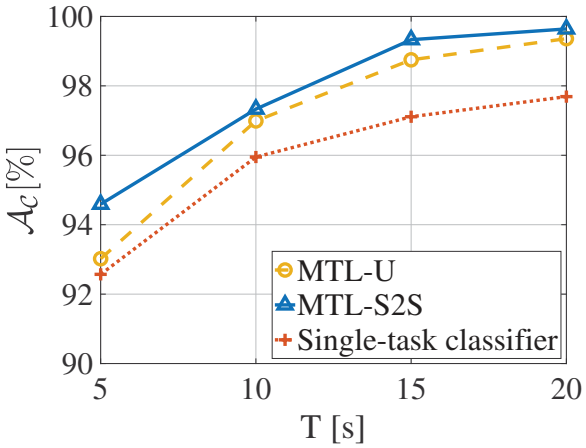


FIGURE 2.13: Classification performance.

TABLE 2.8: F-score Analysis.

Architecture	F-score			
	T=5s	T=10s	T=15s	T=20s
MTL-U	0.9312	0.9755	0.9904	0.9926
MTL-S2S	0.9501	0.9652	0.9927	0.9971
Single-task classifier	0.9198	0.9586	0.9817	0.9866

Classification performance

Figure 2.13 depicts the classification accuracy of the selected architectures as a function of T. As already anticipated, the performance always improves when T increases because all the learning architectures can use a higher number of training data to perform session classification. It is also evident that the single-task approach reaches lower accuracy levels, ranging from 92.52% to 97.73%. On the contrary, better results are obtained by the proposed MTL architectures: in this case, it is possible to reach an accuracy level up to 99.64%. The conducted study also demonstrates that MTL-S2S achieves higher classification accuracy for each T.

Classification performance can be further investigated through the *F-score* [139] index. Theoretically, the higher the *F-score* value, the better the classifier performance. The results summarized in Table 2.8 generally confirm what already discussed. In fact, *F-score* improves when T increases, and the single-task approach always has the lowest *F-score* values. Regarding the proposed MTL architectures, an exception is reported when $T = 10s$: in that case, even if MTL-U has the highest *F-score*, it achieves a lower classification accuracy than MTL-S2S because of a higher error rate for a specific application type (see the study on the confusion matrices proposed below).

To analyze which classes are mismatched in the classification process, the confusion matrices are provided in Figure 2.14 for each T. In general, both MTL architectures misclassify video-streaming sessions with audio-streaming ones. Nonetheless, such an error classification rate decreases when T increases. When $T = 5s$,

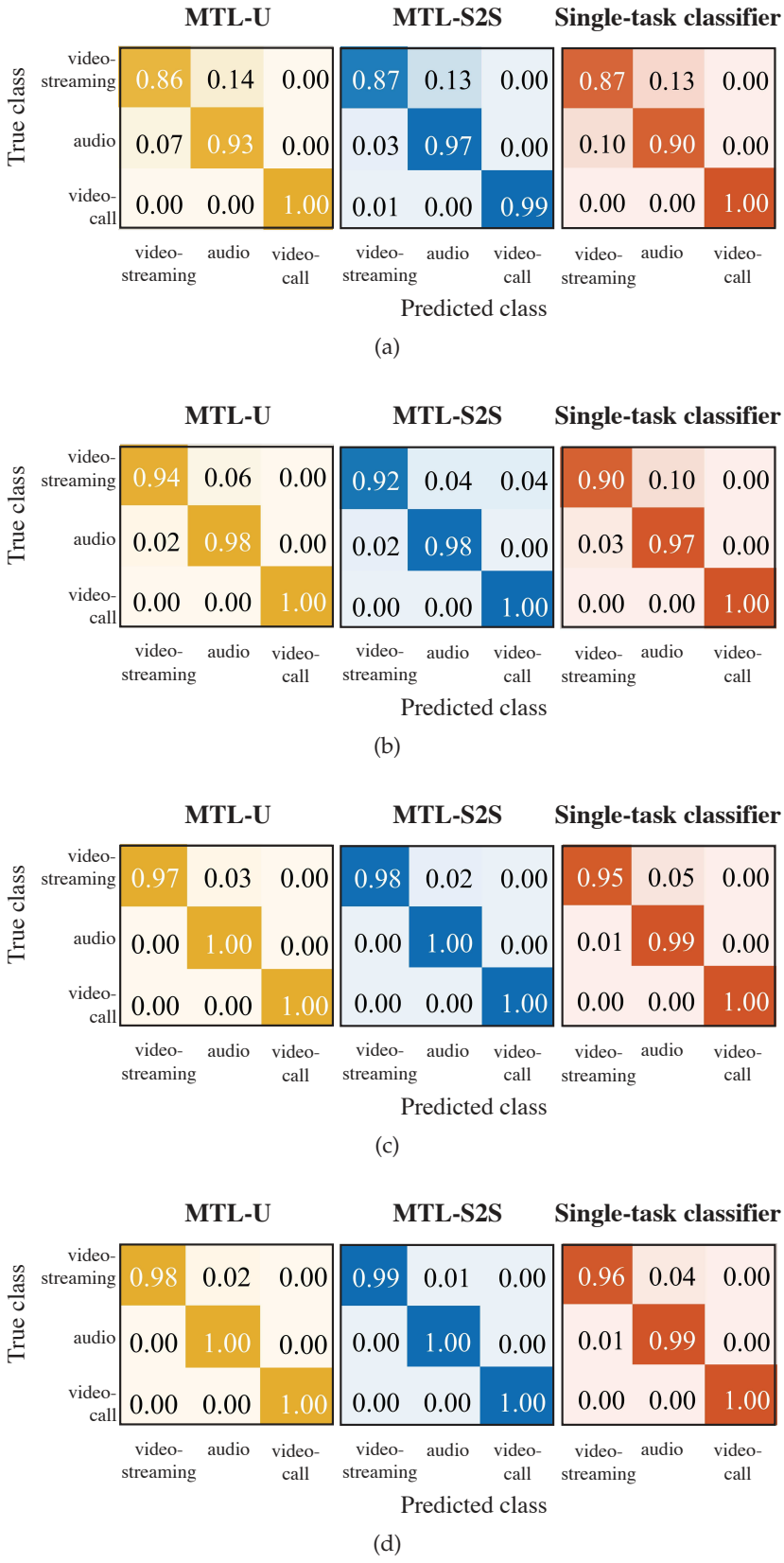


FIGURE 2.14: Confusion matrix for MTL-U, MTL-S2S, and the single-task classifier when (a) $T = 5s$, (b) $T = 10s$, (c) $T = 15s$, and (d) $T = 20s$.

in fact, 14% and 13% of video-streaming sessions are (wrongly) classified as audio-streaming by MTL-U and both MTL-S2S and the single-task classifier, respectively. These percentages decrease to 2% for MTL-U, 1% for MTL-S2S, and 4% for the single-task classifier when $T = 20s$. However, also in this case, it is possible to observe how the proposed MTL architectures always provide better results with respect to those measured for the single-task approach. Going more into detail, MTL-S2S presents the highest percentage of sessions, which are correctly classified, for each T, except for $T = 10s$. When $T = 10s$, as anticipated with the analysis of *F-score*, MTL-U reports a lower $\mathcal{A}_C$ than MTL-S2S. However, MTL-U reports a higher *F-score*. The confusion matrices show the reason why this occurs. The percentages of video-streaming sessions which are correctly classified by MTL-U (see Figure 2.14(b), on the left) and MTL-S2S (see Figure 2.14(b), in the middle) are 94% and 92%, respectively.

Prediction performance

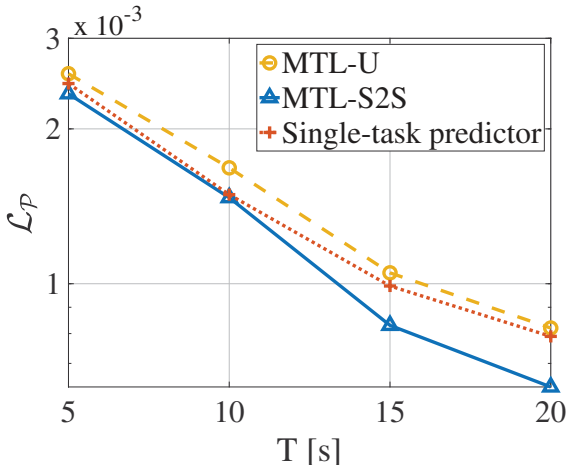
Figure 2.15 shows the prediction loss for the time instants $T + 1s$, $T + 2s$, and $T + 3s$. First of all, it is evident that the curves for $T + 3s$ are incomplete. In this case, the training process always fails when $T = 5s$. As expected, the prediction loss decreases with the observation window T, because the learning architectures have more training data to make a prediction. Regarding the prediction performed at both $T + 1s$ and $T + 2s$, MTL-S2S and MTL-U always achieve the best and the worst performance levels, respectively. On the other hand, when the prediction is done a $T + 3s$, the single-task approach slightly exceeds the prediction losses of MTL-U.

In summary, MTL-S2S always guarantees the lowest prediction losses, at the cost of a higher complexity (see Section 2.2.2). MTL-U has the worst performance when the prediction is done at $T + 1s$ and $T + 2s$. The single-task approach exhibits intermediate performance levels when $T + 1s$ and $T + 2s$, but the prediction accuracy is lowest at $T + 3s$. These results also confirm the ability of LSTM, which is exploited in both MTL-S2S and the single-task scheme, to suitably process time series by taking into account the temporal sequence of TBS values.

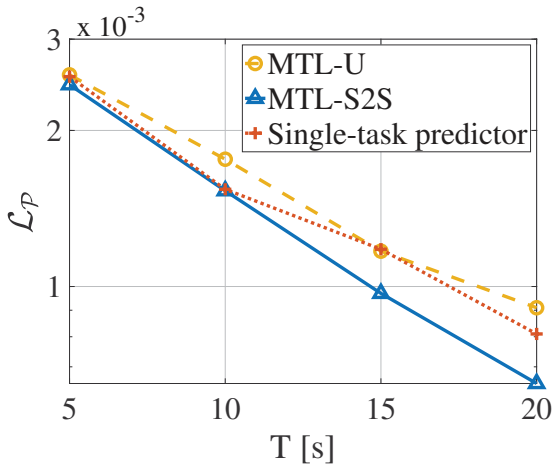
Complexity and convergence analysis

The complexity of selected learning architectures is evaluated by measuring the number of trainable parameters: the higher the number of parameters, the higher the complexity. Firstly, it is evident that the complexity of all the investigated learning architectures increases with an increasing duration T of the observation window. MTL-S2S always has the highest complexity. Also the single-task approach, based on LSTM, has a high complexity because of the structures of LSTM cells. On the contrary, MTL-U guarantees the lowest complexity for each observation window T.

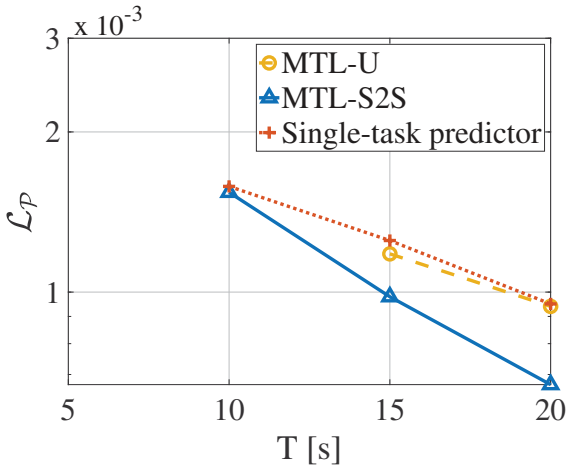
The convergence analysis evaluates the performance of the investigated learning architectures (including autoencoder loss, classification accuracy, and prediction



(a)



(b)



(c)

FIGURE 2.15: Prediction performance of the best configurations of MTL-U and MTL-S2S and the single-task approach: a) prediction loss at $T + 1s$, b) prediction loss at $T + 2s$, and c) prediction loss at $T + 3s$.

loss) as a function of the number of epochs considered during the training phase.

TABLE 2.9: Complexity analysis of learning architectures.

Architecture		# Parameters			
		T=5s	T=10s	T=15s	T=20s
MTL-U	Autoencoder	98	314	805	960
	Classifier	72	189	543	618
	Predictor	43	114	411	486
MTL-S2S	Autoencoder	806	1607	5496	8781
	Classifier	438	665	2513	3603
	Predictor	386	563	2211	3201
Single-task	Classifier	111	513	1068	1068
	Predictor	82	491	1036	1781

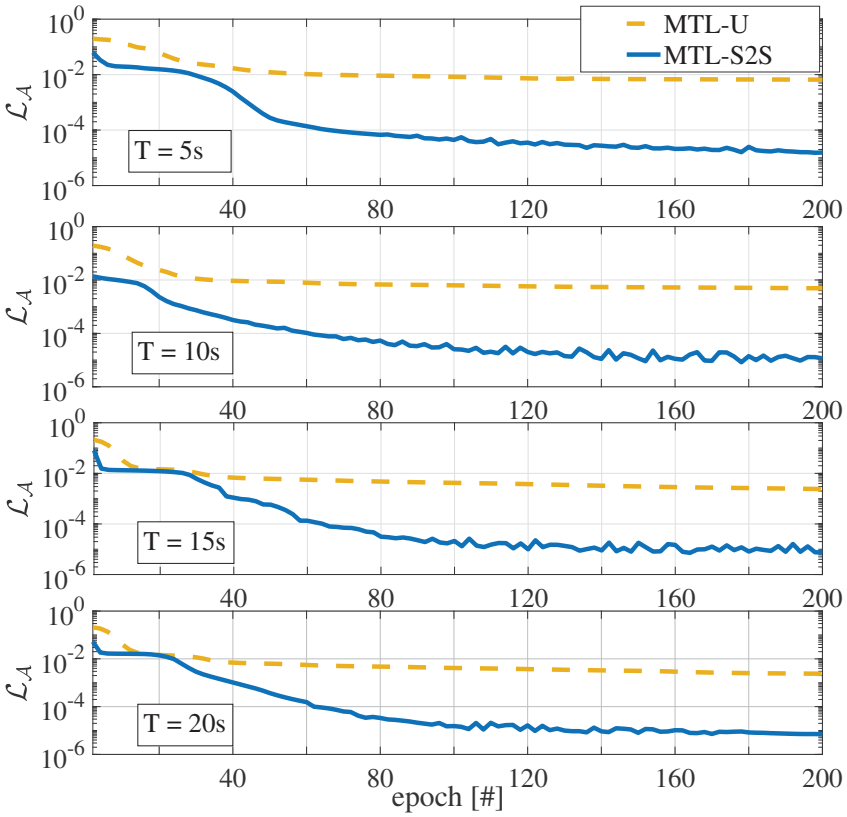


FIGURE 2.16: Autoencoder loss vs number of epochs.

Figure 2.16 shows the autoencoder loss as a function of the number of epochs. MTL-S2S has the slowest convergence time, while providing the lowest autoencoder loss. Figure 2.17 depicts the classification accuracy as a function of the number of epochs. While the proposed MTL architectures reach similar performance, the single-task approach always has the highest convergence time. Figure 2.18 shows the prediction loss as a function of the number of epochs. In this case, it is possible to observe that MTL-S2S achieves lower performance losses, at the cost of a slower convergence time.

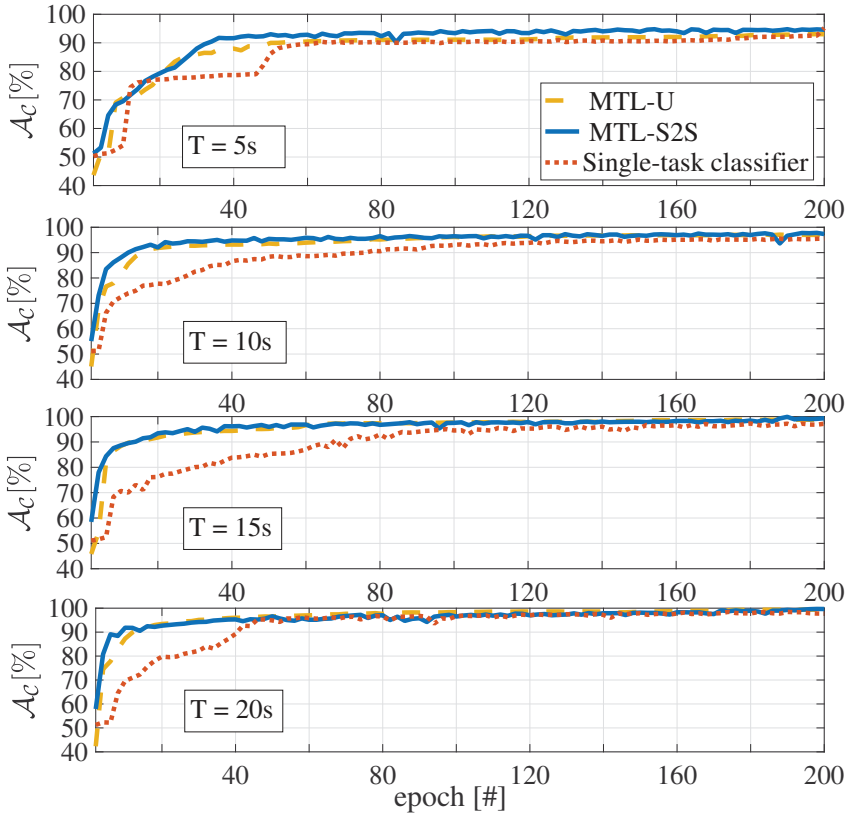


FIGURE 2.17: Classification accuracy vs number of epochs.

A further evaluation with more classes

As described in Section 2.2.1, the proposed MTL approach can be applied to different scenarios with a higher number of classes. To provide further insight, the training dataset considered in this work allowed to evaluate the performance of the proposed methodology when considering the six available classes of applications: YouTube, Vimeo, Spotify, Google Music, Skype, and WhatsApp Messanger. Specifically, differently from the original investigation, the applications belonging to the same service category have been treated as separate classes. The configurations of the MTL-S2S approach that achieved the best performance in the analysis of three service categories only have been tested. Figures 2.19 and 2.20 depict the classification accuracy and the prediction loss of MTL-S2S and the single-task schemes with six classes as a function of T. Obtained results further confirm that the proposed MTL approach outperforms the baseline single-task scheme also in scenarios with a higher number of classes. Differently from the previous case, however, lower accuracy levels are caused by very similar patterns of applications (especially those of audio-streaming type) and it is increasingly difficult to distinguish the different applications when the observation window T decreases.

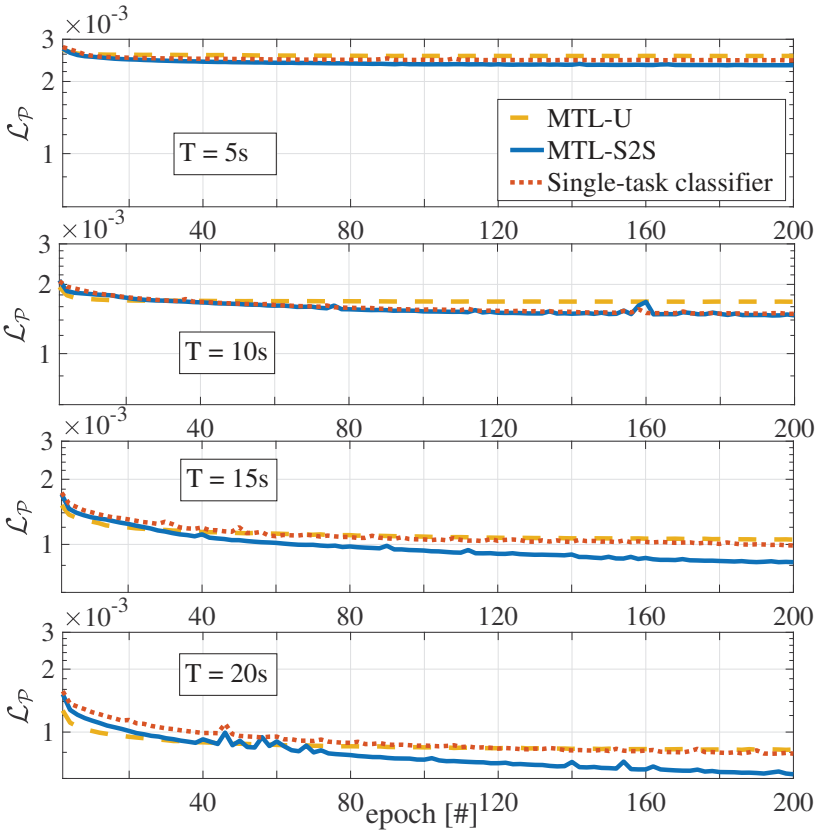


FIGURE 2.18: Prediction loss vs number of epochs.

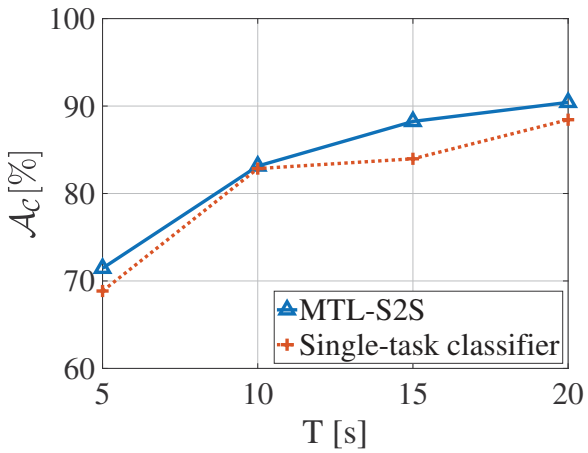


FIGURE 2.19: Classification performance with six application classes.

