ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN

INGEGNERIA ELETTRONICA, TELECOMUNICAZIONI E TECNOLOGIE DELL'INFORMAZIONE

Ciclo 38

Settore Concorsuale: 09/F2 - TELECOMUNICAZIONI

Settore Scientifico Disciplinare: ING-INF/03 - TELECOMUNICAZIONI

ARTIFICIAL INTELLIGENCE TECHNIQUES FOR THE PHYSICAL LAYER OF NON-TERRESTRIAL NETWORKS

Presentata da: Bruno De Filippo

Coordinatore Dottorato

Davide Dardari

Supervisore

Alessandro Vanelli Coralli

Co-supervisore

Alessandro Guidotti

Esame finale anno 2026

Abstract

The upcoming 6th Generation (6G) of wireless communication systems, standardized under the International Mobile Telecommunications 2030 (IMT-2030) framework, aims to extend the capabilities of 5G by enabling immersive, intelligent, and ubiquitous connectivity. Building upon the foundations of 5G, 6G envisions not only incremental performance gains but a paradigm shift driven by two key technological enablers: Artificial Intelligence (AI) and Non-Terrestrial Networks (NTNs). These paradigms are at the basis of the vision of adaptive, data-driven, and globally connected communication systems capable of supporting intelligent and immersive services across all environments. Deep Learning (DL) represents a fundamental departure from traditional model-based wireless design, introducing the ability to learn, infer, and optimize directly from data. Within this context, the 3rd Generation Partnership Project (3GPP) has launched several initiatives to incorporate AI into the New Radio (NR) air interface, focusing on areas such as data collection, model training, and architectural standardization. Notably, studies such as 3GPP TR 38.843 have explored the use of AI for channel state information (CSI) feedback, beam management, and positioning, demonstrating its potential to enhance link-level performance by learning real-world propagation characteristics and mitigating imperfections such as channel aging and synchronization errors. These efforts mark the beginning of a transition toward AI-native radio access networks (RANs), where PHY algorithms can self-optimize and adapt to diverse deployment scenarios. In parallel, NTNs, encompassing satellite, aerial, and high-altitude platform segments, extend the communication infrastructure beyond terrestrial boundaries, enabling seamless global coverage and integration across heterogeneous network layers. Their inherently dynamic topologies, long propagation delays, and hardware constraints, however, present new challenges for physical layer design and optimization. In this context, the application of AI to the PHY of NTNs emerges as a natural convergence point between these two enabling paradigms. By leveraging AI's capacity for pattern recognition, prediction, and decision-making, PHY algorithms can be tailored to the unique characteristics of non-terrestrial channels, enabling a more efficient resource utilization.

This thesis investigates how AI can enhance PHY functionalities in NTNs, with a particu-

lar focus on improving throughput, reliability, and spectral efficiency under the constraints imposed by non-stationary environments and limited computational resources. Three main research directions are addressed: 1) demodulation of Non-Orthogonal Multiple Access (NOMA) schemes, 2) channel prediction for dynamic and resource-constrained NTN scenarios, and 3) equalization of Faster-than-Nyquist (FTN) signaling for bandwidth-efficient transmissions. Each topic is approached through the design and evaluation of DL architectures, emphasizing their ability to learn from realistic channel statistics and approximate optimal PHY behavior while maintaining feasible computational complexity. The first contribution presents a Convolutional Neural Network (CNN)-based receiver for Sparse Code Multiple Access (SCMA), exploiting multi-task learning to specialize different demodulation branches for distinct user codebooks. The proposed model improves the block error rate and aggregated user throughput in NTN-based Multi-Access Edge computing (MEC) systems. The second and third contributions address the challenge of channel prediction: one focuses on cell-free multiple input multiple output NTN systems, where a lightweight DL model predicts temporal channel evolution to mitigate channel aging and enhance beamforming alignment; the other explores frequency-domain channel prediction in OFDM systems to reduce pilot overhead, demonstrating throughput gains in both line-of-sight and non-line-of-sight conditions and real-time inference capabilities under NTN power constraints. The same prediction framework is further extended to NOMA scenarios, enabling the estimation of overlapping users' channels through inter-slot prediction. Finally, this thesis investigates CNN-based equalizers for FTN signaling, showing that DL models can effectively cope with severe intersymbol interference, thus supporting high-throughput single-carrier communications.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The 6th Generation (6G) of wireless communication standards, officially known as International Mobile Telecommunications (IMT)-2030, builds on the foundations of IMT-2020 by extending existing scenarios and introducing new ones to support emerging technologies and applications [1]. Immersive communication evolves from enhanced Mobile Broadband (eMBB) to provide richer interactive experiences requiring synchronized video, audio, and environmental data with high reliability, low latency, and large capacity. Hyper reliable and low-latency communication pushes Ultra Reliable Low Latency Communications (URLLC) further, enabling industrial automation, emergency services, and telemedicine through stricter latency, reliability, and precision positioning requirements. Massive communication expands massive Machine-Type Communications (mMTC) to support billions of Internet of Things (IoT) devices across smart cities, agriculture, energy, and logistics, demanding high connection density, ultra-low power use, and robust security. Ubiquitous connectivity focuses on bridging the digital divide by enhancing coverage in rural and remote regions, enabling both IoT and broadband access where current networks fall short. Beyond these evolutions, IMT-2030 also introduces Artificial Intelligence (AI) and communication, integrating distributed AI, digital twins, and computational offloading across networks to enable autonomous and intelligent services. Integrated sensing and communication adds a new dimension by embedding sensing into IMT systems, supporting applications such as navigation, environmental monitoring, and activity detection through high-precision localization, imaging, and mapping. To enable such advancements, new, fundamental technologies are required. In particular, several IMT-2030 usage scenarios are enabled by the introduction of Non-Terrestrial Networks (NTNs) and AI [2].

1.1 Motivations

To better understand the motivation for this thesis, two main aspects should be highlighted:

1. How can the physical layer (PHY) benefit from AI? And
2. How can NTNs benefit from AI?

1.1.1 AI for PHY

The current decade has been characterized by the rise of AI as a key enabling technology in several fields, spanning from the automotive industry (*e.g.*, autonomous driving [3]) to the biomedical sector (*e.g.*, automatic cancer detection and classification [4]), and the widespread adoption of general-purpose Large Language Models (LLMs). At the basis of the multi-disciplinary success of AI is are the signal processing capabilities of Deep Neural Networks (DNNs), which allow pattern recognition and prediction on data of different nature and origin, including images, text, and historical sensor data. Clearly, this potential can serve well in the context of communication systems and, in particular, to the physical layer. Traditional algorithms typically rely on theoretical frameworks that neglect real-world effects, *e.g.*, imperfect time and frequency synchronization, resulting in suboptimal performance; on the opposite, Deep Learning (DL) models can learn the real data statistics and approximate the optimal receiver's performance. In this context, several AI-related activities have been carried by 3rd Generation Partnership Project (3GPP) working groups, spanning from data collection [5] to architectural aspects for model training [6]. A recent study led by the radio access network (RAN) 1 3GPP working group focused on the application of AI techniques to the 5G New Radio (NR) air interface in Release 18, resulting in the identification of three key use cases [7, 8]:

- **Channel State Information (CSI) feedback**. AI techniques are explored to enhance the CSI feedback between the gNB and a User Equipment (UE) through two main approaches: spatial-frequency domain CSI compression and time-domain CSI prediction. CSI compression uses a two-sided AI model, with encoding at the UE and decoding at the gNB, to reduce feedback data, particularly focusing on compressing the precoding matrix to align with NR's codebook-based framework. However, this setup introduces training and interoperability challenges across vendors, especially when new UE types are introduced. Conversely, time-domain CSI prediction employs a simpler one-sided model to mitigate channel aging caused by CSI reporting delays, while largely reusing the existing NR CSI framework. Further

studies within 3GPP will therefore focus on time-domain CSI prediction at the UE side.

- **Beam management**. Beamforming in NR requires time- and signal-intensive procedures for finding optimal downlink beam pairs. To address this, 3GPP has explored two AI-based approaches: spatial-domain downlink beam prediction, which predicts the best among one set of beams ('Set A') using measurements from another set ('Set B'), and temporal downlink beam prediction, which uses historical data from 'Set B' to forecast optimal beams in 'Set A' for future data exchanges. The AI models typically take layer 1 reference signal received power (L1-RSRP) measurements from 'Set B' as input and output the predicted top-K beams in 'Set A.' Model training and inference can occur at either the gNB or UE, with corresponding differences in what information each side reports.

- **Positioning**. Traditional 5G NR positioning relies on geometry-based methods that estimate an UE location from radio signal measurements, but their accuracy declines in weak line-of-sight (LoS) or dense multipath environments. To enhance performance, 3GPP has explored two AI-based approaches: direct AI positioning, which uses models to directly infer UE location from features like the Channel Impulse Response (CIR) or the Power Delay Profile (PDP), and AI-assisted positioning, where models estimate intermediate metrics such as LoS probability, Angle-of-Arrival (AoA), or Time-of-Arrival (ToA) to support positioning. Training and inference can occur at the UE, Location Management Function (LMF), or gNB, leading to three method categories: UE-based positioning, UE-assisted LMF-based positioning, and gNB-assisted positioning, differing in how each entity contributes to AI-driven location estimation.

Finally, it is worth mentioning that PHY algorithms are typically subject to runtime latency constraints in the order of the millisecond (*e.g.*, for channel estimation and channel decoding), representing strict computational complexity and hardware capability requirements to ensure a real-time inference.

AI for PHY: State-of-the-Art analysis

The areas of applications of AI to the RAN identified by 3GPP are only a few of those explored in the scientific literature, which, thanks to the development of DL techniques and specialized hardware, has flourished in the last decade. Key research areas include the following:

- **Orthogonal Frequency Division Multiplexing (OFDM)**. OFDM is the multiplexing technique at the basis of the current 5G standard waveform and that of its successor, 6G [9]. In OFDM, multiple symbols are carried by a pulse composed of N_{SC} superimposed orthogonal subcarriers, *i.e.*, tones at baseband frequencies that are integer multiples of the reciprocal of the pulse duration. Under ideal conditions, such design allows for a single tap equalization with user symbols being distributed over a grid-like structure, thus greatly simplifying the signal detection process. Nonetheless, real-world conditions, *e.g.*, strong multipath, imperfect time and frequency synchronization, have a detrimental effect on the receiver's capabilities: in such cases, DL techniques can be implemented to approximate the optimal receiver (which formulation may not be mathematically derivable). In this context, several DL-based approaches to the design of Cyclic Prefix (CP)-OFDM receivers have been proposed in the literature. The most promising algorithms are based on Convolutional Neural Networks (CNNs), exploiting temporal and spectral relationships in the OFDM resource grid. [10] proposed DeepRx, a CNN Residual Network (ResNet) for Single-Input Multiple-Output (SIMO) systems based on CP-OFDM, where a CNN performs the channel estimation, equalization, and signal detection stages. The input tensor is obtained by stacking 1) the received OFDM resource grid for each receiving antenna; 2) a resource grid that contains the value of transmitted Demodulation Reference Signals (DM-RSs), *i.e.*, pilot symbols transmitted for channel estimation purposes, in the position corresponding to such DM-RSs, and zeros elsewhere; and 3) as many resource grids as the number of receiving antennas, where each resource grid contains the channel estimate on each DM-RS position obtained by means of Least Squares (LS) and zeros elsewhere for the corresponding receiving antenna. This strategy encodes spatial information (in the DL sense, *i.e.*, temporal and spectral in the communications sense) in the input tensors, thus enhancing the effectiveness of convolution-based receivers. The base model, with over 1M parameters, outperforms the benchmark receiver, using LS channel estimation and Linear Minimum Mean Squared Error (LMMSE) equalization, on a variety of tests. However, the model's complexity is not deeply investigated, with only its linear Big-O complexity in terms of subcarriers and OFDM symbols being reported. The work was later extended to the general Multiple-Input Multiple-Output (MIMO) case in [11], showing similarly promising results. A MIMO receiver for CP-OFDM was also proposed in [12], employing Graph Neural Networks (GNNs) with CNN-based state updates to achieve multi-user and frequency-allocation-independent capabilities (*i.e.*, the model does not require a fixed input size and can be applied to a different number of subcarriers allocated to each user without changing its struc-

ture or retraining). The model's input is composed by the tensor representing the post Fast Fourier Transform (FFT) resource grid on each 5G layer and a positional encoding of the closest pilot's position with respect to each received symbol. The iterative receiver achieves a transport block error rate (BLER) worse by almost 1 dB with respect to a baseline using LMMSE-based channel estimation with K-best detection, but at a significantly lower complexity. An extended version was presented in [13] to support dynamic Modulation and Coding Scheme (MCS) configurations, assessing the tradeoff between inference latency and Signal-to-Noise Ratio (SNR) degradation, and exploring the possibility of site-specific adaptation by means of ray tracing. End-to-end learnable systems have also been investigated, *i.e.*, systems in which DNNs are employed not only at the receiver, but also at the transmitter to jointly learn how to map data on the OFDM resource grid, where and how to implement pilot signaling (*e.g.*, allowing pilots to be superimposed with data), and how to demodulate the received data. In this context, the same authors of [12] proposed a DNN-based transceiver design in [14], showing that superimposed pilots can lead to similar error rates as the pilot-based baseline while also achieving a 7% throughput gains through the increased data allocation.

- **Channel decoding**. Due to the presence of impairments to communication links (*e.g.*, Additive White Gaussian Noise (AWGN), Carrier Frequency Offset (CFO) and time offset due to imperfect synchronization), a statistically error-free link cannot be established. For this reason, data transmissions are encoded at the PHY to introduce redundancy in the sequence and improve the reliability of its transmission. 5G employs Low-Density Parity-Check (LDPC) codes, a class of block codes that can be employed to approach the Shannon capacity bound and boast efficient encoding and decoding processes due to the low density of the generator and parity check matrix. Although near-optimal, typical message-passing decoders are characterized by a large decoding latency and high complexity due to their iterative nature. Given the graph-like association between coded bits and parity checks, GNNs have been considered in [15], showing promising results for Bose–Chaudhuri–Hocquenghem (BCH) codes and regular LDPC codes, but only marginal BLER gains for 5G LDPC codes, with the latter being optimized for decoding with the Belief Propagation (BP) algorithm. It should also be mentioned that the proposed receiver required far higher complexity than the benchmarks. The transformer architecture, popularized by their application in LLMs, has also been successfully applied to short codes decoding in the Error Correction Code Transformer (ECCT) model [16], employing a code-aware Self-Attention (SA) operation that masks bit-wise interactions to mimic the behavior of GNNs. Given the breakthrough nature of this work, sev-

eral variants have been proposed based on the ECCT architecture, among which is worth mentioning the U-Shaped ECCT [17], which takes inspiration from the U-Net architecture presented in [18] to improve the error correction performance at medium code length, and the Multiple-Masked ECCT [19], which includes multiple SA masks based on the systematic parity check matrix to achieve state-of-the-art error correction capabilities with neural decoders for short codes.

