POLITECNICO DI TORINO

ICT for smart societies

Politecnico
di Torino

Master's Degree Thesis

A robust multi-task Representation Learning methodology for sustainable management of Radio Access Networks.

Supervisors

Prof. MICHELA MEO

Prof. DANIELA RENGA

MATTEO BOFFA

GIOACCHINI LUCA

Candidate

MOHAMED AMINE MBAREK

October 2023

Abstract

With advancements in communication technologies, antennas, and solar panel efficiency, high altitude platforms stations (HAPS) have emerged as promising solutions for aerial networks, particularly in radio access networks (RANs). These platforms have the potential to enhance service quality by providing additional capacity while minimizing grid energy consumption. Innovative allocation strategies have been under investigation, with a particular emphasis on the application of deep reinforcement learning (DRL) techniques. By combining deep learning algorithms with reinforcement learning principles, agents can learn optimal decision-making policies through interactions with the environment. However, for DRL to achieve their full potential in optimizing HAP-based aerial networks, a preliminary step of powerful representation learning is necessary. Representation learning aims to extract meaningful patterns and abstractions from raw data, setting the stage for more advanced and efficient learning algorithms. Addressing this need, our research introduces SOM-TCAN, a hybrid model that combines the clustering capabilities of Self-Organizing Maps (SOM) with the forecasting strengths of Temporal Convolutional Attention Networks (TCAN). This integrated approach not only provides precise traffic forecasting but also paves the way for informed and adaptive decision-making in RAN-HAPs environments. Building upon this groundwork, the study further delves into the realm of deep reinforcement learning (DRL) to optimize Weighted Fair Queuing (WFQ) within HAPS. By integrating deep learning algorithms with reinforcement learning principles, the research proposes a shift in focus towards WFQ, highlighting the adaptability and potential of DRL in the landscape of HAPS-integrated aerial networks.

Acknowledgements

I extend my heartfelt appreciation to Matteo Boffa and Luca Gioacchini, who have been invaluable in providing guidance and support throughout my journey. Their insights, knowledge sharing, and relentless assistance significantly contributed to my understanding of the subject matter and its development.

I also express my gratitude to Prof. Michela Meo, for her unwavering support, continuous mentorship, and expert guidance. Her profound insights and supervision have played an instrumental role in shaping the trajectory of this research.

Additionally, I would like to extend my sincere thanks to Prof. Daniela Renga for her valuable suggestions and information.

Special thanks are due to my mother, who has been a pillar of unwavering support throughout this journey. Her encouragement, understanding, and countless sacrifices have been instrumental in my pursuit of knowledge.

I am also grateful to my friends for their encouragement and camaraderie that helped alleviate the challenges of this academic endeavor.

Table of Contents

List of Tables

List of Figures

Acronyms

AI

Artificial Intelligence

ANN

Artificial Neural Network

BS

Base Station

CAGR

Compound Annual Growth Rate

CDF

Cumulative Distribution Function

DRL

Deep Reinforcememt Learning

DTCR

Deep Temporal Clustering Representation

EB

Exa Bytes

EMF

Electromagnetic Field

FWA

Fixed Wireless Access

IOT

Internet of Things

ITU

International Telecommunication Union

LSTM

Long Short-Term Memory

ML

Machine Learning

MAE

Mean Absolute Error

MARE

Mean Absolute Relative Error

MIMO

Multiple Input Multiple Output

MLP

Multilayer Perceptron

MNO

Mobile Network Operator

MSE

Mean Square Error

NB

Narrow Band

RE

Renewable Energy

RL

Reinforcement Learning

RoD

Resource on Demand

RMSE

Root Mean Square Error

RNN

recurrent neural network

SOM

Self Organised Map

SMBS

Super Macro Base Stations

TCN

Temporal Convolutional Network

TCAN

Temporal Convolutional Attention Network

TDMA

Time Division Multiple Access

Chapter 1

Introduction

This initial chapter establishes a foundational understanding of contemporary issues and challenges associated with the Internet. It equips the reader with essential knowledge about the global evolution and significance of wireless networks.

Section 1.1 offers a viewpoint on the worldwide expansion of wireless connectivity and the surge in traffic requirements, presenting contemporary networks with numerous challenges that need to be addressed. Following this, Section 1.2 explores the features and applications of the promising 5th generation, emphasizing its key improvements over the LTE technology. Section 1.3 then outlines the primary challenges faced in 5G development. Section 1.4 offers a glimpse into the realm of 6G networks, setting the stage for the integration of aerial platforms within the conventional terrestrial framework. Concluding this chapter, Section 1.5 introduce HAPS as a super macro base stations (SMBS), emphasizing their transformative and enabling potential in future network architectures.

1.1 Meeting Future Demands for Connectivity

ICT and IoT-enabled applications are nowadays fundamental, and the number of connected things is experiencing unprecedented growth. The Author in [1] emphasized the evolution of the "Internet-of-Things" and its potential to revolutionize various aspects of our lives. The emergence of IoT devices spans a wide range of applications, from optimizing industrial processes and enhancing healthcare services to enabling smart homes and transforming the way we interact with our environment. The projected global economic value of these applications by 2035 is substantial, as illustrated in [2].

In its annual Internet Report, Cisco [3] emphasized that the CAGR of devices and connections globally stands at 10%. This growth rate outpaces that of Internet users and the overall population. Furthermore, the increasing proportion of interconnected devices capable of communicating and transmitting data, indicates the rising prominence of IoT. This emerging trend cements the IoT as one of the most rapidly expanding global phenomena. Supporting this observation, the cited report further notes a shift in terminal connections. Machine-to-Machine (M2M) connections, which are currently the fastest growing at a CAGR of 19%, now account for half of the total connections in the digital landscape. In addition, by inspecting the trends among various connected devices in Fig.1.1 We can notice M2M connections will be the fastest-growing device and connections category. Smartphones will grow the second fastest, at a 7% CAGR .

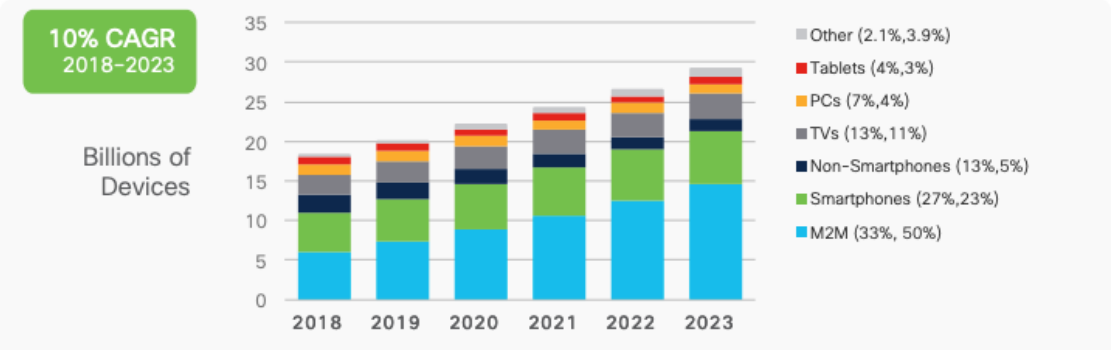


Figure 1.1: Global device and connection growth [3]

The 2021 Erricson Mobility Report highlights that by 2028, global mobile traffic, excluding that from Fixed Wireless Access (FWA), is anticipated to hit 325 EB monthly [4]. Factoring in FWA, the cumulative mobile network traffic could escalate to 453 EB monthly by the close of 2028. These forecasts take into account the phased integration of XR technologies like AR, VR, and MR, primarily towards the forecast's tail end. However, a more rapid adoption of these technologies could lead to data traffic volumes exceeding the current projections. Which could lead to an average usage per smartphone up to 41 Gb per month.

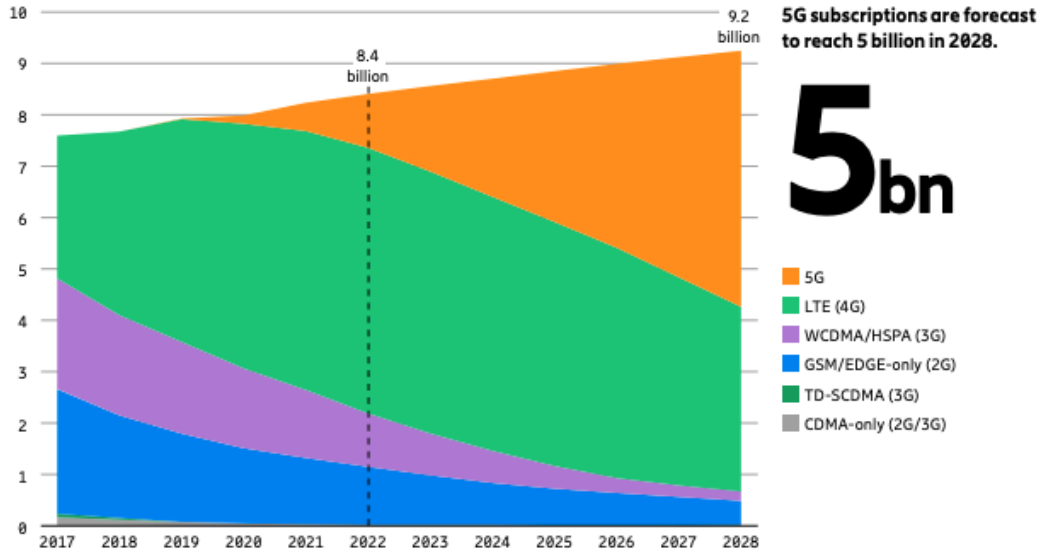


Figure 1.2: Mobile subscription by technology [4]

Furthermore, as 5th-generation wireless systems continue to evolve and proliferate, there's a consistent shift of users transitioning from 4G to 5G-enabled smartphones. In 2022, 5G mobile subscriptions breached the 1 billion mark. With each smartphone averaging 15 GB of data usage monthly, 5G subscriptions are projected to dominate the market. The pace of this growth is anticipated to be so brisk that by 2028, total 5G connections might touch a staggering 5 billion [4] as shown in Fig.1.2.

1.2 5G Networks: Evolution, Advantages, and Key Technologies

The evolution towards the fifth-generation (5G) mobile networks commenced as early as 2019. This shift was driven by the inherent limitations of 4G, such as its high power consumption, cost inefficiencies, and challenges in supporting a vast number of concurrent connections. 5G is not just an incremental upgrade; it's designed to harness the vast potential of the IoT and serve as a catalyst for the development of smart cities. With the digital landscape evolving rapidly, there's an anticipated surge in global mobile data traffic by the end of 2024. This underscores the need for a robust network infrastructure capable of handling such massive data

volumes. This network was designed with eight primary objectives, as outlined by the International Mobile Telecommunications-2020 (IMT-2020) requirements issued by ITU and depicted in Fig.1.3.

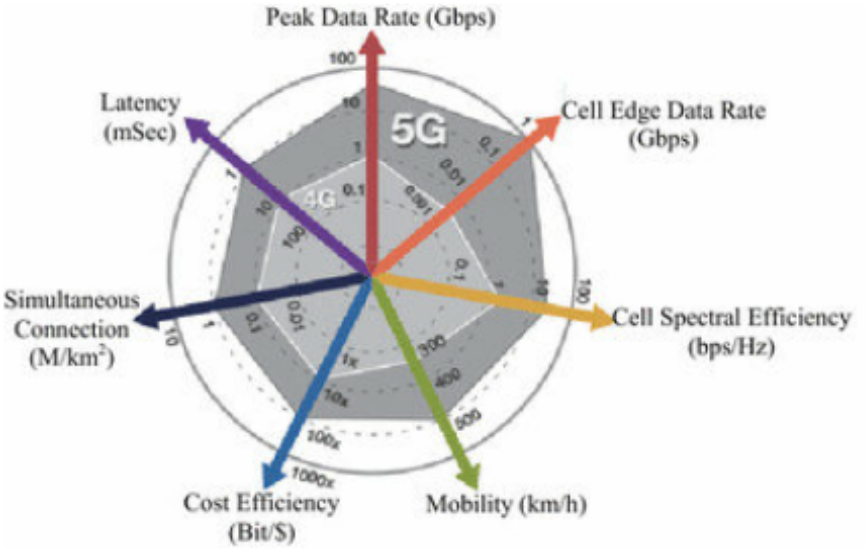


Figure 1.3: KPIs in evolution from 4G to 5G [5]

Specifically, it aims to surpass 4G in terms of latency, targeting under 1ms as a standard. Additionally, it seeks to improve spectrum and energy efficiency, boost mobility, and achieve unparalleled data transfer speeds. One of the defining characteristics of 5G is its operation across an expansive frequency spectrum. This ranges from the low bands, which are below 1 GHz, to the mid-bands that operate below 6 GHz, and extends to the high bands that span between 30Ghz and 300Ghz [6]. Each of these bands has its unique advantages. For instance, the low-frequency waves can travel long distances and penetrate obstacles, making them ideal for widespread coverage. The Third Generation Partnership Project (3GPP) has played a pivotal role in defining these bands specifically for the 5G New Radio (5G-NR) air interface. The 5G-NR is designed to significantly enhance the performance, flexibility, scalability, and efficiency of mobile networks [7]. To ensure comprehensive coverage, high data rates, and minimal latency, it's imperative for 5G deployment to utilize both the lower and higher frequency bands. This also means that the emerging 5G-NR networks, especially those operating at lower frequencies, must harmoniously coexist with the existing 4G networks [8].

5G is defined in a set of standardised specifications that are agreed on by international bodies most notably 3GPP and the ITU. The ITU has defined criteria for IMT-2020 commonly regarded as 5G and selected a set of compatible technologies which will support the following use cases [7]:

- eMBB, or Enhanced Mobile Broadband, focuses on user-centric scenarios involving the access of multimedia content, services, and data. Looking ahead, this vision entails not only enhancing existing mobile broadband applications but also introducing new application areas and requirements to provide better performance and a more seamless user experience. It encompasses various usage scenarios, including scenarios with extensive coverage and those involving densely populated areas known as hotspots. In the context of extensive coverage, the aim is to ensure uninterrupted connectivity even during medium to high mobility scenarios, significantly elevating user data speeds compared to current rates.
- uRLLC, which stands for Ultra-Reliable Low Latency Communication, leverages the network for mission-critical applications demanding continuous and resilient data interchange. Notable examples encompass wireless control of industrial manufacturing or production processes, remote medical surgeries, the automation of distribution within smart grids, and ensuring transportation safety, among others. This utilization scenario imposes rigorous demands on essential capabilities such as data throughput, latency, and availability to guarantee the utmost reliability and performance.
- mMTC, or Massive Machine Type Communication, is characterized by a vast array of interconnected devices, typically engaged in transmitting relatively small volumes of data that are not time-sensitive. This technology is primarily geared towards facilitating connectivity for an extensive number of devices, such as sensors, which typically transmit and receive minimal data quantities. An mMTC network is intentionally designed to be tolerant of latency, efficient for transmitting or receiving small data blocks, and capable of operating on low bandwidth channels. The primary performance requirement for an mMTC network service is to support an exceptionally high connection density, reaching up to 1 million devices per square kilometer.

Finally, mMIMO (massive Multiple Input, Multiple Output) plays a pivotal role. MIMO, a principle integrated into numerous wireless standards, has been instrumental in enhancing the capacity and dependability of wireless systems. It was a significant advancement during the 4G era. The massive MIMO approach, however, takes this a step further. It facilitates the simultaneous transmission and

reception of a larger number of data streams over a single radio channel. 5G infrastructures incorporate mMIMO configurations, such as 8x8 array antennas. These antennas are designed to be more directive, which translates to superior reliability and heightened resistance to deliberate interference or jamming. Furthermore, mMIMO incorporates beamforming, a technique that focuses the signal directly on the intended devices' physical locations. This ensures adaptive signal transmission, as depicted in Fig.1.4. This adaptive approach not only optimizes signal strength but also minimizes interference, ensuring a seamless communication experience.

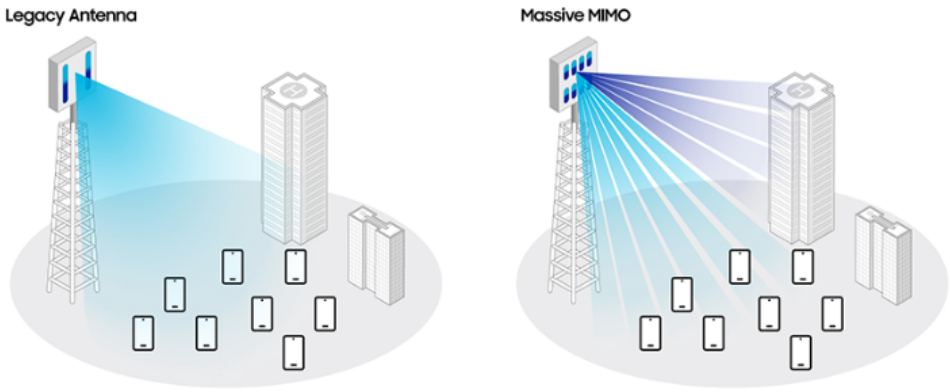


Figure 1.4: Coverage comparison between legacy antenna and Massive MIMO [9]

1.3 5G Challenges

The enhanced capacity and data speeds offered by 5G necessitate a broader spectrum and more advanced spectral technologies than those employed in 3G and 4G systems [10]. A portion of this additional spectrum comes from the mmWave band. However, the inherent properties of millimeter wave propagation present challenges. Specifically, these waves travel over much shorter distances compared to the Low and Mid bands. As a result, providing coverage for a particular area demands a significant number of base stations (BSs). This increases the intricacy of the network infrastructure and amplifies the need for radio equipment installations on street fixtures like poles and traffic lights.

The diverse requirements of various use cases mean that no single solution can

cater to all needs. Given the vast differences in system prerequisites, multiple solutions will be essential. The existing 5G network still falls short in fulfilling some stringent design criteria, including ultra-reliability, ultra-low latency, and ultra-secure networks [11]. This has spurred interest in exploring future physical layers and wireless systems. Beyond terrestrial networks, infrastructure leveraging satellite and aerial platforms will be crucial to meet coverage and capacity demands.

A significant challenge introduced by 5G is the substantial surge in energy consumption, leading to increased energy costs. This issue is a focal point for the networking research community, especially given the alarming annual growth rate of about 12% in the energy consumption of telecommunication networks [12]. This escalation is intrinsically linked to the proliferation of 5G BSs . Numerous studies have delved into methods to reduce BS energy consumption, exploring avenues like Renewable Energy (RE) integration or resource management tactics. Strategies such as Resource on Demand (RoD), which employs sleep modes for BSs during low traffic periods [13], and other sustainable approaches aim to bolster eco-friendliness while curbing expenses.

1.4 The Rising Demand for 6G Connectivity

5G will soon become insufficient in provided throughput, and more than everything, in being able to support a large number of connections. In fact, as already discussed in Section 1.1 connected devices are the category that will experience the highest overall CAGR in the next future, reaching 500 billion by 2030, about 59 times larger than the expected world population which is around 8.5 billion [14], but already being three times larger in 2025. In addition to that, it has to be considered that a typical 5G communication system can support at most 50 thousand IoTs and NB-IoT systems per cell, which is already not enough for many applications. The transformative evolution of wireless systems means that 6G will diverge significantly from its predecessors, marked by its diverse nature across various domains like network structures, radio technologies, computational capacities, storage solutions, and application diversity. Inspired by these trends, in [15] the authors attempt to conceptualize a road-map for 6G, which is based on the strategic plans of various standard bodies and is also projected based on the 5G status. The possible overview of the road-map is depicted in Fig.1.5.