2.2.3 Final considerations

This study, after introducing an unsupervised learning methodology to characterize radio resource utilization patterns, has tailored an MTL model for traffic classification and prediction at the mobile edge, which leverages data mining from the PDCCH and two types of autoencoders (i.e., the Undercomplete Autoencoder and the Seq2Seq Autoencoder) exploited as key building blocks for obtaining common feature representations. Different configurations of neural networks have been trained

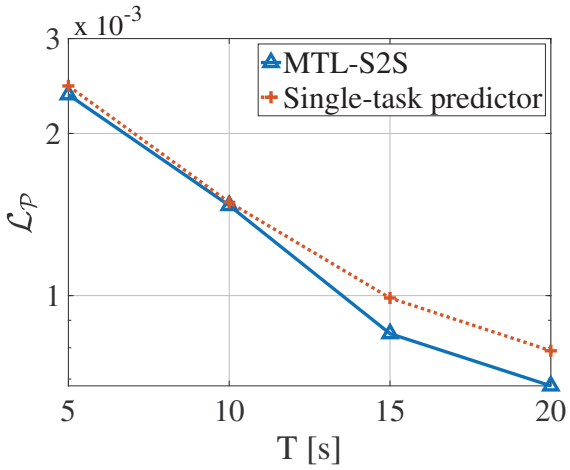


FIGURE 2.20: Prediction performance at $T + 1s$ with six application classes.

with a real dataset collected from an operative mobile network in Spain. Moreover, the performance of the proposed approach has been thoroughly investigated in terms of classification accuracy, prediction loss, complexity, and convergence. A cross-comparison with respect to conventional single-task learning schemes, that do not use autoencoders and that are generally investigated in the current state of the art for traffic classification and prediction, has also demonstrated that: i) the MTL architectures, leveraging the autoencoders, always guarantee higher performance than the single-task learning approach, ii) the MTL architecture based on the Seq2Seq Autoencoder always achieves the highest classification accuracy and the lowest prediction losses, at the cost of a higher complexity and convergence time.

Chapter 3

Anticipatory Resource Allocation at the Edge using Spatio-Temporal Dynamics of Mobile Users

In this Chapter, a novel methodology for anticipatorily allocating communication and computational resources at the network edge is formulated.

Specifically, communication and computational resources available at the network edge should be properly managed to fulfill the spatio-temporal dynamics and the even growing amount of users' requests [13], [14]. Most of the scientific contributions in this context address network resource management, computational resource allocation, and task offloading through optimization algorithms [66]–[75] or iterative procedures based on artificial intelligence [76]–[81]. Unfortunately, these contributions generally consider a static picture of the overall systems and ignore the impact that future spatio-temporal dynamics of mobile users may have on the system behavior. Differently, the knowledge (i.e., prediction) of both users' mobility and communication and computational resources they request over time within a given geographical area could significantly improve network optimization mechanisms [64], [65], [82]. The current state of the art proposes various techniques to forecast the movements of users, their requests, or both [97] (see Section 1.3.2 for more details). Solutions based on machine and deep learning also promise to better anticipate network behaviors and dynamics in heterogeneous and large-scale scenarios characterizing 5G and Beyond systems [86], [87]. Nevertheless, the resulting network optimization problems fail to take advantage of the joint prediction of both users' mobility and service demands over a look-ahead temporal horizon and within a standard compliant ETSI-MEC context.

To bridge this gap, in the following an innovative methodology is formulated for the anticipatory allocation of communication and computational resources at the network edge (i.e., task offloading), based on the knowledge of spatio-temporal dynamics of mobile users. The conceived architecture adopts a SDN approach to monitor users' mobility during time. Then, it exploits a deep learning architecture based

on *Convolutional Long Short-Term Memory (ConvLSTM)* [146] to predict the distribution of users among cells and their related service demands over a look-ahead temporal horizon. A centralized Multi-access Edge Orchestrator uses this information to anticipatorily distribute users' demands among available MEC servers, while satisfying communication and computational constraints at the network edge and the upper bound for latency expected by mobile users. Specifically, the optimal allocation problem is formulated as a sequential decision-making process, which considers future steps in the optimization horizon and it is solved by *Dynamic Programming* [147].

The behavior of the proposed approach is investigated with real mobility traces and realistic network and service settings by using computer simulations. First of all, the presented study underlines that the usage of both *ConvLSTM* and *Dynamic Programming* ensures results comparable with those obtained by the same optimization algorithm running on a perfect knowledge (i.e., ground truth) of spatio-temporal dynamics of mobile users. This demonstrates the high performance of the prediction process. At the same time, the comparison against a baseline approach, which leverages the distribution of users at the current time instant and allocates users' demands to the closest MEC server, reveals that only the conceived anticipatory approach can fairly distribute users' requests among the resources available at the network edge, while ensuring the targeted quality of service level. Finally, a complexity analysis confirms the applicability of the proposed methodology in a pratical network.

3.1 Reference Scenario

The proposed study mainly refers to the task offloading problem, according to which it is necessary to deploy (and properly use) available communication and intensive computational capabilities at the network edge to offer new heavy demanding and latency-critical services with demanding user QoS requirements [9], [12], [148]. E-Health, autonomous driving, and augmented/virtual reality are possible examples of advanced services with very heterogeneous communication and computational demands. For example, bandwidth requirements amount to 100 Mbps, 700 Mbps, and 1 Gbps [148] for e-Health, autonomous driving, and virtual reality use cases, respectively, while their memory requirements range from 8 GB to 32 GB [149]–[151]. This kind of heterogeneity is also expected for the communication latency, ranging from 1 ms (e.g., for e-Health, virtual reality, and robotics) to 100 ms (e.g., for autonomous driving) [148].

The conceived approach can be implemented within the 5G slicing paradigm. In fact, according to 3GPP specifications [152], a slice instance represents a set of network functions and related resources which are arranged and configured in a logical network to meet certain network characteristics. To this end, a Service Provider (or Tenant) declares communication service requirements (e.g., coverage area, number and distribution of users, traffic demand, mobility, latency, etc.) to the Infrastructure

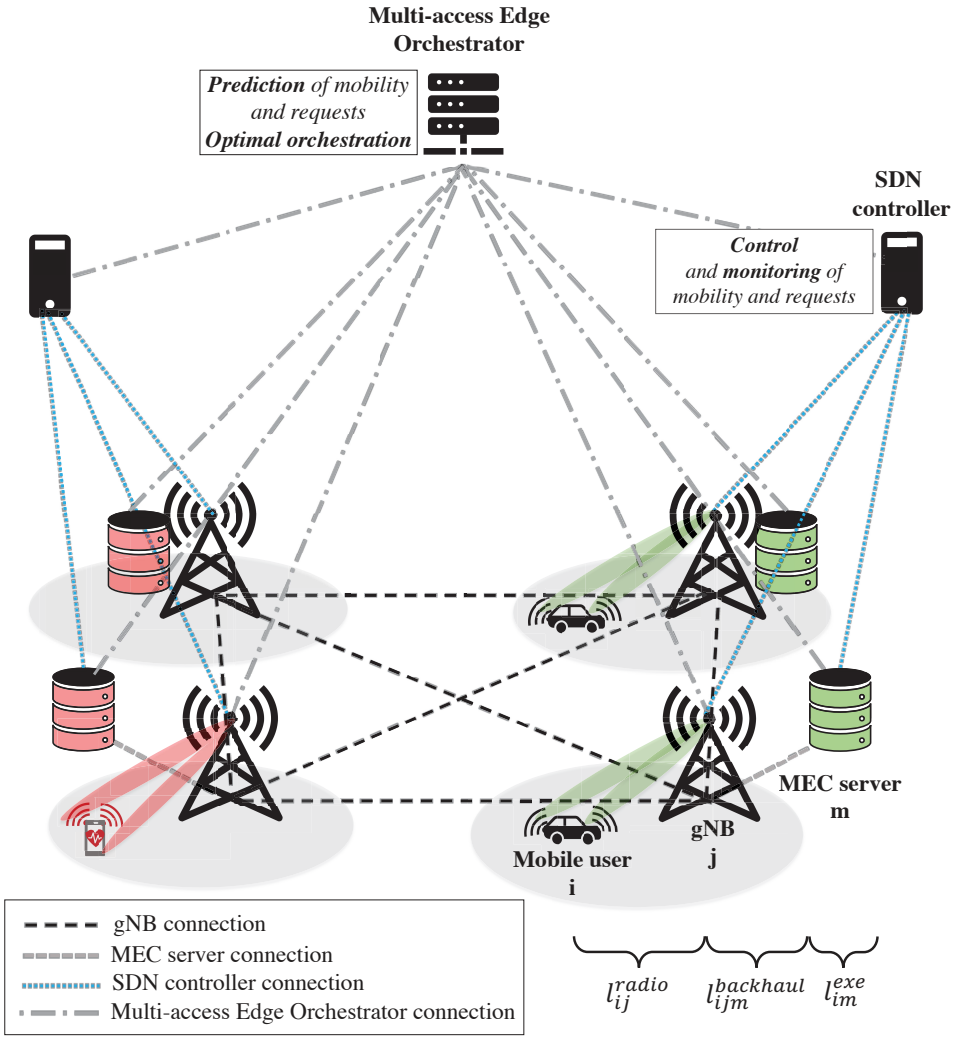


FIGURE 3.1: Reference mobile network with different contributions of latency l_{ijm} in the system model.

Provider. In turn, the Infrastructure Provider configures the corresponding network slice instance, whose preparation phase includes the on-boarding and verification of network function products and the necessary network environment. From this moment on, the Service Provider (or Tenant) can dynamically allocate the resources belonging to the aforementioned slice to the served mobile users (i.e., the task offloading within a specific slice). Note that in complex deployments, where heterogeneous services are offered through different slices, the proposed approach can be replicated for each slice.

In line with 5G specifications, emerging guidelines for the upcoming B5G systems, and the ETSI-MEC standard [153], the mobile network considered in this work embraces mobile users, gNBs, MEC servers, SDN controllers, and a Multi-access Edge Orchestrator (see Figure 3.1). Here, gNBs are part of the 3GPP network integrated within the ETSI-MEC architecture. They provide wireless connectivity to mobile users through heterogeneous technical components at the radio interface [153].

It is important to remark that gNBs can be connected to each other in different ways. Ring, tree, or mesh topologies can be implemented by the Infrastructure Provider [154]. Without loss of generality, a mesh topology is depicted in Figure 3.1 as an example of the backhaul network topology, even if the system model described in Section 3.2.1 will be general enough for capturing the behavior of any topology.

A number of MEC servers (or MEC hosts) expose resources to mobile users, depending on one or more services they use [153]. In this sense, the example reported in Figure 3.1 shows that the green and red blocks of MEC servers are dedicated to autonomous driving and e-Health services, respectively. According to ETSI-MEC specifications, MEC servers can be deployed at the gNBs, at aggregation points, or at the edge of the core network [12]. Independently from their position, however, MEC resources (i.e., memory and computing) can be used by users attached to different cells. This important flexibility, however, requires a careful distribution of users' demands, that should take care of the stringent communication requirements, instead of just considering the computational capabilities of MEC servers.

Network resources are monitored, configured, and orchestrated [153]. To this end, SDN controllers continuously interact with gNBs and MEC servers for monitoring the number of users served by each cell, the computational resources they request, and the amount of resources exposed and/or available in each MEC server. Note that SDN controllers can retrieve useful information from network elements through standardized protocols (i.e., OpenFlow, RESTCONF, etc.) [155]. Specifically, since gNBs know how many users are attached to them, they can retrieve information about the number of users served by each gBN by simply asking for such information to the gNBs.

This information is delivered to the Multi-access Edge Orchestrator for network optimization purposes. It represents a fundamental entity of the ETSI-MEC reference architecture, included in the MEC system level management [153]. The envisaged solution uses Multi-access Edge Orchestrator capabilities for managing a certain number of gNBs and MEC servers in a given geographical area (i.e., the radio access network is divided into clusters, controlled by one orchestrator) in order to optimally allocate computing and communication resources for task offloading, based on the prediction of spatio-temporal users' dynamics. This is done by satisfying heterogeneous traffic demands. The proposed optimization algorithm, which can be aided by exploiting mobility and service requests prediction, is executed by each orchestrator instance in order to minimize the latency (which is one of the most leading performance measures of 5G and B5G [13], [14]) of each service, while jointly considering network communication and computational requirements and satisfying the upper bound of service latency and related network constraints.

Note that the proposed service infrastructure can embrace different domains, whose network equipments are managed by a single SDN controller, namely the domain controller (see Figure 3.2). At the large-scale, the configuration of different domains can be managed through a hierarchical and highly scalable architecture

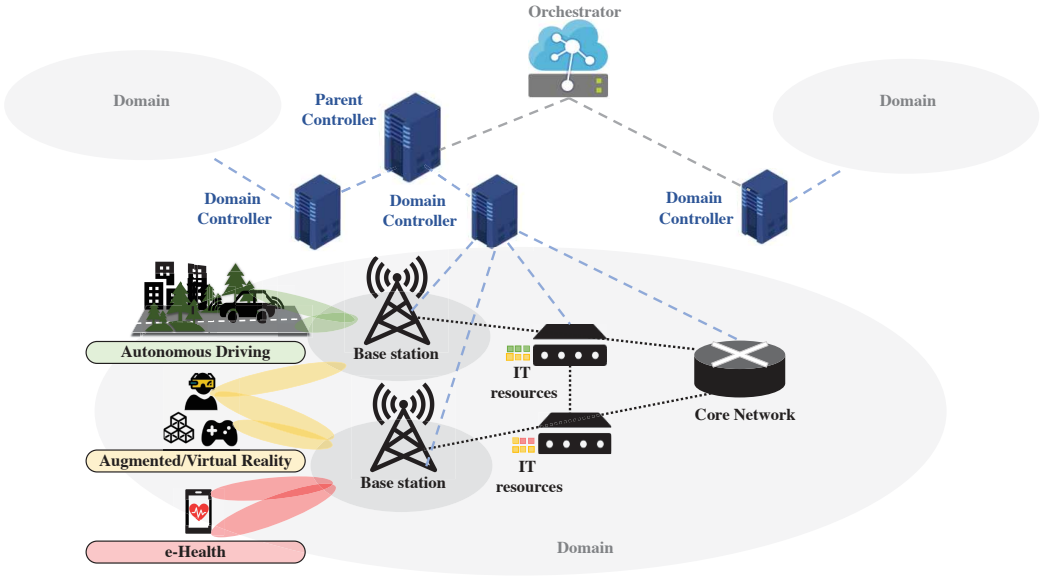


FIGURE 3.2: The proposed service infrastructure.

of controllers [148]. Specifically, multiple domain controllers, that are connected to generic base stations, nodes hosting any kind of IT resources, and network routers of the core network, are specialized controllers in charge of intra-domain services. They are coordinated by a parent controller or directly orchestrated by the Orchestrator, which interoperates with domain controllers to provide end-to-end and inter-domain services.

Moreover, an intrinsic characteristic of many 5G services (e.g., autonomous driving, virtual/augmented reality assisting museum tours) is mobility. Therefore, the communication and computational resources must be managed by using a mobility-aware approach, which is considered to be one of the most critical and challenging issues for network orchestration [14], [90].

3.2 Problem Statement

In this section, the system model is described and the optimization problem for the reference scenario and the adopted mobility prediction model are formulated. To facilitate the understanding of the notations adopted in what follows, a summary of symbols is reported in Table 3.1.

3.2.1 System model

Let $\mathcal{I}$ and $|\mathcal{I}|$ be the set and the number of users moving in the considered geographical area, respectively. According to the target application, the request formulated by the $i-th$ user is characterized by the following communication and computational requirements: the communication bandwidth set to b_i, the upper bound of latency equal to τ_i, the input data size s_i, the memory requirement set to m_i, and the demanded computational capability (expressed in terms of number of CPU cycles)

TABLE 3.1: List of mathematical notation adopted in Chapter 3.

Symbol	Description
i	Index of the $i-th$ user
j	Index of the $j-th$ gNB
m	Index of the $m-th$ MEC server
t_k	Discrete-time interval
$\mathcal{I}(k)$	Set of users
$\mathcal{J}$	Set of available gNBs and related attached cells
$\mathcal{M}$	Set of available MEC servers
$I_j^{gNB}(t_k)$	Set of users attached to the $j-th$ gNB
$I_m^{MEC}(t_k)$	Set of users served by the $m-th$ MEC server
$\|set\|$	Set cardinality
$\|\hat{I}_j^{gNB}(t_k)\|$	Predicted number of users attached to the $j-th$ gNB
b_i	Communication bandwidth of the $i-th$ user
τ_i	Upper bound of latency for the service requested by the $i-th$ user
s_i	Input data size of the $i-th$ user
m_i	Memory requirement of the $i-th$ user
c_i	Computational capability requirement (in CPU cycles) of the $i-th$ user
B_j	Available bandwidth within the cell attached to the $j-th$ gNB
$e_{ij}(t_k)$	Spectral efficiency between the $i-th$ user and the $j-th$ gNB
$M_m(t_k)$	Memory capability of the $m-th$ MEC server
$M_m^{opt}(t_k)$	Memory capability of the $m-th$ MEC server in the optimization problem
$M_m^{cons}(t_k)$	Consumed memory by the $m-th$ MEC server
$F_m(t_k)$	Computing capacity (in CPU cycles/second) of the $m-th$ MEC server
$F_m^{opt}(k)$	Computing capacity (in CPU cycles/second) of the $m-th$ MEC server in the optimization problem
$l_{ijm}(t_k)$	Total latency experienced by the $i-th$ user attached to the $j-th$ gNB and served by the $m-th$ MEC server
$\bar{l}_{ijm}(t_k)$	Average latency per user
$l_{ij}^{radio}(t_k)$	Communication latency experienced between the $i-th$ user and the $j-th$ gNB over the radio interface
$l_{ijm}^{backhaul}(t_k)$	Backhaul latency between the $j-th$ gNB and the $m-th$ MEC server for the $i-th$ user
$l_{im}^{exe}(t_k)$	Execution latency experienced at the $m-th$ MEC server for serving the $i-th$ user
$I_{jm}(t_k)$	Portion of users attached to the $j-th$ gNB and served by the $m-th$ MEC server
$r_{jm}(t_k)$	Capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server
$f_{im}(t_k)$	Number of CPU cycles/second allocated by the $m-th$ MEC server to the $i-th$ user
$\alpha_{im}(t_k)$	Binary decision variable denoting if the $i-th$ user is served by the $m-th$ MEC server
T	Observation window
N	Look-ahead temporal horizon
$t_{k,n}$	Decision cycle
k	Index of the decision epoch t_k
γ	Discount factor
n	Index of the considered time steps in each decision epoch t_k

equal to c_i. Let $\mathcal{J}$ be the set of available gNBs. The number of gNBs is given by $|\mathcal{J}|$, that is the cardinality of $\mathcal{J}$, and B_j represents the amount of bandwidth available within the cell served by the $j-th$ gNB. Let $\mathcal{M}$ and $|\mathcal{M}|$ be the set and the number of the available MEC servers, respectively. M_m and F_m indicate the memory capability and computing capacity (expressed in terms of CPU cycles per second) of the $m-th$ MEC server. I_j^{gNB} is the set of users in the $j-th$ cell, attached to the $j-th$ gNB, and I_m^{MEC} is the set of users served by the $m-th$ MEC server.

The system evolves in discrete-time intervals: every time t_k, a different number of users I_j^{gNB} and I_m^{MEC} is served by the $j-th$ gNB and the $m-th$ MEC server, respectively. It also holds that $|\mathcal{I}(t_k)| = \sum_{j\in\mathcal{J}} |I_j^{gNB}(t_k)| = \sum_{m\in\mathcal{M}} |I_m^{MEC}(t_k)|$ for all time intervals.

The total latency experienced by the $i-th$ user attached to the $j-th$ gNB and served by the $m-th$ MEC server in the $k-th$ time interval is given by:

$$l_{ijm}(t_k) = l_{ij}^{radio}(t_k) + l_{ijm}^{backhaul}(t_k) + l_{im}^{exe}(t_k), \tag{3.1}$$

where $l_{ij}^{radio}(t_k)$ is the communication latency experienced between the $i-th$ user and the $j-th$ gNB over the radio interface, $l_{jm}^{backhaul}(t_k)$ is the backhaul latency experienced between the $j-th$ gNB and the $m-th$ MEC server, and $l_{im}^{exe}(t_k)$ is the execution latency experienced at the $m-th$ MEC server [14], [69], [148]. These different latency contributions are shown in Figure 3.1.

In compliance with ITU specifications, the communication latency over the radio interface, $l_{ij}^{radio}(t_k)$, is expected to be less than 5 ms [123], [148].

The backhaul latency $l_{ijm}^{backhaul}(t_k)$ is obtained by dividing the aggregate traffic load generated by the users attached to the $j-th$ gNB and served by the $m-th$ MEC server, that is $\sum_{i\in\mathcal{I}_{jm}(t_k)} s_i$, and the capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server, $r_{jm}(t_k)$ [43]:

$$l_{ijm}^{backhaul}(t_k) = \frac{\sum_{i\in I_{jm}(t_k)} s_i}{r_{jm}(t_k)}, \tag{3.2}$$

where $I_{jm}(t_k)$ is the portion of users attached to the $j-th$ gNB and served by the $m-th$ MEC server, that share the same backhaul link. The system model described herein is general enough for capturing the behavior of any backhaul topology. Without loss of generality, a mesh topology, with the same capacity for each backhaul link, is considered (see Figure 3.1). MEC servers can be deployed at the gNBs, at aggregation points, or at the edge of the core network. Therefore, the backhaul latency varies depending on the scenario. Specifically, assuming that MEC servers are co-located with gNBs without loss of generality, there are two possibilities when calculating the backhaul latency. No additional delay (i.e., $l_{jm}^{backhaul}(t_k) = 0$) is introduced in the backhaul if the $m-th$ MEC server co-located with the $j-th$ gNB (i.e, $m = j$), to which the user is attached, is the one serving the user. Conversely, the backhaul latency is considered and calculated for the backhaul path connecting the

gNB, to which the user is attached, with a neighboring MEC host, which is serving the user.