- **Non-orthogonal Multiple Access (NOMA)**. The throughput in OFDM systems is limited by the amount of bandwidth that can be allocated to users: to overcome this, NOMA techniques have been proposed as a means of allowing multiple user transmissions over a partially overlapping temporal and frequency span. To correctly receive superimposed user data symbols, high-latency and complex iterative receivers should be implemented, *e.g.*, Successive Interference Cancellation (SIC) for power-domain NOMA techniques and Message Passing Algorithm (MPA) for code-domain NOMA. Furthermore, the superimposition makes channel estimation a non-trivial task. A modular DNN for the joint demodulation of Sparse Code Multiple Access (SCMA) symbols and decoding of non-binary polar-coded bits was presented in [20], alternating MPA modules with sparse BP modules and trainable weights. The model achieved a 2.5 dB gain in BLER over the benchmark under AWGN, while also providing complexity reductions. The authors in [21] proposed a low-complexity sorting MPA algorithm with a DL-based method to limit its search space, reducing the overall computational complexity without significant impact on the BER. Furthermore, code-based NOMA techniques like SCMA have inspired Generative AI approaches as [22, 23, 24]. On the opposite, power-based NOMA techniques can benefit from schemes involving the optimization of the power allocation, *e.g.*, [25, 26].

- **Equalization of Faster-than-Nyquist (FTN) signaling**. FTN signaling, first proposed by J. E. Mazo in 1975, aims at increasing the data rate by violating the Nyquist-Shannon sampling theorem, thus introducing Intersymbol Interference (ISI) in the received symbols sequence [27]. Clearly, the price to pay is an increased receiver complexity: the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm provides excellent performance, but requires building a trellis with a number of hypotheses that grows exponentially with the length of the ISI. Variants with shortened trellis [28] and based on frequency-domain equalization [29] have been proposed, although achieving suboptimal performance; furthermore, the extension to high order modulations also leads to an increased number of hypothesis and complexity, thus introducing a bottleneck for real-time equalization [30]. In this context, DL tech-

niques can provide aid performing multiple hypothesis testing at the same time, thus reducing the overall runtime of the equalizer. In [31], a Fully-Connected neural network (FCNN) was trained to equalize FTN Quadrature Amplitude Modulation (QAM) symbols with a sliding window mechanism. The input of the FCNN spans more symbols than those being equalized, leading to a more accurate equalization and, thus, lower BLER, at the expense of a slight increase in complexity. To identify the temporal patterns introduced by FTN, Recurrent Neural Networks (RNNs) in FTN were investigated in [32] and [33]. On the one hand, [32] implemented both uni- and bi-directional Long Short-Term Memory (LSTM) models to demodulate Binary Phase Shift Keying (BPSK) symbols, observing a reduction in the Bit Error Rate (BER) floor when using the second model over the first. On the other hand, the authors in [33] used a more traditional RNN to equalize FTN BPSK symbols under strong ISI, *i.e.*, with a compression factor $\tau_{FTN} = 0.5$.

1.1.2 AI for NTNs

NTNs play a pivotal role in the future wireless communications standards, enabling unprecedented coverage and a wide gamut of novel services [34]. Global connectivity can be reached by integrating the existing terrestrial infrastructure with spaceborne and airborne nodes, including satellites orbiting around the Earth at several hundreds or thousands of kilometers from ground, Unmanned Aerial Vehicles (UAVs), and High Altitude Platform Stations (HAPS) using transparent payloads, with the possibility of implementing a full gNodeB (gNB) on board of a regenerative payload being considered for the Release 19 of 3GPP standards [35]. Clearly, NTNs are subject to a propagation environment that departs substantially from terrestrial cellular assumptions. Links to moving platforms such as Low Earth Orbit (LEO) satellites and medium-altitude systems are characterized by large relative velocities (resulting in significant Doppler shifts and Doppler rates), wide-ranging delays and path losses, and a light multipath (typically in LoS due to link budget constraints), imprinting complex but well-defined patterns on the propagation channel [36]. Such distinctive propagation characteristics and, more in general, the real-world impairments of satellite links (*e.g.*, power amplifier non-linearities) make NTNs particularly suitable candidates for data-driven optimization approaches. Furthermore, the time-varying nature of satellite constellations poses challenges to the effectiveness of higher layer tasks such as user scheduling and routing: Reinforcement Learning (RL) and Deep RL (DRL) can be ideal solutions in these cases, with AI agents learning to take optimal decisions under challenging conditions. However, the DL models' complexity is a metric that should also be kept into consideration: the design of the satellite payload is subject to strict Size, Weight, and Power (SWaP) constraint, limiting the computational

capabilities that can be assigned to DL acceleration in case of on board models [37].

AI for NTN: State-of-the-Art analysis

The scientific literature on the application of AI to NTNs spans several areas, among which is worth highlighting the following:

- **Channel prediction**. The geometry between Non-Geostationary Orbit (NGSO) satellites, ground stations, and User Terminals (UTs) makes NTN an ideal environment for predicting the CSI. This finds application not only in resource scheduling, *e.g.*, to ensure that a user can experience a high quality link over a certain bandwidth for the entire duration of its allocation, but also in the context of Cell-Free (CF) MIMO NTN systems, where multiple users are served over the same bandwidth by means of digital beamforming. The authors in [38] trained a LSTM model to process a time series of MIMO channel matrices and predict their evolution over time. The work took into account the deterministic LoS component of the propagation channel, as well as random effects due to multiple secondary paths in the non-LoS (NLoS) component. [39] presented a Transformer-based architecture to predict the channel matrix based on recent estimates, achieving a Mean Squared Error (MSE) reduction with respect to RNN-based benchmarks. In [40], a CNN was implemented to transform uplink CSI into downlink CSI, aiming at removing the need of a feedback message. The CNN was shown to be able to provide more accurate predictions for satellites at lower orbits and for smaller distance between the uplink and downlink bands. The Gated Recurrent Unit (GRU) architecture implemented in [41] was able to learn the temporal correlations of a LEO satellite channel based on historical time series, showing the impact of the elevation angle and the LEO altitude, among the others, on the symbol error rate. The authors of [42] implemented a model based on CNNs and LSTM and a second model based on Variational Autoencoders (VAEs) to jointly perform CSI prediction and precoding, thus maximizing the predicted CSI's alignment with respect to the channel encountered during propagation accounting for the precoding module. It must be mentioned that traditional methods have also been proposed for CSI prediction, *e.g.*, the authors in [43] implemented a closed form for the prediction of the NTN channel based on an autoregressive model of order 2 (AR(2)).

- **Doppler shift estimation**. The Doppler shift is a variation of a signal's frequency components caused by the relative movement between transmitter and receiver. From the terrestrial networks' point of view, the gNB is typically stationary and the UE movement is the main responsible for Doppler shifts (multipath components

are also subject to further Doppler shifts due to the movement of the scatterers interacting with the communication link). However, NTNs can introduce far larger Doppler shifts due to the mobility of non-terrestrial nodes. In the case of LEO satellites, the introduced Doppler shift changes over time based on the position of the satellite on its orbit and its relative speed with respect to a reference UE on ground. Given the geometrical nature of such effect, if a UE does not have knowledge of the satellite's ephemeris, DL techniques can be employed to better estimate the Doppler shift and Doppler rate impacting the received waveforms. The literature on the application of AI to Doppler shift estimation in NTNs is still at its early stages. A CNN was employed in [44] as a means of extracting the Doppler shift from an estimate of the channel matrix between a multi-antenna LEO satellite and a single antenna UE, showing the estimation accuracy as a function of the SNR and the number of channel snapshots fed to the AI model. The authors in [45] proposed to train a convolution-based Autoencoder (AE) to extract features from the input Power Spectral Density of the received signals (including not only the target transmission, but also interfering signals); then, the encoder section is extracted and connected to a classifier to perform the coarse Doppler estimation. Nonetheless, DL-based techniques proposed for high mobility terrestrial contexts, *e.g.*, high speed trains, can also find application to NGSO NTNs, *e.g.*, the CNN proposed in [46] to refine LS-based Doppler estimates.

- **Spectrum sharing**. The cooperation between terrestrial and non-terrestrial components of 6G networks, also considering the most general case of multi-layered NTNs (*i.e.*, NTNs where satellites on different orbits and aerial vehicles at various altitudes are interconnected), call for advanced spectrum sharing techniques to ensure the coexistence of all nodes of the unified network. Spectrum allocation plans and interference management techniques are especially important in the case of NGSO satellites, where a service area may temporarily be covered by both a satellite and a set of terrestrial gNBs. In [47] a low-complexity scheme based on CNNs and LSTM to improve the spectrum occupancy detection performance in low SNR conditions based on channel samples was proposed. The CNN and LSTM modules work in parallel, with the first extracting spatial features and the second obtaining temporal features. For a target false alarm probability of 10%, the model is able to approach a 90% correct detection probability at -10 dB of SNR. The authors of [48] proposed a modified Q-learning approach to interference management between NGSO and Geostationary Orbit (GEO) satellites, where each link is optimized by choosing the serving NGSO satellite, the transmission power, the communication rate and the modulation and coding scheme. Furthermore, the detection of NGSO

satellites interference over GSO-based communications has been approached as an anomaly detection problem with an AE in [49] and with a VAE and a Transformer-based architecture in [50]. In both works, a model is trained to bottleneck and perfectly reconstruct a received signal (either its temporal or spectral samples) in absence of interference, thus willingly overfitting the model to such condition; an interference detection event is then triggered when the reconstruction Mean Absolute Error (MAE) on an input sequence is over a properly optimized threshold, caused by the overfit. The outcomes of these studies suggest that temporal sequences lead to better interference detection capabilities with respect to using spectral data, regardless of the chosen architecture; furthermore, the Transformer-based architecture was the one providing the lowest MAE and the best interference detection accuracy and F1 score among the considered models.

- **Resource allocation**. Power and spectrum are the most important resources in NTNs, with their allocation impacting the network's energy efficiency, the interference level, and the overall quality of the links. Deep learning algorithms can learn complex relationships between interference, power levels, and spectrum usage to autonomously allocate bandwidth to different nodes of the network, including terrestrial and non-terrestrial ones in the most general case, and perform power control. In [51], the channel and power allocation task in a multi-satellite scenario is performed through the combination of convex optimization methods and DL methods, including a sequence-to-sequence LSTM-based model denoted Pointer Network to allocate the channels and a FCNN to perform power allocation. The authors in [52] applied a Multi-Agent DRL (MADRL) scheme to minimize the communication delay and system fairness in multi-beam satellite systems under both user-level and beam-level traffic requests. In particular, the strategy adopted in the work is based on a hierarchical view: the channels are initially allocated to beams by one agent accounting for the co-channel interference and the overall traffic requests in each beam; then, a new agent works on each beam independently, allocating sub-channels to each user based on the corresponding traffic demand. A multi-agent approach was also employed in [53], where the joint optimization of the beam hopping patterns together with each beam's bandwidth and power allocation, considering a multi-LEO network cooperating with a GEO satellite, was carried out with a centralized critic multi-agent Deep Deterministic Policy Gradient (DDPG) algorithm. Finally, it should be mentioned that, in the context of Multi-access Edge Computing (MEC), computational resources also come into play, *e.g.*, by offloading tasks between terrestrial and non-terrestrial nodes. As an example, task offloading was jointly optimized with the allocation of computational resources in [54] using a

Double Deep Q-Network (DQN) and DDPG to reduce the cost of task computation.

1.2 Thesis contribution and organization

This thesis aims at advancing multiple PHY aspects of NTNs through the application of AI. While the focus is put on NTNs, the scope of these studies can be extended to that of terrestrial networks and the corresponding challenges (*e.g.*, the increased multipath richness) and benefits (*e.g.*, higher available computational capabilities). Three main areas of application have been identified: 1) NOMA demodulation, with a focus on code-domain NOMA techniques; 2) Channel prediction, tackling both the channel aging effect and the reduction of the pilot overhead; and 3) FTN signaling as a means of improving the spectral efficiency. In most of the cases, AI is a key enabler of throughput improvements in NTNs: this is a key objective for NGSO-based NTNs, where the visibility window can be in the order of minutes and the coverage can be discontinuous (*e.g.*, in the case of sparse constellations). The rest of this thesis is organized as follows:

- Chapter 2 introduces a code-domain NOMA technique, namely SCMA, in the context of NTN-based MEC. Then, it presents a CNN-based architecture for SCMA demodulation employing Multi-Task Learning (MTL) to tailor sections of the demodulator to different users' codebooks. This work represents one of the first contributions to SCMA demodulation in the NTN literature and, in particular, the first DL-based receiver. The Chapter concludes with a link-level evaluation of the end-to-end performance of the neural receiver, highlighting its impact on the BLER and the aggregated user throughput.

- Chapter 3 approaches the CSI prediction task in CF-MIMO NTN systems to counter channel aging. Differently from existing CSI prediction algorithms in the literature, predicted channel gains are obtained through a DNN integrating historical channel data and parameters of the UE-satellite geometry, resulting in an extremely lightweight DL model. The model is tested at the system level, showing that the DNN-based predictions can improve the misalignment between the computed beamforming matrix and the channel encountered at transmission time, thus increasing the per-user capacity.

- Chapter 4 presents the concept of Channel Frequency Response (CFR) prediction in the context of OFDM-based systems, where the CFR estimated over a certain time slot can be used to predict the CFR on the succeeding one, thus avoiding the need to transmit further pilot symbols. Three convolution-based DL architectures are proposed, comparing the MSE achieved on LoS and NLoS 3GPP fading models and

discussing their computational complexity and real time capabilities in the context of NTNs. The most accurate model is then employed in an end-to-end link-level simulation, evaluating the throughput gains deriving from the additional resources available for data transmission (*i.e.*, those typically reserved for pilot transmission). The Chapter also includes a detailed evaluation of the generalization capabilities of the chosen DL model, reporting the throughput in case of channel model and/or UE speed mismatch in the training and test phases. This work marks the first contribution to CFR prediction in NTNs, showing that accurate forecasts can be achieved with low computational complexity even in presence of residual Doppler shifts.