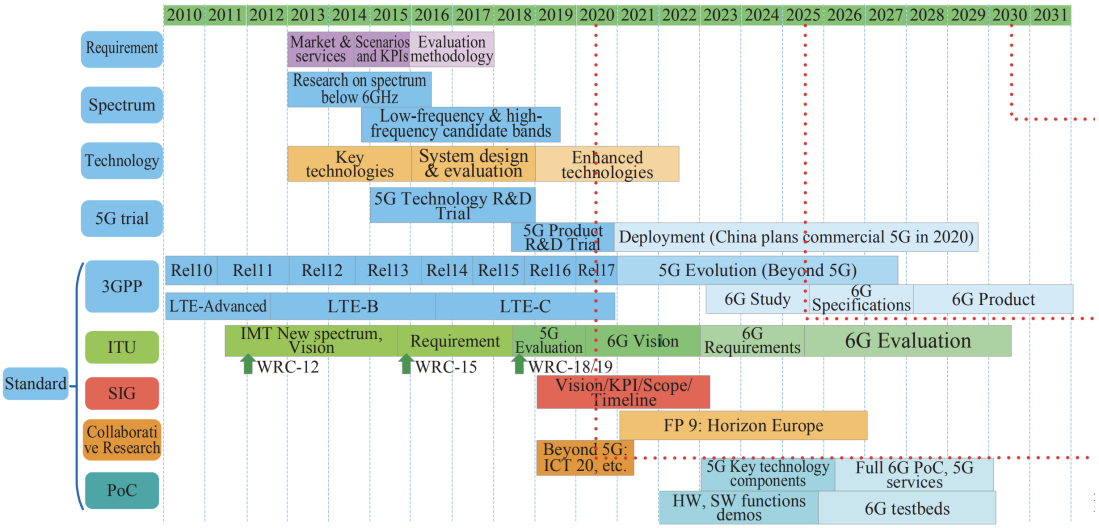


Figure 1.5: The roadmap of 6G [15]

1.4.1 The requirements of 6G

Enhancing network performance like greater capacity and data rates are essential, an example of Key Performance Indicators (KPIs) enhancement in the transition from 5G to 6G is shown in Fig.1.6.

However, we will also mention other that not purely performance oriented criteria. These were termed as societal requirements in [16].

- As 6G aims for higher capacity, it must further reduce energy per bit and per area. The 6G energy efficiency goal should align with capacity gains to maintain overall energy consumption. Comprehensive research is needed, with a holistic approach to energy efficiency across the entire ICT chain, encompassing service design and internet protocols

- Resilience, data security, and privacy have always been pivotal in mobile communication system designs. As telecommunications become integral to daily life and industries, their significance escalates. 6G must ensure heightened resilience, security, and privacy, incorporating advanced protection measures and addressing risks from new 6G features like AI and Thz communications. It should support local network autonomy during network isolation. In potentially untrusted multi-party settings, 6G should foster trustworthy networks, ensuring service and data security across diverse domains. Ultimately, 6G should enable

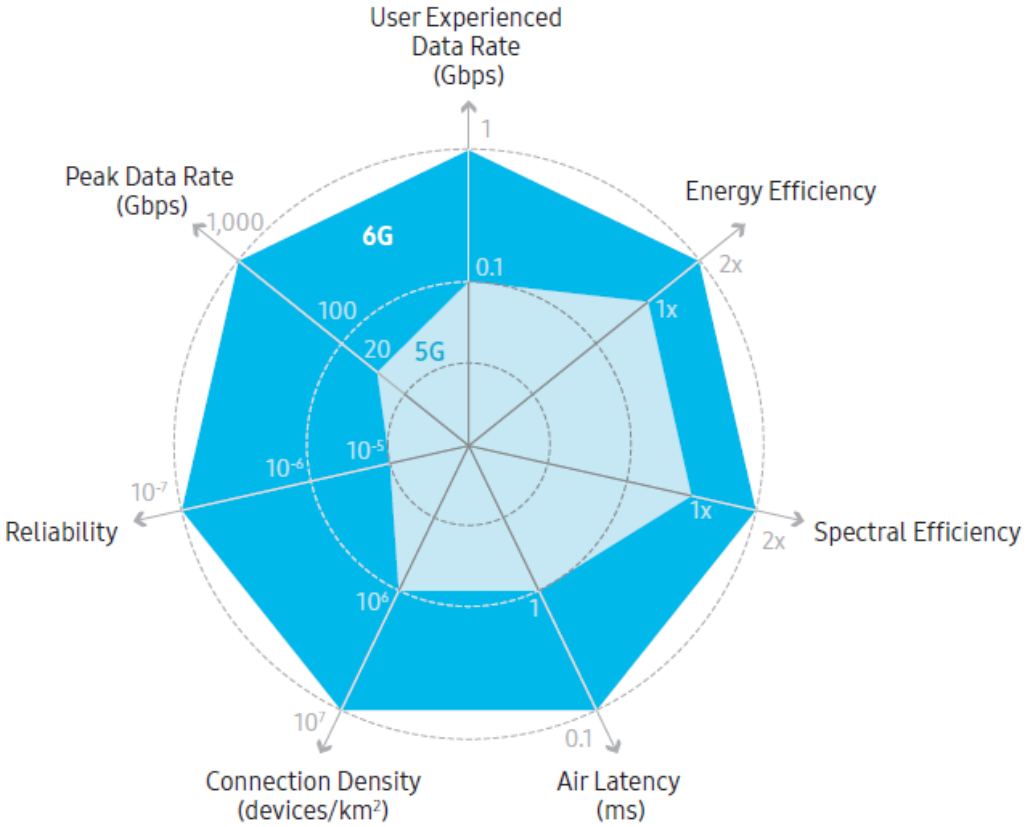


Figure 1.6: KPIs between 5G and 6G [14]

network operators to commit to service level agreements detailing quality, security, and availability standards.

- Expanding network capacity often involves using additional frequency bands, leading to increased Electro-Magnetic Fields (EMF) exposure. Operators must adhere to strict EMF regulations. While many countries base their EMF limits on guidelines from the International Commission on Non-Ionizing Radiation Protection (ICNIRP) [17], some places, like Italy, Poland, Brussels, and Paris, have even stricter limits. Current networks operate well below ICNIRP's recommendations. However, stricter limits may hinder future wireless network expansion using current techniques. Essentially, in areas with tight EMF restrictions, using new frequency bands might require reduced transmit power, impacting coverage. Thus, 6G design must be EMF-aware, enabling capacity growth without a significant rise in EMF exposure.

- New radio access solutions must offer superior performance to provide affordable new services. This includes enhanced data rates, improved latency, and extensive network coverage to maintain service quality. Thus, 6G should incorporate innovative mesh networking and integrated access/backhaul systems for cost-effective, dense network expansion [18]. While 5G uses bands below 6 GHz, a limited new spectrum is anticipated, 6G will need to coexist with previous generations. 6G will build on the 24-52GHz mmWave bands established by 5G and potentially extend to 100GHz or beyond. The sub-terahertz (sub-THz) or THz spectrum is seen as a key innovation for 6G. Its primary benefit is the vast spectral availability, enabling high capacity and throughputs of several $Gbit/s$. However, this high throughput is likely limited to short ranges and requires a direct line of sight between transmitters and receivers. Such bands are ideal for specific scenarios needing substantial data exchange in particular settings, like ultra-fast downloads or telepresence.

- 6G will integrate all physical devices into computation and manage physical systems. Through virtualized computing and storage, it will enhance application performance, reliability, and latency. With more connected objects, the ecosystem will need to evolve, demanding more tailored applications. Beyond standard APIs, fresh programming concepts and streamlined models will emerge. Network computing will drive the development of new software and hardware, with applications spanning the network edge, central cloud, and devices, making 6G a true epicenter of innovation.

1.4.2 Candidate Solutions for 6G

Like each new mobile communication system, 6G will introduce technological innovation to address the challenges associated with the requirements described earlier, and support new innovative usages. In this sub-section, we provide more technical details on key design principles and candidate enablers.

The integration of AI in network operations is anticipated to revolutionize service delivery, ensuring optimal resource utilization and quick responsiveness. For AI to function effectively, it requires specialized architectures and processes to accumulate ample relevant operational data. AI promises to enhance performance and efficiency across all facets of 6G network operations, including the intricate signal processing at the physical layer. Addressing challenges like interference and radio propagation issues demands sophisticated signal processing to maximize radio resource utilization. Current 6G research is delving into the potential of

Machine Learning (ML) to elevate the efficiency of this signal processing in both network infrastructure and end devices. Additionally, the efficiency of AI-focused hardware might enhance the energy conservation of 6G devices and equipment. Machine Learning, when applied to the physical layer, has the potential to intuitively understand radio channel attributes and determine the most effective communication methods, such as modulation patterns. More immediate prospects, like refining massive MIMO beamforming, seem more developed and are already being considered for 5G advancements. On the other hand the author in [15] proposed a potential architecture for 6G as shown in Fig.1.7, in which network intelligentization, subnetwork evolution, and intelligent radio are embraced.

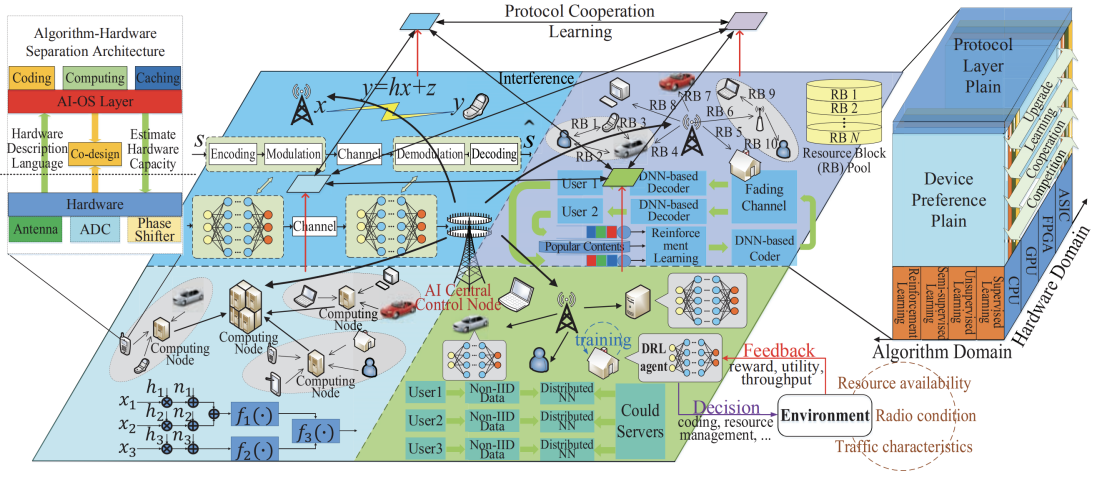


Figure 1.7: The architecture of 6G [15]

Building on the advancements in 6G networks, semantic communication (SC) is emerging as a transformative paradigm that promises to reshape the way we understand and implement wireless communication. Traditional methods, deeply rooted in Shannon's theory of information, have long chosen to sidestep the semantic intricacies of communication. However, as the 6G horizon comes into clearer view, there's a shift towards services that are not just about transmitting data but doing so with an understanding of content, user needs, and semantics [19].

The essence of SC lies in its focus on the message's inherent meaning during its transmission. This becomes especially pivotal for applications like intelligent transportation systems, where the real-time processing of diverse data types, from videos to radar outputs, is crucial. By integrating SC with state-of-the-art machine learning models, including deep learning and neural networks, there's potential to supercharge the efficiency of signal processing across both network infrastructure

and end-user devices [19]. Yet, as with all innovations, challenges arise. The introduction of 'semantic noise', stemming from written discrepancies, ambiguous expressions, or adversarial interventions in the semantic channel, poses a significant hurdle. To navigate these challenges, the development of robust deep learning-enabled semantic communication systems is underway. These systems, equipped with calibrated self-attention mechanisms and adversarial training, aim to fortify the communication process against such semantic disturbances [20].

Another Candidate enabler to provide wireless connectivity with reduced environmental impact and controlled EMF exposure are Reflective Intelligent Surfaces (RIS) . RIS introduces a novel node within the network, distinguished by its "passive" nature, meaning it doesn't produce any supplementary radio waves [21]. When exposed to an incoming radio wave, the RIS reflects it back. It's essential, from our perspective, that the network retains control over the RIS and its induced reflection. One particularly intriguing application for network operators is an RIS that directs the reflected wave purposefully. By manipulating these smart "reflectors", operators can mold wireless propagation, crafting "adaptable, smart, and eco-friendly wireless settings", as articulated in [22]. Studies have shown that such surfaces can bolster antenna beamforming, enhancing the data speed to the intended device while simultaneously reducing EMF exposure near the antenna [23]. RIS present promising solutions to enhance coverage and data speeds in both internal and external settings without the need for extra radio wave transmissions. Present-day challenges revolve around the RIS's reflective capabilities, energy efficiency, cost-effectiveness, RIS control, and the coexistence of RIS operators.

Furthermore, we delve into a unique category of devices known for their energy autonomy, termed Zero-Energy Devices (ZEDs) [24]. Similar to RIS, a ZED operates in a "passive" manner, meaning it doesn't produce supplementary radio waves. Instead, it communicates by reflecting or backscattering ambient waves. During this process, the ZED modulates the reflected signal to embed a message. This method eliminates the need for a power amplifier, making the device's energy consumption minimal. In fact, the energy requirement is so modest that ZEDs can harness power from renewable sources, such as ambient light through photovoltaic cells or even vibrations using piezoelectric materials. This energy-efficient design, combined with energy-harvesting capabilities, positions ZEDs as a sustainable solution in the realm of wireless communication. IoT devices, especially sensors, can benefit from being ZEDs as they often need to be deployed in hard-to-reach or remote areas where changing batteries or providing power is challenging. However, the potential applications of ZEDs, and the design of solutions remain at a research stage.

Finally, With the evolution in satellite engineering and launch methodologies, the expenses associated with deploying low-Earth orbit satellite clusters have been substantially reduced . Coupled with advancements in electronics and antenna technology, this has led to a notable increase in data transmission rates per satellite. In this context, High-Altitude Platforms Stations (HAPS), encompassing various stratospheric airships, are being explored as potential economical means to provide connectivity across expansive regions, catering to conventional devices like smartphones and sensors. Should their cost-effectiveness and performance be validated, satellites and HAPS could play a pivotal role in promoting digital accessibility. Their expansive coverage, combined with the capability to direct beams based on traffic demands, positions HAPS-integrated networks as key assets for enhancing 6G communication's robustness and uninterrupted service.

1.5 HAPS as a Super Macro Base Station

Due to its expansive coverage footprint, augmented capacity featuring multiple-MIMO technology, and the ability to provide integral functions such as data acquisition, computational processing, caching, and data handling, HAPS in its capacity as a SMBS emerges as a viable deployment option, especially well-suited for densely populated urban environments that align with the principles of smart city development. This strategic choice is driven by the pressing need to contend with escalating traffic demands within intricate urban settings, while simultaneously addressing the challenges associated with deploying conventional terrestrial BSs. The forthcoming generation of HAPS-SMBS is poised to equip itself with the requisite capabilities to tackle the mounting requirements for high capacity, minimal latency, and advanced computational capabilities, tailored to the unique demands of densely inhabited metropolitan regions. A representation of the use cases of HAPS-SMBS networks is depicted in Fig.1.8.

HAPS-SMBS offer the potential to complement terrestrial networks in various ways [26]:

- In upcoming urban landscapes, the surge in data traffic and user volume will overwhelm terrestrial BSs. To alleviate this, under-utilised BSs in less dense areas can serve these overloaded regions. This extended connectivity can be facilitated using RIS mounted on HAPS. By adjusting the RIS phases, signals from BSs can be directed to reach underserved users.
- As IoT experiences exponential expansion, it becomes evident that the existing

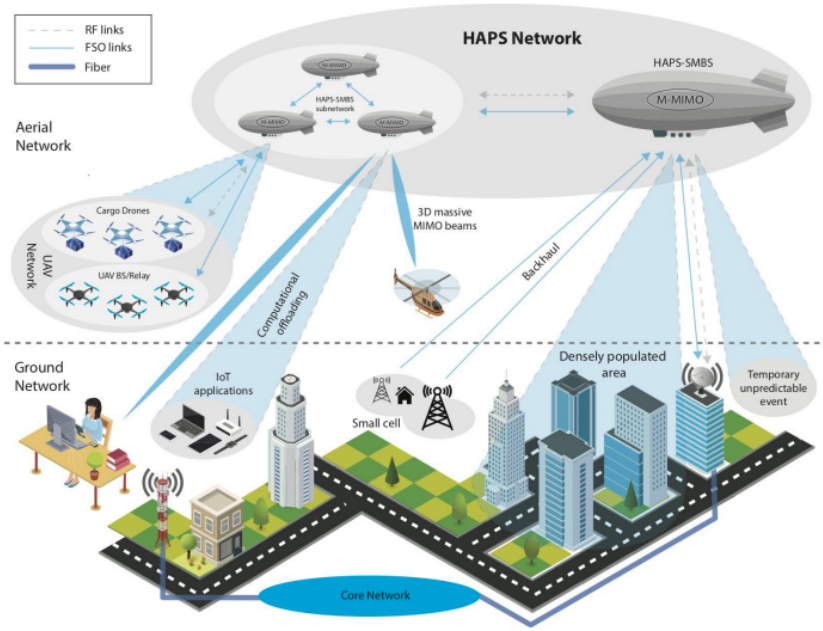


Figure 1.8: HAPS-SMBS networks [25]

infrastructure and traditional wireless access architecture design methods are ill-equipped to handle the escalating demand for wireless systems and services. In this context, the expansive reach of HAPS presents an optimal solution for furnishing extensive coverage to accommodate a multitude of IoT devices. Consequently, HAPS-SMBS emerge as an appealing solution to supplement terrestrial networks, offering the ability to gather data from IoT devices efficiently and establish dependable uplink connections with them.

- Supporting Terrestrial Highways and Users: Connected and autonomous vehicles (CAVs) generate significant deep learning computations for autonomous decision-making. Leveraging HAPS equipped with robust computing power an offload these AI tasks from CAVs. Beyond their computational strengths, HAPS offer advantages over traditional networks (TNs) in overseeing vehicle mobility. Their expansive coverage can notably decrease handover occurrences.

Chapter 2

Related Work

In this section, representation learning is explored, especially its application to time series. Methods that are similar to the proposed approach are highlighted, and a range of related problems within the domain is discussed, positioning the research within the broader context of existing solutions.

Section 2.1 introduces representation learning and outlines its primary objectives. Section 2.2 shifts focus to its application in time series, organizing existing methods into three distinct categories. Sections 2.3 to 2.5 offer a detailed examination of these categories, highlighting their specific characteristics.