During the time interval t_k, the computing capabilities exposed by each MEC host are assumed to be uniformly allocated among served users. Therefore, the execution latency $l_{im}^{exe}(t_k)$ is [69], [73]:

$$l_{im}^{exe}(t_k) = \frac{c_i}{f_{im}(t_k)} = \frac{c_i}{F_m(t_k)/|I_m^{MEC}(t_k)|}, \tag{3.3}$$

where $f_{im}(t_k)$ is the number of CPU cycles per second allocated by the $m-th$ MEC server to the $i-th$ user. The equation above is generic enough for being used in any realistic scenario with homogenous and heterogeneous service requirements: the execution latency refers to the computational capability requirements of users, which can execute a single application task, as well as more heterogeneous application tasks.

3.2.2 Optimization problem

The goal of the proposed optimization framework is to distribute users' requests among the available MEC servers, so that the latency of each considered service is minimized and network outage in terms of memory, computing, and bandwidth resources is avoided. Decisions are made at each time t_k, which is also referred to as decision epoch in the following. This problem is formulated as a sequential decision-making process: at every decision epoch t_k, control actions aiming at assigning users' demands to the best suitable MEC servers are executed, according to their available memory capabilities (i.e., M_m^{opt} for the $m-th$ MEC server) and computing capacities (i.e., F_m^{opt} for the $m-th$ MEC server), in order to minimize latencies experienced by users and to satisfy service latency constraints. At every decision epoch t_k, the requests and, hence, the resources needed to run the user services for the N steps ahead are leveraged and the control is executed based on the optimization problem $P1$ stated in (3.4). The solution of the problem is found by executing the dynamic programming approach [147] at every decision epoch t_k (i.e., each point of the sequential decision-making process where decisions are made), transforming a complex problem into a sequence of simpler problems. In line with the standard discounted dynamic programming framework [147], the discount factor γ, $(0 < \gamma \leq 1)$, is introduced to incorporate the concept of discounting for the look-ahead temporal horizon N. Specifically, besides the decision epoch t_k, the decision cycle $t_{k,n}$, with $n \in \{0, 1, ..., N\}$, is needed. It represents the sequence of the considered time steps (with $t_{k,n} = t_{k+n}$) to reach and implement decisions in each epoch t_k, whose impact is exponentially weighted through γ^n. Thus, it is possible to understand that, when $n = 0$, $t_{k,0}$ is weighted through $\gamma^0 = 1$, while the future time steps in the sequence have a gradually decreasing weight (i.e., from γ^1 for $t_{k,1}$ to γ^N for $t_{k,N}$) in the decision cycle $t_{k,n}$.

$$P1: \min_{\{\alpha_{im}(t_k)\}} \left\{ \sum_{n=0}^{N} \gamma^n \left[\sum_{j \in \mathcal{J}} \sum_{i \in I_j^{gNB}(t_{k,n})} \left(l_{ij}^{radio}(t_{k,n}) + \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) \cdot \left[l_{ijm}^{backhaul}(t_{k,n}) + l_{im}^{exe}(t_{k,n}) \right] \right) \right] \right\}, \forall t_k \tag{3.4}$$

subject to:

$$\sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) \cdot m_i \leq M_m^{opt}(t_{k,n}), \forall m \in \mathcal{M}, \forall n, \sum_{i \in \mathcal{I}(t_{k,n})} m_i \leq \sum_{m \in \mathcal{M}} M_m^{opt}(t_{k,n}), \forall n \tag{3.4a}$$

$$\sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) \cdot f_{im}(t_{k,n}) \leq F_m^{opt}(t_{k,n}), \forall m \in \mathcal{M}, \forall n \tag{3.4b}$$

$$l_{ij}^{radio}(t_{k,n}) + \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) \cdot \left[l_{ijm}^{backhaul}(t_{k,n}) + l_{im}^{exe}(t_{k,n}) \right] \leq \tau_i, \forall i \in \mathcal{I}(t_{k,n}), \forall n \tag{3.4c}$$

$$B_j \cdot e_{ij}(t_{k,n}) \geq b_i, \forall i \in I_j^{gNB}(t_{k,n}), \forall n \tag{3.4d}$$

$$\alpha_{im}(t_{k,n}) \in \{0,1\}, \sum_{m \in \mathcal{M}} \alpha_{im}(t_{k,n}) = 1, \forall i \in \mathcal{I}(t_{k,n}),$$
$$\sum_{i \in \mathcal{I}(t_{k,n})} \alpha_{im}(t_{k,n}) = |I_m^{MEC}(t_{k,n})|, \forall m \in \mathcal{M}, \forall n \tag{3.4e}$$

$$|I_j^{gNB}(t_{k,n})| = \sum_{m \in \mathcal{M}} |I_{jm}(t_{k,n})|, \forall j \in \mathcal{J}, |I_m^{MEC}(t_{k,n})| = \sum_{j \in \mathcal{J}} |I_{jm}(t_{k,n})|, \forall m \in \mathcal{M}, \forall n \tag{3.4f}$$

The implemented control is expressed by a binary decision variable $\alpha_{im}(t_k)$, that is:

$$\alpha_{im}(t_k) = \begin{cases} 1 & \text{if the } i-th \text{ user is served by the } m-th \\ & \text{MEC server, i.e., } \forall i \in I_m^{MEC}(t_k); \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

Note that $\alpha_{im}(t_k)$ only involves the backhaul and the execution latency because they depend on the concerned MEC server, while the radio component is independent thereof. The constraints in (3.4a) consider the memory capabilities and requirements: the memory capability of the $m-th$ MEC server $M_m^{opt}(t_{k,n})$ cannot be exceeded by served users in each decision cycle $t_{k,n}$ and the overall memory capabilities need to be sufficient for satisfying memory requirements. The constraint in (3.4b) regards the CPU ability of the $m-th$ MEC server $F_m^{opt}(t_{k,n})$ in each decision cycle $t_{k,n}$. Because of the definition of the execution latency component in (3.3), computing capabilities are included in the service latency constraint (3.4c), that is valid for each $i-th$ user in the network , where the maximum tolerable latency τ_i is the upper bound of user latency experienced during each decision cycle $t_{k,n}$. If the computing capacities are not enough, (3.4c) is not verified. Bandwidth requirements are considered in (3.4d), where $e_{ij}(t_{k,n})$ is the spectral efficiency between the $i-th$ user and the $j-th$ gNB. Moreover, in every decision cycle $t_{k,n}$ each user can be served by

one and only one MEC server, as reported in (3.4e) and (3.4f), that means the number of users attached to different gNBs should be equal to the number of users served by different MEC hosts.

The solution of the optimization problem $P1$ stated in (3.4) may be anticipatorily found, forecasting the number of users in the coverage area of each gNB. In what follows, the anticipatory optimization approach presented in this work is referred to as *Prediction-based Control (P-C)*. Since the solution of the network optimization problem $P1$ stated in (3.4) may be also found supposing to know the mobility of users in advance, in Section 3.3 also the anticipatory network optimization approach based on ground truth, i.e., *Ground Truth-based Control (GT-C)*, is evaluated.

3.2.3 Mobility prediction model

The users moving in the considered geographical area may pass from one cell to an adjacent one. Accordingly, the number of users attached to each gNB changes over time. The goal of the mobility prediction model described in this section is to anticipatorily discover the distribution of mobile users among available cells, based on the knowledge of the number of users attached to the gNBs that served them in the past. Service demands can be later estimated based on the services requested by the users.

To this aim, this contribution leverages real mobility data from the dataset presented in [156], which reports the movements of around 100 taxi cabs in Rome (Italy), from 1 February 2014 to 2 March 2014, with a granularity of about 15 s. The traces of the published version of the dataset provide information on when the location of the taxis were collected, with a precision of microseconds, and GPS coordinates, in the decimal format. The considered geographical area of the center of Rome has been divided into square cells, covering an area of 1 km $\times$ 1 km each (an example is reported in Figure 3.3). Note that square cells have been considered, but the considerations also hold for arbitrarily shaped cells.

These real traces are used to generate a list of matrices describing the geographical distribution of users over time. For example, Figure 3.4 shows the distribution of the number of users for two cells (i.e., $j = 1$ and $j = 3$) with a low and high number of users, respectively.

The conceived mobility prediction model exploits the ConvLSTM architecture to predict the distribution of mobile users among the available cells and for the upcoming N consecutive time intervals, based on the knowledge of the distribution of users (i.e., retrieved by SDN controllers from gNBs, that know how many users are attached to them) observed during the latest T observation time intervals. As shown in Figure 3.5, the considered ConvLSTM architecture is based on LSTM [137], with the convolution operator as input, forget, and output gates instead of the element-wise or Hadamard product [146]. Therefore, it is able to extract temporal and spatial correlations of data through LSTM memory cells and the convolutional operation, respectively [17], [135]. More specifically, this work conceives a learning architecture

FIGURE 3.3: The considered geographical area with example temporal movements of three taxi cabs.

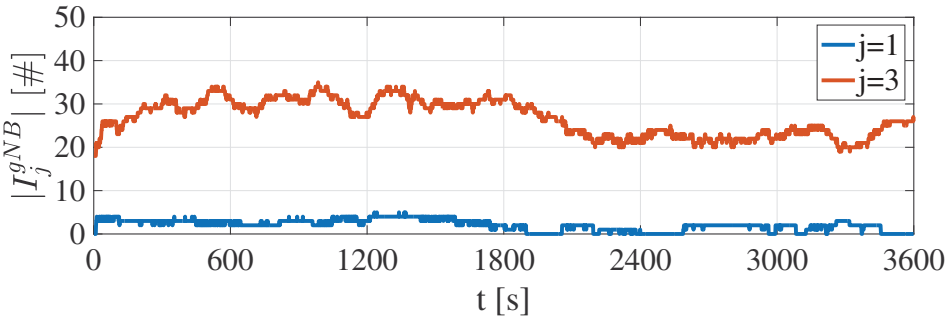


FIGURE 3.4: Distribution of the number of users over time for two cells.

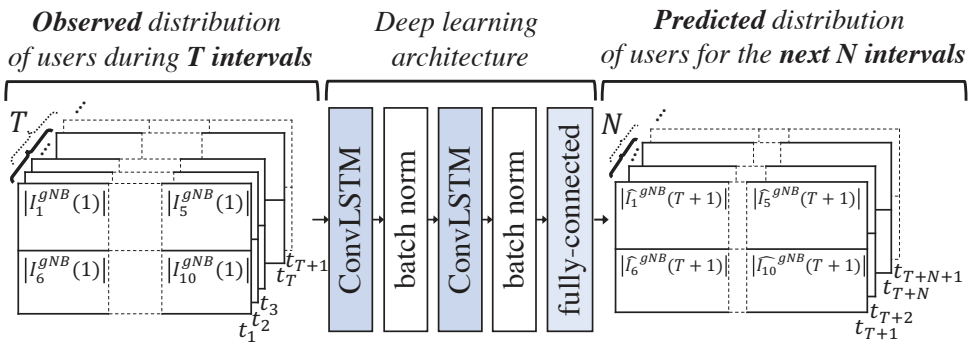


FIGURE 3.5: The conceived mobility prediction model.

embracing two 2-dimensional ConvLSTM layers, after each one a batch normalization layer is used to accelerate deep network training [157]. The number of epochs and the number of filters are set to 30 (see the convergence analysis proposed in Section 3.3.1) and 200, respectively. At the end, there is a fully-connected layer with ReLU activation function [17] to predict the expected distribution of users, after the observation window T, for a specific look-ahead temporal horizon N. The predictor is configured in order to minimize the MSE loss function, which minimizes the difference between the ground truth and the predicted distribution of users [135]. The Adam optimization [143], with a learning rate of 0.001, is used to update the

network weights.

3.3 Performance Evaluation

Herein, the performance of the conceived anticipatory network optimization scheme is evaluated by using computer simulations. Without loss of generality, the study considers an autonomous driving use case (with real mobility traces [156] described in Section 3.2.3 and conceivable network and service settings [72], [123], [148], [150], [158]–[161]). The approach can also be applied to each use case and heterogeneous scenarios by properly adapting the related parameter settings.

A real geographical area of 10 km^2 in Rome (Italy) is considered, divided into 10 square cells (i.e., $|\mathcal{J}| = 10$). According to the autonomous driving use case, for the $i-th$ user the communication bandwidth and the upper bound of service latency are set to $b_i = 700$ Mbps and $\tau_i = 100$ ms, respectively [148], the input data size is set to $s_i = 5$ Mbit [72], [159], and the memory and computational capability requirements are set to $m_i = 16$ GB [150] and $c_i = 300$ Megacycles [159], respectively. The available bandwidth within the cell attached to the $j-th$ gNB and the capacity of the backhaul link between the $j-th$ gNB and the $m-th$ MEC server are set to $B_j = 40$ MHz [160] and $r_{jm} = 10$ Gbps, respectively. Since it is assumed that MEC servers are co-located with gNBs without loss of generality, $|\mathcal{J}| = |\mathcal{M}| = 10$ in the tests. The parameters of MEC servers, whose sizing is a key issue in such systems, are adequately dimensioned with respect to overall requests in each $t_{k,n}$ [161]: the memory capability and the computing capacity of the $m-th$ MEC server are set to $M_m^{opt} = 176$ GB and $F_m^{opt} = 36$ Gigacycles/s, respectively. In simulations, it is considered the upper bound of the communication latency experienced over the radio interface $l_{ij}^{radio}(t_{k,n})$, that means to use constant values for l_{ij}^{radio} (i.e., 5 ms [123], [148]) and the spectral efficiency e_{ij} (i.e., 30 bit/s/Hz [158]) in each $t_{k,n}$. Table 3.2 summarizes the main adopted parameters.

In line with the dynamic programming approach [147], the optimal solution has been found at each decision epoch t_k through the value iteration algorithm implemented by using MATLAB. As anticipated in Section 3.2.2, the optimization problem $P1$ is solved by considering the predicted number of users per cell over a specific temporal horizon, i.e., $|\hat{I}_j^{gNB}(t_{k,n})|$ by using the actual distribution of users for $n = 0$ and the prediction of the number of users for the future time steps in the decision cycle $t_{k,n}$, and the perfect knowledge (ground truth) of users' distributions, i.e., $|I_j^{gNB}(t_{k,n})|$. These anticipatory mechanisms based on prediction and ground truth are denoted with *P-C* and *GT-C*, respectively. Note that the comparison between P-C and GT-C intends to highlight the effectiveness of the prediction procedure and its impact on the overall system performance. The behavior of both P-C and GT-C is studied for different temporal horizons, that are $N = 5$ s, 10 s, 20 s, 40 s, to evaluate their effect on key performance indicators. Moreover, to provide further insight, the anticipatory methods P-C and GT-C are compared against a *Baseline* approach.

TABLE 3.2: Main simulation parameters.

Parameter	Value
Area	10 km^2
$\lvert\mathcal{J}\rvert = \lvert\mathcal{M}\rvert$	10
T	40 s
N	5 s, 10 s, 20 s, 40 s
Learning rate	0.001
γ	0.9
b_i	700 Mbps [148]
τ_i	100 ms [148]
s_i	5 Mbit [72], [159]
m_i	16 GB [150]
c_i	300 Megacycles [159]
B_j	40 MHz [160]
r_{jm}	10 Gbps
M_m^{opt}	176 GB
F_m^{opt}	36 Gigacycles/s
l_{ij}^{radio}	5 ms [123], [148]
e_{ij}	30 bit/s/Hz [158]
Period of time	3600 s

It just leverages the distribution of users at the current time instant t_k and allocates their requests to the closest MEC server (i.e., co-located with the gNB in the related cell), without envisaging optimization problems and constraints. Therefore, since the related literature is missing works that perform network resource optimization based on the prediction of the number of users problem (please see Section 1.3.2 for further details), the proposed anticipatory optimization approach based on mobility prediction is compared with the anticipatory network optimization approach based on ground truth and with the Baseline approach defined above.

Since the system configuration decision and the user latency (as well as the computational requests) are updated with a time granularity of 1 s, the latency constraint (3.4c) is continuously taken into account by P-C and GT-C to consider the impact of users' mobility, handovers, and possible service migrations among MEC servers. As a consequence, the proposed system model and the conceived optimization problem allow to successfully meet the whole service latency constraint. Note that the developed approach has been conceived for 5G and B5G networks. Indeed, it is possible to assume a 0 ms handover latency (namely mobility interruption time in 3GPP specifications) [162]. At the same time, this assumption does not influence the behavior of the proposed approach because resources are optimally allocated on a much higher time granularity than the mobility interruption time allowed in 5G (and Beyond) deployments. For the same reason, the presented approach does not explicitly consider the virtual machine/container migration process. With a time granularity of 1 s, the

Multi-access Edge Orchestrator, that optimally orchestrates requested services and available communication and computational resources, communicates with all the network entities. Therefore, any configuration changes (i.e., on the number of users and related resources to be allocated) are known by MEC servers through the interaction with the orchestrator. Moreover, the delay of task migration between MEC servers can be considered to be negligible in a vehicular context [163], as the analyzed case, or, through prediction information for the look-ahead temporal horizon N, the migration process can eventually occur before it actually happens so that the users do not experience any additional delay due to migration [164].

The measured key performance indicators entail a complete analysis on mobility prediction performance and latency per user, as well as the number of changes among MEC servers, the distribution of users among MEC servers, consumed memory, and CPU usage. The number of changes among MEC servers is included as a key performance metric because, in mobile scenarios, it is important to guarantee service continuity. The changes among MEC servers (and so also the number of potential migrations) imply the establishment of new backhaul connections, having a negative impact on the experienced latency. Also the backhaul connection quality affects the computation execution [11]. Therefore, it is better to avoid changes among MEC servers and keep connectivity between the user and the serving MEC host [11], [165]. Finally, also the complexity of the proposed anticipatory network optimization scheme is evaluated.

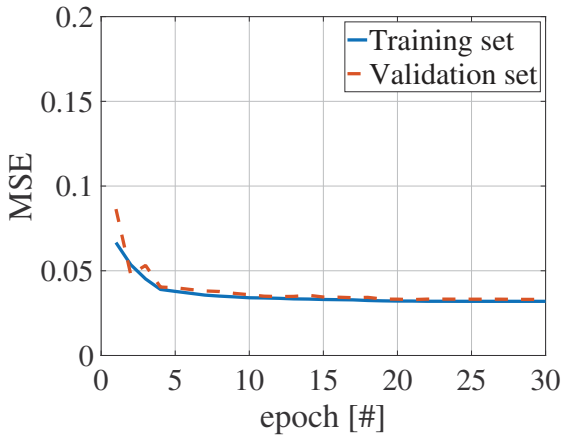
All the results that are reported next have been evaluated in a period of time (i.e., decision epochs) of 3600 s, with a time granularity of 1 s, and have been obtained by averaging the outcomes on the 3600 realizations. Together with average values, the 95%−confidence intervals, computed through the Gaussian statistical distribution, are reported as well for the spatial characterization. For the characterization during $3600s$, the Cumulative Distribution Functions (CDFs) illustrate only P-C and Baseline, because GT-C trends overlap with P-C and they are omitted for the sake of clarity.

3.3.1 Mobility prediction performance

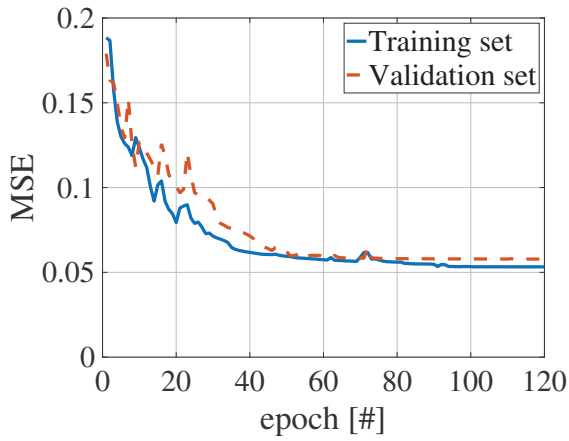
Regarding the prediction procedure (integrated within the P-C scheme), the conceived mobility prediction model exploits the ConvLSTM architecture, as described in Section 3.2.3. To provide further insight, the comparison with a state-of-the-art mobility prediction approach, that uses the LSTM architecture aided by the attention mechanism for capturing long-range dependency [92], is presented as well. In particular, the reference learning architectures selected for the cross-comparison embrace four LSTM layers (i.e., two with 200 and 2 with 100 hidden units, respectively, after each one a batch normalization layer is used) in order to have a comparable complexity (i.e., a similar number of training parameters) of the corresponding ConvLSTM architecture. Note that the mobility prediction architecture needs a different

TABLE 3.3: Complexity analysis of learning architectures.

Architecture	# Parameters			
	N=5s	N=10s	N=20s	N=40s
ConvLSTM	801205	802210	804220	808240
LSTM with attention	799940	801850	804030	808140



(a)



(b)

FIGURE 3.6: Prediction loss (i.e., MSE) vs number of epochs for a) ConvLSTM architecture and b) LSTM architecture with attention.

number of training parameters for each considered temporal horizon N, as summarized in Table 3.3.