- Chapter 5 employs one of the models presented in the previous Chapter in an NTN system based on SCMA, where the non-orthogonal channel estimation problem is solved for the first time in the SCMA literature through channel prediction. In particular, a prediction-oriented slot structure is designed: different users' orthogonal pilots and data symbols are allocated in a first OFDM slot, which enables channel estimation for all users in the corresponding Physical Resoure Blocks (PRBs); then, the estimated CFRs can be employed for channel prediction, thus enabling the equalization of the SCMA symbols carried on a second OFDM slot. The throughput gains over traditional OFDM systems are evaluated by means of numerical simulations, also highlighting the gap between the considered model and one that has perfect channel estimates availability.

- Chapter 6 proposes the application of CNNs to equalize FTN signaling in single carrier systems, a novel approach to FTN equalization in the literature. Due to the investigative nature of the work, under similar assumptions typically taken in the design and evaluation of FTN receivers, only AWGN is here considered. The proposed solution, which is completely open source, is compared against several benchmark algorithms at various FTN compression rates, highlighting the equalization capabilities CNNs even under strong ISI.

- Finally, Chapter 7 draws the conclusions of the thesis, summarizing the advancements in the field of AI for PHY and, in particular, the application of such techniques to NTNs. The most promising research directions that emerged from these studies are also outlined.

- Additionally, Appendix A presents the application of the mechanism at the basis of the Transformer architecture, *i.e.*, the Multi-Headed Self-Attention (MHSA), to the task of Signal-to-Interference-plus-Noise Ratio (SINR) estimation in the context of

user-centric NTNs. This work represents the first contribution to DL-based SINR estimation in the literature. The objective of the DL model is to estimate the SINR achieved by a group of UEs served by means of user-centric beamforming: such information can then be provided to higher layers to perform different tasks, *e.g.*, incorporating it in the scheduler to select the best performing group of users among a set of candidate groups.

- Furthermore, Appendix B reports a general overview of AI and the techniques employed throughout this thesis, including a description of the layers and the training procedure for DL models.

Chapter 2

SCMA Demodulation with Multi-Task Learning

This study was carried out during the first and second year of PhD under the framework of the European Space Agency (ESA)-funded EAGER project. The contents of this chapter are based on the paper "An SCMA Receiver for 6G NTN based on Multi-Task Learning" [55].

NTNs represent a crucial component of modern communication infrastructures, becoming a key technology of the future 6G communication standards [56]. Interconnected satellites at different altitudes provide extensive coverage, especially in remote and underserved regions, integrating with TNs to reach ubiquitous connectivity. In parallel, edge computing frameworks, such as MEC, have emerged as enablers for low latency services and distributed learning paradigms. As an example, nodes distributed on ground can collect local data and train a local version of an AI model, sharing only the model updates with the MEC server in a federated learning fashion [57]. However, the communication link can be a bottleneck to such use cases, as too many users may request access to the network simultaneously. NOMA techniques have been greatly investigated as a mean to enhance the uplink spectral efficiency and accommodate massive connectivity demands. Among such techniques, the codebook-based SCMA provides a good balance between performance and complexity. SCMA enables multiple users to sparsely share the same frequency resources, significantly boosting the spectrum efficiency while managing the generated interference. On the opposite, as the transmissions originating from the same beam exhibit similar received power levels, power-domain NOMA approaches appear as less promising for NTNs [58]. In this framework, the interplay between NTNs, SCMA, and MEC generates a strong synergy. The use of SCMA in NTNs ensures that a vast number of transmissions, *e.g.*, collected data or AI model updates, from Ground Devices (GDs) are supported. LEO satellites, functioning as a MEC servers, processes this data at the

edge, thereby enabling rapid responses and decision-making capabilities [59]. However, the NTN propagation channel poses significant challenges to the communication links, which are typically operating at extremely low received power levels. In order to improve the link budget, more complex ground terminals should be implemented, *e.g.*, employing highly directive and steerable antennas. To keep the terminal complexity at a minimum, the link can be enhanced at the receiver side, *e.g.*, using AI techniques such as NNs. In the time frame when this work was carried out, non-generative approaches to SCMA in NTN had not been evaluated yet. While SCMA can provide the desired spectral efficiency boost to 6G NTNs, it is also imperative to reduce the E_b/N_0 levels required to achieve satisfactory link performance. Thus, aiming at improving the model's performance, MTL is here adopted to let each section of the NN adapt to a specific SCMA codebook. The main contribution of this study are the following:

- An MTL-based SCMA receiver optimized for NTN is presented, requiring no additional complexity at the GDs;
- The performance of the proposed receiver is evaluated in an end-to-end channel-coded NTN link, taking into consideration the challenges related to the implementation of an AI-based receiver on board of a satellite.

2.1 3GPP NOMA techniques

The 3GPP Release 16 NOMA study item [60] focuses on uplink grant-free transmission, which can be either contention-free or contention-based, allowing terminals to operate in RRC connected or idle/inactive states. Several NOMA schemes have been proposed for 5G, mainly distinguished by their multiple access signatures. Symbol-level NOMA techniques use UE-specific spreading, scrambling, or interleaving applied to symbols, and can be categorized into low-density spreading (LDS)-based and non-LDS approaches. LDS schemes, such as LDS - Code Division Multiple Access (CDMA), LDS-OFDM, and SCMA, employ sparse spreading sequences and iterative detection algorithms like MPA to approach near maximum-likelihood performance. Non-LDS schemes, including Pattern Division Multiple Access (PDMA), Multi-User Shared Access (MUSA), and Non-Orthogonal Coded Access (NOCA), rely on SIC or Parallel Interference Cancellation (PIC) detection. While spreading-based NOMA enhances spectrum efficiency and supports grant-free access [61], it demands complex multi-user detection, optimal sequence design, and accurate synchronization, resulting in challenging operation in NTN environments where synchronization and channel knowledge are imperfect. Bit-level NOMA techniques, such as Interleaved Division Multiple Access (IDMA) and Interleave-Grid Division Multiple Access

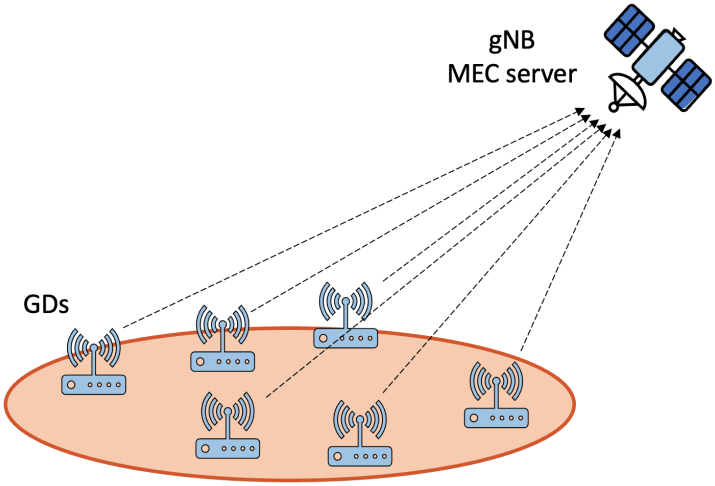


Figure 2.1: Considered system model for SCMA-based MEC in NTN.

(IGMA), separate users through scrambling or interleaving, employing simple estimators for multi-user detection. These methods assume quasi-static channels within an interleaving window, making them unsuitable for fast-varying LEO satellite links. Moreover, slow fading from atmospheric effects further degrades performance of interleaver-based detectors. To improve robustness, resource element (RE) processing-based NOMA introduces sparse RE mapping in schemes like SCMA, PDMA, and IGMA, transmitting zeros in selected REs within each PRB. This hybrid approach combines symbol/bit-level processing with RE-domain sparsity to increase separation among J users sharing K non-orthogonal resources, with sparsity defining the fraction of REs per user. However, it should be stressed that perfect knowledge of channel coefficients and synchronization among the users are required to allow the receiving algorithm to properly work.

2.2 System model

A LEO satellite providing connectivity to a set of GDs within its coverage area is here considered (Figure 2.1). In this NTN-based MEC framework, the satellite payload carries the MEC server. The GDs act as edge nodes that collect, pre-process, and transmit data to it, *e.g.*, model updates for federated learning, or sensor data for internet of things applications. It is assumed that the gNB, which is co-located with the MEC server on board of the satellite, schedules J GDs in connected mode for uplink SCMA transmission over a set of $K < J$ shared frequency resources. 5G NR PRBs, each comprising $N_{SC}^{PRB} = 12$ Orthogonal Frequency Division Multiple Access (OFDMA) subcarriers, are allocated to GDs; thus, K PRBs are considered as frequency resources. Figure 2.2 reports the block diagram of the considered transmitter and receiver structure. The data payload of the

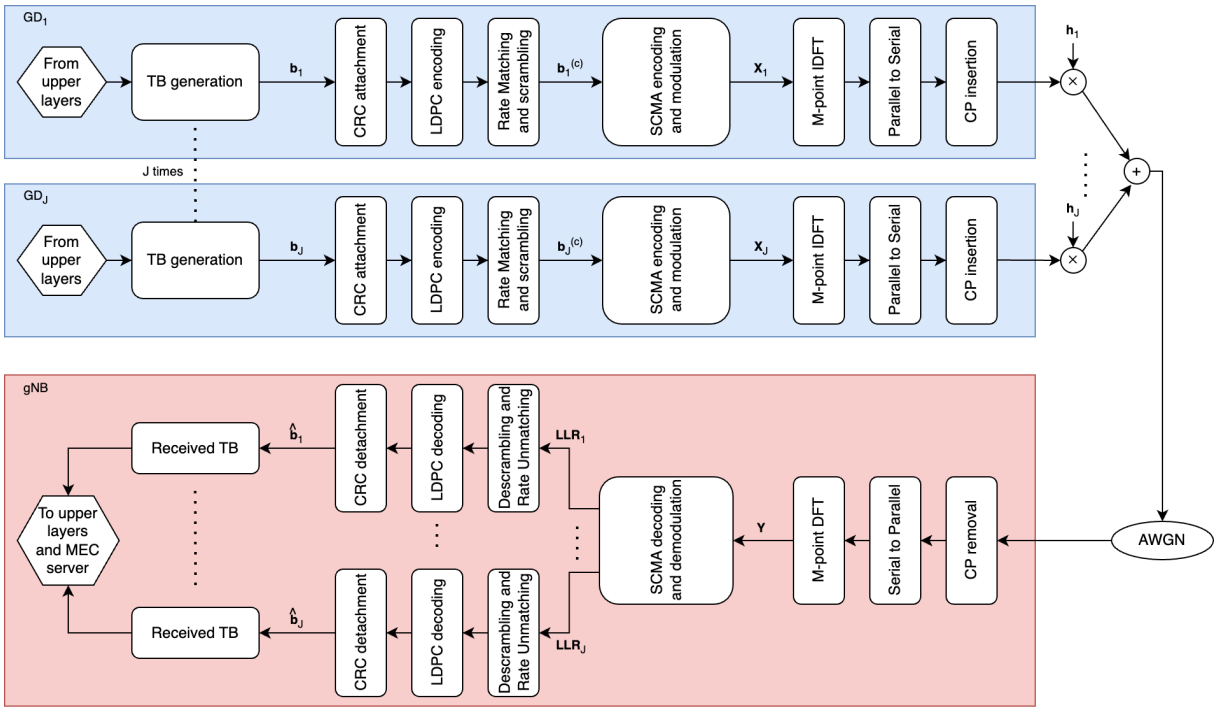


Figure 2.2: Transmitters and receiver block diagram.

i-th GD is mapped to one or more 5G Transport Blocks (TBs) $\mathbf{b}_i$ with a Transport Block Size (TBS) of N_{TBS} bits. Each TB is encoded with an LDPC channel encoder with code rate R_c, resulting in a vector of N_{bits} coded bits $\mathbf{b}_i^{(c)}$. An SCMA encoder and modulator is implemented based on the chosen SCMA codebook $\Gamma_i(\cdot)$, mapping each group of m coded bits in $\mathbf{b}_i^{(c)}$ to one of $M = 2^m$ complex codewords of length K [62]. Such mapping is repeated for each of the N_{SC}^{PRB} groups of K subcarriers. The SCMA codebooks are designed to minimize the overlap between the codewords of different GDs, allowing an increased amount of GDs to be scheduled over the same set of subcarriers with respect to OFDMA. In particular, each codeword has d_v non-zero elements, and the number of codewords of different GDs overlapping over the same resource is limited to d_f. Assuming $K = 4$, $J = 6$, $d_v = 2$ (*i.e.*, two PRBs are allocated to each GD), and $d_f = 3$ (*i.e.*, three GDs transmit over the same PRB), the Factor Graph (FG) in Figure 2.3 can be obtained, corresponding to the following signature matrix:

$$\mathbf{S} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

The resulting overloading factor with respect to orthogonal transmissions (*i.e.*, one user per PRB) is $\frac{J}{K} = 150\%$. At the i-th transmitter, the generated SCMA codewords $\mathbf{x}_i =$

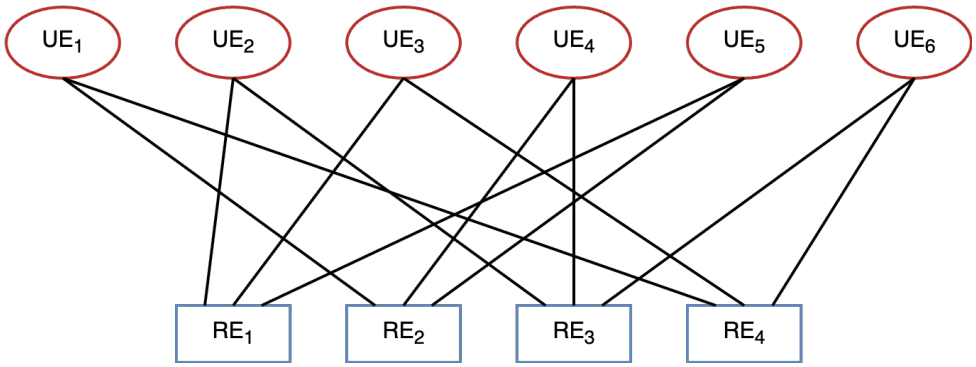


Figure 2.3: SCMA factor graph.