2.1 Representation learning

Representation learning, also known as feature learning, stands as a cornerstone in the expansive landscape of machine learning. Its primary objective is the automated extraction of pertinent features or representations directly from raw data, a concept eloquently explained in [27]. This paradigm shift offers a stark contrast to traditional machine learning techniques, which often hinge on the expertise of domain specialists to meticulously craft features, a process that is not only time-consuming but also requires profound domain-specific expertise.

The allure of representation learning lies in its promise of automation. It alleviates the need for manual feature engineering, empowering models to autonomously go through data and focus on the most relevant features. This self-reliant approach to feature extraction has broader implications. For instance, the learned

representations can be seamlessly repurposed across several tasks, epitomizing the essence of transfer learning. In this context, features learned from one dataset can be leveraged to fortify the training of models on related dataset. Moreover, in the face of high-dimensional data, representation learning emerges as a beacon, offering tools to distill the data, making it more manageable and interpretable, as underscored by [28].

Several methods have been pioneered in the realm of representation learning. Autoencoders, for instance, are a class of neural networks tailored to reconstruct their input data by compressing it into a lower-dimensional latent space and subsequently decompressing it . The compressed representation, or the latent space, captures the most salient features of the data. Variations of autoencoders, such as Variational Autoencoders (VAEs), introduce a probabilistic approach to encoding and decoding, allowing for richer and more diverse representations [29]. On the other hand Word embeddings, like Word2Vec and GloVe, have revolutionized the field of natural language processing. These techniques generate dense vector representations of words based on their contextual semantics. Such embeddings capture syntactic and semantic relationships between words, enabling superior performance in various linguistic tasks [30].

Despite its successes, representation learning faces challenges. Interpretability remains elusive, as the features extracted, especially from deep models, are often hard to decipher. Scalability, especially with increasing data dimensions, is a concern. Moreover, while supervised scenarios have seen significant advancements, unsupervised and semi-supervised representation learning remain active areas of research, with much potential yet to be unlocked [31].

2.2 Time-Series Representation Learning

In the scope of this thesis, we are tackling the domain of time series data, with a specific emphasis on cellular stations traffic. Time series data, characterized by its sequential nature, presents unique challenges and opportunities for representation learning. The learned latent representations can capture potentially valuable information within time series and reveal the underlying mechanisms of the corresponding systems or phenomena. Moreover, these robust representations play a crucial role in various subsequent tasks, such as time series forecasting [32] and clustering [33].

As presented in Fig.2.1, we classify the existing representation learning methods

for time series into 3 categories : hybrid techniques , Supervised learning techniques and unsupervised learning techniques.

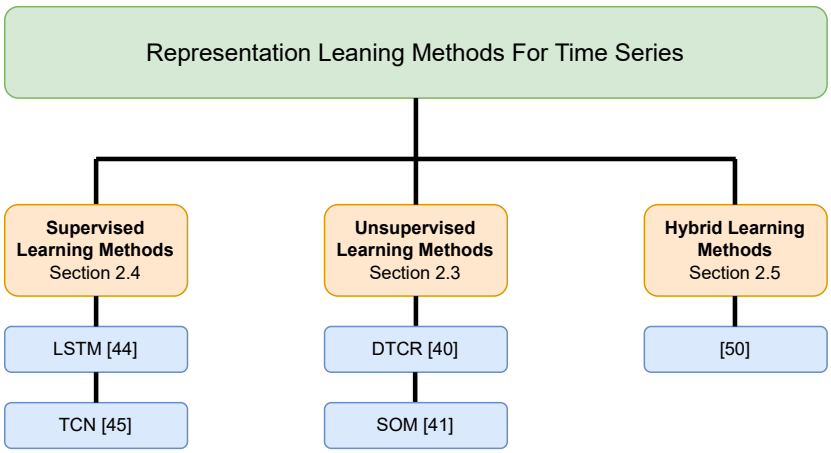


Figure 2.1: Taxonomy of time series representation learning methods

Hybrid Techniques combine elements of both supervised and unsupervised learning to harness the strengths of each. By integrating labeled and unlabeled data, or by combining different learning paradigms, hybrid techniques aim to achieve enhanced performance and more versatile representations suitable for a variety of tasks. In this section we will unravel the work in [34].

Unsupervised Learning operates without the need for labeled data, unsupervised techniques focus on extracting patterns and structures directly from the data itself. Methods such as autoencoders, which learn to reconstruct input sequences, or clustering algorithms that group similar sequences, fall into this category. The goal here is to uncover latent structures and relationships within the time series without any predefined labels. Consequently, unsupervised learning approaches are devised to tackle this challenge by creating various pretext tasks that generate self-generated labels without relying on human supervision. Unsupervised learning for time series representation learning could be divided into three other categories [35]: deep clustering, reconstruction-based, and self-supervised learning methods. However, we will present a method that is associated to Deep clustering a method which typically combine traditional clustering techniques with deep learning neural networks, utilizing clustering results as labels for representation learning

while updating clustering results based on learned representations. Furthermore, we will introduce an artificial neural network (ANN) known as Self Organizing Maps (SOM). These methods use clustering results as labels for representation learning and simultaneously update clustering outcomes based on the learned representations.

Supervised Learning predicates on the availability of labeled data, supervised techniques train models to predict specific outcomes based on past data. In the process, these models learn representations that capture the inherent relationships and dynamics of the time series. Common models in this category include recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks, which are adept at handling sequential data. Convolutional neural networks (CNNs) have shown potential in catching temporal dependencies for forecasting tasks.

2.3 Unsupervised Learning

The authors in [35] describe approaches that combine clustering with feature learning to have demonstrated impressive results in unsupervised representation learning [36] . The core aim is to identify inherent patterns and structures in data without the need for explicit labels. Classic clustering techniques, including flat clustering [33], hierarchical clustering [37], and spectral clustering [38], typically use predefined feature extractors to group similar samples . However, their efficacy diminishes with time series data due to its inherent complexity and high dimensionality. Traditional clustering's reliance on basic distance metrics often falls short in capturing these complexities. Moreover, handcrafted features can be inconsistent and lack automation.

Deep neural networks, with their capacity to discern complex data hierarchies, are apt for handling the nuances of time series data. This has led to the rise of "deep clustering" which merges conventional clustering with deep learning. This fusion blurs the lines between clustering and representation learning, enabling iterative optimization to transform input data into a new latent space [39]. Techniques such as DTCR [40] utilize auto-encoder foundations, optimizing both reconstruction and clustering losses. Finally, SOM, a model based method [41], it is a specific type of ANN used for model-based clustering. Notably, clustering outcomes can act as self-supervised cues, aiding the self-learning mechanism.

2.3.1 Deep Temporal Clustering Representation

Deep Temporal Clustering Representation (DTCR) presents a novel approach to time series clustering by seamlessly blending temporal reconstruction capabilities with the objectives of the K-Means clustering algorithm. At its essence, it leverages a sequence-to-sequence (seq-to-seq) model to produce temporal representations that are specific to each cluster.

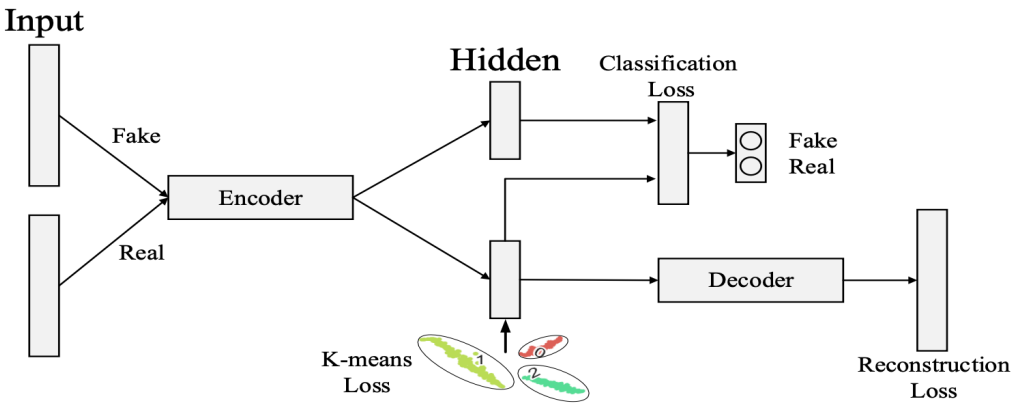


Figure 2.2: The general architecture of the Deep Temporal Clustering Representation (DTCR) [40]

One of the standout features of DTCR is its integration of an auxiliary classification task. This task serves to improve the encoder's ability to discern and differentiate between various temporal patterns. To further refine the learning process, it introduces a fake-sample generation strategy. This strategy involves the random shuffling of certain time steps within the data, challenging the model to recognize and adapt to artificially introduced temporal anomalies. Such a strategy not only augments the robustness of the model but also ensures that it remains sensitive to genuine temporal patterns.

The architectural choice of DTCR is also noteworthy. By employing bidirectional dilated recurrent neural networks as the encoder, it ensures that the resulting representations are both comprehensive and nuanced. This architecture allows the model to capture the inherent temporal dynamics present in time series data. Moreover, the use of dilation ensures that the representations encapsulate multi-scale characteristics, recognizing patterns that manifest over varying time scales.

As illustrated in Fig.2.2, DTCR consists of three losses and the overall training

loss $\mathcal{L}_{DTCR}$ is defined by:

$$\mathcal{L}_{\text{DTCR}} = \mathcal{L}_{\text{reconstruction}} + \mathcal{L}_{\text{classification}} + \lambda \mathcal{L}_{\text{K-means}} \tag{2.1}$$

where λ is the regularization coefficient. Eq.2.1 is minimized to learn the cluster-specific representations. Specifically, $\mathcal{L}_{\text{reconstruction}}$ makes the representations reconstruct the input. $\mathcal{L}_{\text{classification}}$ enhances the ability of the encoder. $\mathcal{L}_{\text{K-means}}$ encourages the representations to form cluster structures.

2.3.2 Self Organized Maps

SOM [41] is a type of ANN that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space, called a map, and is therefore a method to do dimensionality reduction. Self-organizing maps differ from other artificial neural networks as they apply **competitive learning** and **cooperative learning** as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space.

SOM introduced by Kohonen in 1990, is an extension K-means clustering that doesn't require initial projection onto a low-dimensional surface. It produces an easily understandable low-dimensional visualization, even though it represents data across multiple dimensions. It has been found useful to reveal intricate patterns and structure in several applications [42]. It is also used in data mining for its vector quantisation property [43]. As mentioned above, SOM can represent high-dimensional observable data onto lower dimension latent space, typically on a 2D square grid as illustrated in Fig.2.3, while preserving the topology of the original input space. But the map can also be used for projecting new data points and seeing which cluster belongs to the map, too.

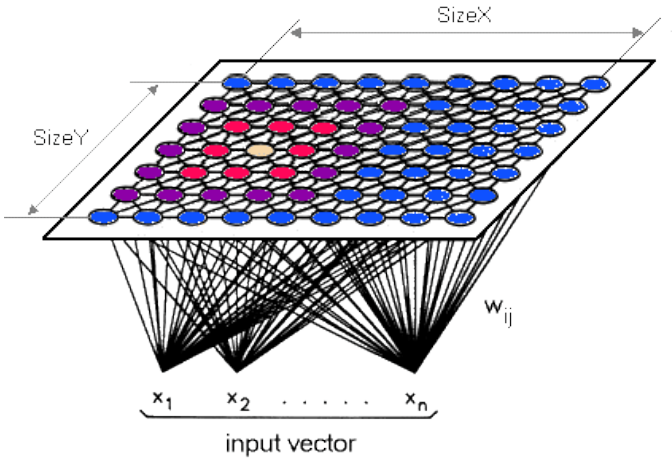


Figure 2.3: Architecture of a SOM network (reference)

Each neuron i in the map has an associated weight vector $w_i \in \mathbb{R}^d$ of the same dimensionality as the input data x. This weight vector is initialized randomly and gets updated during the training process. Each map can be thought as having two sets of coordinates: an input space which is represented by the weigh vector and the output space represented by the position of the map.

In each training step, one sample x is chosen, the distance between x and all the weights vectors w_i are computed. The neuron whose weight vector is closest to the input vector is called the **Best Matching Unit** (BMU) w_{bmu}:

$$w_{BMU} = \arg\min_i \|x - w_i\|_2 \tag{2.2}$$

After finding the BMU, the weight vectors w_{bmu} are updated so that the BMU is moved closer to the input vector in the input space. Also the topological neighbors of the BMU are treated similarly. We say that We say that SOM applies **competitive learning** because the weight vectors most similar to a data vector is modified so that it is even more similar to it and **coorperative learning** because not only the most similar weight vectors in this case BMUs, but also the neighboring weight vectors are moved towards the data vector. the SOM update rule for the weight unit w_i is as follows :

$$w_i^{n+1} = w_i^n + \alpha^n \times h_{BMU,i}^n \times (x - w_i^n) \tag{2.3}$$

$$\alpha^n = \alpha^0(1 - \frac{n}{T}) \tag{2.4}$$

Where α^n is a decreasing learning rate and T is the training length. Topological neighborhood structure is promoted via a neighborhood kernel $h^n_{BMU,i}$ that weights the nodes inversely proportional to their distance with BMU . A Guassian kernel is often used, which weights the node i as following :

$$h^n_{BMU,i} = \exp\left(-\frac{||r_{BMU} - r_i||^2}{2(\sigma^n)^2}\right) \tag{2.5}$$

$$\sigma^n = \sigma^0(1 - \frac{n}{T}) \tag{2.6}$$

Where r_i is the projection of the weight vector w_i to its corresponding coordinate on the grid and σ^0 denotes the initial standard deviation.

2.4 Supervised Learning

Supervised representation learning in the context of time series data is a pivotal approach that has gained prominence for its ability to automatically extract meaningful and informative features from temporal sequences, with guidance from labeled target information. This methodology is particularly vital in dealing with the inherent complexities of time-varying data. A standout technique in this domain is the use of RNNs, notably LSTM networks, which have exhibited exceptional prowess in capturing intricate temporal dependencies and patterns within sequential data [44]. These networks utilize specialized memory cells to maintain and update information over time, making them adept at modeling a wide range of time series phenomena. Additionally, the Temporal Convolutional Network (TCN) has emerged as an alternative approach, employing one-dimensional convolutions to capture hierarchical features within time series data while maintaining computational efficiency [45]. Such supervised representation learning techniques have found applications in diverse domains, including finance, where precise forecasting of market trends is critical, healthcare, for anomaly detection in patient monitoring, and environmental science, for recognizing complex patterns in climate and ecological data.

2.4.1 Long Short Term Memory (LSTM)

The authors in [46] have mentionned that RNNs are purposefully crafted to handle sequential data. A fundamental characteristic shared among these tasks is the presence of temporal dependencies within the data, which traditional feedforward neural networks are ill-equipped to capture. RNNs, as a solution to this issue, are structured to incorporate both past and present data into their architecture. The key feature of an RNN is its recurrent connection, which forms a loop that connects the output of a neuron at one time step to the input of the same neuron at the next time step. This loop enables RNNs to create a form of memory, allowing them to remember information from earlier steps and use it to influence the current step's output. This inherent ability to model sequences makes RNNs exceptionally powerful in tasks like predicting the next word in a sentence, generating sequences of text, recognizing speech patterns, and forecasting future values in time series data.

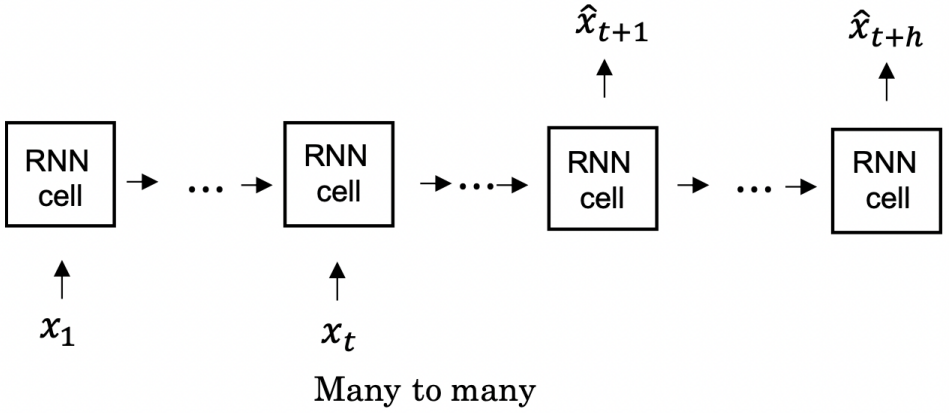


Figure 2.4: Basic architecture of a RNN for time series forecasting [46]

RNN architectures come in various forms, depending on the number of inputs and outputs involved: one-to-one (one input and one output), one-to-many (one input and multiple outputs), many-to-one (multiple inputs and one output), and many-to-many (multiple inputs and outputs). Frequently encountered RNN configurations include many-to-one for classification tasks and many-to-many for applications like machine translation or time series forecasting. Additionally, in the case of time series data, the length of the input sequence often differs from the size of the output sequence, which typically corresponds to the number of samples to be predicted. Fig.2.4 illustrates a basic RNN structure tailored for time series forecasting, where

x_i and $\hat{x}_i$ denote the actual and predicted values at time i respectively, while h represents the prediction horizon .

However, traditional RNNs have some limitations, such as difficulties in learning long-term dependencies, often referred to as the "vanishing gradient" problem. To address these limitations, variations of RNNs, such as LSTM networks and Gated Recurrent Units (GRUs), have been developed. These architectures incorporate mechanisms that better control and manage information flow over longer sequences, making them more effective in capturing complex patterns in sequential data. In order to solve the vanishing gradient problem [44]. Fig.2.5 shows a picture of how a hidden unit works in a LTSM recurrent network.

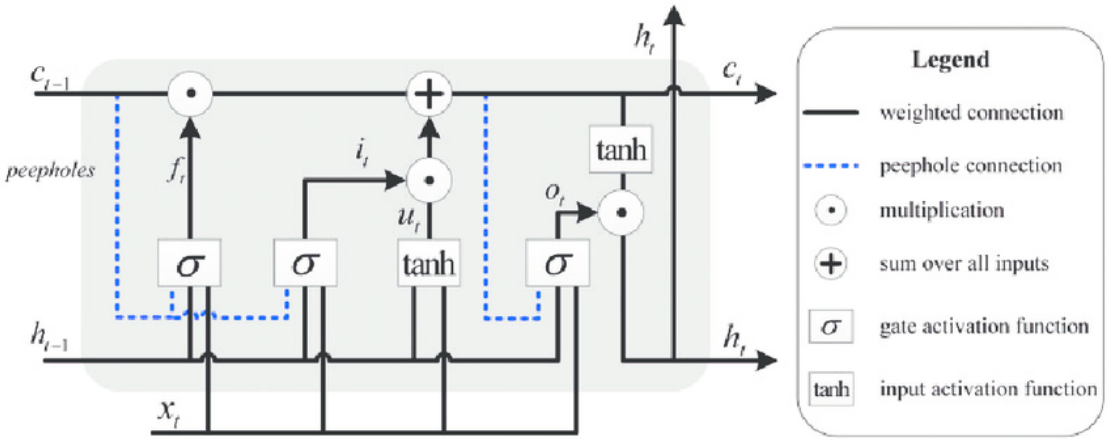


Figure 2.5: Hidden unit in LSTM [47]

LSTM uses three gates to keep relevant information and discard the irrelavant ones. These gates are **forget gate** f. The forget gate is responsible for deciding which information from the previous cell state should be retained and which should be discarded. It takes the previous cell state c^{t-1}, the current input x^t and the previous hidden state h^{t-1} as inputs and produces an output between 0 and 1 for each element in the cell state. A value of 1 indicates that the corresponding information should be preserved, while a value of 0 means it should be forgotten. The forget gate is controlled by a sigmoid activation function. The following equation shows this procedure :

$$f^t = \sigma(W_f \cdot [h^{t-1}, x^t, c^{t-1}] + b_f) \tag{2.7}$$

where w_f is the weight associated with the input x^t, cell value c^{t-1} and hidden state h^{t-1}. While b_f represents the bias vector .