Figure 3.6 shows the prediction loss (i.e., MSE) of the ConvLSTM architecture and the LSTM architecture with attention as a function of the number of epochs for the training set and the validation set, representing 80% and 20% of the adopted dataset, respectively. The reported curves confirm that the developed ConvLSTM architecture reaches lower values of MSE with respect to the LSTM architecture with attention. Moreover, differently from the LSTM architecture with attention, the ConvLSTM architecture fastly converges to stable values, i.e., it does not need a long

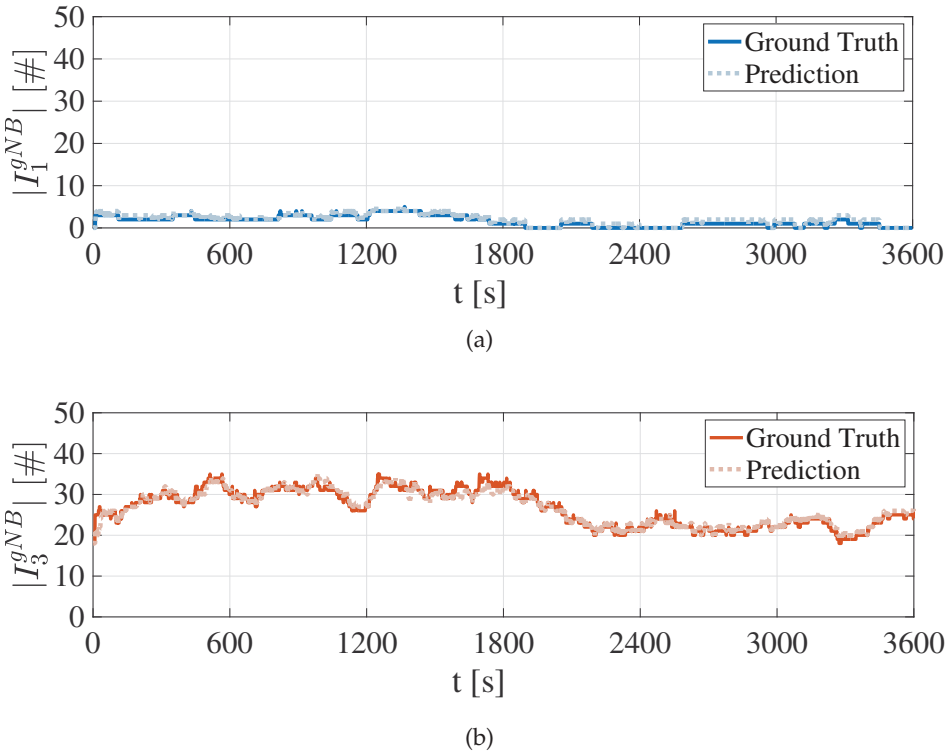


FIGURE 3.7: Prediction of the number of users over time for two cells taken as examples: a) $j=1$ and b) $j=3$.

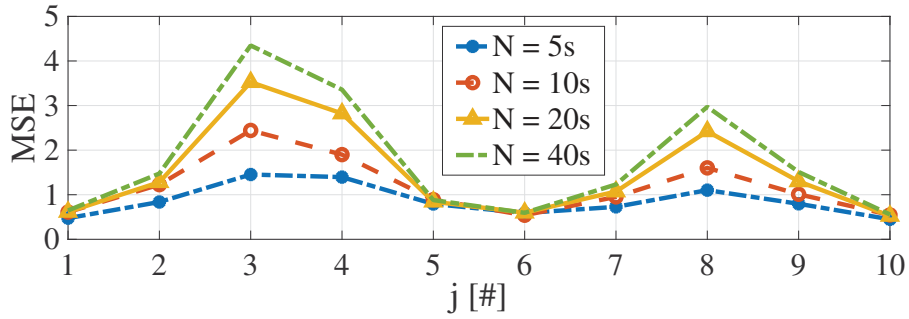


FIGURE 3.8: MSE for each $j-th$ cell, registered with different N.

training period. Accordingly, the ConvLSTM architecture trained during 30 learning epochs, which achieves better results in terms of prediction loss and convergence time/complexity, is considered hereafter.

Figure 3.7 shows the observed and the predicted distribution of users over time for two selected cells that are significant (i.e., $j=1$ and $j=3$ with a low and high number of users, respectively) to evaluate the prediction performance of the conceived model. In particular, the solid lines represent the ground truth of the number of users (integrated within the GT-C scheme), while the red dotted lines describe the predicted number of users (integrated within the P-C scheme), that are rounded up to the nearest integer. It can be noted that the two trends are almost overlapping.

To thoroughly evaluate the mobility prediction performance in the investigated scenario, Figure 3.8 reports the MSE values for each $j-th$ cell, registered with the

(a)

(b)

FIGURE 3.9: Example of taxi distribution at (a) 1:00 pm and (b) 5:00 pm in San Francisco.

different temporal horizons N. Cells $j = 3$ and $j = 8$, with the highest number of users (see the trend of $|I_3^{gNB}|$ in Figure 3.4), reach the greatest prediction losses. Moreover, the MSE tends to increase with N, as expected. In fact, the highest values of MSE are observed for $N = 40$ s, even if in this case MSE is generally lower than 4.

A further evaluation in another scenario

The conceived mobility prediction model exploiting the ConvLSTM architecture can be applied to different scenarios. The different load of cells and the different users' dynamics in the conducted study (see Figure 3.7) help to demonstrate the validity of the proposed approach in changing scenarios. To provide further insight, other real mobility traces from the similar dataset presented in [166], which reports the

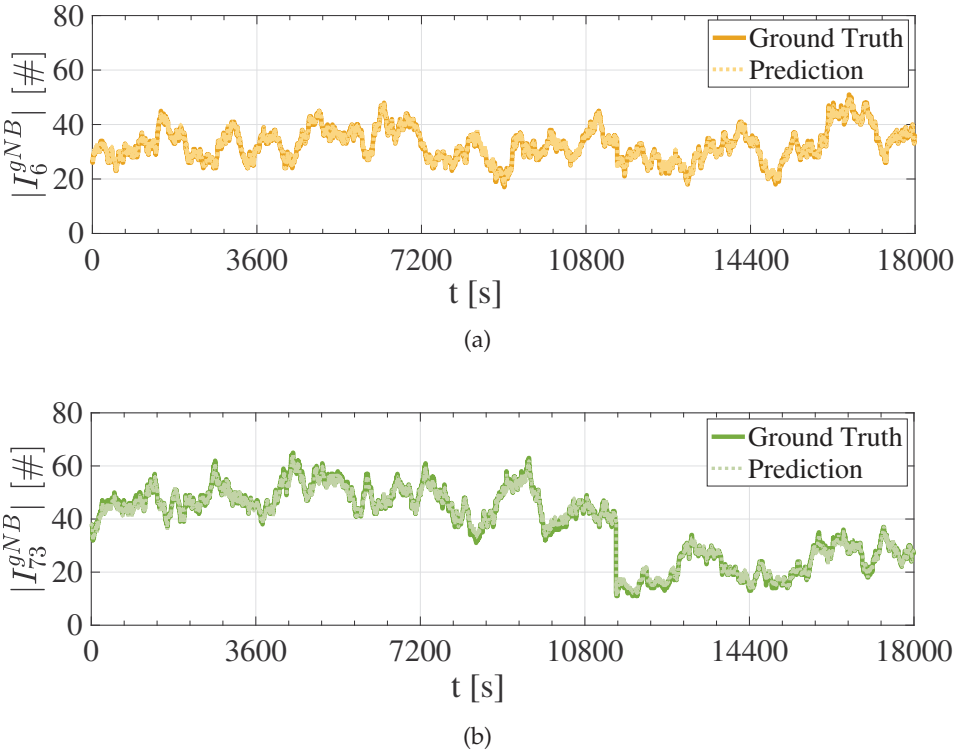


FIGURE 3.10: Prediction of the number of users over time for two cells taken as examples in San Francisco: a) $j=6$ and b) $j=73$.

movements of around 500 taxi cabs in San Francisco (USA), from 17 May 2008 to 10 June 2008, are employed. Figure 3.9 shows an example of taxi distribution at 1:00 pm and 5:00 pm. Similarly to the Rome scenario, the considered geographical area of 100 km^2 in San Francisco has been divided into 100 square cells, covering an area of 1 km $\times$ 1 km each.

Mobility. To evaluate the mobility prediction performance, the same learning setup detailed in Section 3.2.3 and summarized in Table 3.2 is adopted. The results reported below have been evaluated in a period of time of 18000 s. Figure 3.10 shows the observed and the predicted distribution of users over time for two cells taken as examples (i.e., $j=6$ and $j=73$). Obtained results further confirm the effectiveness of the conceived mobility prediction model also in a larger area (100 cells instead of 10) and for a longer period (18000 s instead of 3600 s).

Users' requests. By assuming that all the users execute the same service, the aggregate requests of communication and computational resources to be allocated in each cell can be preliminary estimated by multiplying the predicted number of users in the j-th cell by the related service requirements (see numerical details in Section 3.1). For example, Figures 3.11 and 3.12 show the aggregated communication and computational resources coming from the two selected cells (i.e., $j=6$ and $j=73$) for autonomous driving and virtual reality use cases. Note that the basic multiplicative estimation is focused on bandwidth and memory requirements as examples of communication and computational requirements, respectively. Specifically, according to

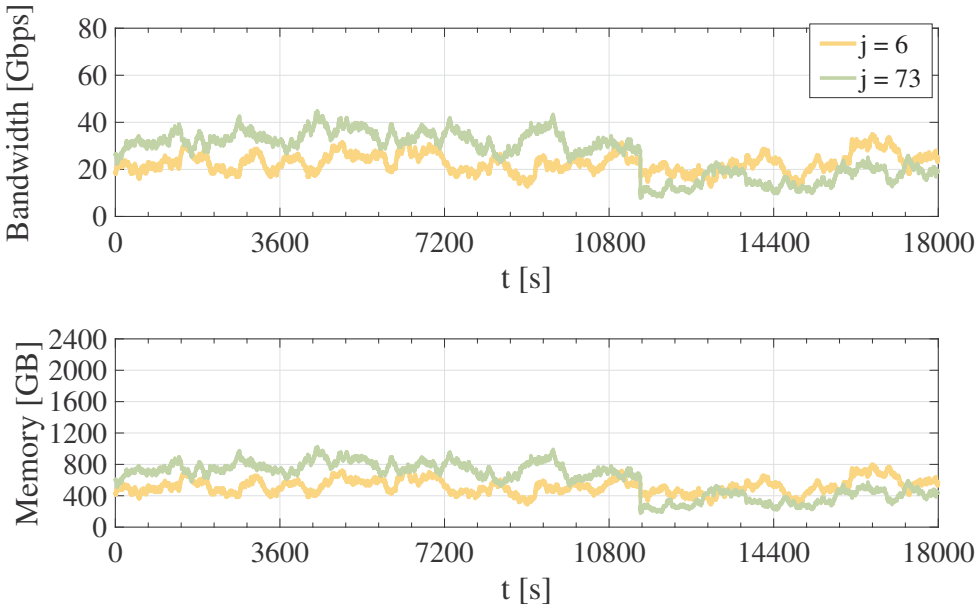


FIGURE 3.11: Aggregated communication and computational resources estimated for autonomous driving use case for $j=6$ and $j=73$.

the autonomous driving use case, for the $i-th$ user the communication bandwidth and memory requirements are set to $b_i = 700$ Mbps [148] and $m_i = 16$ GB [150], respectively (Table 3.2). For the virtual reality use case, for the $i-th$ user the communication bandwidth and memory requirements are set to 1 Gbps [148] and 32 GB [150], respectively. The highly accurate predictions of users' mobility, communication and computational resources over time within a given geographical area could significantly improve network resource optimization (as demonstrated below).

3.3.2 Latency per user

Figure 3.13 depicts the average latency per user served by each MEC server. The average latency per user for both P-C and GT-C schemes is always lower than the maximum tolerable latency τ_i. Furthermore, the proposed optimization approaches, involving a load balancing among MEC servers, keep a nearly stable and uniform latency throughout the network. Thus, they can improve the computation efficiency of MEC servers, avoiding overloaded MEC servers, as well as degraded user experience, balancing MEC servers loads and always satisfying service latency requirements [14], [161].

On the contrary, without the previous implications, the Baseline scheme registers an average latency per user that exceeds the maximum tolerable latency τ_i in high-loaded cells.

The CDFs of all the latencies per user reported in Figure 3.14 thoroughly confirm that only P-C always ensures service latency requirements for all the users in the

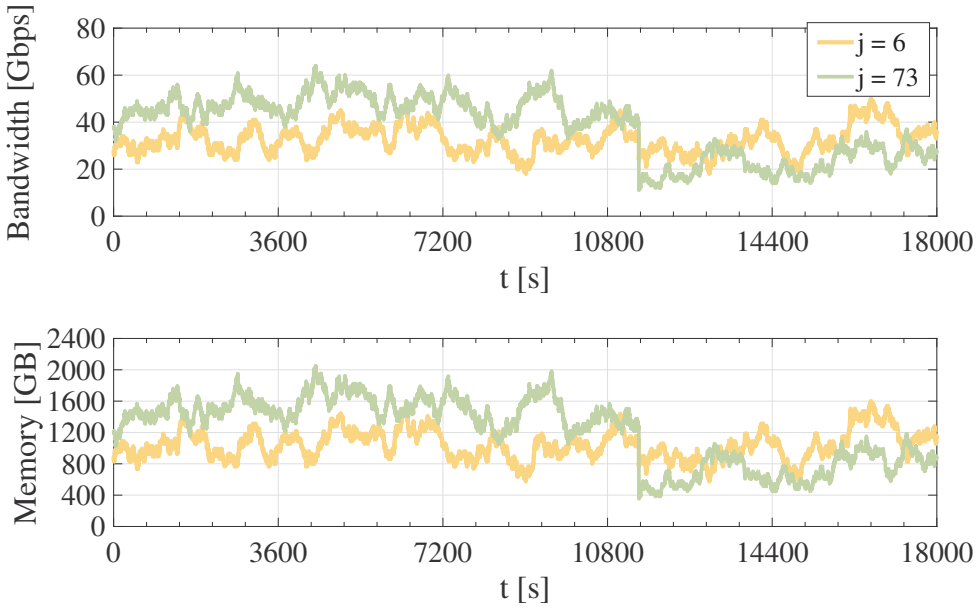


FIGURE 3.12: Aggregated communication and computational resources estimated for virtual reality use case for $j=6$ and $j=73$.

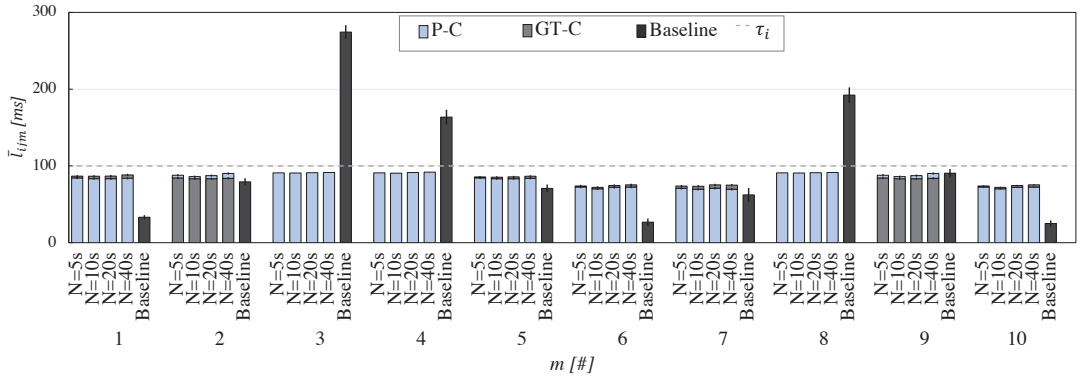


FIGURE 3.13: Average latency per user (with the 95%−confidence intervals) served by each MEC server for P-C, GT-C, and Baseline.

network, differently from the Baseline approach. To analyze the impact of the horizon N, the values of the average latency per user among MEC servers for P-C with each analyzed horizon are reported. They amount to 83.2 ms for the optimization horizon $N = 5$ s, 81.1 ms for $N = 10$ s, 82.7 ms for $N = 20$ s, and 83.6 ms for $N = 40$ s. Obtained results reveal that from $N = 5$ s to $N = 10$ s the average latency per user among different MEC servers decreases, while it tends to increase with values higher than $N = 10$ s, obtaining the highest value of latency for $N = 40$ s.

To conclude, N =10 s is a suitable optimization horizon because of slightly lower values of latency.

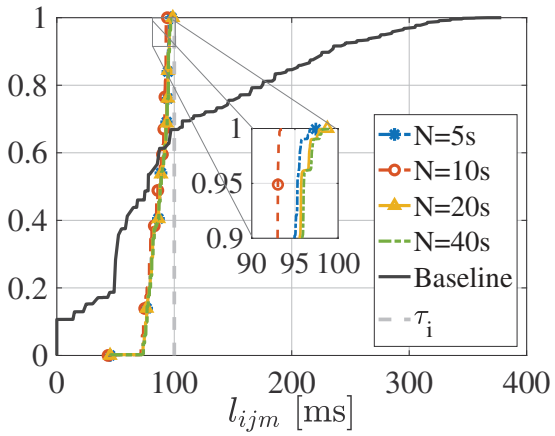


FIGURE 3.14: CDF of the latency per user for P-C and Baseline.

3.3.3 Changes among MEC servers

Figure 3.15 shows the CDFs of the number of changes among MEC servers. Reported curves demonstrate that the presented proposal generally has the highest performance levels: a number of changes equal to 0 is registered by only 28.31% of realizations for the Baseline scheme, whereas the proposed approach presents around 90% of samples with 0 changes. Focusing on the horizon N, from $N = 5$ s to $N = 10$ s the performance improves, while it tends to decrease with values higher than $N = 10$ s. In fact, when $N = 10$ s, the number of changes is always lower and the value of the $95-th$ percentile is 4 changes with respect to 7, 5, and 9 changes measured for $N = 5$ s, $N = 20$ s, and $N = 40$ s, respectively. Thus, increasing the considered optimization horizon (i.e., from $N = 5$ s to $N = 10$ s) in the optimization problem $P1$ reduces the number of changes among MEC servers. However, because of higher variability, for longer temporal horizons (i.e, $N = 20$ s and $N = 40$ s) the anticipatory network optimization approach leads to a higher number of changes among MEC servers and the highest number is reached with $N = 40$ s. The $95-th$ percentile of the presented approach with $N = 10$ s outperforms also Baseline (i.e., 5 changes).

In summary, this analysis further confirms that $N = 10$ s is a suitable optimization horizon since it minimizes the average user latency and the number of changes among MEC servers.

3.3.4 Distribution of users among MEC servers

Figure 3.16 shows the average number of users served by each MEC server. Both anticipatory network optimization methods (P-C and GT-C) are able to fairly distribute users' demands among the different MEC servers, regardless of the gNB with which they are co-located. Moreover, since the ConvLSTM architecture has very high prediction performance, P-C behaves very similarly to GT-C. They have exactly the same behaviors for MEC servers co-located with gNBs having a high

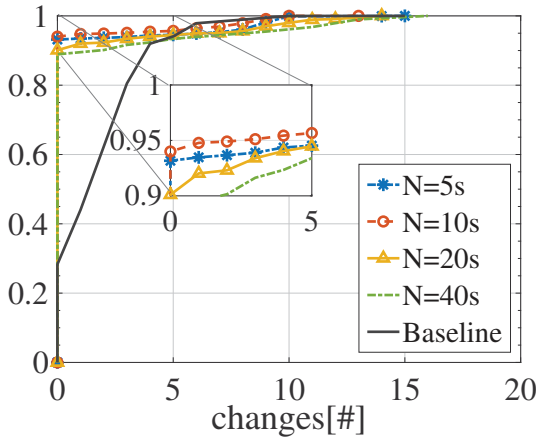


FIGURE 3.15: CDF of the number of changes among MEC servers for P-C and Baseline.

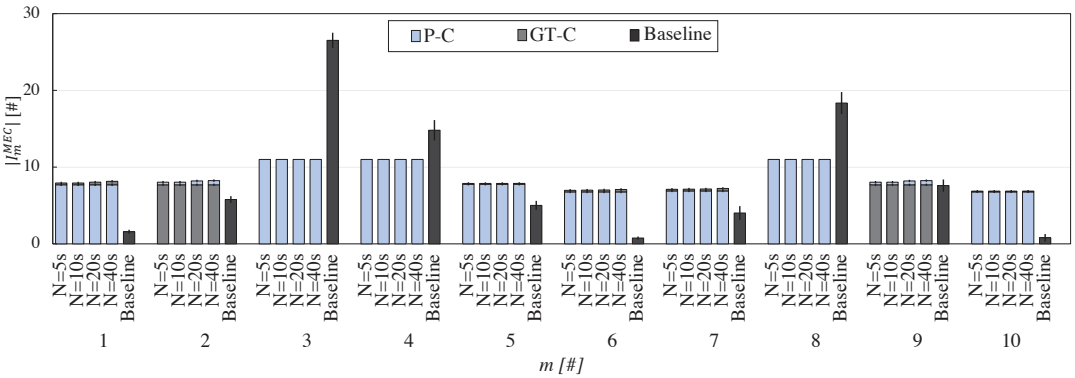


FIGURE 3.16: Average number of users (with the 95%−confidence intervals) served by each MEC server for P-C, GT-C, and Baseline.

number of users (e.g., $j=3$), that are fully used under memory and computing constraints. Also by varying the temporal horizon N, P-C and GT-C achieve a very similar average number of users served by each MEC host.

Instead, the Baseline approach is largely biased by the distribution of the users among cells and, in particular, its policy is to maintain the users at the MEC server co-located with the gNB in which they are attached.

Figure 3.17 illustrates the CDFs of the number of users served by different MEC servers. It further confirms the extremely high similarity between the trends of P-C with different N, that behaves differently from the Baseline scheme having higher variability.

3.3.5 Amount of memory consumed by MEC servers

Figure 3.18 represents the average values of the memory consumed by each MEC server. Also in this case it is possible to observe that both P-C and GT-C methods, with different horizons N, well balance the load among the MEC servers. This result confirms the fairness property investigated in the previous subsection. In high-loaded cells, the MEC servers co-located with the gNBs saturate their available

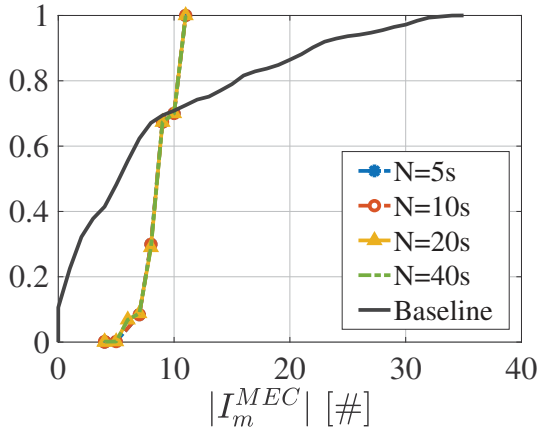


FIGURE 3.17: CDF of the number of users served by MEC servers for P-C and Baseline.