$\Gamma_i(\mathbf{b}_i^{(c)})$ are mapped over the $N_{SC} = N_{SC}^{PRB} \cdot K$ shared subcarriers, repeating the SCMA FG pattern over the available bandwidth and covering a number of symbol times N_{sym}. The resulting grid $\mathbf{X}_i$ of size $[N_{SC} \times N_{sym}]$ is used to generate the CP-OFDM waveform, which is then transmitted at carrier frequency f_c. Assuming line of sight with the satellite, each signal encounters a different propagation channel, represented by the following matrix $\mathbf{H}$ of single tap channel coefficients:

$$\mathbf{H}\left[i;t\right] = G_{i,t}^{(tx)} G_{i,t}^{(rx)} \frac{1}{\sqrt{L_{i,t}}} \frac{\lambda}{4\pi d_{i,t}} e^{-j\frac{2\pi}{\lambda} d_{i,t}}, \tag{2.2}$$

where $\mathbf{H}\left[i;t\right]$ indicates the element at row i and column t of $\mathbf{H}$ (*i.e.*, the channel coefficient of the i-th link at symbol time t), $G_{i,t}^{(tx)}$ and $G_{i,t}^{(rx)}$ represent the transmission and reception gain, respectively, $d_{i,t}$ represents the slant range, $\lambda = \frac{c}{f_c}$ is the carrier wavelength (with c being the speed of light), and $L_{i,t}$ contains all of the power losses in addition to the free space path loss, *i.e.*, atmospheric, scintillation, and shadowing losses. At the receiver, the signals are superimposed one with the other and summed with AWGN with power spectral density N_0. After the FFT, the received grid at symbol time t can be represented as follows:

$$\mathbf{Y} = \sum_{i=1}^{J} \mathbf{H}[i;1,\ldots,N_{sym}] \cdot \mathbf{X}_i + \mathbf{N}, \tag{2.3}$$

where $\mathbf{N}$ represents the matrix of AWGN samples that affects the received OFDMA grid. To separate the SCMA symbols of different GDs over the t-th symbol time and retrieve the J sequences of coded bits $\{\mathbf{b}_i^{(c)}\}_{i=1}^{J}$, a variant of the MPA is typically implemented at the receiver. These algorithms progressively refine the coded bit Log-Likelihood Ratios (LLRs) corresponding to each SCMA transmission, iteratively computing and exchanging messages between factor nodes (representing the K frequency resources) and variable nodes (representing the J GDs) of the SCMA FG. For a more detailed description of the most popular MPA variants, the reader can refer to [63]. Clearly, the process must be repeated N_{SC}^{PRB} times to cover the K PRBs and for each symbol time encompassing the

transmission of the TB. Once the maximum number of iterations is reached and the entire OFDMA grid has been processed, the estimated LLR sequence related to the i-th GD $\mathbf{LLR}_i$ is channel decoded, obtaining the corresponding received TB $\hat{\mathbf{b}}_i$. The data is then forwarded to the upper layers, after which the GDs' payloads can finally be processed by the MEC server. Given the iterative nature of MPA, the computation of the LLRs corresponding to a received symbol vector at symbol time t, $\mathbf{Y}[1,\ldots,N_{SC};t]$, requires long sequential operations that cannot be easily parallelized. Furthermore, variants of the MPA with lower computational complexity, such as the Log-MPA or the Max-log-MPA, are typically preferred over the base algorithm, leading to suboptimal performance. For this reason, a DL-based receiver is implemented to perform SCMA decoding and demodulation. Indeed, DL algorithms can provide significant gains by approximating with matricial operations the optimal SCMA receiver under the considered system model.

2.3 Proposed AI-based SCMA receiver

The task of the proposed receiver consists in estimating $m \cdot J$ coded bit LLRs for each set of K subcarriers at symbol time t, given the corresponding complex vectors of K received symbols $\mathbf{y}_t^{(in)} \subset \mathbf{Y}[1,\ldots,N_{SC};t]$ and J channel coefficients $\mathbf{H}[1,\ldots,N_{SC};t]$. The DL model proposed in this work is a CNN: this architectural choice is driven by the spatial patterns introduced by the SCMA codebooks, filtering out interference from different users' transmissions using convolutional layers. In the first section of the CNN, Furthermore, MTL is also employed to let a subset of layers specialize on different tasks, *i.e.*, decoding different SCMA codebooks, based on the same features extracted by common layers. The structure of the CNN is reported in Figure 2.4, where the task-specific layers are highlighted in light blue. To obtain an image-like input data shape, the user channel coefficients, which are assumed to be known at the receiver, are first pre-processed together with the received grid in a similar fashion to [10]. In particular, each example within a minibatch consists of a real-valued matrix $\mathbf{A}$ of shape $[K \times 2(J+1)]$, initialized with zeros. The first and second columns are filled with the real and imaginary parts of the considered K symbols within the received grid, $\mathbf{y}_t^{(in)}$, respectively. Then, $\mathbf{A}[k;2i+1]$ and $\mathbf{A}[k;(2i+2)]$ are filled with $Re\{\mathbf{H}[i;t]\}$ and $Im\{\mathbf{H}[i,t]\}$, respectively, if $\Gamma_i(\cdot)$ foresees a transmission on subcarrier k. Such operation embeds the SCMA codebook in the NN input, improving the training process. Figure 2.5 represents the obtained matrix considering the codebook reported in [64]. After collecting N_{batch} examples, the resulting input minibatch is a tensor of shape $[N_{batch} \times K \times 2(J+1)]$, where the first dimension refers to the batch size, the second is a spatial dimension related to the subcarrier, and the third is the channels dimension. Before being fed to the CNN, the data in each channel is sep-

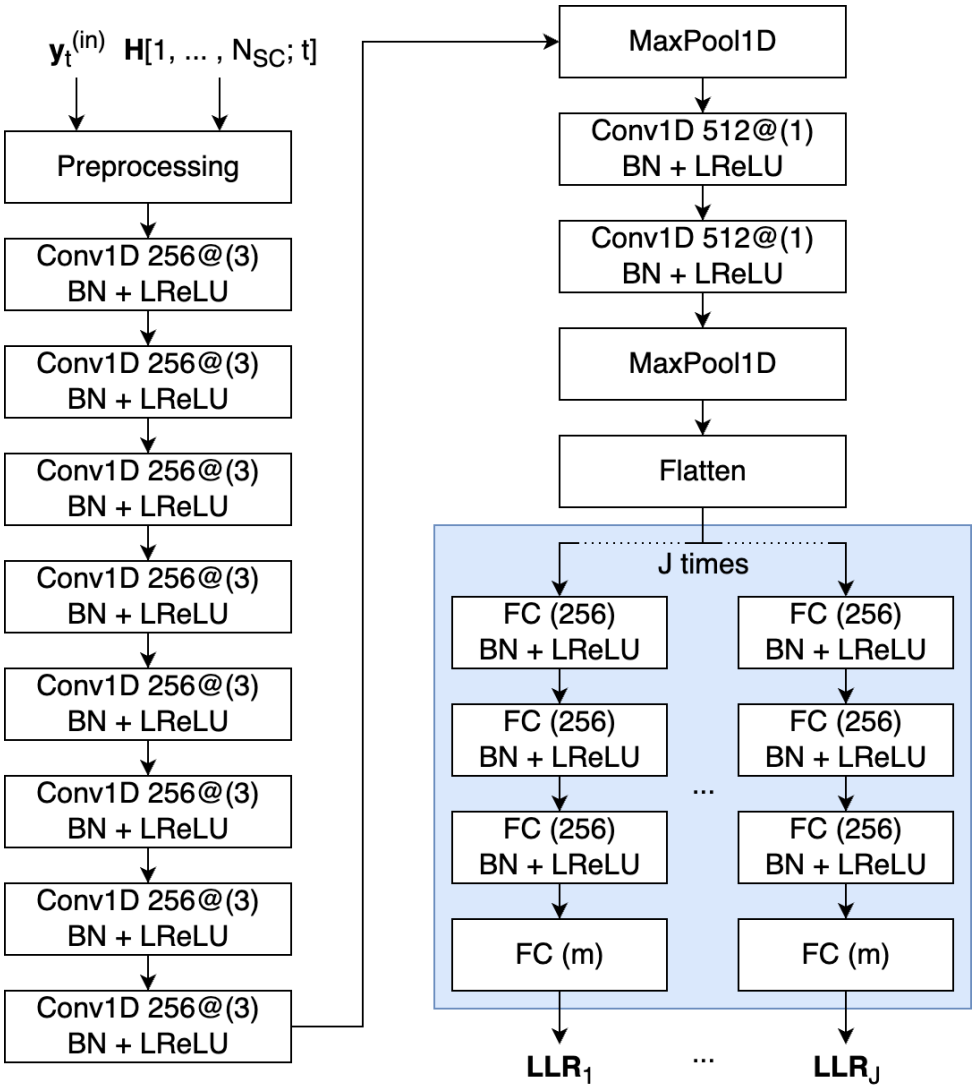


Figure 2.4: Diagram of the proposed CNN-based SCMA receiver.

arately standardized using first and second order statistics, which have been previously estimated on a generated minibatch. To extract and process spatial features from the frequency domain, the input tensor is fed to a set of eight 1-Dimensional Convolutional (Conv1D) layers with 256 kernels of length 3, employing "same" padding to maintain the length of the spatial/frequency dimension. Considering an $[L \times D]$ padded input matrix $\mathbf{I}$, an $[N^{(f)} \times W^{(f)} \times D]$ optimized weight tensor $\mathbf{W}$ (with $W^{(f)}$ odd), and an $[L \times N^{(f)}]$ optimized bias matrix $\mathbf{B}$, a Conv1D layer with "same" padding computes an $[L \times N^{(f)}]$ output matrix $\mathbf{O}$ as follows:

$$\mathbf{O}\left[1,\ldots,L;n\right] = \mathbf{B}\left[1,\ldots,L;n\right] + \sum_{d=1}^{D} \mathbf{I}\left[1,\ldots,L;d\right] * \mathbf{W}\left[n;1,\ldots,W^{(f)};d\right], \tag{2.4}$$

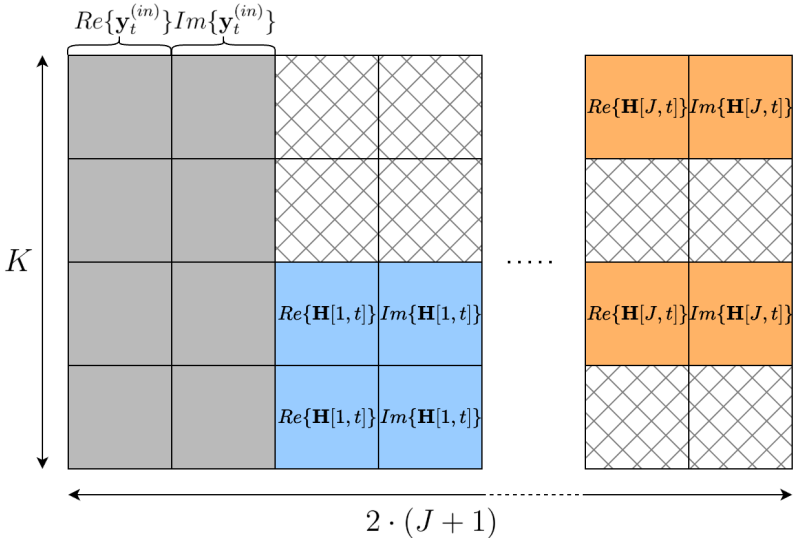


Figure 2.5: NN input after the pre-processing reshaping step.

where the symbol $*$ represents the cross-correlation operator. Each Conv1D layer is followed by a Batch Normalization (BN) layer and a Leaky Rectified Linear Unit (LReLU) activation function:

$$LReLU(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha_{LReLU} \cdot x & \text{otherwise} \end{cases} \tag{2.5}$$

where $\alpha_{LReLU} = 0.3$ is the parameter responsible for a small leak of negative activations. After the first eight layers, a 1-Dimensional Max Pooling (MaxPool1D) layer is employed, halving the tensor dimension by choosing the maximum between each group of two elements over the spatial (*i.e.*, subcarriers) dimension. Two Conv1D layers with 512 kernels of size 1 and "same" padding are then used to refine the extracted features. The common features extraction section of the model terminates with another MaxPool1D layer, flattening the tensor to a vector with 512 elements. The extracted features are fed to J task-specific chains with identical structure, where the i-th chain specializes on the estimation of the m bit LLRs of GD i. Each chain is composed of 3 Fully-Connected (FC) layers with 256 neurons, each activated with LReLU and followed by a BN layer. Each chain is completed by a FC layer with m neurons, leading to a model output of length $m \cdot J$. The receiver should perform a typical classification task, in which each output $\hat{l}_u$, $u \in [1, \ldots, m \cdot J]$, is activated by the sigmoid function $\sigma(\hat{l}_u)$ to obtain the probability of the u-th output bit being 1, $\hat{b}_u = \sigma(\hat{l}_u) = (1 + e^{-\hat{l}_u})^{-1}$. Clearly, this representation carries the same information as the u-th bit LLR, which actually coincide with the non-activated value $\hat{l}_u$ of the output tensor:

$$LLR_u = ln\left(\frac{Pr\{b_u = 1|\hat{b}_u\}}{Pr\{b_u = 0|\hat{b}_u\}}\right) = ln\left(\frac{\hat{b}_u}{1 - \hat{b}_u}\right) = \hat{l}_u, \tag{2.6}$$

where b_u represents the true value of the u-th output bit. For this reason, $\sigma(\cdot)$ is moved to the loss function, resulting in the Binary Cross-Entropy with Logits (L-BCE):

$$\begin{aligned} L-BCE =& \frac{1}{m \cdot J} \sum_{u=1}^{m \cdot J} L-BCE_u, \text{ with} \\ L-BCE_u =& - b_u \cdot ln\left(\sigma\left(\hat{l}_u\right)\right) + \\ & - (1-b_u) \cdot ln\left(1-\sigma\left(\hat{l}_u\right)\right). \end{aligned} \tag{2.7}$$

With little mathematical manipulation, recalling the formulation of $\sigma(\hat{l}_u)$, the $L-BCE_u$ term can be rewritten as:

$$L-BCE_u = \begin{cases} ln\left(1+e^{\hat{l}_u}\right) & \text{if } b_u = 0 \\ ln\left(1+e^{-\hat{l}_u}\right) & \text{if } b_u = 1 \end{cases} \tag{2.8}$$

which corresponds to the softplus function $f_{sp}(a) = ln(1+e^a)$ computed in $a = \pm\hat{l}_u$ based on the value of the true label b_u. Thus, training the CNN with the L-BCE loss results in the vector of m LLRs associated to the i-th GD, $\mathbf{LLR}_i$, to be estimated at the output of the i-th neural chain.