Then comes the **input gate** i determines what new information should be added to the cell state. It consists of two parts: a *sigmoid* layer that decides which values will be updated and a *tanh* layer that creates a vector of new candidate values u. These two parts are then combined to update the cell state. The input gate's output is controlled by the *sigmoid* and *tanh* activation functions.

$$i^t = \sigma(W_i \cdot [h^{t-1}, x^t, c^{t-1}] + b_i) \tag{2.8}$$

$$c^t = f^t * C^{t-1} + i^t * u^t \tag{2.9}$$

Finally, comes the **output gate** o that decides what part of the cell state should be exposed as the network's output. It takes the current cell state c^t and the current input x^t with hidden state h^t as inputs and passes them through a *sigmoid* activation function. The result is then multiplied by c^t after passing through a *tanh* function. This produces the output of the new hidden state h^t for the current time step.

$$o^t = \sigma(W_o \cdot [h^{t-1}, x^t, c^t] + b_o) \tag{2.10}$$

$$h^t = o^t * \tanh(c^t) \tag{2.11}$$

LSTM model serves as a robust foundation for various groundbreaking works in time series representation learning, as the DeepAR model. DeepAR [48] underscores LSTM's prowess as the core architecture for time series forecasting. DeepAR's success in providing probabilistic forecasts relies on autoregressive recurrent networks, primarily built upon LSTM cells, showcasing LSTM's proficiency in capturing intricate patterns and uncertainties within time series data.

In essence, LSTM models, with their capacity to model long-range dependencies and sequential information, have consistently demonstrated their versatility and power as foundational elements.

2.4.2 Temporal Convolutional Networks

TCNs have gained significant attention in recent years as a powerful tool for time series analysis and representation learning. TCNs represent a departure from traditional recurrent and autoregressive models like LSTMs and ARIMA, offering unique advantages for modeling sequential data. Unlike LSTMs, which rely on sequential processing and may suffer from vanishing gradient problems, TCNs

employ one-dimensional convolutions, which allow them to capture long-range dependencies efficiently. This is particularly valuable for time series data, as it often contains complex temporal patterns that can span various time scales. Moreover, TCNs provide parallelism in training and prediction, making them computationally efficient and well-suited for large-scale applications

The standard Convolution considers future information when computing the output of a given time step. This issue can be solved by providing a Causal formulation to convolutions. A formulation in which the present value only depends on past and present inputs. Causality is easily obtained by padding asymetrically the input sequence with zeros.

A basic causal convolution has a limited capability to examine past information, typically constrained to a linear depth within the network. Consequently, its practicality in sequence-related tasks, particularly those demanding extensive historical context, becomes a challenge. In addressing this issue, TCN adopts dilated convolutions. These dilated convolutions offer the advantage of exponentially expanding the network's receptive field, enabling more effective handling of longer historical sequences. For a 1-D sequence input $x \in \mathbb{R}^d$ and a filter $f : \{0, ..., k-1\} \rightarrow \mathbb{R}$, the dilated convolution D on element t of the sequence as following :

$$D(t) = \sum_{i=0}^{K-1} f(i) \cdot x_{t-d \cdot i} \tag{2.12}$$

Where the dilation factor is denoted as d, along with k representing the filter size, and $s - d \cdot i$ accounting for the direction of the past, collectively define the concept of dilation in convolution operations. Essentially, dilation introduces a fixed step between each pair of adjacent filter taps. As depicted in Fig.2.6 When $d = 1$, a dilated convolution behaves identically to a regular convolution. However, by increasing the dilation factor to values greater than 1, the output at the highest level of the network becomes capable of representing a broader spectrum of inputs. This expansion effectively widens the receptive field of a ConvNet, allowing it to capture more extensive contextual information.

TCN employs a generic residual module because it effectively allows layers to learn modifications to identity mapping rather than the entire transformation , which has shown to be effective in very deep networks The residual block for our baseline TCN is shown in Fig.2.7

TCNs and LSTMs are both advanced architectures tailored for sequence modeling, yet they present distinct advantages and challenges. TCNs are renowned

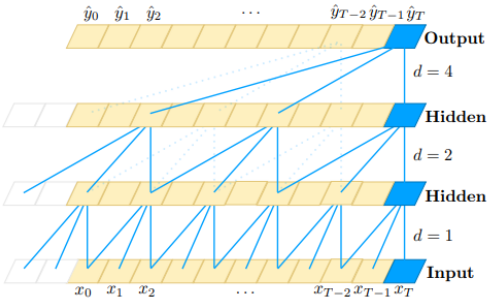


Figure 2.6: A dilated causal convolution [45]

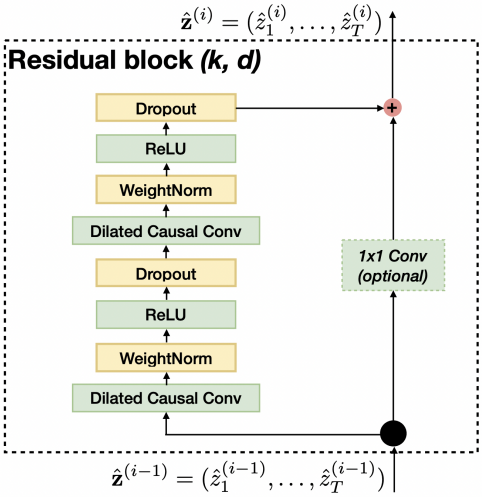


Figure 2.7: TCN residual block [45]

for their ability to process data in parallel, a feature that stands in contrast to the sequential nature of LSTMs. This parallelism, coupled with dilated convolutions, equips TCNs to adeptly capture long-range dependencies in data without a significant increase in parameters. Furthermore, the gradient stability during TCN training, bolstered by residual connections, often outperforms that of LSTMs, which, despite their gating mechanisms, can still encounter the vanishing gradient problem in certain scenarios. TCNs also demonstrate flexibility in handling sequences of varying lengths and can be more memory-efficient since they don't maintain continuous hidden states across time steps, unlike LSTMs.

However, LSTMs possess their own set of strengths. Their intricate gating mechanisms, including forget, input, and output gates, allow them to regulate information flow, making them particularly adept at retaining crucial long-term dependencies and discarding irrelevant data. This inherent memory mechanism can be invaluable for tasks where context from distant past time steps is pivotal. Moreover, the sequential processing of LSTMs can offer a more intuitive and interpretable temporal progression for certain tasks. For short-sequence tasks or scenarios where long-term dependencies are not as critical, LSTMs might present a more parameter-efficient alternative.

In essence, while TCNs offer advantages like parallelism, gradient stability, and flexibility in sequence length, LSTMs shine with their sophisticated gating mechanisms and inherent memory capabilities.

2.5 Hybrid Learning

In the realm of time series data analysis, hybrid machine learning models have emerged as a potent tool for representation learning, seamlessly blending the strengths of multiple algorithms to capture both linear and non-linear patterns inherent in sequential data. By integrating architectures like ANNs with classical time series models such as ARIMA, or combining feature-based representations with model ensembles, these hybrid approaches offer enhanced accuracy, robustness, and generalization capabilities. While they harness the pattern recognition prowess of deep learning and the statistical rigor of traditional models, challenges like model interpretability and computational efficiency remain. Nevertheless, the strategic fusion of models in a hybrid framework presents a promising avenue for advancing time series representation, paving the way for more insightful and accurate data analyses. In this section, we provide a comprehensive examination of the hybrid models as delineated by [34].

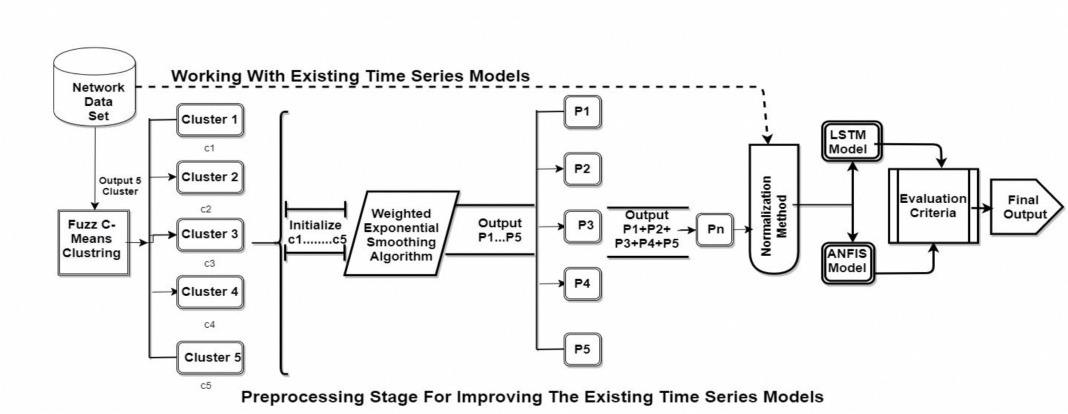


Figure 2.8: Hybrid model for time series forecasting [34]

Fig.2.8 delineates the architecture of the proposed framework aimed at enhancing the efficacy of time series models for network traffic prediction. This advanced methodology refines both the Long Short-Term Memory (LSTM) and Adaptive Neuro-Fuzzy Inference System (ANFIS) time series models. Empirical data, sourced from diverse network backbones, serve as the foundation for evaluating the proposed framework's methodology. The fuzzy c-means clustering algorithm is employed to segment the network data, facilitating the identification of homogenous object characteristics. These homogenous entities are subsequently grouped, with a focus on five distinct clusters (C1 to C5) based on the inherent data distribution. These clusters act as preliminary inputs for the Weighted Exponential Smoothing (WES)

method, which is applied individually to each cluster. From this process, five predictors (P1 to P5) are derived, which are then amalgamated to produce a composite predictor, Pn. This predictor, Pn, serves as the input for both the LSTM and ANFIS time series models. This preprocessing phase is integral to augmenting the predictive capabilities of the LSTM and ANFIS models for network traffic. Notably, the proposed framework demonstrates a marked reduction in prediction errors, yielding results that align more closely with expectations. As a concluding step, evaluative metrics are utilized to rigorously assess the refined methodology's capacity to optimize advanced machine learning algorithms, specifically LSTM and ANFIS, for network traffic prediction.

Chapter 3

Time Series Analysis

This chapter delves into an analysis of the available data and thoroughly describes the foundational scenario. It emphasizes the importance of examining the provided data, aiming to identify specific patterns and overarching trends

Section 3.1 details the terrestrial network for each district of the Milan municipality. It further offers a visual representation of LTE BSs' placement within the city, linking examined clusters to specific Milan neighborhoods. Additionally, the selection of traffic zones for analysis is determined based on varying activities and anticipated patterns. Following this, Section 3.2 is dedicated to deriving insightful statistics regarding the selected areas. The primary objective is to identify overarching trends, patterns, and behaviors during the examined period, especially during peak hours and days.

3.1 Spatial Distribution of Base Stations in Milan's Traffic Zones

Our data is characterized by its geographical separation where each zone consists of 7 BS, leading to an aggregate of 112 cells (equivalent to 112 BSs) spread across 16 distinct clusters. To visualize the spatial distribution and scale of these sample clusters within Milan, the BSs were plotted on a city map. Fig.3.1 illustrates the positioning of the studied cells within the city's municipal regions.

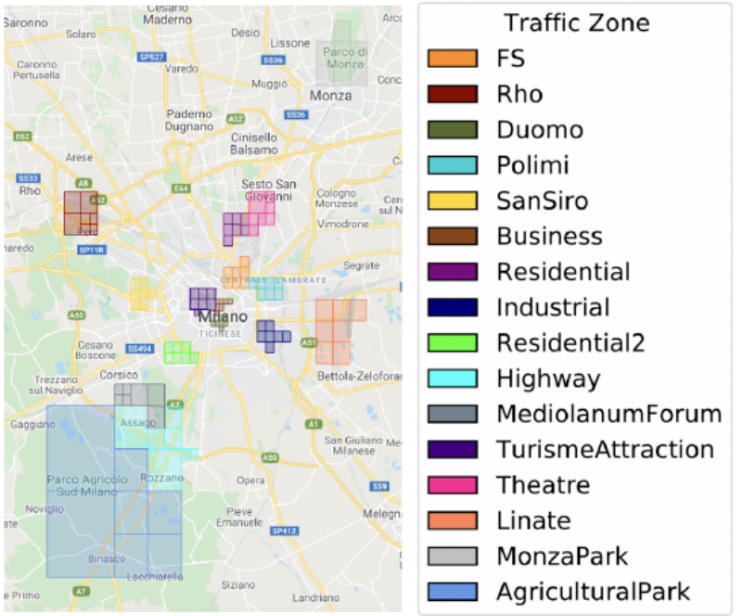


Figure 3.1: location of different BS zones in Milan [49]

In the next section, we will conduct a statistical analysis on the data traffic, based on several zones in order to better characterize the traffic in different areas. The traffic zones we've opted for are **business**, **Rho**, **San Siro**, **FS**, and **Residential** areas. This selection is intended to compare areas that are distinctly different, reflecting the varied activities and habits of the individuals within them.

3.2 Statistical Properties per zone

The data sets being examined are sourced from a leading Mobile Network Operator (MNO) in Italy, specifically related to Milan. These data sets are based on the 5G technology utilized by this major operator, detailing the traffic demand in bits for 1,420 time instants (observations) per base stations across various districts of Milan, recorded every hour. This data covers a period of approximately two months in 2020, beginning on April 1st and concluding at 11 pm on Mai 30th.

Fig.3.2 presents the outcomes of the calculations for the mentioned clusters. It emphasizes the differences in the Cumulative Distribution Functions (CDFs). The results align with expectations. Specifically, the FS district records the highest traffic volume, attributed to the Milan Central Train Station, a key mobility hub

in the city. The Residential area, where people reside, displays more consistent demand patterns, with a pronounced curve at lower traffic values and a short tail. Which means that the Residential area does not go through sudden traffic peaks and that low traffic volumes is more common in the Residential area compared to the other areas. On the other hand, Rho and SanSiro known for events and fairs, occasionally experience significant traffic spikes. This is evident, especially for the Rho district and SanSiro since they are characterised with a very long tail that reaches high traffic volumes.

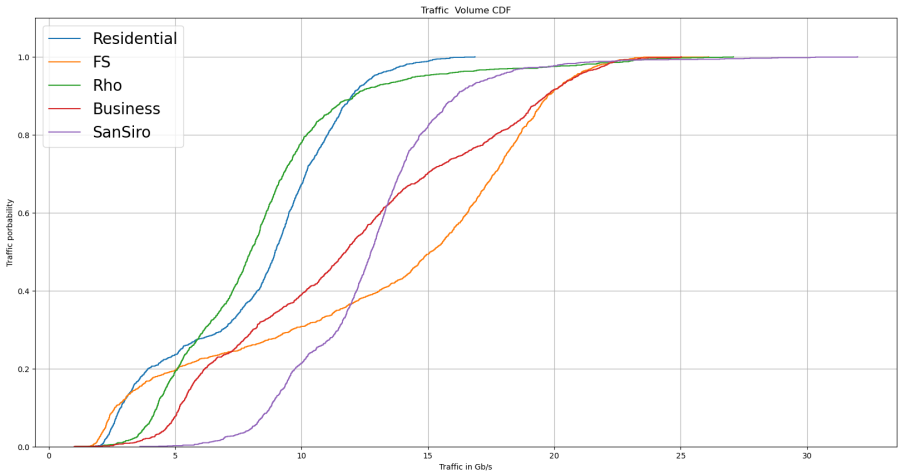


Figure 3.2: CDFs of different traffic areas in Milan

The next step involves calculating the average traffic in each area, aggregating data on both an hourly and daily basis. Fig.3.3 displays the results for the primary traffic aggregation in each district. A consistent trend across all curves is the evident drop in traffic volume during the night. From 1 AM on-wards, even the busiest district sees minimal traffic. Notably, the FS district consistently records high traffic, a reflection of its significance as previously discussed. Its traffic peaks at 5 PM, likely due to people starting to return from work. Supporting this observation, the Business district sees a decline in traffic post-3 PM, with its peak at 2 PM. Conversely, residential areas experience the highest traffic at 9 PM when residents return home. Similarly, SanSiro's peak hours align with this, likely due to football matches drawing large crowds. Lastly, the Rho district sees its maximum traffic in the afternoon.

Finally, Fig.3.4 showcases the average daily traffic for each cluster under consideration along with the hourly traffic along April and Mai. On one hand, This

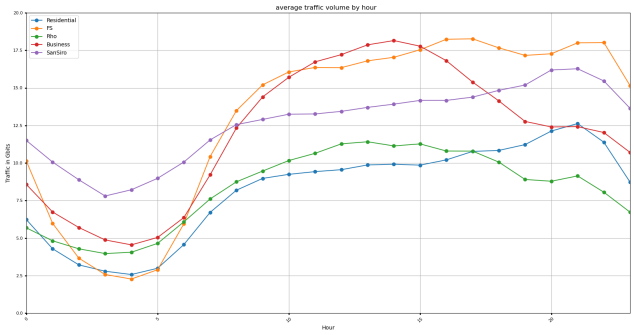


Figure 3.3: Daily traffic average for the chosen districts

visualization emphasizes the fluctuations in daily traffic demand (right). A prominent pattern observed from the graph is the distinction between weekday and weekend traffic along different zones , with a notable drop in traffic during the weekends in the Business and FS area. However, the district with the stadium deviates from this general trend. This is expected since football matches typically occur on weekends. As evidence, this district sees traffic surges on either Sundays or Saturday when local teams play. Another unique pattern is seen in the residential district, where the usual weekday-weekend alternation is inverted. Here, Sundays and Saturdays generally see higher traffic. On the other hand, the hourly traffic (left) reaffirms the claims made in the CDF analysis and the interpretations made by visually inspecting the daily traffic per zones (right). The variance in the Residential area is less pronounced compared to others, likely due to its dense population, ensuring consistent traffic. However Rho and SanSiro are characterized by sudden peaks that far exceed the average traffic, which is due the weekly events in SanSiro and Fairs in Rho.