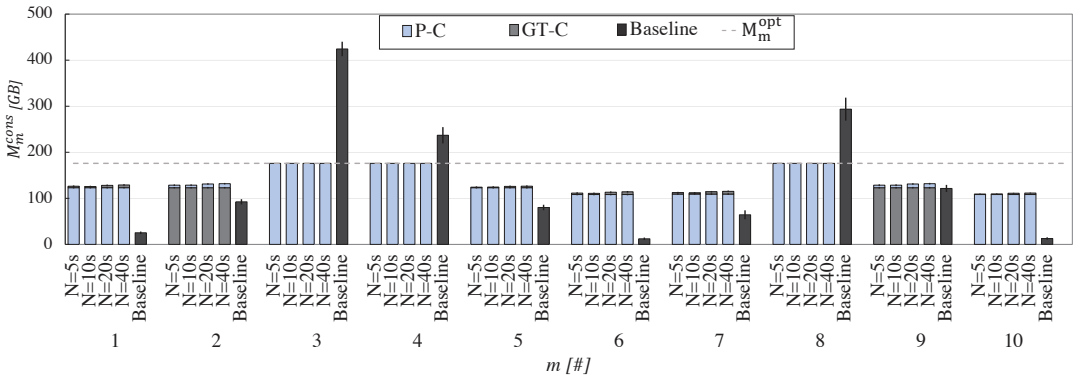


FIGURE 3.18: Average amount of memory (with the 95%−confidence intervals) consumed by each MEC server for P-C, GT-C, and Baseline.

memory. As a consequence, the proposed approaches redirect some of the requests generated within these cells towards other MEC servers, thus always satisfying the constraint reported in (3.4a).

Instead, the consumed memory M_m^{cons} for Baseline, without memory constraints, reaches an average value of around 400 GB for MEC servers corresponding to cells with a high number of users (e.g., $m = j = 3$), as demonstrated by the reported results.

As an additional confirmation, the CDFs reported in Figure 3.19 describe how the Baseline approach registers peak usage of memory equal to around 600 GB. On the contrary, the anticipatory optimization scheme developed in this work guarantees quite balancing of the amount of memory consumed in the available MEC servers, which is always below the target upper bound.

3.3.6 CPU usage of MEC servers

According to the definition in (3.3) and the related constraint (3.4b) of the formulated optimization problem $P1$, the CPU capability of each MEC server is completely consumed in all the implemented approaches. Note that the computing capacity of the $m-th$ MEC server in the optimization problem $P1$, i.e., $F_m^{opt} = 36$ Gigacycles/s, is

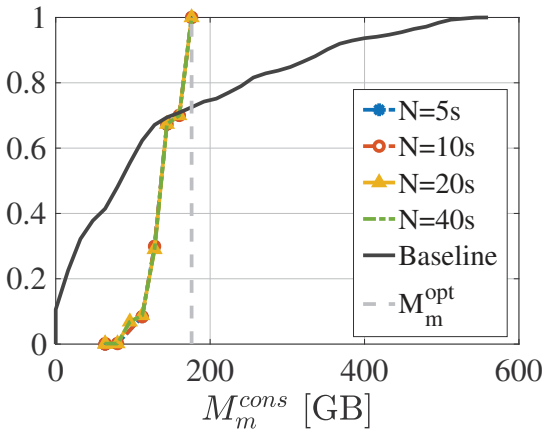


FIGURE 3.19: CDF of the amount of memory consumed by MEC servers for P-C and Baseline.

adequately set with respect to overall requests and it is lower compared to typical values of the MEC server ability. In fact, they can be greater than 1000 Gigagycles/s [167] and, with that assumption, the vast majority of the available CPU resources could be dedicated to other services and purposes.

Of course, the CPU ability of MEC server affects the execution latency experienced by each user, which is the most significant component of user latency. In fact, because of the computing sizing of MEC servers (i.e., F_m^{opt}), the average total latency per user performed through the optimization approach is generally closer to the maximum tolerable latency τ_i (as detailed in Section 3.3.2) and it validates the current assumption in considering the radio component as constant. Without load balancing among MEC servers, the same computing capacities (i.e., $F_m = F_m^{opt}$) are not sufficient to always satisfy the upper bound of service latency τ_i in the Baseline case. In particular, the average number of CPU cycles/second allocated by the $m-th$ MEC server to the $i-th$ user, i.e., $\bar{f}_{im}$, is generally lower for Baseline compared to P-C and GT-C, as demonstrated in Figure 3.20. Therefore, the Baseline case has a higher execution latency because of lower values of $\bar{f}_{im}$. Furthermore, since $\bar{f}_{im}$ is inversely related to the number of users served by the $m-th$ MEC server $|I_m^{MEC}|$, the Baseline scheme registers the lowest and the highest values of $\bar{f}_{im}$ for MEC servers co-located with gNBs having a high and a low number of users, respectively.

The related CDFs reported in Figure 3.21 illustrate the similar behaviors of P-C with different horizons N, that generally has higher values of $\bar{f}_{im}$ with respect to Baseline. Moreover, this plot confirms that the maximum possible value for the number of CPU cycles per second allocated by the $m-th$ MEC server to the $i-th$ user (i.e., f_{im}) is the CPU ability of MEC server F_m^{opt}.

3.3.7 Complexity analysis

Despite the overall better performance reported above, introducing the anticipatory methods increases the complexity in the network management system, basically due

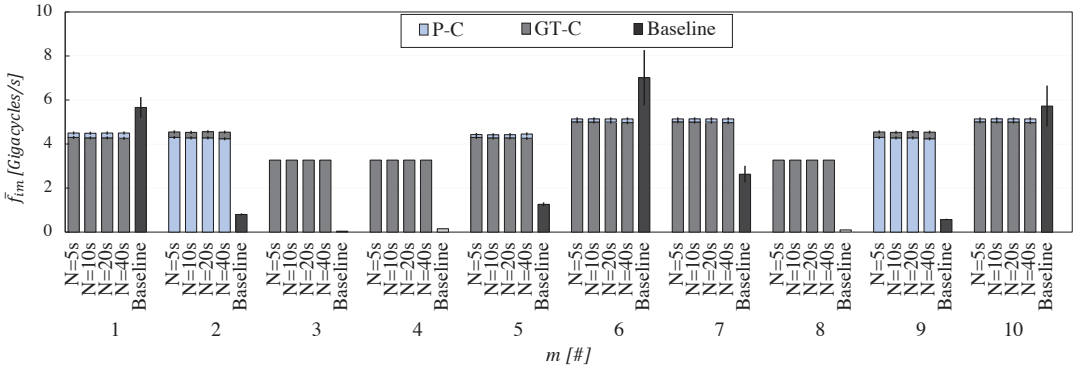


FIGURE 3.20: Average number of CPU cycles/second (with the 95%−confidence intervals), allocated by each MEC server to served users, for P-C, GT-C, and Baseline.

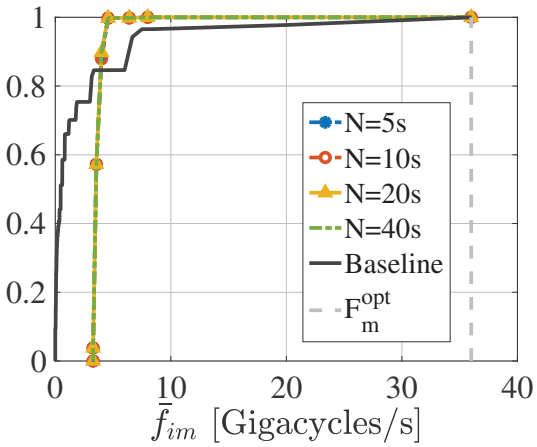


FIGURE 3.21: CDF of the average number of CPU cycles/second per user for P-C and Baseline.

TABLE 3.4: Complexity analysis per decision epoch.

Complexity parameter	Approach				
	P-C				Baseline
	N=5s	N=10s	N=20s	N=40s	
Running time	4.37s	7.81s	34.78s	78.62s	0.04s
# Objective function evaluations	5321	10125	21369	43791	-

to finding the solution to the optimization problem $P1$. In Table 3.4, P-C and Baseline are compared in terms of the average running time of each decision epoch t_k and the average total number of objective function evaluations needed for each decision epoch t_k, characterizing only P-C solved through value iteration, which is $\sum_{j\in\mathcal{J}} |I_j^{gNB}(t_{k,n})| \cdot |\mathcal{M}| \cdot N$ [168]. GT-C is omitted because the cost of the optimization process is analogous to P-C. However, here it is highlighted that the mobility prediction model required by P-C needs an extra-training phase, which early converges anyhow (Section 3.3.1). Without the optimization problem, Baseline has an extremely lower running time because it does not implement any controls and does

not anticipatorily evaluate the user distributions. For P-C, it is evident that the complexity increases with N. In fact, the larger the look-ahead horizon N, the deeper in future in the objective function of each $k-th$ optimization problem (i.e., by considering N steps ahead in each decision cycle $t_{k,n}$). Thus, P-C with $N = 5$ s has the lowest average running time and the lowest number of objective function evaluations per decision epoch. Intermediate values are reached when $N = 10$ s and $N = 20$ s and P-C with $N = 40$ s has the highest complexity. Again, $N = 10$ s is the best trade-off between performance and complexity.

Note that simulations have been executed on an Intel Core i5 CPU quad-core with 8 GB of RAM and the running time will be extremely reduced on a powerful machine with GPU, by improving the efficiency of the proposed approach [169], [170]. In particular, GPU server is at least 4-5 times faster than CPU server (with 16/24 cores) [170]. The significant profit of using a powerful machine makes the running time not only comparable to but much lower than the optimization epoch of the optimization algorithm. The effectiveness can be further enhanced (i.e., much shorter running time) by using a GPU server with more features [170], that are actually used by network operators.

It is remarked here that the encouraging results achieved by the proposed anticipatory network optimization approach open future research directions aiming at decreasing the computational complexity of the proposed solution based on dynamic programming while maintaining the same performance. To this aim, solutions based on DRL [22], [171], [172] and distributed training [173]–[176] seem interesting areas to be further explored.

3.4 Final considerations

This study has presented a novel methodology for anticipatorily allocating communication and computational resources at the network edge, and over different look-ahead temporal horizons. Specifically, the ConvLSTM has been used to predict the number of users served within a given number of cells and their related service demands, and the Dynamic Programming has been exploited to optimally allocate users' requests among MEC servers for better managing task offloading within a network slice created into a 5G system. By focusing on the autonomous driving use case, computer simulations have demonstrated the capability of the proposed solution to fairly distribute users' requests at the network edge, while satisfying communication and computational constraints, as well as meeting latency constraints of the considered service.

Chapter 4

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

A Tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is proposed in this Chapter to achieve an important step forward in the management of network slicing at the RAN level.

B5G networks are expected to support various new use cases from vertical industries, which impose a wide range of performance and requirements. Network slicing is 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 [8], [177]–[181]. At the same time, it promises to open new business models for all the interested stakeholders (that are Infrastructure Providers and Service Providers or Tenants), while intensifying the collaboration among all the involved parties and keeping their requirements distinct [119], [182]. 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, and configure (potentially on-demand) the corresponding network slice instances. On the other hand, Tenants should be able to submit their requests, without having a complete understanding of the network itself. The management of network slicing in the core network has been intensely investigated in the current scientific literature. On the contrary, handling network slicing in the 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 QoS requirements of different services pose significant technical challenges on the management and provisioning of RAN slicing [180], [183]. To this end it would be essential to pervasively adopt AI within the slicing paradigm other than to anticipatorily allocate communication and computational resources at the network edge for solving the task offloading problem (as analyzed in the previous

Chapter), especially in view of the ever-increasing network complexity of future mobile systems due to resource sharing among multiple entities [13], [100]. Most of the contributions employing AI-based methods for channel estimation and the management of network slicing in the core network and RAN, propose centralized solutions based on deep learning and RL/DRL, where the network status is fully observable (please see Section 1.3.3 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 view of the network status [8], [9], [184]. 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, a novel Tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence, that can be fully implemented in the business and privacy-preserving vision of network slicing, is proposed in this Chapter. Thanks to the Tenant-driven nature of the proposal, Tenants can operate independently (by only use their own partial view of the network status) on the underlying infrastructure to dynamically adapt bandwidth requests and guarantee their expected service performance [8], [185]. This aspect also provides higher scalability with respect to any centralized method [119]. Moreover, according to the Pervasive Intelligence paradigm, both Infrastructure Provider and Tenants exploit AI to accomplish their tasks. This is fully aligned with the pervasive intelligent endogenous design of future generations of mobile networks [100]. 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 DDPG algorithm 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. 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 over-provisioning.

The efficiency of the devised Tenant-driven RAN slicing enforcement scheme based on Pervasive Intelligence is investigated for the eMBB and Remote Driving use cases, by using computer simulations with real and conceivable network and QoS settings in compliance with ITU and 5G specifications [186]–[188]. 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

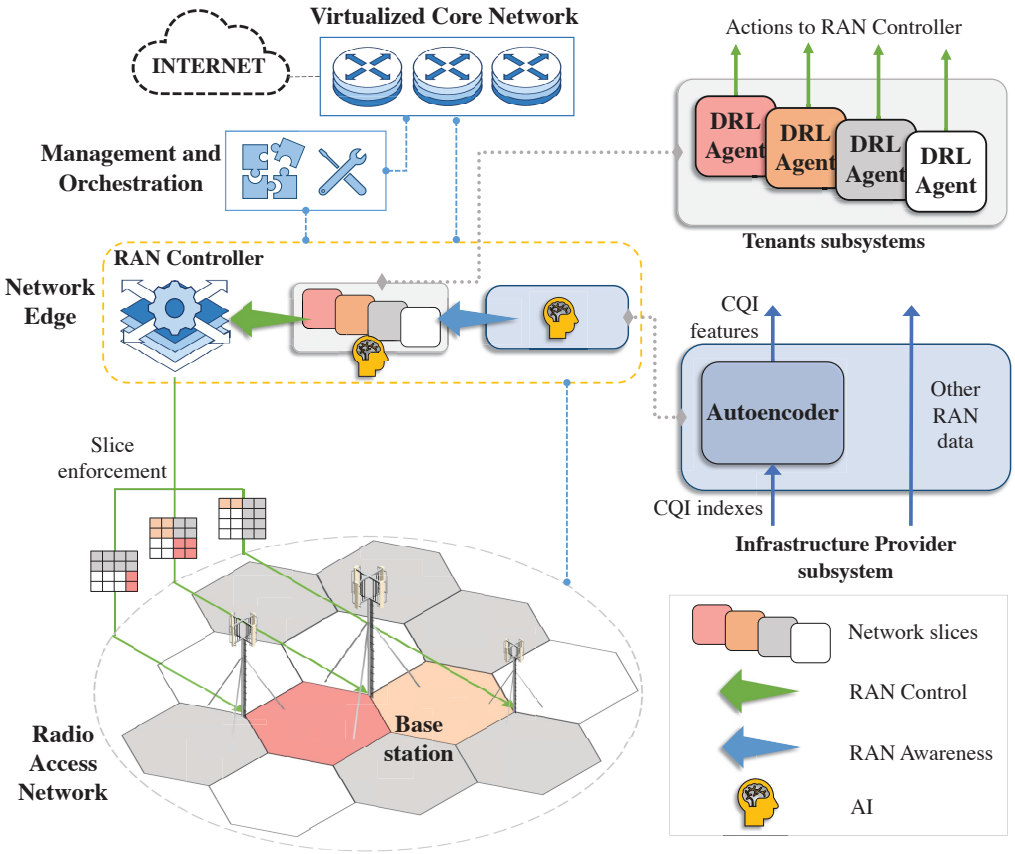


FIGURE 4.1: The reference architecture.

target Service Availability.

4.1 The proposed scheme

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 Tenants for the creation of independent slices [189]. 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 [8], [9], [178]. To this end, different logical entities and networking functions are exploited across the overall communication infrastructure, including core network, edge network, and RAN (see Figure 4.1).

Specifically, the Management and Orchestration entity is in charge of configuring the entire system. 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 ensures a dynamic RAN slicing enforcement strategy. 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 [190]. However, as already mentioned, the discussion will consider the run-time phase only, dedicated to the slicing enforcement.

As expected, the Infrastructure Provider has a comprehensive view of the network and it can access to data not natively accessible for Tenants. For example, the Infrastructure Provider subsystem is the only entity able to retrieve information from the RAN. 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 [191], in fact, it is assumed to be used in 6G & Beyond as well. Among the other parameters carried by the CSI feedback, the CQI provides information about the current communication channel quality.

The methodology presented in this work assumes that the Tenant subsystem may control its slice 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 [103]. 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 dimensionality of data, the autoencoder is used to generate a feature vector representing the CQI feedbacks, without requiring the knowledge of data distribution nor the explicit identification of a certain structure [37], [192]. 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 4.1.1.

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. 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. The requests that the Tenant

subsystem may submit to the RAN controller include bandwidth allocation, variation of radio resource scheduling algorithms, Hybrid Automatic Repeat reQuest (HARQ) configurations, channel coding schemes, power control strategies, multicast/broadcast activation, or beam management [193], [194]. Also, these requests must be issued in real-time, to successfully meet end-users requirements under the current RAN conditions.

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 [184]. In this context, the scenario appears as a classical MDP:

- the *environment* is the cellular network architecture;
- the *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 for supporting the slice enforcement strategy. 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 locally without exchanging information among each other. This reduces communication overheads and also improves the security and robustness of the networks [22]. 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 move to a new state $\mathbf{s}$ [195]. Section 4.1.2 will provide more details about the implemented DRL framework.

To facilitate the understanding of the notations adopted in what follows, a summary of symbols is reported in Table 4.1.

TABLE 4.1: List of mathematical symbols adopted in Chapter 4.

Symbol	Description
$\mathbf{Y}$	Input spatial snapshot of CQI indexes
K	Number of rows of $\mathbf{Y}$
L	Number of columns of $\mathbf{Y}$
N_l	Number of filters used by the $l-th$ convolutional layer of the autoencoder
r_l	Number of rows of the $l-th$ convolutional layer of the autoencoder
c_l	Number of columns of the $l-th$ convolutional layer of the autoencoder
v_l	Vertical step size of the $l-th$ convolutional layer of the autoencoder
h_l	Horizontal step size of the $l-th$ convolutional layer of the autoencoder
ch_l	Number of channels for the channel-wise normalization of the $l-th$ convolutional layer of the autoencoder
$\mathbf{f}$	Feature learning representation vector
F	Dimension of $\mathbf{f}$
$\hat{\mathbf{Y}}$	Reconstructed input snapshot
β_s	Mini-batch size
a	Action
$\mathcal{A}$	Action space
$\mathbf{s}$	State
$\mathcal{S}$	State space
R	Reward
$\mathbf{u}$	Number of users per sector
W	Number of cell sectors in the system
w	Sector index
σ	Service Availability
$\hat{Q}(s,a\|\theta^Q)/\hat{Q}'(s,a)$	Approximation through which the critic/target critic network evaluates the Q function
$\mu(s\|\theta^\mu)/\mu'(s)$	Policy modeled by the actor/target actor network
$\theta^Q/\theta^{Q'}$	Parameters of the critic/target critic network
$\theta^\mu/\theta^{\mu'}$	Parameters of the actor/target actor network
L_c	Loss minimized by the critic network
M	Number of experiences sampled from the experience replay
J	Environment start distribution
λ_n	Number of neurons of the $n-th$ fully connected layer of the actor and critic networks
P_{LOS}	LOS probability
PL_{LOS}/PL_{NLOS}	Pathloss model in the case of LOS/NLOS propagation condition
$\mathcal{T}$	Downlink target rate
B	Maximum bandwidth that each Tenant subsystem may request for every sector
p	Percentile of the SINR distribution
$\mathcal{E}$	Episode Availability Indicator
N_E	Number of test episodes
ϵ_t	Parameter related to the target Service Availability σ
t	Test episode index

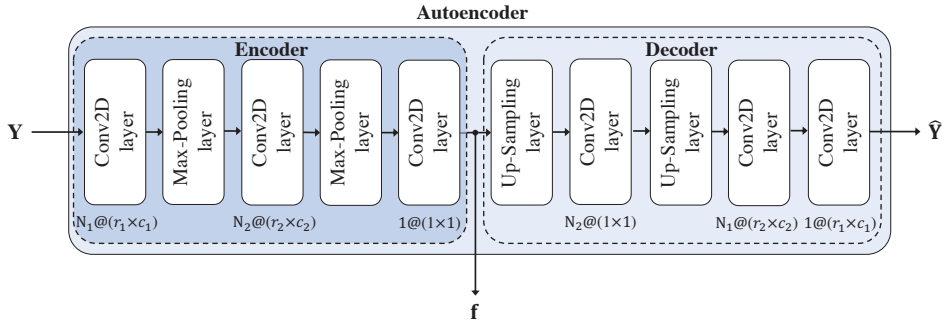


FIGURE 4.2: Architecture of the adopted convolutional autoencoder.

4.1.1 Design of the Autoencoder used by the Infrastructure Provider subsystem

As anticipated in Section 2.2.1, the autoencoder is a particular 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. As depicted in Figure 4.2, the investigated encoder is made of three chained 2-dimensional convolutional layers to extract spatial correlations of CQI indexes snapshots [135]. Each layer comprises a set of filters which are convolved with the CQI indexes snapshot to extract the features of a certain input region and with the ReLU activation function. 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 [17], [41]. 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 [196]. 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 [17], [41], and three convolutional layers, with the ReLU activation function, except for the output layer, which uses the sigmoid activation function [17]. 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 [134]. 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 MSE loss function. Formally, the MSE loss function is defined as [135], [138]:

$$MSE = \frac{1}{\beta_s} \sum_{b=1}^{\beta_s} \sum_{k=1}^{K} \sum_{l=1}^{L} \left(\hat{y}_{b,k,l} - y_{b,k,l} \right)^2, \tag{4.1}$$

where β_s 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 $\mathbf{f}$) [37] 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.