2.4 Results

In this section, the link level performance achieved by the proposed AI algorithm are analyzed. Table 2.1 provides an overview of the simulation parameters. First, a train and test dataset of channel coefficients was obtained by simulating the orbital movement of a LEO satellite at 600km of altitude on the MATLAB computing environment. Then, the CNN presented in Section 2.3 was implemented using Python with the TensorFlow library [65]. A data generator was developed, where user bits are continuously randomly generated, mapped to SCMA codewords, and superimposed one with each other and AWGN considering the training channel coefficients dataset. The model was trained using the popular Adam optimizer, setting the maximum number of epochs to 2000. Each epoch consisted of 256 minibatches of data, each comprising of $N_{batch} = 3000$ superimposed SCMA symbols and the corresponding bits as input data and labels, respectively. To improve the learning process, the learning rate, initially set to 10^{-3}, was reduced by a factor 10 every time a L-BCE improvement was not observed for 50 training epochs. Early stopping was also implemented to interrupt the training process after 200 epochs from the last observed loss decrease. Once trained, the model was imported in MATLAB in a complete end-to-end link-level simulation, including LDPC encoding/decoding and OFDMA waveform generation, based on the 5G Toolbox. A Monte Carlo simulation

Table 2.1: Simulation parameters

Parameter	**Value**
Satellite altitude	600 km
Carrier frequency	$f_c = 2$ GHz
SCMA codebook	As in [64]
Number of Log-MPA iterations	$N_{iter} = 10$
5G numerology	$\mu = 0$
Code rate	$R_c = \{602, 193\}/1024$
TBS	$N_{TBS} = \{168, 48\}$ bits
Coded bits per TBS	$N_{TBS}^{(c)} = 288$ bits
E_b/N_0 range	$[-15, -14, \ldots, 9, 10]$
Monte Carlo Iterations	$N_{MC} = 10^4$
Channel model	Based on [36]

was run encompassing $N_{MC} = 10^4$ iterations, during which each of the $J = 6$ GDs was generating a TB of either $N_{TBS} = 168$ bits ($R_c = 0.588$) or $N_{TBS} = 48$ bits ($R_c = 0.188$). In both cases, after rate matching the coded TB was composed of $N_{TBS}^{(c)} = 288$ bits. First, the BLER is assessed as the average number of correctly decoded TBs averaged over the Monte Carlo iterations and the GDs, without hybrid automatic repeat request. Such metric is critical to evaluate the effectiveness of the proposed receiver, reporting the E_b/N_0 level required to achieve a target error rate on the transmitted TBs.

2.4.1 Block error rate

Figure 2.6 reports the BLER achieved by the proposed SCMA receiver as a function of the E_b/N_0 computed on the received waveform. Log-MPA with $N_{iter} = 10$ iterations is also shown as a benchmark. The plot shows that the AI model provides excellent performance at low E_b/N_0 levels. In particular, the CNN achieves a target BLER of 10% at around -2.75dB of E_b/N_0 for $R_c = 0.588$ and -5.25dB for $R_c = 0.188$. In comparison, Log-MPA requires a 3.25dB and 0.5dB higher E_b/N_0 to reach the same BLER target, respectively. The reason for the large gap at high code rate is twofold: firstly, CNNs typically provide good performance in pattern recognition tasks in presence of noise; secondly, Log-MPA introduces approximations to MPA to lower its computational complexity, which in turn

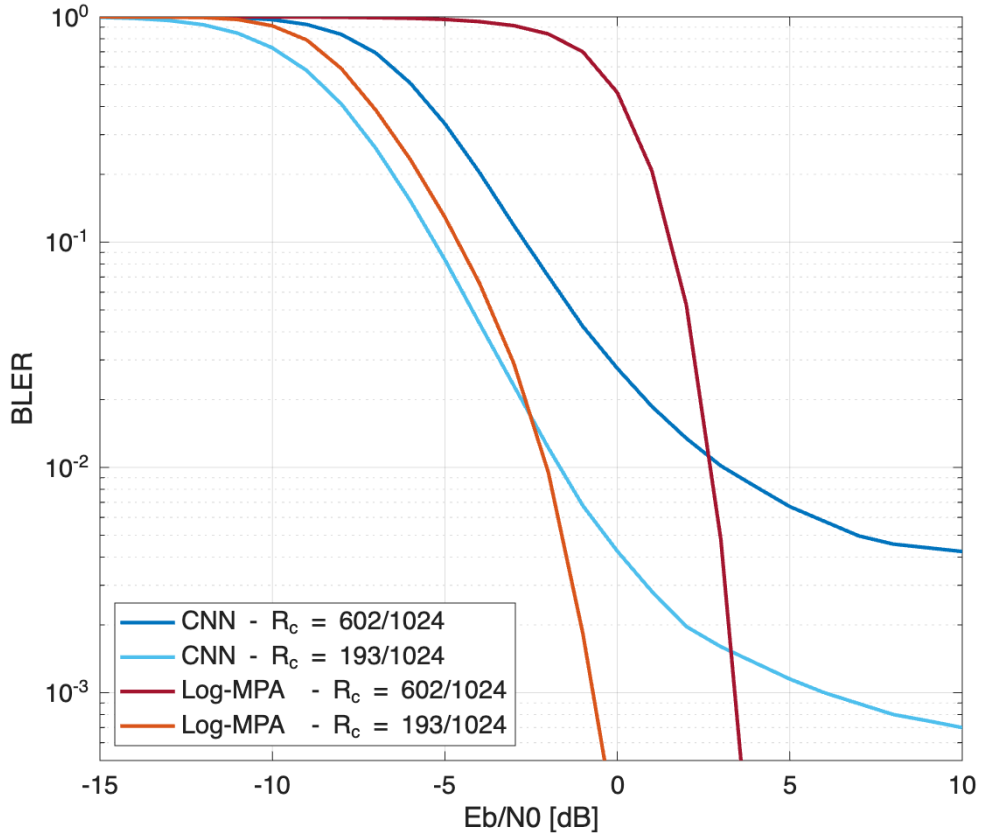


Figure 2.6: BLER as a function of the E_b/N_0.

leads to suboptimal performance. Due to the training process of the CNN, the layers can approximate the MPA solution with higher accuracy, improving the BLER under low E_b/N_0 conditions. Clearly, when robust codes are implemented, more demodulation errors can be corrected, leading to marginal gains with respect to Log-MPA. The BLER performance of the AI-based receiver shows an error floor, resting above $3 \cdot 10^{-3}$ and $5 \cdot 10^{-4}$ for high and low code rate, respectively, instead of the classic waterfall trend. As the AI model operates as a function approximator, residual errors are to be expected even at high E_b/N_0 levels, leading to the observed behavior. It must be noted that an increase in the model complexity may result in a lower error floor. Nonetheless, operating at a target 10% BLER with a relaxed E_b/N_0 requirement is a good trade-off for non-critical applications. Indeed, the achieved gain can be exploited by reducing the transmit power of each GD, increasing the devices' expected battery life.

2.4.2 Aggregated theoretical throughput

While the BLER analysis provides an insight into the error frequency, it fails at highlighting the rate at which information bits are correctly received. From the point of view of the satellite payload, it is of interest to determine the aggregated throughput to be

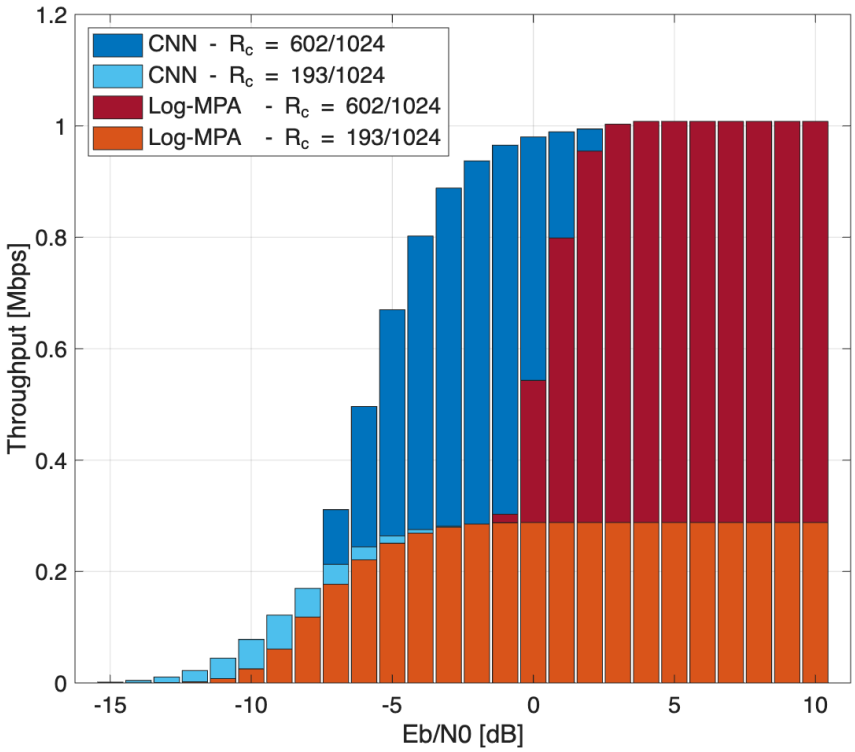


Figure 2.7: Aggregated theoretical throughput as a function of the E_b/N_0.

expected at the gNB by the J scheduled GDs. Indeed, apart from the implications from the communications point of view, such quantity also affects computational capabilities requirements for the MEC server. The Aggregated Theoretical Throughput (ATT) can be obtained from the BLER as follows:

$$ATT = \frac{J \cdot N_{TBS}}{T_{TB}} \cdot (1 - BLER), \tag{2.9}$$

where T_{TB} represents the duration of the transmission of a TB. Fixing $m = 2$ bits per SCMA codeword, and recalling that a codeword spans $K = 4$ subcarriers, each GD is able to transmit the entire TB in $N_{sym} = N_{TBS}^{(c)}/(m \cdot K) = 12$ symbol times. Thus, considering regular cyclic prefix length and 5G numerology $\mu = 0$, accounting for typical pilot overhead, each TB is transmitted over one slot, *i.e.*, $T_{TB} = T_{slot} = 1ms$. The bar plot reported in Figure 2.7 shows the ATT as a function of the E_b/N_0 ratio. One can notice that the BLER improvement provided by the AI-based receiver at high code rate result in significant gains in ATT. Indeed, the proposed algorithm provides 800kbps of aggregated throughput at -4dB of E_b/N_0, while Log-MPA is not able to operate with such a high noise level, resulting in less than 50kbps of ATT. On the other hand, with $R_c = 0.188$ the CNN can provide 170kbps of ATT at -8dB of E_b/N_0, a 1dB gain over Log-MPA. Overall, the CNN enables higher code rates to be used at lower E_b/N_0 levels, *e.g.*, $R_c = 0.588$ at -5dB of E_b/N_0 instead of being limited to $R_c = 0.188$ with the traditional receiver,

resulting in an ATT gain of 420kbps.

2.5 Computational complexity

Clearly, the CNN-based SCMA receiver can provide considerable communication gains to the MEC NTN system. However, computational complexity is also a key parameter to take into consideration. The number of multiply-and-accumulate (MAC) units for Conv1D layers with $N^{(f)}$ kernels of length $W^{(f)}$ applied to an input matrix of shape $[L \times D]$ can be computed as:

$$MAC_{Conv1D} = LDW^{(f)}N^{(f)}, \tag{2.10}$$

while the same metric for FC layers with $N^{(out)}$ neurons applied to an input vector of length $N^{(in)}$ can be evaluated as:

$$MAC_{FC} = N^{(in)}N^{(out)}. \tag{2.11}$$

Considering the model reported in Figure 2.4 and the chosen SCMA codebooks, the total amount of MAC operations amounts to 7.9M. On the opposite, recalling the parameters in Table 2.1, Log-MPA performs 23k multiplications [63], limiting the number of required MAC units to such value, two orders of magnitude lower than the proposed CNN. However, three key considerations should be taken into account. 1) Given the vast amount of tunable parameters in AI, it is possible that specific parameter combinations providing lower computational complexity with comparable, if not higher, link-level performance were not tested. 2) Complexity reduction and model compression techniques, which are out of the scope of this work, can be applied to greatly reduce the number of MAC units in the proposed model with negligible performance losses or, in some cases, even marginal gains [66]. 3) Specialized hardware accelerators have been successfully implemented on board of power-constrained devices, running inference with large AI models in a timely manner [67]. Hence, the adoption of hardware accelerators, together with model compression techniques and technological advancements, can make the proposed SCMA receiver a feasible solution to enable an efficient massive access in future 6G NTNs.

2.6 Conclusions and future works

In this work, SCMA was integrated in an NTN-based MEC framework to enhance the spectral efficiency of uplink transmissions. To improve the BLER of traditional SCMA receivers at low Eb/N0 levels, where NTN links typically operate, a CNN with MTL was trained to demodulate SCMA signals. The AI model was evaluated in a link-level

simulation, considering both the gNB and the MEC server on board of a LEO satellite. Compared to Log-MPA, the CNN achieves a target 10% BLER with a gain of up to 3.75dB of Eb/N0. The proposed receiver allows higher code rates to be used at low Eb/N0 levels, resulting in a higher ATT at the MEC server. While the adopted approach is theoretically sound for both NTNs and TNs, its generalization capabilities should be evaluated in future activities. Due to the graph-like structure of the SCMA signature matrix, the application of GNNs to SCMA demodulation is promising, possibly resulting in a lower computational complexity. However, SCMA is limited by the channel estimation accuracy over non-orthogonal pilots, an open issue in the SCMA literature. This is exacerbated in LEO-based NTNs due to the large Doppler shifts and Doppler rates. This consideration motivated the study of the application of channel prediction techniques to SCMA systems in NTNs, reported in Chapter 5. Nonetheless, joint channel estimation and demodulation techniques with SCMA signals should also be studied, possibly considering convolutional GNNs. Generative AI approaches could also be investigated to optimize the pilot overhead and the SCMA codebooks in NTNs. Indeed, the design of SCMA codebooks is a key research area in the literature due to its impact on the error rates and Peak-to-Average Power Ratio (PAPR) of the generated waveform. Nonetheless, most of the proposed codebooks have been designed without consideration of NTN scenarios, *e.g.*, [68, 69, 70, 71], or the dynamics of NGSO-based NTNs, *e.g.*, [72].