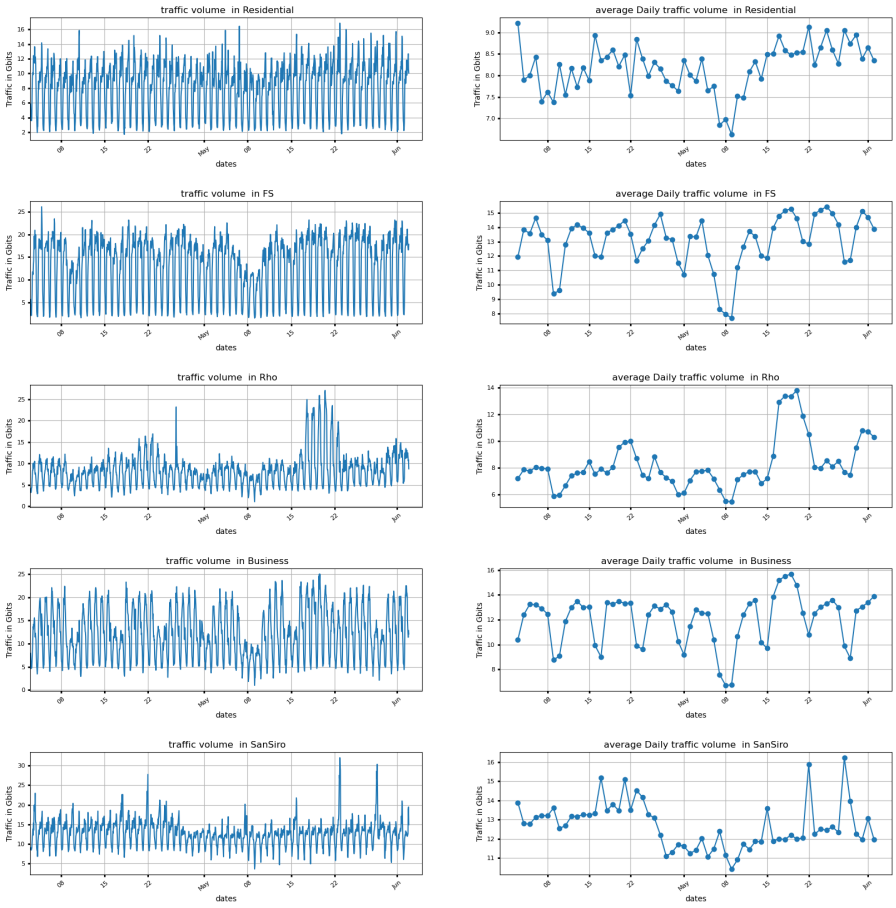


Figure 3.4: Hourly traffic (right) daily average traffic (left)

Chapter 4

Methodology

This chapter delves into the models and parameters employed in our methodology.

Section 4.1 provides a comprehensive overview of the standard terrestrial network scenario for each district of the Milan municipality, detailing both the ground infrastructure and the on-board capabilities of the HAPS. Furthermore, Section 4.2 details the hybrid model, SOM-TCAN, which integrates a Self-Organized Mapping backbone for clustering with a Temporal Convolutional Attention Network for forecasting. Then Section 4.3 introduces an LSTM-based forecasting model, designed for individual time series training. This model serves as a baseline for comparison with our primary approach, offering a more direct and straightforward methodology.

4.1 Scenario Configuarion

In this research, we propose a fusion of the terrestrial network infrastructure of a densely populated region with an aerial network, consisting of a HAPS that functions as a HAPS-SMBS. This concept is illustrated in Fig.4.1.

Our design aims to provide coverage to 16 traffic zones in Milan, Italy, using a singular HAPS equipped with Massive-MIMO technology. This allows for a dedicated coverage beam for each traffic zone. In Section 3.1, we delved into an in-depth analysis of network traffic spanning 16 distinct zones in Milan. Each of these zones is equipped with 7 BSs, cumulatively generating a dataset of 112 unique time series. The core tenets of our research are:

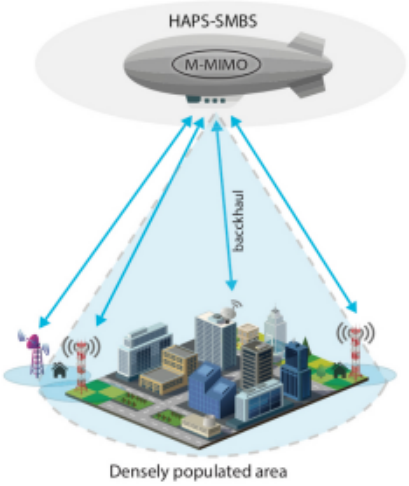


Figure 4.1: HAPS-SMBS scenario [50]

- **Traffic Pattern Analysis**: By examining the intricate traffic patterns, we aim to cluster the BSs. This clustering not only aids in understanding the underlying network dynamics but also provides insights into peak traffic times and usage trends.
- **Behavioral Representation**: Beyond clustering, our goal extends to crafting a precise representation for each BS.

4.2 SOM-TCAN

Our proposed model, visually represented in Fig.4.2, is crafted to minimize the number of time series models required for both accurate representation and future traffic predictions. For the clustering phase, we've chosen SOM algorithm. Given our data's distribution across the 16 zones, we found that a 4x4 neuron grid for the SOM was optimal. One of the standout attributes of SOM is its data-driven approach to determining cluster count, a stark contrast to methods like k-means which require predefined cluster numbers.

Upon completion of the SOM clustering, each BS's traffic is mapped to a Best Matching Unit (BMU) or "winning neuron". It's essential to note that BSs mapped to the same BMU exhibit similar traffic patterns, thereby justifying their grouping

into a cohesive cluster. The BMU's weights are dynamically adjusted in line with Eq.2.3 through-out the training phase. This continuous adjustment ensures that each BMU evolves to become a representative model, accurately reflecting the traffic patterns of its associated BSs.

Emerging from this methodology are n well-defined clusters. The characteristic weights of these clusters, ranging from W_1 to W_n, form the foundational dataset for the development of n specialized traffic prediction models based on Temporal Convolutional Attention Networks (TCAN). This approach, while being resource-efficient, is also designed with foresight, accommodating the potential for future expansions or modifications to the BS network in Milan.

To underscore the reliability and precision of our methodology, we've incorporated a comprehensive suite of evaluation metrics. These metrics provide both quantitative and qualitative assessments, ensuring a multi-faceted evaluation of our approach's performance.

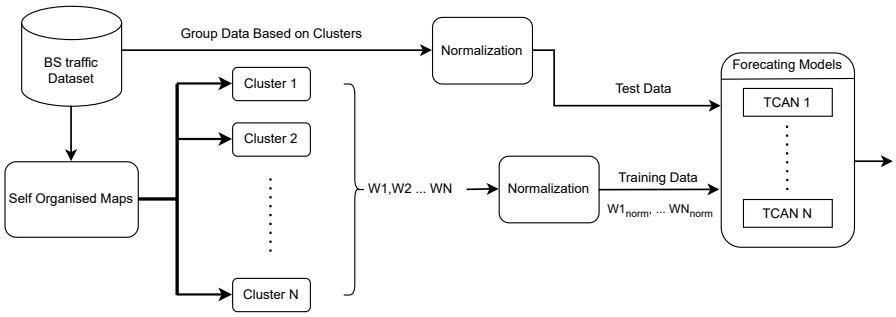


Figure 4.2: SOM-TCAN

4.2.1 SOM Clustering

Building on our earlier discussion, the adoption of the SOM offers distinct advantages over traditional clustering techniques. One of the most salient benefits of SOM is its inherent ability to determine the optimal number of clusters based on the data's topology. This feature obviates the often challenging and somewhat arbitrary task of pre-specifying the cluster count, a common requirement in many clustering algorithms.

For our study, we have chosen a 4x4 neuron grid for the SOM. This grid size aligns with the distribution of our data across the 16 zones, ensuring that each zone can potentially map to a unique neuron.

However, it's crucial to acknowledge that while SOM offers flexibility in cluster determination, certain parameters need to be predefined to ensure the algorithm's effective operation. Specifically, the parameters α and σ, as delineated in Eq.2.4 and Eq.2.6 respectively, require prior specification. After careful consideration and based on preliminary tests, we have chosen $\alpha = 0.05$ and $\sigma = 0.3$ as the optimal values for our dataset.

Furthermore, the training duration, represented by T, plays a pivotal role in the convergence and stability of the SOM. A longer training duration often leads to a more refined clustering outcome. For our study, we have determined that a training length of $T = 50000$ steps strikes the right balance between computational efficiency and clustering accuracy.

Following the training of the SOM on our dataset, we observed a convergence towards 8 distinct winning neurons. These neurons, having adapted their weights over the course of the training, effectively represent the primary traffic patterns within our data. For a detailed breakdown of each winning neuron, we refer to Table 4.1. This table lists the coordinates of each winning neuron, the number of samples associated with it, as well as the mean and standard deviation of the traffic patterns they represent. Complementing this, Fig.4.3 offers a visual representation of the weights of these neurons. The plot highlights the weight adjustments and showcases the distinct patterns that each neuron has captured over the training duration. Together, the table and plot furnish a comprehensive understanding of the SOM's clustering results and the inherent traffic patterns within our dataset.

Neurons	Number of Samples	mean (Gbits)	std (Gbits)
(0,1)	2	1.27	0.76
(1,0)	16	3.37	1.55
(1,1)	15	27.81	8.92
(1,2)	15	5.83	2.07
(2,0)	18	13.88	4.49
(2,1)	1	68.90	43.03
(2,2)	16	19.04	8.18
(3,1)	29	9.04	3.45

Table 4.1: Winning Neuron's Statistics

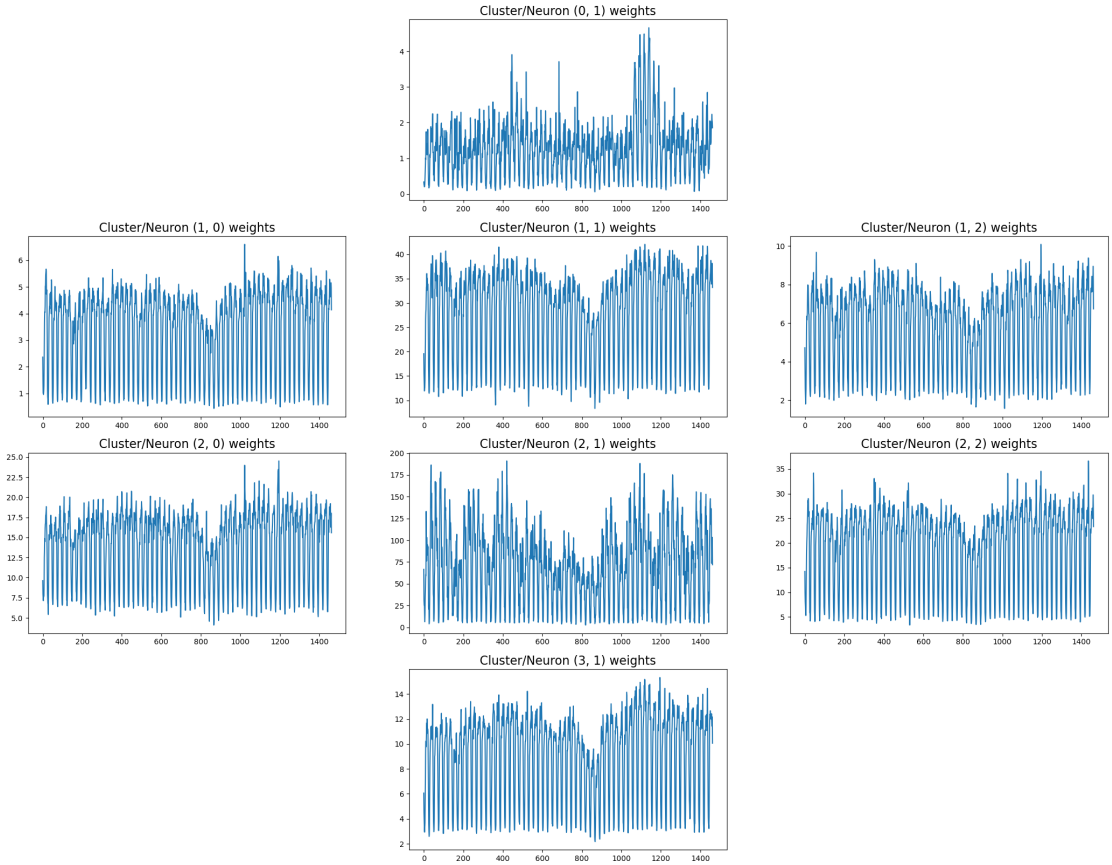


Figure 4.3: Weights of winning neurons

Upon examining Table 4.1 and referencing Fig.4.3, it becomes evident that Neuron (2,1) is uniquely associated with a singular time series. This particular assignment can be attributed to the pronounced traffic demands exhibited by this time series during specific time-steps. These demands significantly surpass the traffic patterns observed in many of the BSs that are mapped to adjacent neurons.

One might wonder why this data sample wasn't grouped with a neighboring neuron. The rationale lies in the distribution of the data samples. The neighboring neurons, each accommodating more than 15 samples, exhibit a certain stability in their weight profiles. Introducing a time series with such high traffic peaks to these neurons could disrupt this stability. This poses a challenge for subsequent forecasting models. Since these models rely on neuron weights for training, the presence of an anomalous high-peak sample could skew predictions, leading to sub-optimal performance, especially when confronted with high traffic demands.

A similar observation can be made for neuron (0,1). This neuron is associated with two time series samples, both of which are characterized by notably low traffic demands. Such outliers, if not appropriately clustered, can adversely impact the efficacy of forecasting models, underscoring the importance of the SOM's clustering approach and the selected parameter values.

This distribution and clustering outcome are heavily influenced by the σ and α values we previously selected. By decreasing the α value, the learning rate would be reduced, leading to a potential decrease in the number of distinct neurons. Consequently, the high and low peak data samples might not be isolated as it currently is, and could instead be grouped with neighboring neurons.

To critically assess the efficacy of the clustering produced by our SOM clustering, it's essential to employ metrics that capture the nuances of data structures, both at a local and global scale. This leads us to the pivotal metrics of **trustworthiness** and **neighborhood preservation** [51].

Trusworthiness serves as a safeguard against the potential pitfalls of dimensionality reduction. When data is projected from a high-dimensional space to a lower-dimensional one, there's a risk that points which were distant from each other in the original space might end up being close in the reduced space. Trustworthiness quantifies this distortion. For each data point, it identifies those points that have become its neighbors in the reduced space but weren't originally its neighbors in the high-dimensional space. The metric then assigns a penalty based on how close these new neighbors have become, with closer neighbors incurring a higher penalty. The formula for trustworthiness, for a given neighborhood size k, is:

$$T_k = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in U_k(i)} (r(i,j) - k) \tag{4.1}$$

Where $U_k(i)$ represents the set of these new neighbors for point i and r represents the rank of the distance between points i and j in the original high-dimensional space.

Conversely, **Neighborhood Preservation** is concerned with the maintenance of local structures. It evaluates how well the original neighbors of a data point remain its neighbors after the projection. For each point, it identifies those neighbors from the high-dimensional space that are no longer its neighbors in the reduced space. Similar to trustworthiness, it assigns a penalty based on how distant these original neighbors have become in the reduced space. The formula capturing this is:

$$P_k = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in V_k(i)} (r'(i, j) - k) \tag{4.2}$$

where $V_k(i)$ denotes the set of original neighbors that are no longer neighbors in the reduced space for point i and r' denotes the rank of the distance between points i and j but in the reduced-dimensional space.

For a comprehensive evaluation, we've calculated trustworthiness and neighborhood preservation for various values of k. The results are tabulated in Table 4.2. It's worth noting that values closer to 1 for both metrics indicate better preservation of the data's structure, signaling the effectiveness of our SOM clustering.

In essence, by employing trustworthiness and neighborhood preservation, we are ensuring a dual-layered evaluation of our SOM results. While trustworthiness ensures that no false proximities are introduced, neighborhood preservation guarantees that intrinsic local relationships are upheld. Together, these metrics provide a robust and comprehensive evaluation framework, ensuring the integrity and quality of our SOM clustering.

K	trustworthiness	neighborhood preservation
2	0.863	0.831
3	0.87	0.865
4	0.876	0.88
5	0.882	0.90

Table 4.2: Trustworthiness and neighborhood preservation for SOM evaluation

4.2.2 Normalization

Normalization is a pre-processing technique employed to adjust and modify real network data. Its primary purpose is to improve time series models by scaling the data and converting it from one format to another. By using normalization, one can ensure that larger numeric values don't overshadow smaller ones. The min-max approach is specifically used to scale network traffic data, converting it to a scale ranging from 0 to 1.

$$z = \left(\frac{x - x_{min}}{x_{max} - x_{min}} \right) \times (\max - \min) + \min \tag{4.3}$$

4.2.3 Temporal Convolutional Attention Network

Having successfully clustered our data using the SOM, we now transition to the subsequent phase of our pipeline: forecasting. As previously alluded to, the weights of each neuron, fine-tuned through the SOM process, will serve as the foundation for our training. Correspondingly, the time series assigned to each neuron will be harnessed for testing purposes, ensuring our model's predictions are grounded in real-world data patterns.

For this forecasting endeavor, we've chosen to employ a **Temporal Convolutional Attention Network (TCAN)**. This model, renowned for its capability to capture intricate temporal patterns, leverages both convolutional layers to detect local patterns and an attention mechanism to weigh the significance of different time steps. By integrating these components, we aim to achieve precise and robust predictions, further enhancing the utility and effectiveness of our comprehensive pipeline. There have been attempts to implement such architectures, showcasing their potential in forecasting [52, 53]

The core of our TCAN is its convolutional layers, which are specifically designed to capture temporal patterns in time series data. A crucial parameter in these layers is the dilation rate, which determines how far apart the input values are taken for each convolution. For our model, we will discuss the dilation rate sequence in the next chapter since the choice of dilation rate sequence is intrinsically tied to how far back we wish to look into the past in our input data. Essentially, this means our model will capture patterns at various scales, from immediate neighbors to broader, more spaced-out intervals. A higher dilation rate allows the model to consider more distant past values without increasing the kernel size. Speaking of which, our chosen kernel size is 2, ensuring that each convolution considers pairs of data points. For a more in-depth discussion on dilation rates and kernel sizes, readers are referred to Section.2.4.2. The sequence of dilations directly dictates the number of stacked residual blocks, with each rate corresponding to a specific block layered on top of the previous ones.

To visually grasp the architecture and the interplay of these parameters, one can refer to Fig.4.4, which provides a comprehensive visualization of the TCAN structure.