4.1.2 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 with 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 allowable bandwidth. It is a continuous value between 0 and 1 (i.e., 0% and 100% of

the bandwidth made available to the Tenant).

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

$$\mathbf{s} = (\mathbf{u}, \mathbf{f}, \sigma) \tag{4.2}$$

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, divided by the amount of time the Tenant is expected to deliver the service [187].

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 only rely on the compressed version of the radio link conditions of their users (i.e., compressed network status) [191], [197].

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:

$$R = \begin{cases} 1 - a, & \text{if the target Service Availability } \sigma \text{ is} \\ & \text{guaranteed;} \\ -1, & \text{otherwise.} \end{cases} \tag{4.3}$$

Thus, according to the Tenant service and the agreed QoS, 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 [195], [198]. It is an actor-critic, model-free, off-policy DRL method which computes an optimal policy that maximizes the long-term reward [22], [199]. DDPG primarily uses two neural networks, one for the actor and one for the critic, as illustrated in Figure 4.3. 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.

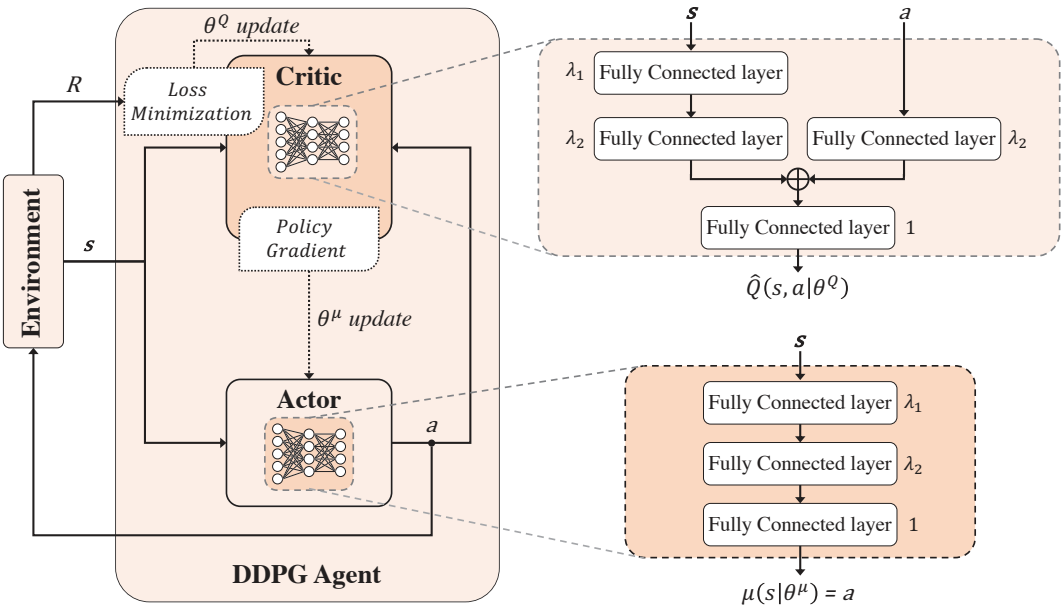


FIGURE 4.3: Architecture of the adopted DDPG algorithm.

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_c:

$$L_c = \frac{1}{M}\sum_{i=1}^{M}\left(y_i - \hat{Q}(s_i, a_i|\theta^Q)\right)^2, \tag{4.4}$$

where M is the number of experiences sampled from the experience replay (i.e., where the agent stores each of its experiences during training) [200], $y_i = R_i + \gamma\hat{Q}'(s_i, \mu'(s_i'|\theta^\mu)|\theta^{Q'})$ is the Q function target approximated through bootstrapping [195], γ is the future reward discount factor [195], 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:

$$\nabla_{\theta^\mu} J \approx \frac{1}{M}\sum_{i=1}^{M}\nabla_{\mu(s_i)}\hat{Q}\left(s_i, \mu(s_i)|\theta^Q\right)\nabla_{\theta^\mu}\mu(s_i|\theta^\mu), \tag{4.5}$$

where J is the environment start distribution as defined in the *policy gradient theorem* [195].

To further elaborate, the *state* **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 Figure

4.3, 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.

4.2 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 [186]. 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), is 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 a network layout configuration that wraps around the simulation map as one gets to one of its borders. 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 ITU guidelines [186]. 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 P_{LOS} is given by the following:

$$P_{LOS} = \begin{cases} 1, & d_{2D} \leq 18m; \\ P'_{LOS}, & d_{2D} > 18m; \end{cases} \tag{4.6}$$

where

$$P'_{LOS} = \left[\frac{18}{d_{2D}} + e^{-d_{2D}/63}\left(1 - \frac{18}{d_{2D}}\right)\right]\left(1 + C'(H_{UE})\frac{5}{4}\left(\frac{d_{2D}}{100}\right)^{3} e^{-d_{2D}/150}\right), \tag{4.7}$$

with

$$C'(H_{UE}) = \begin{cases} 0, & H_{UE} \leq 13m; \\ \left(\frac{H_{UE}-13}{10}\right)^{1.5}, & 13m < H_{UE} \leq 23m. \end{cases} \tag{4.8}$$

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:

$$PL_{LOS} = \begin{cases} PL_1, & d_{2D} < d_{bp1} \\ PL_2, & d_{2D} > d_{bp1} \end{cases} \tag{4.9}$$

where

$$d_{bp1} = 4(H_{gNB} - 1)(H_{UE} - 1)(\omega/c) \tag{4.10}$$

and

$$PL_1 = 28.0 + 22\log_{10}(d_{3D}) + 20\log_{10}(\omega), \tag{4.11}$$

$$PL_2 = 40\log_{10}(d_{3D}) + 28.0 + 20\log_{10}(\omega) - 9\log_{10}\left({d_{bp1}}^2 + (H_{gNB} - H_{UE})^2\right). \tag{4.12}$$

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

$$PL_{NLOS} = max(PL_{LOS}, PL'_{NLOS}), \tag{4.13}$$

where

$$PL'_{NLOS} = 161.69 + \left(43.42 - 3.1\log_{10}(H_{gNB})\right)\left(\log_{10}(d_{3D}) - 3\right) + 20\log_{10}(\omega) + \\ -\left(24.37 - \frac{1480\log_{10}(H_{gNB})}{H_{gNB}^2}\right) - 0.6(H_{UE} - 1.5) - \left(3.2(\log_{10}(17.625))^2 - 4.97\right). \tag{4.14}$$

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.

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, which is exploited to estimate the upper bound of the performance. 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, Figure 4.4 shows the CDF 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 [201].

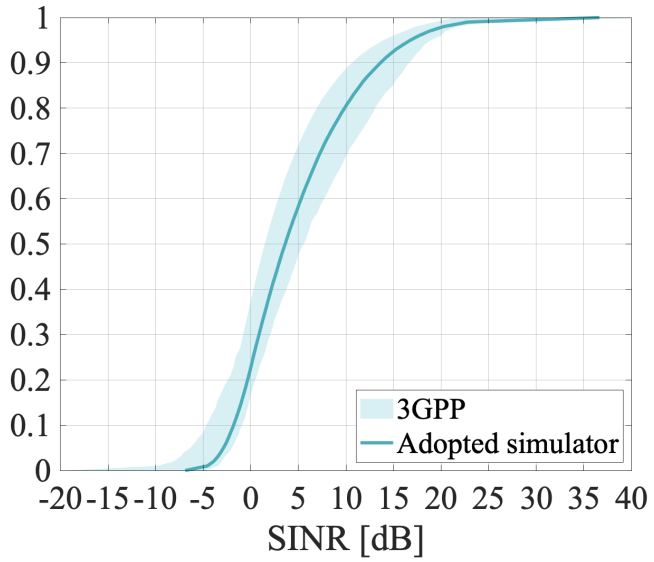


FIGURE 4.4: CDF of the wideband SINR of the developed simulator with respect to 3GPP Phase 1 dense-urban (macro-layer) system-level calibration for multi-antenna systems.

TABLE 4.2: Scenarios.

	eMBB	**Remote Driving**
Network deployment	ITU Dense Urban eMBB [186]	ITU Urban Macro URLLC [186]
UE speed	3 km/h [186]	30 km/h [186]
UE density	2000 UE/km^2 [187]	1200 UE/km^2 [188]
Downlink target rate $\mathcal{T}$	50 Mbps [187]	400 kbps [188]
Service Availability σ	90%	99%
Traffic model	Full-buffer [186]	Full-buffer [186]

Without loss of generality, the Tenant subsystems are assumed to operate two types of network slices, i.e., eMBB and Remote Driving slices, with real and conceivable network and service settings. Of course, the whole scheme can be applied to each type of slice and scenario by correctly adapting the related parameter settings.

For the eMBB scenario, the speed and density of UEs are set to 3 km/h [186] and 2000 UE/km^2 [187], respectively, and the downlink target rate and the Service Availability are set to $\mathcal{T} = 50$ Mbps [187] and $\sigma = 90\%$, respectively, according to the ITU Dense Urban eMBB deployment [186]. For the Remote Driving scenario, in line with the ITU Urban Macro URLLC deployment [186], the speed and density of UEs are set to 30 km/h [186] and 1200 UE/km^2 [188], respectively, and the downlink target rate and the Service Availability are set to $\mathcal{T} = 400$ kbps [188] and $\sigma = 99\%$, respectively. Then, the full-buffer traffic model is implemented for both scenarios [186]. Table 4.2 summarizes the parameter settings.

TABLE 4.3: 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

4.2.1 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 4.2, 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 to have snapshots of the same dimensions. Thus, the autoencoder must not be retrained if the number of UEs changes in the network scenario.

Different configurations of convolutional autoencoders, characterized by different values of parameters listed in Section 4.1.1, 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 4.3 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 4.3, that is $F = 6$ (i.e., a sufficient number of representative features on radio channel conditions).

The training set, whose performance is listed and evaluated in Table 4.2, 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, lasts 100 epochs (i.e., complete passes

through the training data [142] such that each example has been seen once) for all the evaluated configurations of convolutional autoencoders. The Adam optimization [143], with a learning rate of 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. The RMSE represents the root of the MSE, as defined in (4.1), 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. 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 4.3 for all the evaluated configurations of convolutional autoencoders. The second-last configuration, which is highlighted in Table 4.3, represents a good trade-off between 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. Figure 4.5 shows the autoencoder loss as a function of the number of epochs for the training set and the validation set. The validation set reaches slightly lower values of loss with respect to the training set because it is not hard to reconstruct. 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, Figure 4.6 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.

4.2.2 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. Specifically, a DDPG algorithm is implemented by Tenant subsystems. As anticipated, two different Tenant subsystems are taken into account. They are assumed to provide eMBB services and Remote Driving services, whose target Downlink rate $\mathcal{T}$ and Service Availability σ are reported in Table 4.2). Each episode lasts 1 s, i.e., the DRL agent performs its actions every second. This

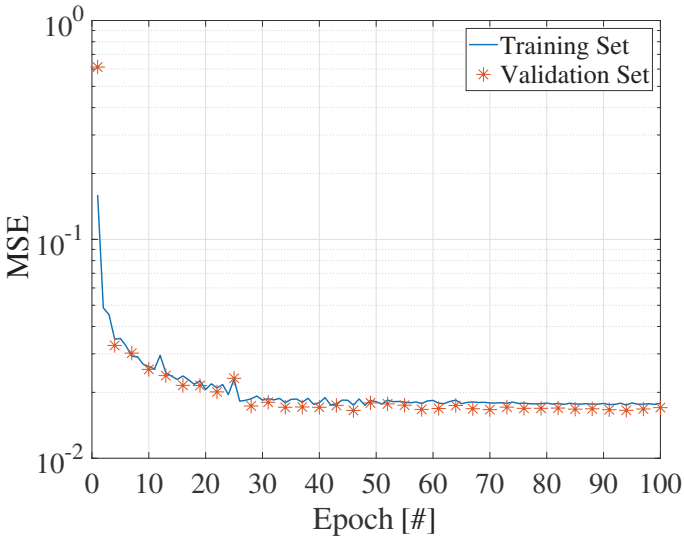


FIGURE 4.5: Autoencoder loss (i.e., MSE) vs number of epochs.

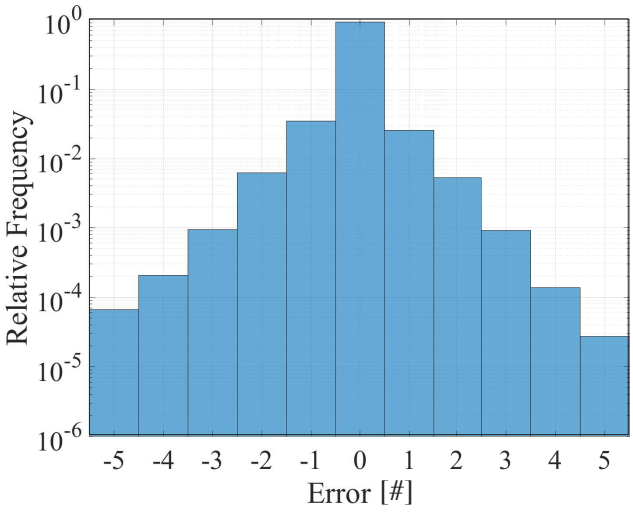


FIGURE 4.6: Relative frequency of the reconstruction errors on the test set.

means that the Tenant subsystems can update their bandwidth allocation requests every second (i.e., with an allocation period of 1 s). Each Tenant subsystem may request a maximum bandwidth B of 100 MHz for every considered sector. Note that the sum of requested bandwidth cannot exceed the bandwidth of the Infrastructure Provider. As for the actor and critic neural networks, the learning rate is set to 0.001 and 0.0001, respectively; after some attempts, 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 to 50.

Figure 4.7 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 Remote Driving scenarios converges to stable values after 20 and 15

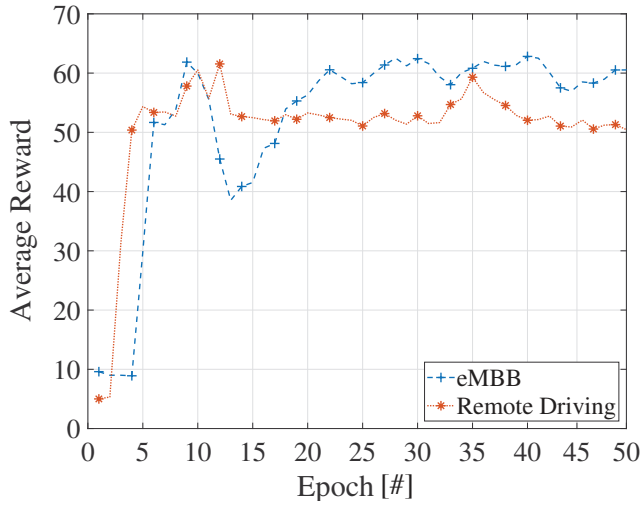


FIGURE 4.7: Average episode reward vs number of epoch (with 1 epoch corresponding to 100 training episodes) for eMBB and Remote Driving scenarios.

training epochs, respectively. Thus, 50 training epochs are sufficient for convergence (and that is the reason why the number of training epochs is set to 50).

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 is important to note that the bandwidth, in this case, is determined through iterative adjustments during simulations. Therefore, the Genie approach is infeasible in actual deployments;

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

- *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 Remote Driving use cases), the action a is computed at each step according to the following:

$$a = \min\left\{\left(\max_{1\leq w\leq W} u_w \cdot \frac{\mathcal{T}}{B\log_2(1+SINR_p)}\right), 1\right\} \tag{4.15}$$

where u_w is the number of UEs in the $w-th$ 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 [202], and $p = 50\%$, namely the median SINR [202], are considered. Thus, the Heuristic approach allocates a

TABLE 4.4: Episode Availability Indicators $\mathcal{E}$ for the analyzed approaches.

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

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:

$$\mathcal{E} = \frac{1}{N_E} \sum_{i=1}^{N_E} \epsilon_i \cdot 100 \tag{4.16}$$

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

$$\epsilon_t = \begin{cases} 1, & \text{if the Service Availability } \sigma \text{ is guaranteed} \\ & \text{in the } t-th \text{ test episode, } \forall t; \\ 0, & \text{otherwise.} \end{cases} \tag{4.17}$$

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. Note that the total number of test episodes N_E is set to 500.

Table 4.4 reports the Episode Availability Indicators $\mathcal{E}$ performed by the analyzed approaches for eMBB and Remote Driving 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 $\mathcal{E}$, i.e., it allows to always provide the service with 90% and 99% Service Availability σ for eMBB and Remote Driving 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 $p = 50\%$), and it is never obtained by the *Random* approach. In the Remote Driving 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.

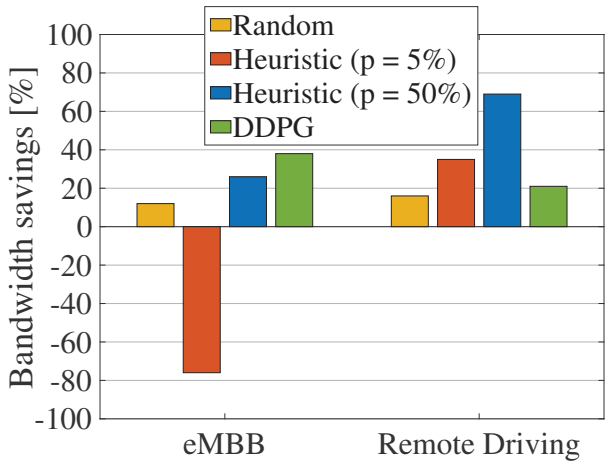


FIGURE 4.8: Comparison among different approaches with respect to the Genie in terms of bandwidth savings.

Figure 4.8 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 requires a greater amount of bandwidth than the *Genie*. In the case of Remote Driving, 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 Availability Indicator $\mathcal{E}$ equal to 100%. Thus, it can be considered as the winning approach also for this scenario.

To sum up, the DRL agents used by the Tenant subsystems, which implement the DDPG algorithm, actually learn to save bandwidth, 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 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.

4.3 Final considerations

This Chapter has presented a novel Tenant-driven RAN 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 RAN. 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 DRL algorithm (i.e., DDPG) to dynamically adapt bandwidth requests. Finally, the resulting outcomes are employed by the Infrastructure Provider to effectively enforce the RAN slicing. The comparison with conventional resource allocation methods has demonstrated that the proposed approach ensures the best trade-off between bandwidth savings and bandwidth over-provisioning, while always guaranteeing the target Service Availability.

Conclusions

This thesis described innovative solutions conceived for the analysis and the optimal management of mobile networks that adopt Machine Learning techniques towards Pervasive Intelligence.

In particular, after tailoring an unsupervised learning methodology to characterize radio resource utilization patterns, a Multi-Task Learning model, running directly at the network edge, was effectively trained with a real dataset collected from the control channel of an operative mobile network. It jointly anticipated information on the type of traffic to be served and the radio resource utilization pattern requested by each service. To this end, it exploited the Undercomplete and the Sequence to Sequence autoencoders as key building blocks for obtaining common feature representations for traffic classification and prediction. A cross-comparison with respect to conventional single-task learning schemes demonstrated that the Multi-Task Learning architectures always guarantee higher performance than the single-task learning approach.

This thesis also demonstrated that Machine Learning techniques can effectively aid the anticipatory allocation of communication and computational resources at the network edge, and over different look-ahead temporal horizons. Specifically, in a realistic autonomous driving use case, the number of users served within a given number of cells and their related service demands were predicted through the Convolutional Long Short-Term Memory and the Dynamic Programming was exploited to optimally allocate users' requests among Multi-access Edge Computing servers for better managing task offloading.

Also the management of network slicing in the Radio Access Network can benefit from the ubiquitous and pervasive adoption of Artificial Intelligence mechanisms, avoiding the radio resources over-provisioning while saving bandwidth. At the basis of the proposed Tenant-driven Radio Access Network slicing enforcement scheme, a convolutional autoencoder was exploited by the Infrastructure Provider to compress network status and share it with Tenants. In turn, in realistic enhanced Mobile BroadBand and Remote Driving scenarios, a Deep Deterministic Policy Gradient algorithm was implemented by each Tenant to dynamically adapt the amount of bandwidth to assign to its slice. At the end, the Infrastructure Provider employed the resulting outcomes to effectively enforce the Radio Access Network slicing.

The investigation in real scenarios and the comparison against conventional

approaches adopted for the analysis and the optimal management of mobile networks demonstrated the effectiveness of the proposed Machine Learning-based approaches. Moreover, the proposed solutions were evaluated in scenarios with changing conditions to encompass a variety of deployments (e.g., in terms of different number of classes for traffic analysis or different areas for mobility).

The conceived methodologies could be merged and jointly exploited to properly design the future mobile networks, where Artificial Intelligence should be natively and pervasively integrated into various layers of the network for enabling full network automation. Mobile traffic classification and prediction will be jointly performed with the mobility prediction by considering the even-more complex network scenarios with heterogeneous services sharing a variable amount of resources at the network edge. Future research activities will further extend also the proposed Tenant-driven Radio Access Network slicing enforcement scheme by predicting the network status, in addition to its compression, to boost the presence and the exploitation of Pervasive Intelligence in the network. Finally, recent Machine Learning deployments, i.e., solutions based on distributed learning, need to be considered in order to decrease the computational complexity and to address critical issues such as data privacy, data security, and data access rights to heterogeneous data. In contrast to traditional centralized learning techniques, distributed learning approaches will train Machine Learning algorithms across multiple local datasets, managed by decentralized edge devices or servers holding local data samples. In this context, Federated Learning with its privacy and security preservation nature is particularly attractive to achieve ubiquitous Artificial Intelligence in 6G and Beyond mobile networks. Thus, it will be investigated to further reduce the monitoring system complexity and the storage capacity of the Multi-Task Learning model at the edge, or bandwidth, computational complexity, and energy consumption in the task offloading problem, or to enhance the privacy preservation as well as the independence of Tenant actions in multiple slices scenarios. In fact, the efficiency of the Machine Learning models is an open issue to be taken into account since Machine Learning carbon footprint is starting to be high as its adoption in many fields is getting so widespread. Thus, distributed implementations as Federated Learning will become more and more important because, by having similar performance, they could save energy with respect to their correspondent centralized versions.