Chapter 3

CSI Prediction in Cell-free NTNs

This study was carried out during the first year of PhD under the framework of the ESA-funded EAGER project. The contents of this chapter are based on the paper "Cell-Free MIMO in 6G NTN with AI-predicted CSI" [73].

Thanks to digital beamforming, NTN systems can dynamically generate beams pointing towards a service area or, more specifically, towards single users in a cell-free fashion with full frequency reuse. In the NTN context, CF-MIMO systems may be implemented with multiple satellites, in a federated fashion, or with a single satellite, moving from the traditional fixed beam lattice to a user-centric coverage. Such technique has been demonstrated successful in improving the spectral efficiency in NTNs, resulting in an increased data rate experienced by the users. With the next generation 3GPP standard, 6G, foreseeing the integration of NTNs and terrestrial networks, CF-MIMO may be employed to enable higher spectral efficiency in NTNs, aiming at coping with the large expected traffic demand [74]. However, CF-MIMO systems typically rely on a downlink CSI feedback from the UTs to the ground station where digital beamforming coefficients are computed. Due to the intrinsic propagation distance that characterizes NTNs, the link is subject to strong delays, resulting in the CSI being partially outdated once they are received by the ground station. This effect, known as "channel aging", affects system performance, as the feedback CSI accuracy with respect to the propagation channel at transmission time is crucial to generate well-aligned CF beams. AI algorithms have been proposed in the scientific literature as a means to predict the CSI based on previous estimates, *e.g.*, [38, 40, 41]; however, while having the advantage of being able to learn from real data and perform more accurate predictions than traditional methods, these techniques typically require large computational capabilities.

This study focuses on a low-complexity AI-based model to perform CSI prediction based on the downlink CSI feedback, the user location, and the LEO satellite's ephemeris. Differently from previous works, the proposed model is trained to extract temporal statistics

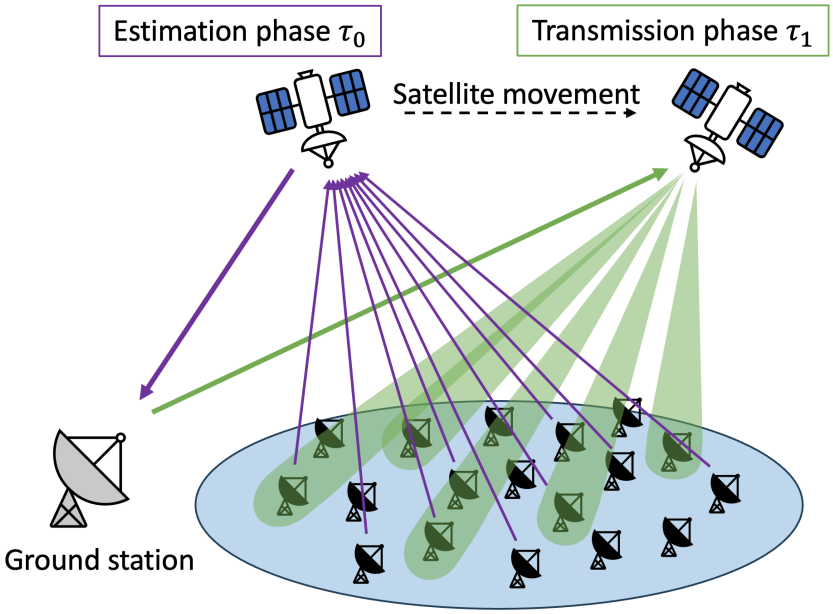


Figure 3.1: System architecture.

of not only the contributes due to the satellite's movement, but also due to correlated LoS shadowing and further additional losses that characterize NTN propagation channels. The impact of the CSI prediction on a LEO-based CF-MIMO system is assessed by means of simulations, reporting key percentiles of the capacity achieved by each UT, as well as the outage probability.

3.1 System Model

In this work, a LEO satellite is assumed to be providing connectivity from orbit to a set of UTs in a CF fashion using Feed Space (FS) digital beamforming (Figure 3.1). The satellite is equipped with an N_F-feed Uniform Planar Array (UPA), represented in Figure 3.2, with the antenna boresight directed towards the sub-satellite point. The array response of the UPA with respect to UT i, *i.e.*, in the direction represented by the pair of angles (ϑ_i, φ_i), can be written as:

$$\mathbf{a}(\vartheta_i, \varphi_i) = g_E(\vartheta_i, \varphi_i)(\mathbf{a}_H(\vartheta_i, \varphi_i) \otimes \mathbf{a}_V(\vartheta_i)), \tag{3.1}$$

where $g_E(\vartheta_i, \varphi_i)$ is the radiation pattern of an array feed, $\otimes$ represents the Kronecker product, and $\mathbf{a}_H(\vartheta_i, \varphi_i)$ and $\mathbf{a}_V(\vartheta_i)$ are the steering vectors of two uniform linear arrays, with N_H feeds for the y axis and N_V feeds for the z axis. The two steering vectors can

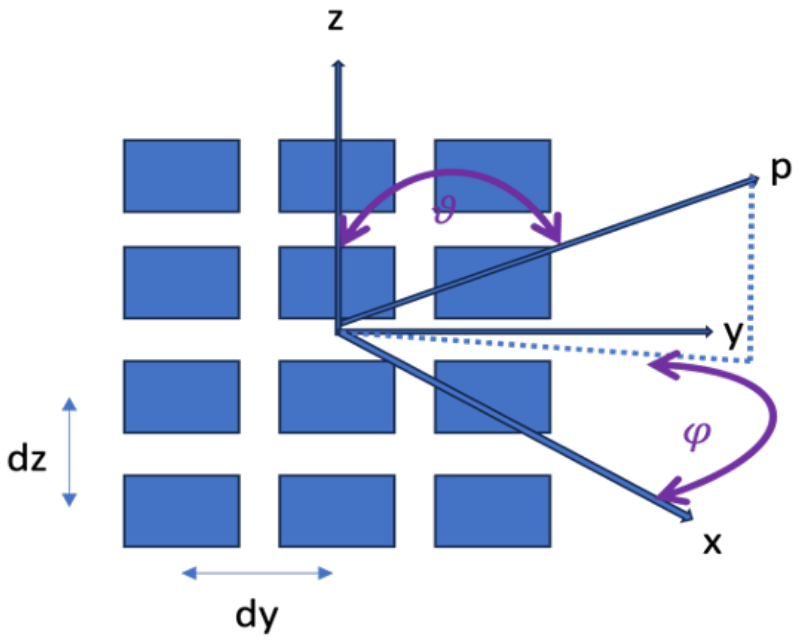


Figure 3.2: UPA antenna model [75].

be computed as:

$$\mathbf{a}_H(\vartheta_i,\varphi_i) = \left[1, e^{jk_0 d_H \sin\vartheta_i \sin\varphi_i}, \ldots, e^{jk_0 d_H (N_H-1)\sin\vartheta_i \sin\varphi_i}\right] \tag{3.2}$$

$$\mathbf{a}_V(\vartheta_i) = \left[1, e^{jk_0 d_V \cos\vartheta_i}, \ldots, e^{jk_0 d_V (N_V-1)\cos\vartheta_i}\right] \tag{3.3}$$

An on-ground gNB, which is assumed to have continuous visibility of the satellite, performs scheduling to select a subset of N_{UT} users to serve within time slot t. A coverage with a fixed N_{tier}-tiered beam pattern would allow to schedule at most one user per beam; thus, the maximum number of schedulable users per time slot is here fixed to $N_{UT} = 1 + 3N_{tier} \cdot (N_{tier} + 1)$. As the scheduling algorithm is out of the scope of this paper, a random scheduler is employed for the sake of simplicity. The gNB transmits downlink pilot symbols at time τ_0, which are used by the scheduled UTs to estimate the downlink FS CSI vector with respect to the N_F on-board feeds, $\hat{\mathbf{h}}_i$. The channel model is based on the system level 3GPP LoS propagation model [36]:

$$\hat{\mathbf{h}}_i = \tilde{h}_i^0 \times \mathbf{a}(\vartheta_i^0, \varphi_i^0), \tag{3.4}$$

$$\tilde{h}_i^0 = G_i^{(rx)} \frac{\lambda}{4\pi d_i^0} \frac{1}{\sqrt{L_i^0 \kappa B T_{sys}}} e^{-j\frac{2\pi}{\lambda} d_i^0}, \tag{3.5}$$

where $G_i^{(rx)}$ represents the i-th UT receiving gain, $\lambda = 1/f_c$ is the carrier wavelength corresponding to the carrier frequency f_c, and d_i^0 is the slant range at time τ_0. The CSI are normalized with respect to the noise power, represented by Boltzman's constant κ, the

channel bandwidth B and the receiver noise temperature T_{sys} (assumed to be the same for each UT). The term L_i^0 includes all additional losses that may incur within the i-th propagation channel at time τ_0, *e.g.*, atmospheric attenuation L_{atm}, scintillation losses L_{scint}, and LoS shadowing L_{sh} (*e.g.*, due to foliage). In general, each contribution to the additional losses is characterized by different statistics. The atmospheric attenuation in NTN represents gaseous absorption and is thus dependent on the density and type of gas molecules encountered during propagation. Similarly, scintillation losses are also linked to physical interactions between the radiated electromagnetic wave and the gas molecules in the atmosphere, in particular in the troposphere for transmissions at a carrier frequency above 6 GHz [36]. For these two contributes, the considered channel model associates empirical constant attenuation values to different ranges of elevation angles, as these type of losses vary slowly with the satellite movement. On the other hand, LoS shadowing is modeled as a log-normal zero-mean random variable with an elevation-angle-varying standard deviation. However, each sample is typically uncorrelated from another; hence, the temporal correlation of time-consecutive shadowing samples would not be represented in this model, making it not suitable to generate time series. To take the temporal correlation into account, it is assumed that each shadowing sample exhibits a correlation ρ with respect to its antecedent. Hence, the shadowing time series experienced by each link is represented by an autoregressive model of order 1 (AR(1)), *i.e.*,

$$\Lambda^t = \rho \times \Lambda^{t-1} + \epsilon^t, \tag{3.6}$$

where ϵ^t is the realization of a zero-mean Gaussian random process with variance $\sigma_\epsilon^2 = 1 - \rho^2$, such that the generated process Λ has unit variance. The covariance of the process can then be written as $[\mathbf{\Sigma}]_{i,j} = \rho^{|i-j|}$ [76], and the unit-variance shadowing time series can be obtained by sampling a zero-mean multivariate normal distribution with covariance matrix $\mathbf{\Sigma}$. Finally, the length-ν shadowing time series for UT i $\{L_{sh_i}\}_{1,\dots,\nu}$ can be obtained by multiplying each shadowing sample in the unit-variance sequence by the shadowing standard deviation specified in the system level 3GPP model, according to the corresponding elevation angle.

After the CSI vectors are reported to the ground station, the FS beamforming matrix can be computed and used to transmit user data with space division multiplexing at time $\tau_1 = \tau_0 + \Delta\tau$. $\Delta\tau$ represents the delay between the channel estimation phase and the actual data transmission:

$$\Delta\tau = \tau_{\mathrm{UT,max}} + 2\tau_{\mathrm{feeder}} + \tau_p + \tau_{ad}, \tag{3.7}$$

with $\tau_{\mathrm{UT,max}}$ being the maximum delay between the set of scheduled UTs and the satellite, τ_{feeder} represents the delay contribution on the feeder link, τ_p including all processing delays

and τ_{ad} representing possible additional losses. Clearly, as $\Delta\tau$ increases, the distance traveled by the satellite during such time interval makes the channel aging effect stronger, making the beamformed transmissions less accurate. Hence, in the proposed method, the AI-based channel prediction block is employed to project the outdated CSI amplitude $|\tilde{h}_i^0|$ to transmission time τ_1:

$$\tilde{h}_i^{AI} = f(|\tilde{h}_i^0|, d_i^1, \alpha_i^1, \mathbf{\Theta}) \times e^{-j\frac{2\pi}{\lambda}d_i^1}, \tag{3.8}$$

where d_i^1 and α_i^1 are the slant range and the elevation angle, respectively, at transmission time, and $f(\cdot)$ is a properly trained AI function with parameters $\mathbf{\Theta}$. Assuming no user mobility, d_i^1 and α_i^1 are geometric terms that can be estimated on the network side from the position of UT i and the satellite ephemeris. It must be noted that, if user mobility was to be taken into account, the message carrying the CSI feedback should also include the UT position. As in Equation 3.4, the CSI vector $\mathbf{h}_i^{AI}$ can be obtained by evaluating the UPA array response in the updated UT i direction $(\vartheta_i^1, \varphi_i^1)$. At transmission time, the N_{UT} CSI vectors, either estimated at τ_0 or predicted, are collected in a $N_{UT} \times N_F$ complex CSI matrix $\mathbf{H}$. The gNB then computes the complex $N_F \times N_{UT}$ beamforming matrix $\mathbf{W}$ using the selected beamforming algorithm. Such matrix is applied to the N_{UT} user symbols to project them onto the feed space, resulting in the formation of a dedicated beam towards each scheduled UT. After propagation, the signal received by UT i can be written as [77]:

$$y_i = \mathbf{h}_i^1 \mathbf{w}_i s_i + \sum_{\substack{k=1 \\ k\neq i}}^{N_{UT}} \mathbf{h}_i^1 \mathbf{w}_k s_k + z_i, \tag{3.9}$$

where $\mathbf{s} = [s_1, s_2, \ldots, s_{N_{UT}}]^\intercal$ is the vector of user symbols to be transmitted, $\mathbf{w}_i$ represents the beamformer dedicated to UT i (*i.e.*, the i-th column of $\mathbf{W}$), $\mathbf{h}_i^1$ represents the i-th CSI vector at transmission time τ_1, and z_i represents AWGN. Clearly, the first term of the equation represents the desired contribute, while the summation contains the co-channel interference contributes deriving from beams directed towards other UTs. From these observations, the SINR at UT i can be assessed as follows:

$$\mathrm{SINR}_i = \frac{\left\|\mathbf{h}_i^1 \mathbf{w}_i\right\|^2}{1 + \sum\limits_{\substack{k=1 \\ k\neq i}}^{N_{UT}} \left\|\mathbf{h}_i^1 \mathbf{w}_k\right\|^2}. \tag{3.10}$$