Following the convolutional layers, we introduce the Bahdanau attention mechanism as a subsequent layer [54]. This attention mechanism aims to :

1. Refine the model's focus on specific intervals within the input sequence,

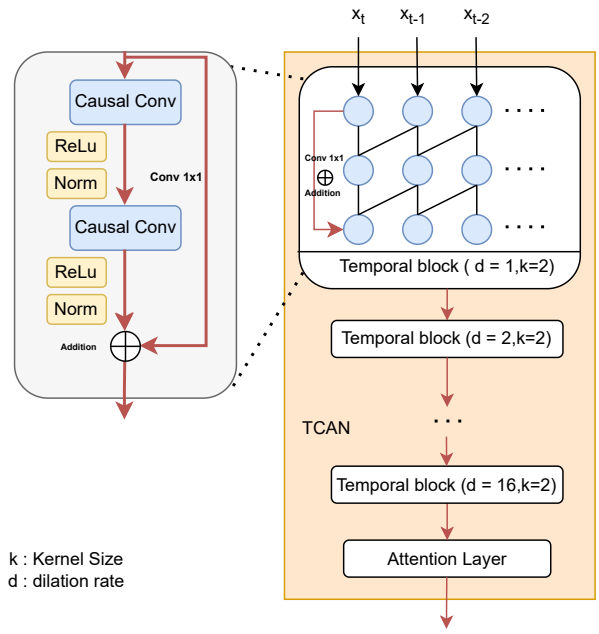


Figure 4.4: TCAN architecture

ensuring that the most pertinent temporal patterns are emphasized.

2. Expand receptive fields without the need to add more convolutional layers.

For each time step t in the output of the TCN, an attention score is computed, capturing the significance of that particular step:

$$score(h_t, s_{t-1}) = v^T \tanh(W_1 h_t + W_2 h_T) \tag{4.4}$$

where h_t represents the output from the TCN at the time step t, and the internal state h_T is the output form TCN at time step T. The trainable weight matrices W_1,W_2 and v adjust the model's focus on temporal patterns detected by the TCN. These scores are then normalized using the $softmax$ function to yield attention weights:

$$\alpha_t = \frac{\exp(score(h_t, h_T))}{\sum_{j=1}^{T} \exp(score(h_t, h_T))} \tag{4.5}$$

Subsequently, a context vector is derived for each time step:

$$c_t = \sum_{j=1}^{T} \alpha_j h_j \tag{4.6}$$

Given the context vector c_t and the output from TCN h_t, the attention vector a_t is formed using a fully connected layer with a $tanh$ activation function :

$$a_t = \tanh(W_c c_t + W_h h_t) \tag{4.7}$$

where Wc and W_h are the weights associated with the context vector and TCN output. This attention vector encapsulates the salient information from the entire input sequence, emphasizing the most relevant temporal patterns detected by the TCN. In essence, the Bahdanau attention layer dynamically adjusts the model's focus across the input sequence, ensuring that the most relevant temporal patterns, as identified by the TCN, are accentuated, thereby enhancing the precision of the forecasts.

4.3 Anticipating the Comparative Analysis: Individual LSTM Training

As we delve deeper into the validation of our SOM-TCAN model, it's crucial to set the stage for a comparison. To this end, we've developed an LSTM-based forecasting model, tailored to be trained on each individual time series. This approach, while more direct and straightforward, serves as a foundational baseline against which the merits of our hybrid model can be evaluated. The rational for the LSTM development is the following :

- **Baseline Evaluation:** Using the individually trained LSTM models provides a foundational baseline. This allows us to clearly delineate the advantages our SOM-TCAN model might offer and identify areas where it might fall short.
- **Complexity vs. Performance:** The SOM-TCAN introduces an intricate layer through its combination of clustering and representation learning. It's vital to determine whether this sophistication indeed offers performance benefits over the more straightforward LSTM training method.

- **Scalability Insights:** Comparing the two strategies offers insights into their scalability. As the volume and complexity of time series data grow, understanding how each method adapts becomes crucial.
- **Generalization Capabilities:** A head-to-head comparison provides clarity on how well our proposed model generalizes across diverse time series data, especially when benchmarked against the LSTM based model we've developed.

Fig.4.5 describes the model's architecture. which consists of a two-layered LSTM cells with size 128 connected to a MLP.

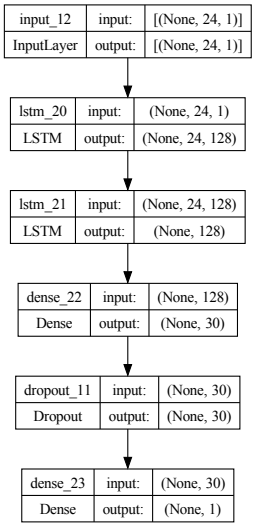


Figure 4.5: LSTM based model architecture

Chapter 5

Results

This chapter presents the thesis's experimental findings.

Section 5.1 outlines preliminary concepts, detailing the training and validation data split, the model's hyperparameters, and the employed loss function. Section 5.2 centers on the key performance indicators used to assess the accuracy of the proposed model. Section 5.3 discusses the configuration of the TCAN-based forecasting model, emphasizing the importance of selecting an appropriate look-back period. Furthermore, Section 5.4 evaluates the performance of the TCAN model for time series forecasting by contrasting it with two notable models: the LSTM network and TCN.

Finally, Section 5.5 offers a detailed comparison between the proposed SOM-TCAN framework and the Baseline approach which consists of a traditional method using individual LSTM models for each time series. The aim is to highlight the benefits of the integrated approach, especially in computational efficiency and forecasting accuracy.

5.1 Dataset Allocation, Validation Split, and Model Hyper-parameters

As previously discussed in Section 4.2, our methodology leverages the weights obtained from the SOM as the training dataset, while the time series corresponding to each neuron serve as the test set. This approach ensures that our model is

trained on generalized patterns and then evaluated on specific real-world data points.

For the training phase, it's essential to further split the SOM weights into training and validation sets to fine-tune the model and prevent overfitting. To this end, we allocate 85% of the weight's observations for training and reserve the remaining 15% for validation. This split ensures that the model has a substantial amount of data for learning while also having a separate subset to validate its performance during the training iterations. By monitoring the model's performance on the validation set, we can make necessary adjustments to achieve optimal forecasting accuracy.

The hyper-parameters for the the proposed forecasting models, play a pivotal role in influencing their performance. A comprehensive summary of these hyper-parameters can be found in table 5.1.

Hyper-parameter	**Value**
Number of Epochs	200
Batch Size	32
Loss Function	MSE
Optimizer	Adam
Early Stopping	20
Learning Rate	10^{-3}

Table 5.1: Summary of Model Hyper-parameters

5.2 Evaluation metrics

Evaluating the performance of forecasting models is paramount to understanding their efficacy and areas of improvement. Various metrics and evaluators are employed to assess the accuracy, precision, and reliability of predictions. Here, we discuss some of the pivotal evaluators used in our analysis:

- **Mean Square Error (MSE)**:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.1}$$

- **Root Mean Square Error (RMSE)**:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5.2}$$

- **Mean Absolute Error (MAE)**:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.3}$$

- **Mean Absolute Relative Error (MARE)**:

$$\text{MARE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{5.4}$$

Where $\hat{y}_i$ is the predicted value, y_i is the actual value and n is the number of observations.

5.3 Optimizing Look-back Period

In the process of configuring our forecasting models, a pivotal decision was the selection of the appropriate look-back period. The look-back period determines how many previous time steps the model should consider to make a future prediction. For our experiments, we specifically evaluated look-back periods of 6, 12, 18, 24, 36, and 48 time steps.

Given the temporal convolutional nature of the TCAN model, the dilation rate is intrinsically linked to the look-back period. As we varied the look-back, the dilation rate within the TCAN layers was adjusted to ensure that the model can effectively capture long-range dependencies and patterns in the data. Notably, with the change in dilation rate, there was also a corresponding change in the number of residual blocks in the model. This relationship between the look-back period, dilation rate, and the number of residual blocks is detailed in Table 5.2.

To empirically determine the optimal look-back period, we conducted experiments with the aforementioned look-back lengths. The performance for each was evaluated based on the MSE. The results are visualized in Fig.5.1, which plots the MSE against the different look-back periods for TCAN and LSTM. By analyzing this trend and pinpointing the look-back that minimized the MSE, we have chosen 24 the most suitable look-back for our forecasting tasks.

look back	dilation sequence	number of residual blocks
6	[1,2,4]	3
12	[1,2,4,8]	4
18	[1,2,4,8,16]	5
24	[1,2,4,8,16]	5
32	[1,2,4,8,16,32]	6
48	[1,2,4,8,16,32]	6

Table 5.2: dilation for various look back periods

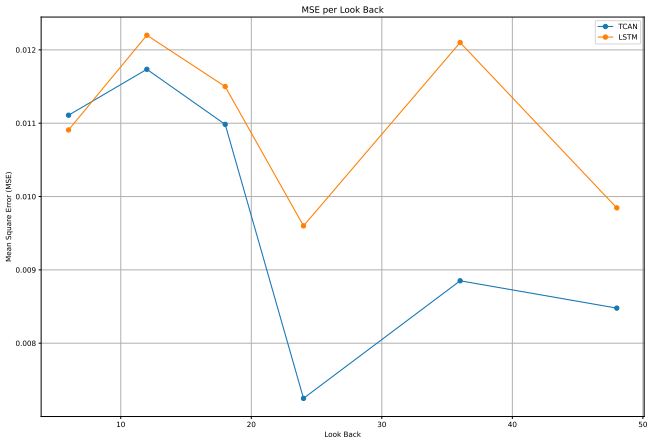


Figure 5.1: MSE per look-back

5.4 Benchmarking TCAN Against State-of-the-Art Models

In order to validate the performance of the TCAN model for time series forecasting, we compared it against two other prominent models: the LSTM network, and TCN. Building upon our earlier discussions, it's pivotal to reiterate that, post-SOM application, we identified 8 distinct winning neurons, each representing a unique traffic pattern. For each of these neurons, we trained a dedicated forecasting model using the corresponding neuron's weights. This approach ensured that each model was tailored to the specific characteristics and patterns inherent to the time series data associated with its respective neuron.

To provide a global view of our methodology's overall performance, we aggregated the results from all 8 models into single values. This aggregation offers a consolidated perspective, enabling us to interpret the overall efficacy of our approach Table 5.3 summarizes the empirical result of the proposed TCAN against TCN and LSTM. The implementation has been done in two scenarios. First, the weight of every winning Neuron is divided into training and validation. TCAN shows the best results in both scenarios, during the training phase, where the proposed results are MSE = 0.00.21, RMSE = 0.044, MAE = 0.033 and MARE = 0.123 for training. Then we calculated the mean value of all the metrics for the separate models. In the second scenario, the time series linked to each neuron (or cluster) was input into its corresponding model for testing purposes. The results of TCAN are MSE = 0.007, RMSE = 0.08, MAE = 0.061 and AMARE = 0.347.

	MSE(std)	**RMSE(std)**	**MAE(std)**	**MARE(std)**
			Training	
TCAN	**0.0021(0.0012)**	**0.044(0.012)**	**0.033(0.008)**	**0.123(0.06)**
TCN	0.0027(0.00130)	0.058(0.0115)	0.0387(0.0086)	0.164(0.084)
LSTM	0.0035(0.0014)	0.058(0.0124)	0.0438(0.00921)	0.127(0.87)
			Validation	
TCAN	**0.0028(0.00174)**	**0.051(0.0145)**	**0.038(0.0113)**	**0.121(0.068)**
TCN	0.0036(0.0022)	0.058(0.016)	0.045 (0.013)	0.173(0.118)
LSTM	0.0036(0.00162)	0.059(0.0129)	0.0499(0.0106)	0.126(0.0857)
			Testing	
TCAN	**0.007(0.0028)**	**0.08(0.017)**	**0.061(0.013)**	**0.347(0.623)**
TCN	0.011(0.064)	0.101(0.027)	0.0782(0.0217)	0.69(2.5)
LSTM	0.98(0.0041)	0.097(0.02)	0.075(0.016)	0.69(3.38)

Table 5.3: Empirical results of different forecasting models

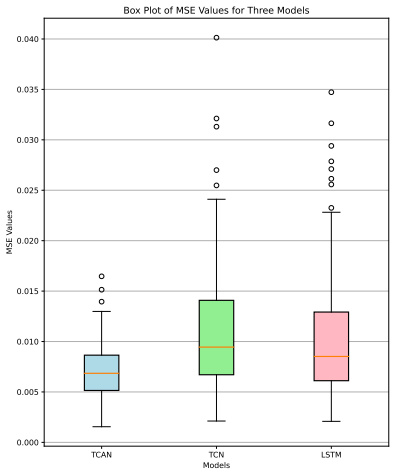


Figure 5.2: Boxplots of test's data MSE

Fig.5.2 displays the box-plots representing the MSEs of test predictions. It's evident that TCAN has a reduced median, as well as lower boundaries for both the upper and lower quartiles. Referencing Fig.5.3, we present boxplots that visualize the distribution of performance metrics for samples assigned to each neuron determined by the SOM clustering. These visualizations encompass results from the three forecasting models we've employed. As depicted in Fig.5.3, the boxplot for TCAN consistently outperforms others across all neurons. However, upon closer examination, the LSTM model demonstrates superior lower and upper quartiles, as well as a maximum value for Neuron (1,1). Furthermore, Neuron (3,1) under LSTM exhibits more favorable overall statistical measures compared to TCAN.

Scatter analysis is a graphical method used to depict the relationship between two sets of data points, often representing actual and predicted values. In the context of predictive modeling, the slope of the line of best fit (often denoted as m) in a scatter plot provides insight into the model's performance. A slope value of 1 indicates perfect prediction, where predicted values match the actual values. Values close to 1 suggest that the model's predictions are closely aligned with the actual outcomes, while values significantly different from 1 indicate potential discrepancies.

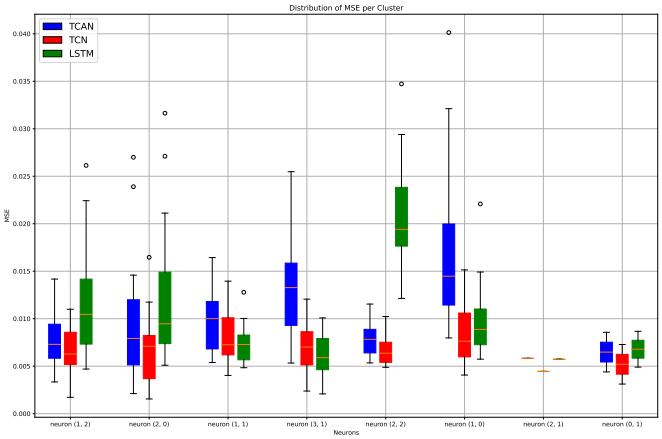
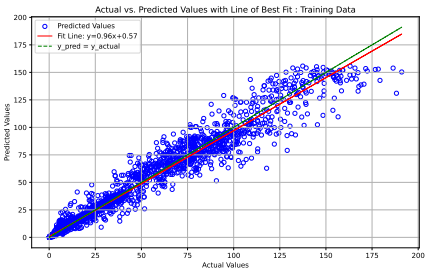
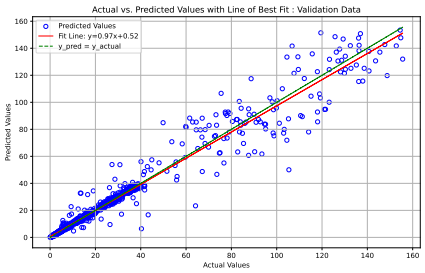


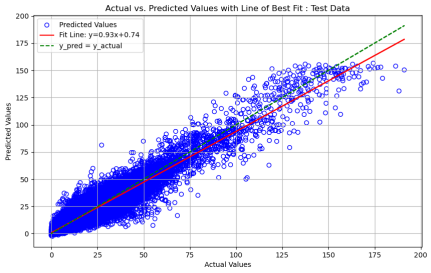
Figure 5.3: Boxplots of test's data MSE per cluster



(a) Training Scatter Plot



(b) Validation Scatter Plot



(c) Test Scatter Plot

Figure 5.4: Scatter plots for training, validation, and test data (TCAN).

For the analysis conducted with TCAN, the obtained slope values are quite promising. A slope of 0.93 for test values indicates that the model's predictions on unseen data are highly aligned with the actual outcomes, though there's a slight

deviation from the ideal. The training data, with a slope of 0.97, showcases an even closer alignment, suggesting that the model has learned the training patterns effectively. The validation results, with a slope of 0.96, further confirm the model's robustness, as it performs consistently on data it hasn't been trained on. These observations are visually evident in Fig.5.4. Overall, these results suggest that the model provides reliable predictions across different data subsets, with minor deviations from the ideal scenario.

5.5 SOM-TCAN vs. Baseline: A Comparative Analysis

In this section, we embark on a detailed comparison between our proposed integrated model, SOM-TCAN, and the traditional approach of employing individual LSTM models for each time series. Notably, each LSTM model was trained on a 60% data cut for training and tested on the remaining 40%. This is in contrast to the SOM-TCAN approach, where testing was conducted on all timesteps of the samples assigned to their respective cluster. While this might not be the ideal scenario, it provides a valuable insight into the performance of our model. The overarching goal of this comparison is to underscore the benefits of our hybrid model, both in terms of computational efficiency and forecasting accuracy, over the conventional method of treating each time series independently. Through this comparison, we emphasize the advantages of harnessing the inherent patterns and relationships in the data via SOM clustering, subsequently complemented by the forecasting capabilities of TCAN.

Turning our attention to accuracy metrics, Table 5.4 provides an aggregated perspective on the average and standard deviation of forecasting accuracy for both methodologies. An initial glance reveals that the SOM-TCAN model slightly edges out the individual LSTM models in terms of MSE, RMSE, MAE and MARE. Furthermore, SOM-TCAN demonstrates a reduced standard deviation across these metrics, suggesting a more consistent performance. This consistency might be attributed to the integrated nature of the SOM-TCAN approach, which capitalizes on the patterns identified by the SOM and the forecasting prowess of TCAN. On the other hand, the heightened standard deviation observed for the LSTM models could hint at certain time series having pronounced values. To further understand this observation, we propose visualizing the distribution of the MSE for both strategies using box plots, as depicted in Fig.5.5.

From the boxplots in Fig.5.5, the values between the LSTM baseline and

	Test			
	MSE	**RMSE**	**MAE**	**MARE**
SOM-TCAN	**0.0071(0.0028)**	**0.08(0.017)**	**0.061(0.013)**	**0.347(0.623)**
LSTM	0.0074(0.0041)	0.083(0.023)	0.061(0.018)	0.0.363(1.044)

Table 5.4: Comparative Mean and Std accuracy metrics of SOM-TCAN vs. Individual LSTM Forecasting

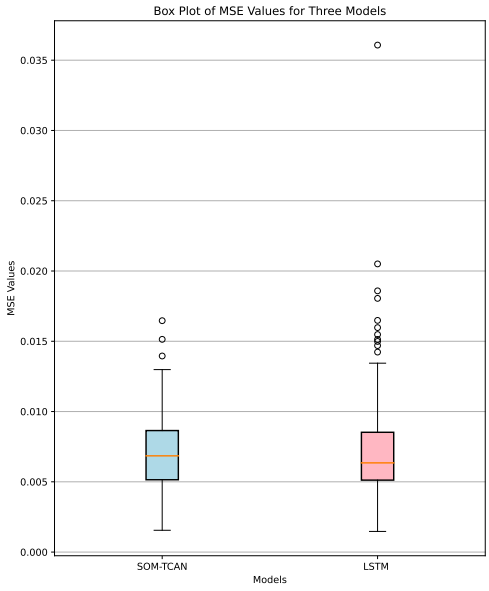


Figure 5.5: MSE' boxplot of SOM-TCAN vs. Individual LSTM forecasting

our SOM-TCAN model are largely comparable. However, the LSTM baseline is distinctly characterized by a higher maximum, suggesting a broader range of errors. This observation aligns with the higher standard deviation noted earlier, further emphasizing the variability in the LSTM baseline's performance.