To conclude, Artificial Intelligence techniques are and will increasingly be the basis of advanced techniques for mobile network optimization, ranging from radio resource scheduling and admission control, resource and mobility management and energy saving mechanisms, to network slicing and dynamic placement of virtualized functions, so as to achieve the Pervasive Intelligence paradigm in next generation mobile systems.

Acknowledgements

The achievement of this important goal would not have been possible without the help of many.

I would like to thank my Ph.D. supervisor Prof. Gennaro Boggia and Prof. Giuseppe Piro. Their precious guidance encouraged me during my Ph.D. course, by giving me the opportunities for personal growth and to achieve the results presented in this thesis. I would like to express my sincere gratitude to Dr. Paolo Dini for his continuous advice, unwavering support, and assistance at every stage of the research project. I wish to extend my special thanks to Prof. Andrea Abrardo, who helped me finalize my project, and to Prof. Alfredo Grieco and the Telematics Lab group for their teachings. I am also indebted to the Referees, Prof. Michele Rossi and Dr. Marco Miozzo, for their treasurable comments and useful suggestions that allowed me to improve the quality of my work.

I would like to thank my colleagues and my friends (especially those who are both) for sharing the most challenging and exciting moments of the last years together. I also thank all the people I met during my way who spurred me on. Finally, my biggest thanks go to my family for continuously supporting me and for always being next to me.

Bibliography

[1] NGMN Alliance, "5G White Paper", *White Paper*, 2015.

[2] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, "Toward Edge Intelligence: Multiaccess Edge Computing for 5G and Internet of Things", *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6722–6747, 2020. DOI: 10.1109/JIOT.2020.3004500.

[3] A. Dogra, R. K. Jha, and S. Jain, "A Survey on Beyond 5G Network With the Advent of 6G: Architecture and Emerging Technologies", *IEEE Access*, vol. 9, pp. 67 512–67 547, 2020. DOI: 10.1109/ACCESS.2020.3031234.

[4] 3GPP, "5G; NR; Physical channels and modulation (Release 15)", 3rd Generation Partnership Project, Tech. Rep. 38211, 2020.

[5] ——, "5G; NR; Overall description (Release 15)", 3rd Generation Partnership Project, Tech. Rep. 38300, 2020.

[6] 5G PPP Architecture Working Group, "View on 5G Architecture", *White Paper*, 2019, V3.0.

[7] R. Alvizu *et al.*, "Comprehensive Survey on T-SDN: Software-Defined Networking for Transport Networks", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2232–2283, 2017. DOI: 10.1109/COMST.2017.2715220.

[8] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges", *Computer Networks*, vol. 167, p. 106 984, 2020. DOI: 10.1016/j.comnet.2019.106984.

[9] Q. Pham *et al.*, "A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art", *IEEE Access*, vol. 8, pp. 116 974–117 017, 2020. DOI: 10.1109/ACCESS.2020.3001277.

[10] ETSI, "MEC in 5G Networks", *White Paper*, no. 18, 2018.

[11] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017. DOI: 10.1109/COMST.2017.2682318.

[12] ETSI, "Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements", European Telecommunications Standards Institute, Group Specification (GS) V2.1.1, 2018.

[13] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine Learning for 5G/B5G Mobile and Wireless Communications: Potential, Limitations, and future Directions", *IEEE Access*, vol. 7, pp. 137 184–137 206, 2019. DOI: 10.1109/ACCESS.2019.2942390.

[14] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017. DOI: 10.1109/COMST.2017.2745201.

[15] K. Zheng *et al.*, "Big Data-Driven Optimization for Mobile Networks Toward 5G", *IEEE Network*, vol. 30, pp. 44–51, 2016. DOI: 10.1109/MNET.2016.7389830.

[16] C. Jiang *et al.*, "Machine Learning Paradigms for Next-Generation Wireless Networks", *IEEE Wireless Communications*, vol. 24, no. 2, pp. 98–105, Apr. 2017, ISSN: 1536-1284. DOI: 10.1109/MWC.2016.1500356WC.

[17] C. Zhang, P. Patras, and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019, ISSN: 2373-745X. DOI: 10.1109/COMST.2019.2904897.

[18] M. Paolini, *Mastering Analytics: How to benefit from big data and network complexity: An Analyst Report*. RCR Wireless News, 2017. [Online]. Available: http://content.rcrwireless.com/20170620_Mastering_Analytics_Report.

[19] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues", *IEEE Communications Surveys & Tutorials*, 2019. DOI: 10.1109/COMST.2019.2924243.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[21] 5G PPP Technology Board, "AI and ML – Enablers for Beyond 5G Networks", *White Paper*, 2021, V3.0.

[22] N. C. Luong *et al.*, "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019. DOI: 10.1109/COMST.2019.2916583.

[23] S. Zhang, J. Liu, T. K. Rodrigues, and N. Kato, "Deep Learning Techniques for Advancing 6G Communications in the Physical Layer", *IEEE Wireless Communications*, pp. 1–7, 2021, to be published. DOI: 10.1109/MWC.001.2000516.

[24] P. Torres *et al.*, "Data analytics for forecasting cell congestion on LTE networks", in *Proc. of IEEE Network Traffic Measurement and Analysis Conference (TMA)*, 2017, pp. 1–6. DOI: 10.23919/TMA.2017.8002917.

[25] M. Z. Shafiq *et al.*, "A first look at cellular network performance during crowded events", in *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, 2013, pp. 17–28. DOI: 10.1145/2494232.2465754.

[26] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile Encrypted Traffic Classification Using Deep Learning", in *Proc. of IEEE Network Traffic Measurement and Analysis Conference (TMA)*, 2018, pp. 1–8. DOI: 10.23919/TMA.2018.8506558.

[27] T. Stöber, M. Frank, J. Schmitt, and I. Martinovic, "Who do you sync you are?: Smartphone fingerprinting via application behaviour", in *Proc. of 6th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2013, pp. 7–12. DOI: 10.1145/2462096.2462099.

[28] Y. Liu, S. Zhang, B. Ding, X. Li, and Y. Wang, "A cascade forest approach to application classification of mobile traces", in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6. DOI: 10.1109/WCNC.2018.8377311.

[29] Q. Wang, A. Yahyavi, B. Kemme, and W. He, "I know what you did on your smartphone: Inferring app usage over encrypted data traffic", in *IEEE Conference on Communications and Network Security (CNS)*, 2015, pp. 433–441. DOI: 10.1109/CNS.2015.7346855.

[30] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, "Robust Smartphone App Identification via Encrypted Network Traffic Analysis", *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 1, pp. 63–78, 2018. DOI: 10.1109/TIFS.2017.2737970.

[31] M. Conti, Q. Q. Li, A. Maragno, and R. Spolaor, "The Dark Side(-Channel) of Mobile Devices: A Survey on Network Traffic Analysis", *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2658–2713, 2018. DOI: 10.1109/COMST.2018.2843533.

[32] P. Wang, X. Chen, F. Ye, and Z. Sun, "A Survey of Techniques for Mobile Service Encrypted Traffic Classification Using Deep Learning", *IEEE Access*, vol. 7, pp. 54 024–54 033, 2019. DOI: 10.1109/ACCESS.2019.2912896.

[33] J. G. Andrews *et al.*, "What Will 5G Be?", *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, 2014, ISSN: 1558-0008. DOI: 10.1109/JSAC.2014.2328098.

[34] N. Bui *et al.*, "A Survey of Anticipatory Mobile Networking: Context-Based Classification, Prediction Methodologies, and Optimization Techniques", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1790–1821, 2017. DOI: 10.1109/COMST.2017.2694140.

[35] Z. M. Fadlullah *et al.*, "State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrow's Intelligent Network Traffic Control Systems", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2432–2455, 2017. DOI: 10.1109/COMST.2017.2707140.

[36] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Towards the Deployment of Machine Learning Solutions in Network Traffic Classification: A Systematic Survey", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1988–2014, 2019. DOI: 10.1109/COMST.2018.2883147.

[37] D. Gündüz *et al.*, "Machine Learning in the Air", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2184–2199, 2019. DOI: 10.1109/JSAC.2019.2933969.

[38] S. Rezaei and X. Liu, "Deep Learning for Encrypted Traffic Classification: An Overview", *IEEE Communications Magazine*, vol. 57, no. 5, pp. 76–81, 2019. DOI: 10.1109/MCOM.2019.1800819.

[39] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end Encrypted Traffic Classification with One-dimensional Convolution Neural Networks", in *Proc. of IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2017, pp. 43–48. DOI: 10.1109/ISI.2017.8004872.

[40] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep Packet: A Novel Approach for Encrypted Traffic Classification Using Deep Learning", *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020. DOI: 10.1007/s00500-019-04030-2.

[41] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile Encrypted Traffic Classification Using Deep Learning: Experimental Evaluation, Lessons Learned, and Challenges", *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 445–458, 2019. DOI: 10.1109/TNSM.2019.2899085.

[42] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep Learning Based Encrypted Network Traffic Classification in SDN Home Gateway", *IEEE Access*, vol. 6, pp. 55 380–55 391, 2018. DOI: 10.1109/ACCESS.2018.2872430.

[43] T. Subramanya, D. Harutyunyan, and R. Riggio, "Machine Learning-driven Service Function Chain Placement and Scaling in MEC-enabled 5G Networks", *Computer Networks*, vol. 166, p. 106 980, 2020. DOI: 10.1016/j.comnet.2019.106980.

[44] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide Cellular Traffic Prediction based on Densely Connected Convolutional Neural Networks", *IEEE Communications Letters*, vol. 22, no. 8, pp. 1656–1659, 2018. DOI: 10.1109/LCOMM.2018.2841832.

[45] J. Wang *et al.*, "Spatiotemporal Modeling and Prediction in Cellular Networks: A Big Data Enabled Deep Learning Approach", in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, 2017, pp. 1–9. DOI: 10.1109/INFOCOM.2017.8057090.

[46] J. Feng, X. Chen, R. Gao, M. Zeng, and Y. Li, "DeepTP: An End-to-End Neural Network for Mobile Cellular Traffic Prediction", *IEEE Network*, vol. 32, no. 6, pp. 108–115, 2018. DOI: 10.1109/MNET.2018.1800127.

[47] Y. Hua *et al.*, "Deep Learning with Long Short-Term Memory for Time Series Prediction", *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019. DOI: 10.1109/MCOM.2019.1800155.

[48] T. Dlamini, Á. F. Gambín, D. Munaretto, and M. Rossi, "Online Resource Management in Energy Harvesting BS Sites Through Prediction and Soft-Scaling of Computing Resources", in *Proc. of 29th IEEE Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1820–1826. DOI: 10.1109/PIMRC.2018.8580912.

[49] Á. F. Gambín and M. Rossi, "A Sharing Framework for Energy and Computing Resources in Multi-Operator Mobile Networks", *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 1140–1152, 2019. DOI: 10.1109/TNSM.2019.2962725.

[50] L. Chen, D. Yang, D. Zhang, C. Wang, J. Li, *et al.*, "Deep Mobile Traffic Forecast and Complementary Base Station Clustering for C-RAN Optimization", *Journal of Network and Computer Applications*, vol. 121, pp. 59–69, 2018. DOI: 10.1016/j.jnca.2018.07.015.

[51] C.-W. Huang, C.-T. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting", in *Proc. of IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1–6. DOI: 10.1109/PIMRC.2017.8292737.

[52] C. Zhang and P. Patras, "Long-Term Mobile Traffic Forecasting Using Deep Spatio-Temporal Neural Networks", in *Proc. of 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2018, pp. 231–240. DOI: 10.1145/3209582.3209606.

[53] C. Zhang, M. Fiore, and P. Patras, "Multi-Service Mobile Traffic Forecasting via Convolutional Long Short-Term Memories", in *Proc. of IEEE International Symposium on Measurements Networking (M&N)*, 2019, pp. 1–6. DOI: 10.1109/IWMN.2019.8804984.

[54] Y. Zhang and Q. Yang, "A Survey on Multi-Task Learning", *IEEE Transactions on Knowledge and Data Engineering*, to be published. DOI: 10.1109/TKDE.2021.3070203.

[55] X. Song, H. Kanasugi, and R. Shibasaki, "DeepTransport: Prediction and Simulation of Human Mobility and Transportation Mode at a Citywide Level", in *Proc. of 25th International Joint Conference on Artificial Intelligence (IJCAI)*, ser. IJCAI'16, vol. 16, New York, New York, USA: AAAI Press, 2016, pp. 2618–2624, ISBN: 9781577357704. DOI: 10.5555/3060832.3060987.

[56] S. Rezaei and X. Liu, "Multitask Learning for Network Traffic Classification", in *Proc. of 29th IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–9. DOI: 10.1109/ICCCN49398.2020.9209652.

[57] H. Sun *et al.*, "Common Knowledge Based and One-Shot Learning Enabled Multi-Task Traffic Classification", *IEEE Access*, vol. 7, pp. 39 485–39 495, 2019, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2904039.

[58] D. Naboulsi, R. Stanica, and M. Fiore, "Classifying call profiles in large-scale mobile traffic datasets", in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, 2014, pp. 1806–1814. DOI: 10.1109/INFOCOM.2014.6848119.

[59] D. Naboulsi, M. Fiore, S. Ribot, and R. Stanica, "Large-Scale Mobile Traffic Analysis: A Survey", *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 124–161, 2016, ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2491361.

[60] A. Furno, M. Fiore, and R. Stanica, "Joint spatial and temporal classification of mobile traffic demands", in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, 2017, pp. 1–9. DOI: 10.1109/INFOCOM.2017.8057089.

[61] F. Meneghello, M. Rossi, and N. Bui, "Smartphone Identification via Passive Traffic Fingerprinting: A Sequence-to-Sequence Learning Approach", *IEEE Network*, vol. 34, no. 2, pp. 112–120, 2020. DOI: 10.1109/MNET.001.1900101.

[62] H. D. Trinh, A. F. Gambin, L. Giupponi, M. Rossi, and P. Dini, "Mobile Traffic Classification through Physical Control Channel Fingerprinting: a Deep Learning Approach", *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1946–1961, 2021. DOI: 10.1109/TNSM.2020.3028197.

[63] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile Traffic Prediction from Raw Data Using LSTM Networks", in *Proc. of IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018, pp. 1827–1832. DOI: 10.1109/PIMRC.2018.8581000.

[64] A. Azari, P. Papapetrou, S. Denic, and G. Peters, "User Traffic Prediction for Proactive Resource Management: Learning-Powered Approaches", in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9014115.

[65] B. Ma, W. Guo, and J. Zhang, "A Survey of Online Data-Driven Proactive 5G Network Optimisation Using Machine Learning", *IEEE Access*, vol. 8, pp. 35 606–35 637, 2020, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2975004.

[66] Y. Yue *et al.*, "Resource Optimization and Delay Guarantee Virtual Network Function Placement for Mapping SFC Requests in Cloud Networks", *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1508–1523, 2021. DOI: 10.1109/TNSM.2021.3058656.

[67] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing", *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4188–4200, 2019. DOI: 10.1109/JIOT.2018.2875246.

[68] M. Berno, J. J. Alcaraz, and M. Rossi, "On the Allocation of Computing Tasks under QoS Constraints in Hierarchical MEC Architectures", in *Proc. of 4th IEEE International Conference on Fog and Mobile Edge Computing (FMEC)*, 2019, pp. 37–44. DOI: 10.1109/FMEC.2019.8795345.

[69] S. Sardellitti, M. Merluzzi, and S. Barbarossa, "Optimal Association of Mobile Users to Multi-Access Edge Computing Resources", in *Proc. of IEEE International Conference on Communications Workshops (ICC Workshops)*, 2018, pp. 1–6. DOI: 10.1109/ICCW.2018.8403594.

[70] F. Zhao, Y. Chen, Y. Zhang, Z. Liu, and X. Chen, "Dynamic Offloading and Resource Scheduling for Mobile Edge Computing With Energy Harvesting Devices", *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2154–2165, 2021. DOI: 10.1109/TNSM.2021.3069993.

[71] L. Ferdouse, A. Anpalagan, and S. Erkucuk, "Joint Communication and Computing Resource Allocation in 5G Cloud Radio Access Networks", *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 9122–9135, 2019. DOI: 10.1109/TVT.2019.2927904.

[72] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency and Mobility-Aware Service Function Chain Placement in 5G Networks", *IEEE Transactions on Mobile Computing*, to be published. DOI: 10.1109/TMC.2020.3028216.

[73] K. Guo, R. Gao, W. Xia, and T. Q. S. Quek, "Online Learning based Computation Offloading in MEC Systems with Communication and Computation Dynamics", *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 1147–1162, 2021. DOI: 10.1109/TCOMM.2020.3038875.

[74] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. S. Monroy, "Multi-Objective Computation Sharing in Energy and Delay Constrained Mobile Edge Computing Environments", *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 2992–3005, 2021. DOI: 10.1109/TMC.2020.2994232.

[75] H. Peng, Q. Ye, and X. S. Shen, "SDN-Based Resource Management for Autonomous Vehicular Networks: A Multi-Access Edge Computing Approach", *IEEE Wireless Communications*, vol. 26, no. 4, pp. 156–162, 2019. DOI: 10.1109/MWC.2019.1800371.

[76] S. Deng *et al.*, "Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence", *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7457–7469, 2020. DOI: 10.1109/JIOT.2020.2984887.

[77] P. Roy *et al.*, "User mobility and Quality-of-Experience aware placement of Virtual Network Functions in 5G", *Computer Communications*, vol. 150, pp. 367–377, 2020. DOI: 10.1016/j.comcom.2019.12.005.

[78] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "LORM: Learning to Optimize for Resource Management in Wireless Networks With Few Training Samples", *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 665–679, 2020, ISSN: 1558-2248. DOI: 10.1109/TWC.2019.2947591.

[79] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep Reinforcement Learning for Offloading and Resource Allocation in Vehicle Edge Computing and Networks", *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 158–11 168, 2019. DOI: 10.1109/TVT.2019.2935450.

[80] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach", *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1529–1541, 2021. DOI: 10.1109/TETC.2019.2902661.

[81] G. Wang, F. Xu, and C. Zhao, "Multi-Access Edge Computing Based Vehicular Network: Joint Task Scheduling and Resource Allocation Strategy", in *Proc. of IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6. DOI: 10.1109/ICCWorkshops49005.2020.9145277.

[82] X. Wang *et al.*, "Convergence of Edge Computing and Deep Learning: A Comprehensive Survey", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020. DOI: 10.1109/COMST.2020.2970550.

[83] W. Zhan *et al.*, "Mobility-Aware Multi-User Offloading Optimization for Mobile Edge Computing", *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3341–3356, 2020. DOI: 10.1109/OJCOMS.2020.3008485.

[84] X. Yu, M. Guan, M. Liao, and X. Fan, "Pre-Migration of Vehicle to Network Services Based on Priority in Mobile Edge Computing", *IEEE Access*, vol. 7, pp. 3722–3730, 2019. DOI: 10.1109/ACCESS.2018.2888478.

[85] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic Resource Allocation Exploiting Mobility Prediction in Mobile Edge Computing", in *Proc. of 27th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2016, pp. 1–6. DOI: 10.1109/PIMRC.2016.7794955.

[86] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing", *IEEE Access*, vol. 6, pp. 23 511–23 528, 2018. DOI: 10.1109/ACCESS.2018.2828102.

[87] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine Learning Meets Computation and Communication Control in Evolving Edge and Cloud: Challenges and Future Perspective", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 38–67, 2020, ISSN: 2373-745X. DOI: 10.1109/COMST.2019.2943405.

[88] C. Nguyen, C. Klein, and E. Elmroth, "Multivariate LSTM-based Location-aware Workload Prediction for Edge Data Centers", in *Proc. of 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2019, pp. 341–350. DOI: 10.1109/CCGRID.2019.00048.

[89] W.-C. Chien *et al.*, "Multiple Contents Offloading Mechanism in AI-enabled Opportunistic Networks", *Computer Communications*, vol. 155, pp. 93–103, 2020. DOI: 10.1016/j.comcom.2020.02.084.

[90] C.-L. Wu, T.-C. Chiu, C.-Y. Wang, and A.-C. Pang, "Mobility-Aware Deep Reinforcement Learning with Glimpse Mobility Prediction in Edge Computing", in *Proc. of IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7. DOI: 10.1109/ICC40277.2020.9149185.

[91] H. Ma, Z. Zhou, and X. Chen, "Leveraging the Power of Prediction: Predictive Service Placement for Latency-Sensitive Mobile Edge Computing", *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6454–6468, 2020. DOI: 10.1109/TWC.2020.3003459.

[92] Y. Chen, C. Long, G. Cong, and C. Li, "Context-aware deep model for joint mobility and time prediction", in *Proc. of 13th ACM International Conference on Web Search and Data Mining (WSDM)*, 2020, pp. 106–114. DOI: 10.1145/3336191.3371837.

[93] A. Dalgkitsis, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Data Driven Service Orchestration for Vehicular Networks", *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4100–4109, 2021. DOI: 10.1109/TITS.2020.3011264.

[94] I. Labriji *et al.*, "Mobility Aware and Dynamic Migration of MEC Services for the Internet of Vehicles", *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 570–584, 2021. DOI: 10.1109/TNSM.2021.3052808.