The capacity achievable by UT i can then be computed as:

$$\Gamma_i = B \cdot \log_2 (1 + \mathrm{SINR}_i). \tag{3.11}$$

Clearly, the choice of the beamforming algorithm affects the SINR, which in turn has a strong impact on the achievable user capacity. This work employs the LMMSE equation

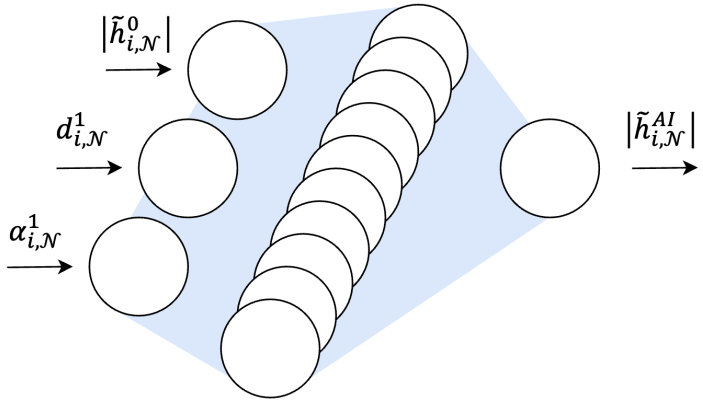


Figure 3.3: AI model for CSI prediction.

to compute $\mathbf{W}$ [77]:

$$\mathbf{W} = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H + \alpha_r \mathbf{I}_{N_{UT}})^{-1}, \tag{3.12}$$

where $(\cdot)^H$ is the Hermitian operator, $\mathbf{I}_{N_{UT}}$ is the $N_{UT} \times N_{UT}$ identity matrix, and α_r is the regularization factor, here set to 1. The beamforming matrix requires a normalization step to ensure that the payload power constraints are not broken [78]. In particular, the Maximum Power Constraint (MPC) normalization is here employed to limit the power supplied to each feed to the most stringent feed power constraint:

$$\tilde{\mathbf{W}} = \frac{\sqrt{P_{tot}}\mathbf{W}}{\sqrt{N_F \max_j [\mathbf{W}\mathbf{W}^H]_{j,j}}}. \tag{3.13}$$

While this power normalization maintains the column-wise orthogonality and ensures that all power constraints are respected, the limitation to the same power level results in the total available payload power not being fully utilized.

3.2 AI-based Channel Prediction

As highlighted in the previous section, the delay $\Delta\tau$ between channel estimation and data transmission results in a mismatch between the estimated CSI matrix $\mathbf{H}^0$, which is used in Equation 3.12 to compute the beamforming matrix, and the CSI matrix that is encountered at transmission time $\mathbf{H}^1$. In particular, considering Equation 3.4, two main sources of mismatch can be identified:

- The variation of the slant range d_i, which affects both the amplitude and the phase of the CSI;
- The losses L_i, which affect the CSI amplitude only.

Given the satellite position, which can be obtained from ephemeris data, and the UTs positions, which are here assumed to be constant and known at the network side, the gNB can estimate the slant range for each scheduled UT. Indeed, the literature has shown that similar techniques can be employed to perform MIMO beamforming without the need of a CSI feedback from the UTs, *e.g.*, [78]. Furthermore, it must be noted that eventual phase mismatches due to imperfect slant range estimations do not affect the SINR, as the complex scalar term is canceled by the norm operator in Equation 3.10. However, a CSI feedback is necessary to assess the additional propagation losses. As mentioned in the previous section, each contribute to L_i exhibits a different temporal statistic, which can be learnt and partially compensated by a deep learning algorithm. A FCNN with a single hidden layer is considered, taking as input the amplitude of the i-th CSI at estimation time at the output of the UPA $|\tilde{h}_i^0|$, and the slant range d_i^1 and elevation angle α_i^1 experienced by UT i at transmission time τ_1. The model is depicted in Figure 3.3. The input data, originated from a synthetic dataset, is standardized using the first and second order statistics of each input estimated over the entire dataset, obtaining the input vector $\mathbf{x}_{\mathcal{N}} = \left[|\tilde{h}_{i,\mathcal{N}}^0|,\ d_{i,\mathcal{N}}^1,\ \alpha_{i,\mathcal{N}}^1\right]$, where the subscript $\mathcal{N}$ indicates the data standardization. The hidden layer, composed by 10 fully-connected neurons, extracts complex features from the inputs as follows:

$$\mathbf{y}_H = \mathbf{W}_H \mathbf{x}_{\mathcal{N}} + \mathbf{b}_H, \tag{3.14}$$

where $\mathbf{x}_{\mathcal{N}}$ represents the model's input vector, and $\mathbf{W}_H$ and $\mathbf{b}_H$ are the weights matrix and bias vector obtained through the training process. The features are then fed to a BN layer, which task is to estimate mean and standard deviation of $\mathbf{y}_H$ during each training mini-batch and standardize the features. To introduce non-linearities in the network, the standardized features are processed through a LReLU activation function. Finally, the activated features are combined together in a single dense output neuron $\mathbf{y}_O$ with weights vector $\mathbf{w}_O$ and bias b_O, representing the (standardized) predicted CSI amplitude at transmission time τ_1. Assuming that the CSI amplitude has mean value μ_{CSI} and standard deviation σ_{CSI} over the training dataset, the predicted CSI amplitude of the i-th link has the following overall formulation:

$$\tilde{h}_i^{AI} = \tilde{h}_{i,\mathcal{N}}^{AI} \times \sigma_{CSI} + \mu_{CSI}, \tag{3.15}$$

$$\tilde{h}_{i,\mathcal{N}}^{AI} = \mathbf{w}_O \times LReLU(BN(\mathbf{W}_H \mathbf{x} + \mathbf{b}_H)) + b_O. \tag{3.16}$$

3.3 Simulations and Results

In the following section, the results of numerical simulations assessing the impact of the CSI prediction module on CF-MIMO systems are reported. The FCNN was implemented

Table 3.1: Simulation parameters.

Parameter	**Value**
Carrier frequency	$f_c = 20$ GHz
System band	Ka ($B = 400$ MHz)
Receiver type	VSAT
Receiver antenna gain	$G_i^{(rx)} = 39.7$ dBi
Noise figure	1.2 dB
System scenario	Sub-urban
Available power per beam	2.21 dB
Number of tiers	$N_{tier} \in \{2, 4\}$
User density	0.05 users/km^2
Total delay	$\Delta\tau = 20$ ms
Number of transmitters	$N_F = 1024$ (32×32 UPA)

Table 3.2: AI-predicted and Feedback CSI amplitude error.

Method	**Mean Error**	**Error Standard Deviation**
AI-predicted CSI	0.00 dB	2.16 dB
Feedback CSI	0.00 dB	2.49 dB

using Python's TensorFlow library [65] and trained for 100 epochs using the popular Adam optimizer, aiming at minimizing the MSE between the predicted CSI and its true value. The training, validation, and test datasets were synthetically generated using the parameters reported in Table 3.1, based on [36, 79]. It is assumed that UTs can be scheduled for an integer number of consecutive 5G NR frames of length 10 ms, and that pilot symbols for channel estimation are transmitted at the beginning of each frame. Considering Equation 3.7, the total delay for a LEO system can be assumed to be 16.7 ms [80]; hence, the total delay is set to the duration of two frames, *i.e.*, $\Delta\tau = 20$ ms.

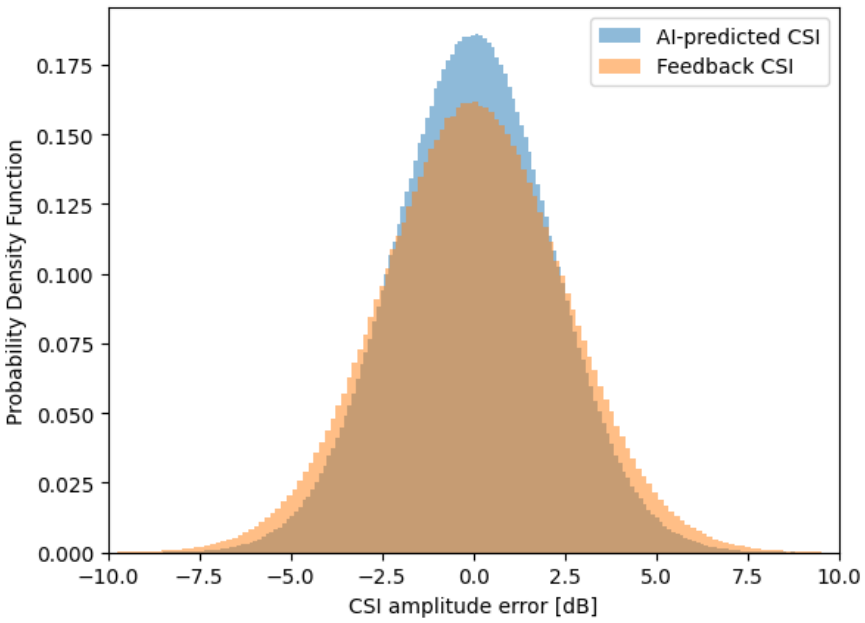


Figure 3.4: Distribution of the CSI amplitude error at τ_1.

Figure 3.4 reports the distribution of the CSI amplitude error on the test dataset, encompassing more than 4 million tests, for both the feedback CSI and the predicted CSI. The plot shows that the FCNN is able to reduce the standard deviation of the CSI amplitude error: indeed, as reported in Table 3.2, this value decreases from 2.49 dB to 2.16 dB using the CSI prediction module. The mean error remains at 0 dB for both solutions, as expected. However, further analysis is needed to assess the impact of the AI model on the CF-MIMO performance.

To this aim, system level numerical simulations of a LEO-based CF-MIMO system were carried out on MATLAB, evaluating the capacity achieved by the UTs using Equation 3.11. Figure 3.5 reports the 10, 50, and 90%-iles of the achieved user capacity. Regardless of N_{tier}, the AI model is able to greatly improve the capacity low 10%iles, increasing the capacity achieved by users with low SINR. In particular, the CSI prediction increases the achievable capacity from 107 Mbps to 119 Mbps in a 2-tiered system (an 11% improvement) and from 78 Mbps to 90 Mbps in a 4-tiered system (a 15% improvement). UTs with large SINR also experience a slight increase in capacity (90%-ile), corresponding to 1.55% and 1.52% in systems with 2 and 4 tiers, respectively. On the opposite, the median capacity is reduced from 351 Mbps to 349 Mbps in the 4-tiered system and from 278 Mbps to 275 Mbps in the 2-tiered system. However, given the magnitude of the reduction, these last results may be within margin of error. Finally, the AI channel predictions result in lower outage probabilities, *i.e.*, the probability that the SINR falls under a minimum threshold (here set to -10 dB), from 3.02% to 1.53% with 2 tiers and from 5.47% to 3.22%

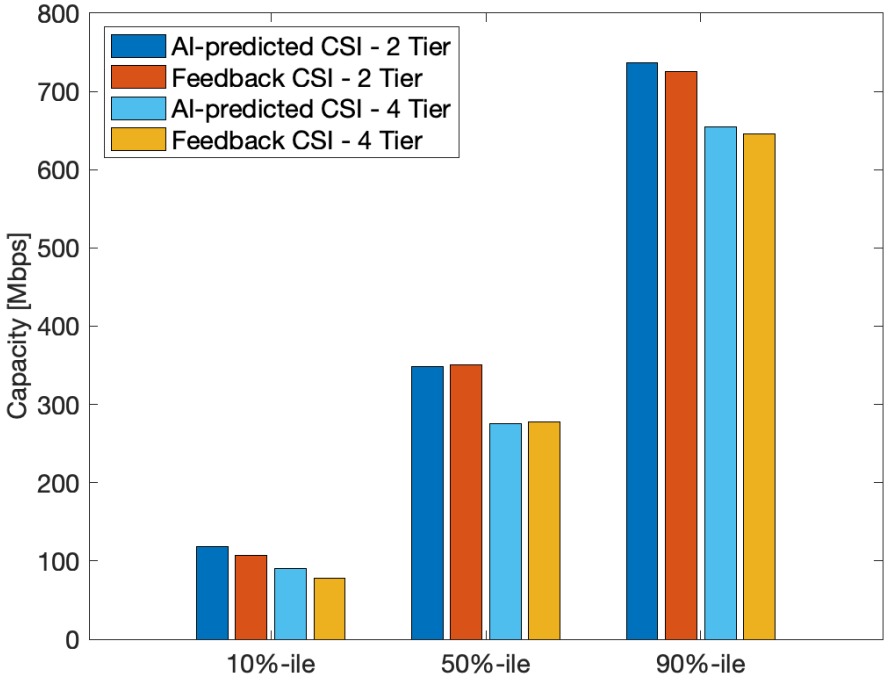


Figure 3.5: Comparison of the user capacity percentiles.

with 4 tiers.

It is clear that the AI-based channel prediction module introduces an advantage within the CF-MIMO system while requiring only a fractional increase in computational complexity. Indeed, the AI model employs only 91 parameters to predict the CSI amplitude, making it extremely lightweight. Furthermore, the prediction can be carried out for multiple UTs in a single batch, exploiting the parallelization capabilities of AI techniques. It must be mentioned that, while this work is based on a synthetic dataset, real data would be necessary to ensure that the AI model is able to cope with the real additional losses statistics. Moreover, a real implementation of the proposed technique may benefit from online learning, *i.e.*, continuously or periodically fine tuning the model using recent data. Indeed, while offline learning makes the model learn general short-term statistics, online learning allows the training process to learn and take into account local variations, *e.g.*, temporary fades due to clouds or rain. Nonetheless, this work showed that the AI model is able to learn NTN channel statistics, resulting in performance gains in LEO-based CF-MIMO systems over the simple CSI feedback.