Finally, we delve deeper into the performance nuances of our hybrid model by examining the boxplots for each time series associated with a specific neuron, as illustrated in Fig.5.6. This granular perspective allows us to directly compare the performance of the various TCAN models within our hybrid model against the results obtained from individually training each time series with an LSTM model. Upon visually inspecting Fig.5.6, it becomes evident that the boxplots are

largely comparable. Particularly in densely populated neurons containing many time series, such as (1,2), (2,0), (1,1), (3,1), (2,2), and (1,0), distinguishing between the performances of the two approaches becomes challenging. In some neurons, the LSTM based baseline model exhibits superior performance, while in others, the SOM-TCAN model takes the lead. This variability is further compounded by the fact that different aspects of the boxplots might favor different models, making a clear-cut determination difficult. This observation underscores the competitive nature of SOM-TCAN , even in regions with a high concentration of time series data.. Notably, for neurons (2,1) and (0,1), our hybrid model outperforms the LSTM. This superior performance is somewhat expected given that these neurons have only 1 and 2 time series assigned to them, respectively.

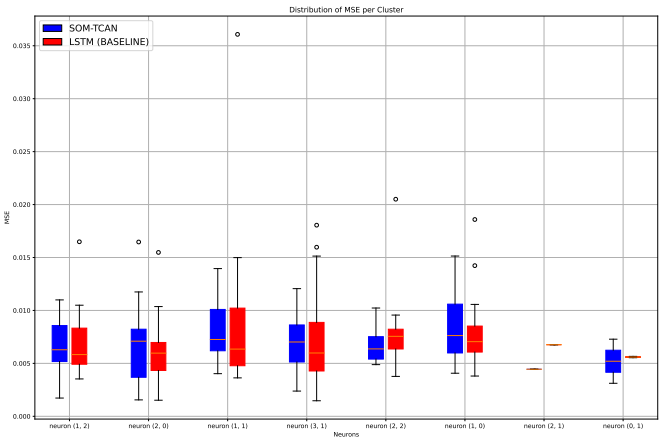


Figure 5.6: MSE' boxplot of SOM-TCAN vs. Individual LSTM forecasting across different clusters

Upon evaluating the prediction accuracy of both the SOM-TCAN and the baseline models, the results are notably close. However, the SOM-TCAN model slightly edges out the Baseline model. Even though the evaluation metrics are comparable SOM-TCAN shows a reduced standard deviation error, indicating a more consistent and narrower range of errors.

However, it's important to contextualize this comparison in terms of model complexity and scalability. As illustrated in Table 5.5, our SOM-TCAN model comprises a mere 8 models, each tailored to a specific cluster. In stark contrast, the Baseline approach necessitates the deployment of 112 distinct models, one for each individual time series. This disparity underscores the efficiency and compactness of our proposed hybrid model, especially when considering the negligible difference in prediction accuracy.

Table 5.5: Model Count Comparison: SOM-TCAN vs. Baseline

	SOM-TCAN	**BASELINE**
Number of Models	8	112

This comparison invites a deeper consideration of the balance between model simplicity and forecasting accuracy. While the SOM-TCAN might offer slightly better results, its efficiency and reduced complexity, make a strong argument for its use in practical scenarios, especially where scalability and ease of deployment are crucial. To illustrate, consider a scenario where the city decides to add new BS. With the Baseline approach, a new forecasting model would need to be developed for each new station. In contrast, with the SOM-TCAN model, the new BS's data can simply be plugged into the existing SOM. The BS would then be assigned to an existing cluster, and the forecasting model corresponding to that cluster can be immediately utilized. This exemplifies the inherent scalability and adaptability of the SOM-TCAN approach, making it more suitable for dynamic environments where the network infrastructure might evolve.

Chapter 6

Rethinking Deep Reinforcement Learning for Weighted Fair Queuing

One approach to dynamically manage and optimize the synergy between RANs and HAPS is the application of deep reinforcement learning. DRL offers the prospect of an intelligent, adaptive system capable of making real-time decisions that can adapt to fluctuating network conditions, ensuring optimal resource allocation and user experience.

However, as with many ventures, challenges abound. Representing the entire RAN-HAPS environment within a DRL framework is a task of formidable complexity. The environment encapsulates several variables: fluctuating traffic loads, diverse user demands, interference, infrastructure constraints, and the innate unpredictability of wireless channels, to name just a few. An accurate representation requires a delicate balance, too simplistic, and it fails to capture the nuances; too intricate, and it becomes computationally infeasible and slow to adapt.

It is within this context that we propose a shift in perspective, a rethinking of our DRL embedding. Instead of attempting to model the intricate web of the entire RAN-HAPS environment, could we narrow our focus to a more contained yet impactful aspect? This work delves into this concept, zooming in on the idea of tuning the weights of Weighted Fair Queuing (WFQ) within HAP using DRL. By refining our approach, we aspire to achieve the adaptability and efficiency promised by reinforcement learning, without the drawbacks of an overly complex

representation.

6.1 Weighted Fair Queueing (WFQ)

Demers et al [55]. developed the advanced WFQ algorithm, drawing inspiration from Nagle's work [56]. WFQ addresses congestion issues in datagram network gateways. This issue arises when network participants send data packets, with some consistently sending larger packets than their counterparts. Instead of the conventional FIFO (first in, first out) approach, WFQ systematically categorizes datagram sources based on packet size and allocates them to specific queues with set weights [57]. The decision on which packet to process next is based on its weight, size, and the queue's past activity. WFQ ensure fair bandwidth allocation among different flows while also allowing differentiated services based on assigned weights, which brings enhanced granularity and adaptability, an example of WFQ is depicted in Fig.6.1.

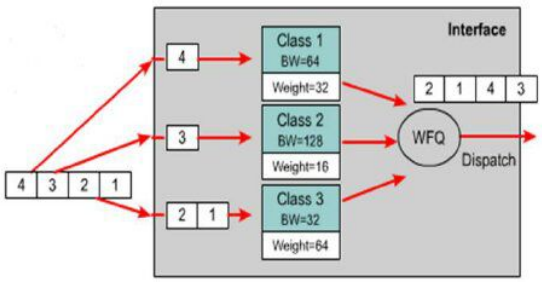


Figure 6.1: WFQ

At its core, WFQ is designed to ensure that each flow gets its fair share of the bandwidth, as defined by its weight. If there are N flows, each with an assigned weight w_i, the fraction of the link bandwidth that flow i should receive is given by:

$$f_i = \frac{w_i}{\sum_{j=1}^{N} w_j} \tag{6.1}$$

where f_i is the fraction of the link bandwidth for flow i. Each arriving packet is timestamped based on its expected departure called **virtual finish time**. This timestamping ensures that packets from a flow with a higher weight (i.e., higher

priority) get an earlier **finish time** and hence are scheduled before packets from flows with lower weights.

$$FT_{i,k} = \max(AT_{i,k}, FT_{i,k-1}) + \frac{L_{i,k}}{f_i} \tag{6.2}$$

where $FT_{i,k}$ is the virtual finish time of k^{th} packet of flow i , AT_i is the arrival time of packet and $L_{i,k}$ is the length in bits of the k^{th} in flow i.

Determining the optimal weights for flows remains an ambiguous task, as there isn't a universally accepted method for their selection. The choice of weights often relies on the specific network parameters at hand. Moreover, each fairness metric introduced in prior research calls for a distinct approach to weight allocation. An RL agent can be employed to learn and adapt these weights dynamically, optimizing the network's performance based on ongoing conditions and objectives.

6.2 Rule-Based Systems for Traffic Prioritization in Terrestrial-HAPS Integration

As we delve deeper into the integration of terrestrial networks with HAPs, efficient traffic management emerges as a pivotal concern. This challenge encompasses not only handling the sheer data volume but also intelligently prioritizing it to ensure optimal resource utilization and an enhanced user experience. This brings us to the concept of a Rule-Based System for Traffic Prioritization. Leveraging the forecasting capabilities of our SOM-TCAN framework, we aim to segment the forecasted traffic into distinct priority levels: low, mid, and high traffic. These segments serve as the guiding hand for directing traffic into the appropriate queues within the WFQ system, ensuring that packets are forwarded based on their priority level. In the subsequent sections, we will elucidate three strategies meticulously designed to manage and allocate traffic, optimizing the network for varying traffic intensities and user demands.

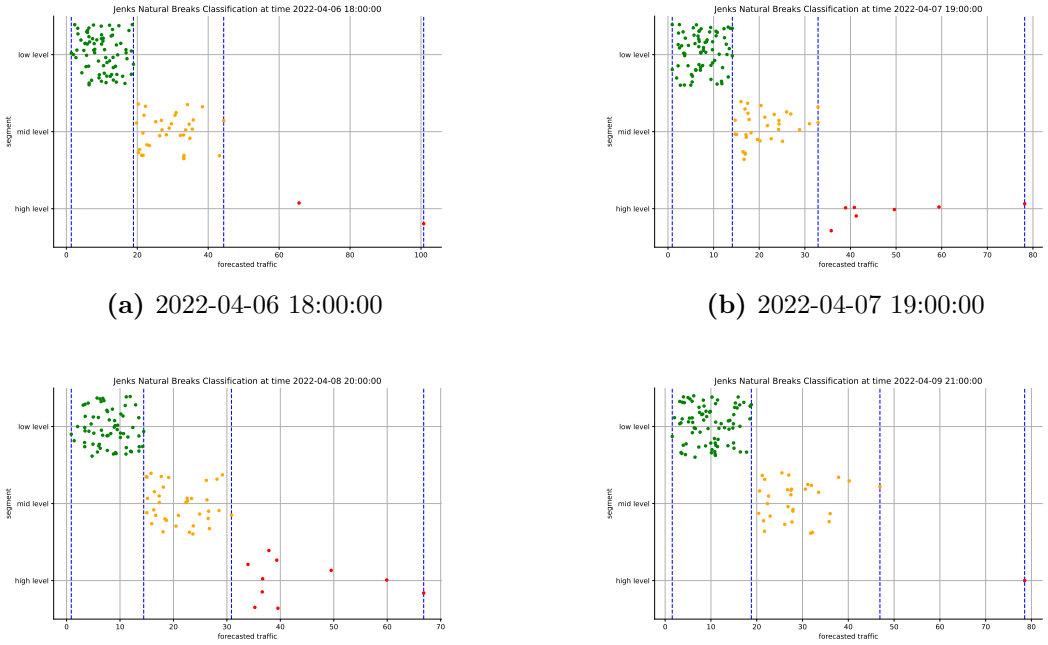
- **Low Priority level:** One of the primary strategies we propose is the **RoD** approach proposed in [13]. The core idea behind this strategy is to optimize network energy consumption, especially during periods of low traffic. In traditional terrestrial networks, BS continue to operate even when the traffic demand is minimal, leading to unnecessary energy consumption. With the

RoD strategy, when traffic falls below a specified threshold, the corresponding base station is turned off. Instead of leaving the users in that area without service, their traffic is seamlessly forwarded to the HAPS. This not only conserves energy but also ensures uninterrupted service for the users.

- **High Priority level:** Another pivotal strategy we introduce focuses on enhancing the QoS while simultaneously optimizing the energy consumption of the base stations. In many scenarios, base stations can become overwhelmed due to sudden surges in traffic demand or deteriorating performance metrics. Such situations can lead to degraded service quality, manifested as increased packet drops or prolonged wait times for connections. To counteract this, our strategy monitors both the traffic demand and KPIs such as packet drop rates and connection wait times. If a base station exceeds a predefined threshold for these metrics, it indicates potential service degradation. Instead of allowing the base station to continue operating under high demand, a portion of its traffic is offloaded to the HAPS. This offloading not only alleviates the load on the base station, ensuring better QoS for its users, but also optimizes its energy consumption by reducing the processing demands.

- **Middle Priority level:** Given that the HAPS can harness solar energy, it is at its peak power capacity during the afternoon. This provides an opportune window to offload traffic from terrestrial base stations to the HAPS. The forecasted traffic is segmented into three clusters : low, mid, and high traffic. For the mid-traffic cluster, we can identify base stations whose traffic is increasing at a rate of α compared to previous periods. Since these stations are experiencing a noticeable increase in demand but might not yet be at their capacity limits, they are ideal candidates for partial offloading to the HAPS. This ensures that these stations don't get overwhelmed if the traffic surge continues.

as visualized in Fig.6.2. For this clustering of 1-dimensional data, we employed the Jenks Natural Breaks Classification Method. It is a technique for grouping data to find the optimal classification of values into distinct categories. The goal is to minimize the average difference of values within each category from its category mean, while also maximizing the difference of each category from the means of other categories [58] .



(a) 2022-04-06 18:00:00

(b) 2022-04-07 19:00:00

(c) 2022-04-08 20:00:00

(d) 2022-04-09 21:00:00

Figure 6.2: Clustering of forecasted traffic across various hours and days

From the plot, it's evident that the high-traffic cluster comprises fewer data points. Given this observation, it's feasible to offload some of their traffic to the HAPS. Even though it's the afternoon and terrestrial base stations might have stored energy from solar panels, redirecting traffic to the HAPS is advantageous. The HAPS, due to its elevated altitude, potentially has a larger energy storage capacity. Furthermore, if the BSs are connected to the power grid, offloading traffic to HAPS can help in reducing the load on the grid, leading to more efficient energy utilization.

However, it's worth noting that this strategy is aggressive in its approach. Several parameters, such as the cost of energy, consumption metrics per base station, and the specific energy storage capabilities of the HAPS, need careful consideration. While the strategy provides a foundational approach, future work should delve deeper into these parameters. Through rigorous research and analysis, the strategies can be fine-tuned to achieve optimal results while ensuring sustainability and efficiency.

6.3 Reinforcement Learning

Reinforcement Learning (RL) is a subfield of machine learning that focuses on training agents to make sequences of decisions by interacting with an environment. Unlike supervised learning, where the model is trained with explicit input-output pairs, RL operates in situations where the correct decision (or action) isn't known in advance. Instead, an agent learns from trial and error, receiving feedback in the form of rewards or penalties. Fig.6.3 visually represents the interacting components.

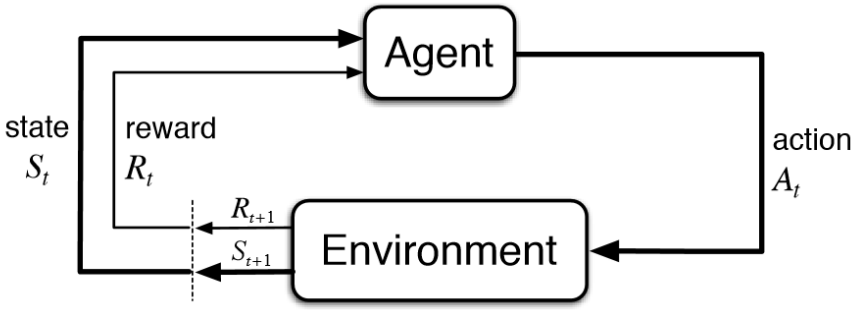


Figure 6.3: Learning cycle of RL

At its core, RL involves several key components [59]:

1. **Agent:** The decision-maker that interacts with the environment.
2. **Environment:** The external system the agent interacts with.
3. **State:** A representation of the environment at a particular point in time.
4. **Action:** A decision or move made by the agent.
5. **Reward:** Feedback received after taking an action in a particular state.

like after-mentions , the aim of the agent is to maximize the reward which is expressed :

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{6.3}$$

where γ is the discount factor. To express the anticipated total future reward after taking an action in state s_t , it's beneficial to introduce the concept known as

the Q-function:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[R_t + \gamma \sum_{a'} \pi(a'|s') Q^{\pi}(s',a') | S_t = s, A_t = a\right] \tag{6.4}$$

The Q-function is a fundamental concept in reinforcement learning as it provides a mechanism to estimate the value of an action without having to know all the future states and rewards. To determine the best action to take in a given state, one would typically employ a policy that maximizes the Q-function. This policy, often denoted as π^* , is defined as:

$$\pi^*(s) = \arg\max_a Q(s,a) \tag{6.5}$$

A fundamental challenge in RL is the trade-off between exploration and exploitation. Exploration involves trying out new actions to discover their effects, while exploitation means choosing actions that are known to yield good rewards. Striking the right balance is crucial for effective learning.

There are various algorithms and methods in RL, but two of the most prominent are Q-learning and policy gradient methods:

- **Q-learning:** This is a value-based method where the agent learns a value function that estimates the expected cumulative reward of taking an action in a given state. The Q-value function is updated iteratively based on the Bellman equation.

- **Policy Gradient:** instead of learning a value function, policy gradient methods directly optimize the policy. These methods estimate the gradient of the expected reward concerning the policy parameters and adjust the policy in the direction of increasing rewards.

Finally, with the advent of deep learning, neural networks have been integrated into RL, leading to what's known as DRL. DRL methods, like Deep Q-Networks (DQN), combine the representational power of deep neural networks with traditional RL algorithms as illustrated in Fig.6.4, allowing agents to handle environments with high-dimensional state spaces, like video games or robotic control tasks.

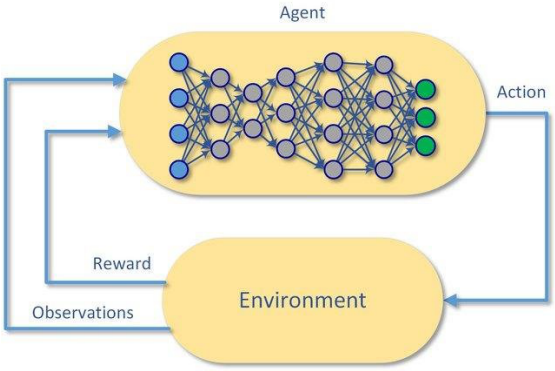


Figure 6.4: Learning cycle of DRL

6.4 DRL-WFQ

In the evolving landscape of RANs, optimizing traffic management has taken center stage. A prominent advancement in this realm has been the incorporation of HAPS to improve RANs' operations. There have been proposals to utilize DRL models to harness the potential of these HAPS in augmenting RAN performance [60, 55]. While this methodology has its merits, it simultaneously faces a series of complexities and challenges. To address these intricacies, our perspective shifts towards refining the internal mechanisms of HAPS, specifically targeting the WFQ system as depicted in Fig.6.5. By shifting our focus on the WFQ within the HAPS, we aim to simplify the problem, offering a more direct approach to enhancing network traffic management. The approach in question might provide a couple of advantages :

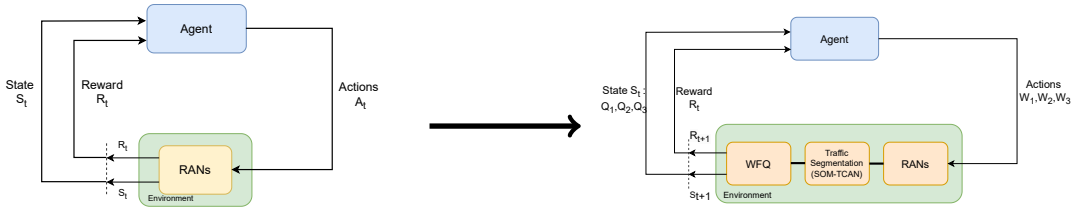


Figure 6.5: from DRL-RANs to DRL-WFQ

1. **Simplifying the Problem Space**: Leveraging HAPS to aid RANs inherently broadens the problem space. It encompasses not just the communication between user equipment and BSs but also the coordination, communication, and management of HAPS. By shifting the focus of DRL from managing HAPS

to directly tuning WFQ weights in RANs, we can narrow down and target a specific optimization problem: ensuring efficient and fair traffic management.