[95] M. Polese *et al.*, "Machine Learning at the Edge: A Data-Driven Architecture with Applications to 5G Cellular Networks", *IEEE Transactions on Mobile Computing*, to be published. DOI: 10.1109/TMC.2020.2999852.

[96] O. Narmanlioglu, E. Zeydan, M. Kandemir, and T. Kranda, "Prediction of Active UE Number with Bayesian Neural Networks for Self-Organizing LTE Networks", in *Proc. of 8th IEEE International Conference on the Network of the Future (NOF)*, 2017, pp. 73–78. DOI: 10.1109/NOF.2017.8251223.

[97] L. Chen *et al.*, "Data-Driven C-RAN Optimization Exploiting Traffic and Mobility Dynamics of Mobile Users", *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1773–1788, 2021. DOI: 10.1109/TMC.2020.2971470.

[98] F. B. Mismar, B. L. Evans, and A. Alkhateeb, "Deep Reinforcement Learning for 5G Networks: Joint Beamforming, Power Control, and Interference Coordination", *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1581–1592, 2019. DOI: 10.1109/TCOMM.2019.2961332.

[99] F. Jiang, K. Wang, L. Dong, C. Pan, and K. Yang, "Stacked Autoencoder-Based Deep Reinforcement Learning for Online Resource Scheduling in Large-Scale MEC Networks", *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9278–9290, 2020. DOI: 10.1109/JIOT.2020.2988457.

[100] Y. Chen, W. Liu, Z. Niu, Z. Feng, Q. Hu, and T. Jiang, "Pervasive intelligent endogenous 6G wireless systems: Prospects, theories and key technologies", *Digital Communications and Networks*, vol. 6, no. 3, pp. 312–320, 2020. DOI: 10.1016/j.dcan.2020.07.002.

[101] S. Zhang and D. Zhu, "Towards artificial intelligence enabled 6G: State of the art, challenges, and opportunities", *Computer Networks*, p. 107 556, 2020. DOI: 10.1016/j.comnet.2020.107556.

[102] C. Gutterman, E. Grinshpun, S. Sharma, and G. Zussman, "RAN Resource Usage Prediction for a 5G Slice Broker", in *Proc. of 20th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2019, pp. 231–240. DOI: 10.1145/3323679.3326521.

[103] S. Bakri, P. A. Frangoudis, A. Ksentini, and M. Bouaziz, "Data-Driven RAN Slicing Mechanisms for 5G and Beyond", *IEEE Transactions on Network and Service Management*, pp. 1–1, 2021, to be published. DOI: 10.1109/TNSM.2021.3098193.

[104] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez, "DeepCog: Optimizing Resource Provisioning in Network Slicing With AI-Based Capacity Forecasting", *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 361–376, 2019. DOI: 10.1109/JSAC.2019.2959245.

[105] M. R. Raza, C. Natalino, P. Öhlen, L. Wosinska, and P. Monti, "Reinforcement Learning for Slicing in a 5G Flexible RAN", *Journal of Lightwave Technology*, vol. 37, no. 20, pp. 5161–5169, 2019. DOI: 10.1109/JLT.2019.2924345.

[106] H. D. R. Albonda and J. Pérez-Romero, "An Efficient RAN Slicing Strategy for a Heterogeneous Network With eMBB and V2X Services", *IEEE Access*, vol. 7, pp. 44 771–44 782, 2019. DOI: 10.1109/ACCESS.2019.2908306.

[107] R. Li, C. Wang, Z. Zhao, R. Guo, and H. Zhang, "The LSTM-Based Advantage Actor-Critic Learning for Resource Management in Network Slicing With User Mobility", *IEEE Communications Letters*, vol. 24, no. 9, pp. 2005–2009, 2020. DOI: 10.1109/LCOMM.2020.3001227.

[108] B. Khodapanah, A. Awada, I. Viering, A. N. Barreto, M. Simsek, and G. Fettweis, "Slice Management in Radio Access Network via Deep Reinforcement Learning", in *Proc. of 91st IEEE Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–6. DOI: 10.1109/VTC2020-Spring48590.2020.9128982.

[109] X. Chen, Z. Zhao, C. Wu, M. Bennis, H. Liu, Y. Ji, and H. Zhang, "Multi-Tenant Cross-Slice Resource Orchestration: A Deep Reinforcement Learning Approach", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2377–2392, 2019. DOI: 10.1109/JSAC.2019.2933893.

[110] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-Powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing", *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 334–349, 2019. DOI: 10.1109/JSAC.2019.2959185.

[111] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, and H. Mineno, "Flexible Resource Block Allocation to Multiple Slices for Radio Access Network Slicing Using Deep Reinforcement Learning", *IEEE Access*, vol. 8, pp. 68 183–68 198, 2020. DOI: 10.1109/ACCESS.2020.2986050.

[112] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. Shen, "Joint RAN Slicing and Computation Offloading for Autonomous Vehicular Networks: A Learning-Assisted Hierarchical Approach", *IEEE Open Journal of Vehicular Technology*, 2021. DOI: 10.1109/OJVT.2021.3089083.

[113] Z. Wang, Y. Wei, F. R. Yu, and Z. Han, "Utility Optimization for Resource Allocation in Edge Network Slicing Using DRL", in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9322481.

[114] W. Wu, N. Chen, C. Zhou, M. Li, X. Shen, W. Zhuang, and X. Li, "Dynamic RAN Slicing for Service-Oriented Vehicular Networks via Constrained Learning", *IEEE Journal on Selected Areas in Communications*, 2020. DOI: 10.1109/JSAC.2020.3041405.

[115] J. Mei, X. Wang, K. Zheng, G. Boudreau, A. B. Sediq, and H. Abou-zeid, "Intelligent Radio Access Network Slicing for Service Provisioning in 6G: A Hierarchical Deep Reinforcement Learning Approach", vol. 69, no. 9, pp. 6063–6078, 2021. DOI: 10.1109/TCOMM.2021.3090423.

[116] W. Guan, H. Zhang, and V. C. Leung, "Slice Reconfiguration Based on Demand Prediction with Dueling Deep Reinforcement Learning", in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, 2020, pp. 1–6. DOI: 10.1109/GLOBECOM42002.2020.9322180.

[117] R. Li, Z. Zhao, Q. Sun, I Chih-Lin, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep Reinforcement Learning for Resource Management in Network Slicing", *IEEE Access*, vol. 6, pp. 74 429–74 441, 2018. DOI: 10.1109/ACCESS.2018.2881964.

[118] F. Mason, G. Nencioni, and A. Zanella, "Using distributed reinforcement learning for resource orchestration in a network slicing scenario", 2021. [Online]. Available: https://arxiv.org/abs/2105.07946.

[119] X. Zhang, B. Li, J. Peng, X. Pan, and 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*, vol. 39, no. 1, pp. 4–16, 2021. DOI: 10.1109/JLT.2020.3023693.

[120] N. Bui and J. Widmer, "OWL: A Reliable Online Watcher for LTE Control Channel Measurements", in *Proc. of ACM Workshop on All Things Cellular: Operations, Applications and Challenges (ATC)*, New York, NY, USA, 2016, pp. 25–30. DOI: 10.1145/2980055.2980057.

[121] H. D. Trinh, N. Bui, J. Widmer, L. Giupponi, and P. Dini, "Analysis and Modeling of Mobile Traffic Using Real Traces", in *Proc. of IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1–6. DOI: 10.1109/PIMRC.2017.8292200.

[122] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G", *IEEE Communications Magazine*, vol. 52, no. 2, pp. 74–80, 2014. DOI: 10.1109/MCOM.2014.6736746.

[123] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture", *IEEE Communications Magazine*, vol. 52, no. 11, pp. 65–75, 2014. DOI: 10.1109/MCOM.2014.6957145.

[124] R. Caruana, "Multitask learning", *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997. DOI: 10.1023/A:1007379606734.

[125] M. McClellan, C. Cervelló-Pastor, and S. Sallent, "Deep Learning at the Mobile Edge: Opportunities for 5G Networks", *Applied Sciences*, vol. 10, no. 14, 2020. DOI: 10.3390/app10144735. [Online]. Available: https://www.mdpi.com/2076-3417/10/14/4735.

[126] K. Bian *et al.*, "Learning at the Edge: Smart Content Delivery in Real World Mobile Social Networks", *IEEE Network*, vol. 33, no. 4, pp. 208–215, 2019. DOI: 10.1109/MNET.2019.1800294.

[127] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical layer procedures", 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 36.213, 2016.

[128] D. Steinley, "K-means clustering: A half-century synthesis", *British Journal of Mathematical and Statistical Psychology*, vol. 59, no. 1, pp. 1–34, 2006.

[129] A. J. Izenman, "Modern multivariate statistical techniques", *Regression, classification and manifold learning*, 2008.

[130] G. Menardi, "Density-based silhouette diagnostics for clustering methods", *Statistics and Computing*, vol. 21, no. 3, pp. 295–308, 2011. DOI: 10.1007/s11222-010-9169-0.

[131] J. Garcia and A. Brunstrom, "Clustering-based separation of media transfers in DPI-classified cellular video and VoIP traffic", in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6. DOI: 10.1109/WCNC.2018.8377027.

[132] Ericsson, *Ericsson Mobility Report*, 2019. [Online]. Available: https://www.ericsson.com/en/mobility-report/reports/november-2019.

[133] Cisco, "Cisco Annual Internet Report (2018–2023)", *White Paper*, 2020.

[134] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, p. 436, 2015.

[135] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, http://www.deeplearningbook.org.

[136] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks", in *Proc. of ACM Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 3104–3112.

[137] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[138] S. Hashem and B. Schmeiser, "Improving model accuracy using optimal linear combinations of trained neural networks", *IEEE Transactions on neural networks*, vol. 6, no. 3, pp. 792–794, 1995. DOI: 10.1109/72.377990.

[139] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[140] H. Feng and Y. Shu, "Study on network traffic prediction techniques", in *Proc. of International Conference on Wireless Communications, Networking and Mobile Computing*, vol. 2, 2005, pp. 1041–1044. DOI: 10.1109/WCNM.2005.1544219.

[141] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[142] J. G. Carney and P. Cunningham, "The epoch interpretation of learning", *IEEE Transaction on Neural Networks*, vol. 8, pp. 111–116, 1998.

[143] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *Proc. of 3rd International Conference on Learning Representations (ICLR)*, 2015, pp. 1–15.

[144] Y. Liao, S. Kodagoda, Y. Wang, L. Shi, and Y. Liu, "Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks", in *Proc. of IEEE international Conference on Robotics and Automation (ICRA)*, 2016, pp. 2318–2325.

[145] R. Cipolla, Y. Gal, and A. Kendall, "Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics", in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7482–7491. DOI: 10.1109/CVPR.2018.00781.

[146] S. Xingjian *et al.*, "Convolutional LSTM network: A Machine Learning Approach for Precipitation Nowcasting", in *Proc. of ACM Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 802–810.

[147] D. P. Bertsekas, *Dynamic programming and optimal control*. Belmont, MA, USA: Athena Scientific, 2005, vol. 1.

[148] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A Survey on Low Latency Towards 5G: RAN, Core Network and Caching Solutions", *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018. DOI: 10.1109/COMST.2018.2841349.

[149] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg, "Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach", *Future Generation Computer Systems*, vol. 78, pp. 641–658, 2018, ISSN: 0167-739X. DOI: 10.1016/j.future.2017.02.014.

[150] M. Jung *et al.*, "Driving into the Memory Wall: the Role of Memory for Advanced Driver Assistance Systems and Autonomous Driving", in *Proc. of ACM International Symposium on Memory Systems (MEMSYS)*, 2018, pp. 377–386. DOI: 10.1145/3240302.3240322.

[151] R. Albert, A. Patney, D. Luebke, and J. Kim, "Latency requirements for foveated rendering in virtual reality", *ACM Transactions on Applied Perception (TAP)*, vol. 14, no. 4, pp. 1–13, 2017. DOI: 10.1145/3127589.

[152] 3GPP, "5G; Management and orchestration; Concepts, use cases and requirements", 3rd Generation Partnership Project, Tech. Specification (TS) 28.530, 2019, V15.2.0.

[153] ETSI, "Multi-access Edge Computing (MEC); Framework and Reference Architecture", European Telecommunications Standards Institute, Group Specification (GS) MEC 003, 2020, V2.2.1.

[154] A. H. Mousa, N. T. Mohammed, and E. A. Mohammed, "EFCNT: An evaluation framework for computer's network topologies", *AIP Conference Proceedings*, vol. 2144, no. 1, p. 050 010, 2019. DOI: 10.1063/1.5123126.

[155] D. Kreutz *et al.*, "Software-Defined Networking: A Comprehensive Survey", *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014. DOI: 10.1109/JPROC.2014.2371999.

[156] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi, *CRAWDAD dataset roma/taxi (v. 2014-07-17)*, Downloaded from https://crawdad.org/roma/taxi/20140717/taxicabs, 2014. DOI: 10.15783/C7QC7M.

[157] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", in *Proc. of 32nd ACM International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[158] ITU-R, "Minimum requirements related to technical performance for IMT-2020 radio interface(s)", ITU Radiocommunication Sector, Report M.2410, 2017.

[159] W. Almughalles, R. Chai, J. Lin, and A. Zubair, "Task Execution Latency Minimization-based Joint Computation Offloading and Cell Selection for MEC-Enabled HetNets", in *Proc. of 28th IEEE Wireless and Optical Communications Conference (WOCC)*, 2019, pp. 1–5. DOI: 10.1109/WOCC.2019.8770582.

[160] J. Xiong, H. Guo, and J. Liu, "Task Offloading in UAV-Aided Edge Computing: Bit Allocation and Trajectory Optimization", *IEEE Communications Letters*, vol. 23, no. 3, pp. 538–541, 2019. DOI: 10.1109/LCOMM.2019.2891662.

[161] M. Bouet and V. Conan, "Mobile Edge Computing Resources Optimization: A Geo-clustering Approach", *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 787–796, 2018. DOI: 10.1109/TNSM.2018.2816263.

[162] 3GPP, "5G; Study on scenarios and requirements for next generation access technologies", 3rd Generation Partnership Project, Tech. Rep. (TR) 38.913, 2020, V16.0.0.

[163] T. Lian, Y. Zhou, X. Wang, N. Cheng, and N. Lu, "Predictive Task Migration Modeling in Software Defined Vehicular Networks", in *Proc. of 4th IEEE International Conference on Computer and Communication Systems (ICCCS)*, 2019, pp. 570–574. DOI: 10.1109/CCOMS.2019.8821707.

[164] K. Gilly, S. Filiposka, and S. Alcaraz, "Predictive Migration Performance in Vehicular Edge Computing Environments", *Applied Sciences*, vol. 11, no. 3, p. 944, 2021. DOI: 10.3390/app11030944.

[165] ETSI, "Multi-access Edge Computing (MEC); MEC 5G Integration", European Telecommunications Standards Institute, Group Report (GR) MEC 031, 2020, V2.1.1.

[166] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, *CRAWDAD dataset epfl/mobility (v. 2009-02-24)*, Downloaded from https://crawdad.org/epfl/mobility/20090224, 2009. DOI: 10.15783/C7J010.

[167] H. Mazouzi, K. Boussetta, and N. Achir, "Maximizing Mobiles Energy Saving Through Tasks Optimal Offloading Placement in two-tier Cloud: A Theoretical and an Experimental Study", *Computer Communications*, vol. 144, pp. 132–148, 2019. DOI: 10.1016/j.comcom.2019.05.017.

[168] N. Shimkin, *Learning in Complex Systems: Dynamic Programming – Finite Horizon*, 2011. [Online]. Available: https://webee.technion.ac.il/shimkin/LCS11/ch2_DP_finite.pdf.

[169] C. Cullinan, T. R. Frattesi, and C. Wyant, *Computing Performance Benchmarks among CPU, GPU, and FPGA*, MathWorks, 2013. [Online]. Available: https://m.wpi.edu/Pubs/E-project/Available/E-project-030212-123508/unrestricted/Benchmarking_Final.pdf.

[170] E. Buber and D. Banu, "Performance Analysis and CPU vs GPU Comparison for Deep Learning", in *Proc. of 6th IEEE International Conference on Control Engineering & Information Technology (CEIT)*, 2018, pp. 1–6. DOI: 10.1109/CEIT.2018.8751930.

[171] V. Mnih *et al.*, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, no. 7540, pp. 529–533, 2015. DOI: 10.1038/nature14236.

[172] M. Miozzo, N. Piovesan, and P. Dini, "Coordinated Load Control of Renewable Powered Small Base Stations through Layered Learning", *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 1, pp. 16–30, 2020, ISSN: 2473-2400. DOI: 10.1109/TGCN.2019.2938860.

[173] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, Methods, and Future Directions", *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020. DOI: 10.1109/MSP.2020.2975749.

[174] W. Y. B. Lim *et al.*, "Federated Learning in Mobile Edge Networks: A Comprehensive Survey", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020. DOI: 10.1109/COMST.2020.2986024.

[175] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications", *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020. DOI: 10.1109/ACCESS.2020.3013541.

[176] M. Miozzo, Z. Ali, L. Giupponi, and P. Dini, "Distributed and Multi-Task Learning at the Edge for Energy Efficient Radio Access Networks", *IEEE Access*, vol. 9, pp. 12 491–12 505, 2021. DOI: 10.1109/ACCESS.2021.3050841.

[177] S. Zhang, "An Overview of Network Slicing for 5G", *IEEE Wireless Communications*, vol. 26, no. 3, pp. 111–117, 2019. DOI: 10.1109/MWC.2019.1800234.

[178] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, "A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 196–248, 2020. DOI: 10.1109/COMST.2019.2933899.

[179] M. Maule, J. Vardakas, and C. Verikoukis, "5G RAN Slicing: Dynamic Single Tenant Radio Resource Orchestration for eMBB Traffic within a Multi-Slice Scenario", *IEEE Communications Magazine*, vol. 59, no. 3, pp. 110–116, 2021. DOI: 10.1109/MCOM.001.2000770.

[180] J. Mei, X. Wang, and K. Zheng, "An intelligent self-sustained RAN slicing framework for diverse service provisioning in 5G-beyond and 6G networks", *Intelligent and Converged Networks*, vol. 1, no. 3, pp. 281–294, 2020. DOI: 10.23919/ICN.2020.0019.

[181] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, and Y.-D. Lin, "Security and privacy for 6G: A survey on prospective technologies and challenges", *IEEE Communications Surveys & Tutorials*, 2021, to be published. DOI: 10.1109/COMST.2021.3108618.

[182] X. Zhang, W. Lu, B. Li, and 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.

[183] S. E. Elayoubi, S. B. Jemaa, Z. Altman, and A. Galindo-Serrano, "5G RAN Slicing for Verticals: Enablers and Challenges", *IEEE Communications Magazine*, vol. 57, no. 1, pp. 28–34, 2019. DOI: 10.1109/MCOM.2018.1701319.

[184] S. D'Oro, F. Restuccia, and T. Melodia, "Toward Operator-to-Waveform 5G Radio Access Network Slicing", *IEEE Communications Magazine*, vol. 58, no. 4, pp. 18–23, 2020. DOI: 10.1109/MCOM.001.1900316.

[185] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks", *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, 2016. DOI: 10.1109/MCOM.2016.7509393.

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

[187] 3GPP, "Service requirements for the 5G system; Stage 1 (Release 17)", 3rd Generation Partnership Project, Technical Specification (TS) 22.261, 2019, V17.1.0.

[188] 5GAA Automotive Association, "C-V2X Use Cases Volume II: Examples and Service Level Requirements)", *White Paper*, 2020.

[189] O. U. Akgul, I. Malanchini, and A. Capone, "Dynamic Resource Trading in Sliced Mobile Networks", *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 220–233, 2019. DOI: 10.1109/TNSM.2019.2893126.

[190] 3GPP, "Study on management and orchestration of network slicing for next generation network (Release 15)", 3rd Generation Partnership Project, Technical Report (TR) 22.261, 2018, V15.1.0.

[191] ——, "5G; NR; Physical layer procedures for data (Release 15)", 3rd Generation Partnership Project, Technical Specification (TS) 38.214, 2018, V15.3.0.

[192] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey", *Computer Communications*, 2021. DOI: 10.1016/j.comcom.2021.01.021.

[193] 3GPP, "5G; NR; Physical layer; General description (Release 15)", 3rd Generation Partnership Project, Technical Specification (TS) 38.201, 2017, V15.0.0.

[194] ——, "5G; NR; Medium Access Control (MAC) protocol specification (Release 15)", 3rd Generation Partnership Project, Technical Specification (TS) 38.214, 2020, V15.8.0.

[195] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 2018.

[196] Z. Dai and R. Heckel, "Channel normalization in convolutional neural network avoids vanishing gradients", 2019. [Online]. Available: `https://arxiv.org/abs/1907.09539`.

[197] 5G PPP, "5G PPP architecture working group: View on 5G architecture", *White Paper*, 2020, V3.0.

[198] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning", 2015. [Online]. Available: `https://arxiv.org/abs/1509.02971`.

[199] P. Henderson *et al.*, "Deep reinforcement learning that matters", in *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[200] D. Zha, K.-H. Lai, K. Zhou, and X. Hu, "Experience replay optimization", in *Proc. of 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*, International Joint Conferences on Artificial Intelligence Organization, Jul. 2019, pp. 4243–4249. DOI: `10.24963/ijcai.2019/589`.

[201] 3GPP, "Evaluation assumptions for Phase 1 NR MIMO system level calibration", 3rd Generation Partnership Project, TSG RAN WG1 Meeting R1-1701824, 2017.

[202] M. Shi, K. Yang, Z. Han, and D. Niyato, "Coverage Analysis of Integrated Sub-6GHz-mmWave Cellular Networks With Hotspots", *IEEE Transactions on Communications*, vol. 67, no. 11, pp. 8151–8164, 2019. DOI: `10.1109/TCOMM.2019.2939802`.