2. **Cost-Effectiveness and Enhanced Qos**: Optimizing WFQ within HAPS not only brings about an enhanced QoS but also offers significant financial savings. During peak demand periods, a substantial fraction of the traffic can be managed by HAPS, reducing the pressure on groud BSs. This removes the immediate need to deploy additional BSs, which can be a substantial investment both in terms of finances and logistics. By ensuring efficient traffic management through WFQ in HAPS, service providers can achieve better user experience while also optimizing their infrastructure investments.

In the context of our WFQ system, the DRL agent continuously interacts with its environment, making decisions, observing outcomes, and refining its strategies. As we mentioned in Section.6.3, this interaction cycle can be broken down into four main components: actions taken by the agent, the environment in which these actions are executed, the states that provide the agent with context, and the rewards that guide the agent's learning. Each of these components is critical to the agent's effective operation and learning trajectory.

6.4.1 States

In DRL, the state provides an agent with a snapshot of its environment, offering context to make informed decisions. For a DRL agent working within a WFQ system, the state must accurately capture the current dynamics of the priority queues. Various state representations can be considered, The following are some proposed state representation for each queue:

- **Queue Lengths**: The number of packets in each of the three priority queues.
- **Average Wait Time**: Average time the packets have been waiting in the queue. This can help in understanding congestion or potential for packet drops due to timeout.
- **Incoming Traffic Rate**: The rate at which packets are arriving at each queue. This could give the agent an indication of impending congestion or potential queue starvation.
- **Service Rate**: The rate at which packets are being processed and leaving each queue. This can give insights into how well the system is coping with the incoming traffic.

- **Packet Drop Rate**: If any packets are being dropped due to a full queue, this could be an important metric to consider.

The state representation should ideally encompass the attributes of the packets residing in the queues. One significant metric to consider is the **Packet Size Distribution**. This involves capturing both the average size of the packets and its variance within a queue. Nevertheless, incorporating such details might introduce additional complexity to the model.

6.4.2 Rewards

Designing a suitable reward function is crucial for the effective training of reinforcement learning agents. Given the context an agent aiming to adjust the weights for WFQ which contains three priority queues , the following are some potential reward mechanisms:

1. **Queue Stability**:
   - **Positive Reward**: If none of the queues overflow in a given time step.
   - **Negative Reward**: For every packet that's dropped due to an overflow.
2. **Queue Delay**:
   - **Positive Reward**: If the average waiting time of packets in each queue is below a certain acceptable threshold.
   - **Negative Reward**: Deduct from the reward based on how much the average waiting time exceeds the threshold.
3. **Priority Adherence**:
   - **Positive Reward**: If high-priority queues are indeed given more bandwidth and low-priority queues are not starved.
   - **Negative Reward**: If a low-priority queue gets more bandwidth than a higher-priority queue beyond acceptable limits.
4. **Throughput**:
   - **Positive Reward**: For every packet successfully processed and sent out from the queues.

- **Negative Reward**: If the throughput falls below a certain threshold, indicating potential underutilization or inefficiency.

It's worth noting that these reward strategies can be used independently, tailoring the agent to focus on a specific aspect of the WFQ system. Alternatively, multiple reward strategies can be combined, allowing the agent to learn and optimize based on a compound set of criteria. The combination of two or more rewards can provide a more general view, but care should be taken to balance the relative importance of each reward to prevent overshadowing or conflicting objectives.

6.4.3 Actions

Finally, the actions determine how the agent modifies or sets the weights for the queues. For our proposed design, the action space is represented by three continuous values, each ranging from 0 to 1. These values correspond to the weight assignments for specific priority queue. By constraining these action values within this range, we maintain a standardized framework for weight assignments. The DRL agent, through its interactions with the environment, will learn to select optimal values within this range to achieve the desired balance and efficiency in the WFQ system.

Nevertheless, Using a range of 0 to 1 for action values in a DRL framework tailored for WFQ optimization presents both benefits and challenges. The simplicity of this range is undeniably attractive, making it easy to implement and interpret. Coupled with certain activation functions like sigmoid in ANNs, this range facilitates stable training due to natural normalization. Furthermore, the relative differences between values in this range could be more essential if the weights function as "priority levels". However, potential issues arise from this fixed boundary. The range might be restrictive if the WFQ system requires weights with broader flexibility or if weights have meaningful absolute values for exemple bandwdwidth alocation in *Mbps*. Thus, while 0 to 1 is a commonly used range, aligning it effectively with the specifics of the WFQ system is crucial to ensure optimal outcomes.

Chapter 7

Conclusion

The field of wireless communication is currently experiencing a significant shift, with the integration of RANs and HAPS being a central focus. This research has delved into the challenges and opportunities presented by this integration, offering fresh solutions and insights to navigate its complexities.

Central to our exploration is the SOM-TCAN. This hybrid model brings together the strengths of Self-Organizing Maps and Temporal Convolutional Attention Networks. Designed specifically for traffic forecasting, this hybrid model showcases the benefits of combining unsupervised learning techniques with the power of deep learning. By effectively segmenting and forecasting traffic, SOM-TCAN provides a foundation for more informed decision-making in the dynamic RAN-HAP environment.

Our comparative analyses highlighted the effectiveness of TCAN when pitted against other forecasting models. When we compared the results with the baseline model, it became clear that there's a balance to strike between the simplicity of a model and its forecasting accuracy. While the baseline model showed results that were closely matched in some areas, SOM-TCAN stood out in terms of its efficiency, reduced complexity, and potential for scalability. This makes it a strong contender for practical applications, especially in scenarios where the network infrastructure is constantly evolving and adapting to new challenges.

Recognizing the potential of DRL in this domain, we revisited its application, specifically focusing on WFQ in RANs. The traditional approach of leveraging HAPS, while promising, introduced layers of complexity. Our proposition was to narrow our focus, targeting the optimization of WFQ weights within a HAPS

using DRL. This shift not only simplified the problem space but also offered direct avenues for enhancing performance.

Moving forward, there are numerous avenues to explore. Exploring other forecasting models, such as N-BEATS [61] and DeepAR [62], will not only enrich our comparative analysis but also offer a broader framework to assess the efficacy of TCAN. The availability of energy consumption data could pave the way for innovative strategies, optimizing both performance and energy efficiency. And in the vast expanse of DRL, finding the optimal model remains an exciting endeavor, promising further enhancements in our approach.

Bibliography

[1] Rahmani et al. «Next-Generation IoT Devices: Sustainable Eco-Friendly Manufacturing, Energy Harvesting, and Wireless Connectivity». In: (Jan. 2023). URL: https://ieeexplore.ieee.org/ielx7/9171629/10007536/10007697.pdf?tag=1&tag=1 (cit. on p. 1).

[2] Qualcomm Inc. «"Intelligently connecting our world in the 5G era," May 2020.» In: (May 2020). URL: https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/intelligently_connecting_our_world_in_the_5g_era_web_1.pdf (cit. on p. 1).

[3] Cisco. «Cisco Annual Internet Report (2018–2023) White Paper». In: (Mar. 2020). URL: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html (cit. on p. 2).

[4] Erricson. «Ericsson Mobility Report». In: (Nov. 2022). URL: https://d110erj175o600.cloudfront.net/wp-content/uploads/2022/11/29180338/Ericsson-Mobility-Report-November-2022.pdf (cit. on pp. 2, 3).

[5] Mohammed H. Alsharif and Rosdiadee Nordin. «Evolution towards fifth generation (5G) wireless networks: Current trends and challenges in the deployment of millimetre wave, massive MIMO, and small cells». In: *Telecommunication Systems* (Apr. 2017). DOI: 10.1007/s11235-016-0195-x (cit. on p. 4).

[6] GSMA. «5G Spectrum - GSMA Public Policy Position». In: (June 2022). URL: https://www.gsma.com/spectrum/wp-content/uploads/2022/06/5G-Spectrum-Positions.pdf (cit. on p. 4).

[7] Sacha Kavanagh. «What is 5G New Radio (5G NR).» In: (Mar. 2020). URL: https://5g.co.uk/guides/what-is-5g-new-radio/ (cit. on pp. 4, 5).

[8] Zhiheng Guo Lei Wan and Xiang Chen. «Enabling Efficient 5G NR and 4G LTE Coexistence». In: (Mar. 2019). DOI: 10.1109/MWC.2019.8641417 (cit. on p. 4).

[9] Samsung. «Massive MIMO for New Radio, White Paper». In: (Dec. 2020). URL: https://images.samsung.com/is/content/samsung/p5/global/business/networks/insights/white-papers/1208_massive-mimo-for-new-radio/MassiveMIMOforNRTechnicalWhitePaper-v1.2.0.pdf (cit. on p. 6).

[10] ITU. «5G - Fifth generation of mobile technologies». In: (Dec. 2019). URL: https://www.itu.int/en/mediacentre/backgrounders/Pages/5G-fifth-generation-of-mobile-technologies.aspx (cit. on p. 6).

[11] Behnaam Aazhang et al. «Key drivers and research challenges for 6G ubiquitous wireless intelligence (white paper)». In: (Sept. 2019) (cit. on p. 7).

[12] Ward Heddeghem et al Sofie Lambert. «Worldwide electricity consumption of communication networks». In: *Optics express 20* (Dec. 2012). DOI: 10.1364/OE.20.00B513 (cit. on p. 7).

[13] M. Meo M. Dalmasso and D. Renga. «Radio resource management for improving energy self-sufficiency of green mobile networks». In: *ACM SIGMETRICS Performance Evaluation Review* 44 (2016) (cit. on pp. 7, 59).

[14] Samsung Research. «6g the next hyper connected experience for all (White Paper)». In: (2021). URL: https://cdn.codeground.org/nsr/downloads/researchareas/20201201_6G_Vision_web.pdf (cit. on pp. 7, 9).

[15] Wei Chen et al Khaled B. Letaief. «The Roadmap to 6G: AI Empowered Wireless Networks». In: *IEEE Communications Magazine* (2019) (cit. on pp. 7, 8, 11).

[16] Orange. «Orange's vision for 6G (White Paper)». In: (Mar. 2022). URL: https://hellofuture.orange.com/app/uploads/2022/03/White-Paper-%E2%80%93-Oranges-vision-for-6G-%E2%80%93-March-2022.pdf (cit. on p. 8).

[17] ICNIRP. «RF EMF GUIDELINES 2020». In: (Mar. 2020). URL: https://www.icnirp.org/en/activities/news/news-article/rf-guidelines-2020-published.html (cit. on p. 9).

[18] Michele Polese et al Marco Giordani. «Toward 6G Networks: Use Cases and Technologies». In: *IEEE Communications Magazine* (Mar. 2020). URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9040264 (cit. on p. 10).

[19] Shiva Raj Pokhrel and Jinho Choi. «Understand-Before-Talk (UBT): A Semantic Communication Approach to 6G Networks». In: (Sept. 2022). URL: https://arxiv.org/pdf/2210.06943.pdf (cit. on pp. 11, 12).

[20] Xiang Peng et al. «A Robust Deep Learning Enabled Semantic Communication System for Text». In: (June 2022). URL: https://arxiv.org/pdf/2206.02596.pdf (cit. on p. 12).

[21] M. Di Renzo et al. «Smart radio environments empowered by reconfigurable AI meta-surfaces: an idea whose time has come». In: (2019) (cit. on p. 12).

[22] E. Calvanese Strinati et al. «Reconfigurable, intelligent, and sustainable wireless environments for 6G smart connectivity». In: 59 (Oct. 2021) (cit. on p. 12).

[23] D. -T. Phan-Huy N. Awarkeh and R. Visoz. «Electro-Magnetic Field (EMF)-aware beamforming assisted by Reconfigurable Intelligent Surfaces». In: (2021) (cit. on p. 12).

[24] D. -T. Phan-Huy et al. «Ambient Backscatter Communications in Mobile Networks: Crowd-Detectable Zero-Energy-Devices». In: (2021) (cit. on p. 12).

[25] Md Sahabul Alam et al. «High Altitude Platform Station based Super Macro Base Station (HAPS-SMBS) Constellations». In: (Sept. 2020). URL: https://arxiv.org/pdf/2007.08747.pdf (cit. on p. 14).

[26] Omid Abbasi et al. «HAPS for 6G Networks: Potential Use Cases, Open Challenges, and Possible Solutions». In: (Apr. 2023) (cit. on p. 13).

[27] Aaron Courville et al Yoshua Bengio. «Representation Learning: A Review and New Perspectives». In: (Apr. 2014) (cit. on p. 15).

[28] Yoshua Bengio Ian Goodfellow and Aaron Courville. «Deep learning». In: (2016), p. 800 (cit. on p. 16).

[29] Max Welling Diederik P Kingma. «Auto-Encoding Variational Bayes». In: (Dec. 2022). URL: https://arxiv.org/pdf/1312.6114.pdf (cit. on p. 16).

[30] Tomas Mikolov et al. «Efficient Estimation of Word Representations in Vector Space». In: (Sept. 2013). URL: https://arxiv.org/pdf/1301.3781.pdf (cit. on p. 16).

[31] Y.Bengio Yann LeCun and Geoffrey Hintonn. «Deep learning». In: (Oct. 2015). DOI: 10.1038/nature14539. URL: https://www.researchgate.net/publication/277411157_Deep_Learning (cit. on p. 16).

[32] J.F.Torres et al. «Deep learning for time series forecasting: A survey». In: *Big Data* 9 (2021) (cit. on p. 16).

[33] J. Zheng et al Q. Ma. «Learning representations for time series clustering,» in: *NeurIPS* (2019) (cit. on pp. 16, 18).

[34] Theyazn H.H Aldhyani et al. «Intelligent Hybrid Model to Enhance Time Series Models for Predicting Network Traffic». In: *IEEE* (July 2020). URL: https://ieeexplore.ieee.org/document/9380673 (cit. on pp. 17, 28).

[35] Qianwen Meng et al. «Unsupervised Representation Learning for Time Series: A Review». In: *Journal of Latex class files* 14 (8 2021) (cit. on pp. 17, 18).

[36] X. Zhan et al. «Online deep clustering for unsupervised representation learning». In: *CVPR. Computer Vision Foundation / IEEE* (2020) (cit. on p. 18).

[37] K. Song S. Shin and I. Moon. «Hierarchically clustered representation learning,» in: *AAAI. AAAI Press* (2020) (cit. on p. 18).

[38] J. Wang C. Zhang H. Fu. «Tensorized multi-view subspace representation learning». In: *Int. J. Comput. Vis* 128 (2020) (cit. on p. 18).

[39] Z. Zheng et al S. Zhou H. Xu. «"A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions». In: *CoRR* (2022) (cit. on p. 18).

[40] J. Zheng et al Q. Ma. «Learning representations for time series clustering». In: *NeurIPS* (2019) (cit. on pp. 18, 19).

[41] T. Kohonen. «The Self-organizing Map». In: *Proceedings of the IEEE* 78 (1990) (cit. on pp. 18, 20).

[42] H Yin. «The self-organizing maps: background, theories, extensions and applications». In: *Computational intelligence: A compendium* (2008) (cit. on p. 20).

[43] Cottrell et al de Bodt E. «On the use of self-organizing maps to accelerate vector quantization». In: *Neurocomputing* 56 (2004) (cit. on p. 20).

[44] Hochreiter S and Schmidhuber J. «Long short-term memory». In: *Neural computation* 9 (1997) (cit. on pp. 22, 24).

[45] Bai et al. «An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling». In: (2018) (cit. on pp. 22, 27).

[46] José Francisco Torres et al. «Deep Learning for Time Series Forecasting: A Survey». In: *Big Data* 9 (2020). DOI: 10.1089/big.2020.0159 (cit. on p. 23).

[47] Huiwei Zhou. «Exploiting syntactic and semantics information for chemical–disease relation extraction». In: *Database* (Apr. 2016). DOI: 10.1093/database/baw048 (cit. on p. 24).

[48] et al Salinas D. «DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks». In: (2017) (cit. on p. 25).

[49] Scarpa F. «An in-depth analysis of future sixth generation networks with integration of aerial platforms». In: (2021) (cit. on p. 31).

[50] Gunes Kurt et al. «A Vision and Framework for the High Altitude Platform Station (HAPS) Networks of the Future». In: *IEEE* (2020). URL: https://ieeexplore.ieee.org/document/9380673 (cit. on p. 36).

[51] J. et al Venna. «neighborhood preservation in nonlinear projection methods: An experimental study». In: (Sept. 2001). DOI: 10.1007/3-540-44668-0_68 (cit. on p. 40).

[52] Renzhuo Wan et al. «A Multivariate Temporal Convolutional Attention Network for Time-Series Forecasting». In: *Electronics* 11 (10). URL: https://doi.org/10.3390/electronics11101516 (cit. on p. 42).

[53] Yang Lin et al. «Temporal Convolutional Attention Neural Networks for Time Series Forecasting». In: (July 2021). DOI: 10.1109/IJCNN52387.2021.9534351 (cit. on p. 42).

[54] Cho et al Bahdanau D. «Neural Machine Translation by Jointly Learning to Align and Translate». In: *In Proceedings of the 3rd International Conference on Learning Representations (ICLR)* (2015) (cit. on p. 42).

[55] S. Shenker A. Demers S. Keshav. «Analysis and simulation of a fair queueing algorithm». In: (1989). URL: http://portal.acm.org/citation.cfm?doid=75247.75248 (cit. on pp. 58, 64).

[56] J. Nagle. «Congestion control in IP/TCP internetworks». In: 14 (4 Oct. 1984). URL: https://dl.acm.org/doi/10.1145/1024908.1024910 (cit. on p. 58).

[57] Tuncer Haslak. «Weighted Fair Queuing as a Scheduling Algorithm for Deferrable Loads in Smart Grids». In: *International Symposium on Energy System Optimization* (2020) (cit. on p. 58).

[58] George F Jenks. «The Data Model Concept in Statistical Mapping». In: *International Yearbook of Cartography* (July 1967), pp. 186–190 (cit. on p. 60).

[59] Richard S. Sutton and Andrew G. Barto. «Reinforcement Learning: An Introduction. Second». In: *The MIT Press* (2018) (cit. on p. 62).

[60] Yuejiao Xie et al. «Online Bipartite Matching for HAP Access in Space-Air-Ground Integrated Networks using Graph Neural Network-Enhanced Reinforcement Learning». In: *IEEE International Conference on Communications* () (cit. on p. 64).

[61] Boris N. Oreshkin et al. «N-BEATS: Neural basis expansion analysis for interpretable time series forecasting». In: (May 2019) (cit. on p. 69).

[62] Valentin Flunkert David Salinas and Jan Gasthaus. «DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks». In: (Feb. 2019) (cit. on p. 69).