3.4 Conclusions and future works

This study presented a lightweight AI model that refines outdated CSI estimates by learning and applying NTN channel statistics, projecting the CSI estimates to transmission

time. The performance impact of the model was evaluated at system level on a LEO-based CF-MIMO system. Numerical analyses proved that CSI prediction enables an improvement of the user capacity 10%-iles by up to 15% and a reduction of the outage probability, regardless of the number of beams that are generated by the multi-beam satellite. Future works should aim at validating the AI model performance on real data, also providing a comparison in terms of accuracy and complexity with other AI-based techniques. Further activities may also assess the performance impact of CSI prediction on user scheduling. Finally, this study motivated the investigation of CFR prediction techniques reported in the next Chapter.

Chapter 4

Channel Prediction for Pilot-free NTNs

This study was carried out during the third year of PhD under the framework of the Smart Networks and Services Joint Undertaking (SNS-JU) 6G-NTN project. The contents of this chapter are based on the papers "Channel Prediction in 6G Non-Terrestrial Networks With Deep Learning: a Physical Layer Analysis" [81], "Uplink OFDM Channel Prediction with Hybrid CNN-LSTM for 6G Non-Terrestrial Networks" [82], and "In-Orbit Channel Prediction: Deep Learning Architectures for 6G Non-Terrestrial Networks" [83], the book chapter "Deep Learning Techniques for Channel Prediction in Non-Terrestrial Networks" [84], and the project deliverable "D4.5 - Report on unified and data driven air-interface for 6G-NTN" [85].

NTNs play a pivotal role in the future wireless communications standards, enabling unprecedented coverage and a wide gamut of novel services. Due to the introduction of inherently moving nodes, NTNs are subject to a propagation environment that departs substantially from terrestrial cellular assumptions. Links to moving platforms such as LEO satellites and medium-altitude systems are characterized by large relative velocities (resulting in significant Doppler shifts) and wide-ranging delays and path losses, imprinting complex but well-defined patterns on the propagation channel. Such distinctive propagation characteristics make NTNs particularly suitable candidates for data-driven optimization approaches. In this context, AI has emerged as a key enabler to improve the efficiency of the radio resources utilization by exploiting temporal and frequency correlations in the propagation channel. Indeed, the 3GPP activities on AI for the NR air interface investigated channel prediction as a means of minimizing the need for frequent pilot signals transmission and counter channel aging. While the study focuses on CSI, which refers to specific indicators, *e.g.*, the channel quality indicator and the precoding matrix indicator, the same principle can be applied to the channel frequency response employed for data equalization at the receiving chain. Several demodulation reference signals (DM-RS) must be periodically transmitted to ensure that the receiver can main-

tain an accurate estimate of the channel: this overhead is particularly detrimental for LEO-based NTNs, where the data exchange may be required to terminate within the few minutes of visibility window of a LEO satellite, *e.g.*, due to the discontinuous coverage provided by a sparse satellite constellation. Nonetheless, the CFR presents temporal and spectral patterns that can be identified using DL-based techniques and used to predict its state in an upcoming time slot: with an accurate prediction of the CFR, no DM-RS transmission is required during the succeeding time slot, thus reducing the pilot overhead and improving the overall user throughput. This motivates a comprehensive study of the capabilities of DL architectures for channel prediction in NTNs, encompassing not only the accuracy of DL models, but also their impact on the end-to-end communication chain. In particular, a prediction-oriented resource grid is considered in an NTN system to take full benefit from the increased freedom from pilot signaling. In this framework, three candidate DL architectures are presented and their prediction accuracy and computational complexity is evaluated, including the potential for real-time inference feasibility under tight power budget constraints as in the context of NTNs. The most promising DL model is then selected and its impact on the entire communication chain in several scenarios is assessed, including those where the pre-trained channel predictor is employed under unseen channel conditions. The objective of these analysis is to show that channel prediction is a powerful technique that can enable considerable throughput gains in the next generation of non-terrestrial communication systems.

Prior works in the context of NTNs have mainly focused on the prediction of CSI, especially in the context of CSI aging and systems based on digital beamforming [86, 87, 73, 40, 38, 88]. Nonetheless, an accurate channel prediction can also enable the minimization of the pilot overhead by reducing the frequency of channel estimation. This approach was initially explored in [81] and [82], showing limited but promising results in the direction of prediction-oriented NTN systems. In these papers, two different DL architectures were tested in a simplified system model, focusing on ideal conditions for channel prediction in [81] and on key differences between the channel conditions considered during training and test in [82]. Extending these previous works, the objective of this study is to:

- Present, train, and compare the accuracy and computational complexity of three DL architectures for channel prediction, selecting the best performing one for further evaluations;
- Evaluate the accuracy drift over time obtained with the selected model, highlighting the impact of key design choices in the DL architecture;
- Assess the throughput achieved by the selected model in an end-to-end physical layer simulation considering the 5G DM-RS allocation on the OFDM resource grid;

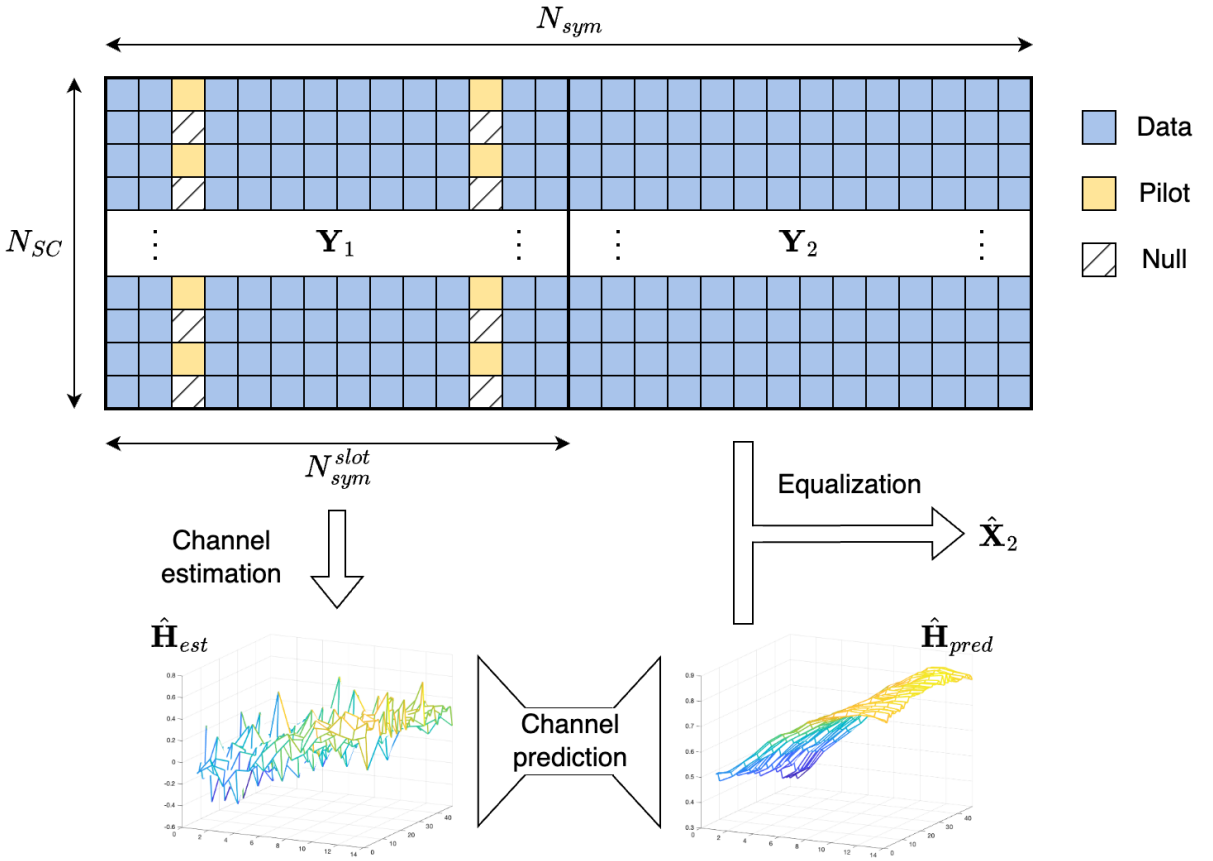


Figure 4.1: System model for channel prediction.

and

- Investigate the generalization capabilities of the selected model considering a wide gamut of tests under propagation conditions unseen during the training phase.

4.1 System model

A LEO-based NTN with a single LEO satellite hosting a full gNB on board of its regenerative payload is considered, providing connectivity to a set of UEs located on ground. Focusing on the uplink, the transmission chain at each UE is assumed to include a channel encoder, a bit interleaver, an M-ary mapper, and a CP-OFDM multiplexer. The high mobility of the NTN node results in large Doppler shifts and Doppler rate, i.e., the rate at which the Doppler shift changes [34]; under the assumption of global navigation satellite system availability at the UEs, such effects are considered to be estimated by each UE (based on the satellite's ephemeris, which can be obtained from the downlink system information blocks, and the UE position) and pre-compensated before transmission. Focusing on a single UE, a frequency allocation of N_{PRB} PRBs (composed by $N_{SC}^{PRB} = 12$

subcarriers each) for the temporal span of $N_{slot} = 2$ slots (composed by $N_{sym}^{slot} = 14$ OFDM symbols each) is assumed. Such allocation results in a transmitted resource grid $X \in \mathbb{C}^{N_{SC} \times N_{sym}}$, with $N_{SC} = N_{SC}^{PRB} N_{PRB}$ subcarriers and $N_{sym} = N_{sym}^{slot} N_{slot}$ OFDM symbols, assuming retro-compatibility with the 5G numerology (Figure 4.1). In 5G systems, at least one OFDM symbol every slot is dedicated to the transmission of DM-RS pilots, with dynamic channel conditions requiring an additional pilot OFDM symbol to enable the temporal interpolation of the channel estimates on the pilot positions. Following a similar approach as in [81] and [82], the 3rd and 11th OFDM symbol (assuming 1-based indexing) of the first slot is allocated to the transmission of pilots, with Quadrature Phase Shift Keying (QPSK) DM-RS symbols and nulls being interleaved in frequency, leading to boosted DM-RS and increased channel estimation accuracy; as a result, the first slot contains $(N_{sym}^{slot} - N_{\pi}^{slot}) \cdot N_{SC}$ data symbols, where $N_{\pi}^{slot} = 2$ is the number of OFDM symbols reserved for pilot and null transmission. On the opposite, the second slot does not contain any pilot signaling, allocating all $N_{sym}^{slot} \cdot N_{SC}$ resource elements to data transmission: their equalization will rely on the predicted CFR. The increase in number of data resource elements achieved by the considered scheme with respect to an OFDM system transmitting pilots during each of the two slots can be computed as:

$$\eta = \frac{N_{SC} \cdot (N_{sym} - N_{\pi}^{slot})}{N_{SC} \cdot N_{slot}(N_{sym}^{slot} - N_{\pi}^{slot})} - 1 = \frac{N_{\pi}^{slot}(N_{slot} - 1)}{N_{slot}(N_{sym}^{slot} - N_{\pi}^{slot})} = 8.33\%. \tag{4.1}$$

The CP-OFDM waveform carrying the resource grid is generated and transmitted by the UE. The propagation channel is modeled as a frequency-selective Tapped Delay Line (TDL) in accordance with 3GPP specifications [36]. The channel impulse response of a baseband TDL model with N_{taps} can be written as [89]:

$$h(t, \tau) = e^{j2\pi\xi t} \sum_{n=1}^{N_{taps}} h_n(t, \tau)\delta(\tau - \tau_n)e^{j(2\pi f_c \tau_n + \phi_n(t,\tau))}, \tag{4.2}$$

where the generic n-th tap is characterized by a channel gain $h_n(t, \tau)$ and a normalized delay τ_n (*i.e.*, the delay with respect to the first tap, which has absolute delay τ_0), $\delta(\cdot)$ represents the Dirac's delta function, $\xi \sim \mathcal{N}(0, \sigma_D^2)$ represents the carrier frequency offset with standard deviation σ_D due to imperfect frequency synchronization, and $\phi_n(t, \tau)$ represents additional phase offsets on the n-th path. At the receiver, which is located on board of the satellite, the waveform is further corrupted by AWGN; then, the received OFDM grid is demultiplexed, resulting in the following received grid:

$$\mathbf{Y} = \mathbf{H} \odot \mathbf{X} + \mathbf{W}, \tag{4.3}$$

where $\mathbf{H} = FFT[h(t, \tau)] \in \mathbb{C}^{N_{SC} \times N_{sym}}$ is the CFR matrix corresponding to the impulse response $h(t, \tau)$ on the considered set of subcarriers and slots, with $FFT[\cdot]$ representing

the FFT operation; $\mathbf{W} \sim \mathcal{N}(\mathbf{0}_{N_{SC}N_{sym}\times 1}, \frac{N_0}{2} \cdot \mathbf{I}_{N_{SC}N_{sym}\times N_{SC}N_{sym}})$ represents the effect of AWGN on the received grid, where $\mathbf{0}_{a\times b}$ and $\mathbf{I}_{a\times b}$ are the $a \times b$ zero matrix and identity matrix, respectively, and N_0 is the noise power spectral density; finally, $\odot$ represents the Hadamard (*i.e.*, element-wise) product. The LS channel estimator is applied on the received and transmitted symbols at the pilot positions (represented by the vectors $\mathbf{y}_p$ and $\mathbf{x}_p$, respectively), with spline interpolation being used to obtain an initial CFR estimate on all positions in the first slot [90]:

$$\hat{\mathbf{H}}_{LS} = Interp[\mathbf{y}_p \oslash \mathbf{x}_p] \in \mathbb{C}^{N_{SC}\times N_{sym}^{slot}}, \tag{4.4}$$

where $\oslash$ represent the element-wise division operation. The resulting CFR $\hat{\mathbf{H}}_{LS}$ is applied to the corresponding portion of the received grid $\mathbf{Y}_1 = \mathbf{Y}[1, \ldots, N_{SC};\ 1, \ldots, N_{sym}^{slot}]$ to equalize the symbols carried by the first OFDM slot by means of Minimum Mean Squared Error (MMSE) equalization, populating the first slot of the equalized resource grid, $\hat{\mathbf{X}}_1$:

$$\hat{\mathbf{X}}_1[m;n] = \frac{\hat{\mathbf{H}}_{LS}[m;n]^*}{|\hat{\mathbf{H}}_{LS}[m;n]|^2+N_0}\mathbf{Y}_1[m;n], \text{ with } m \in 1, \ldots, N_{SC},\ j \in 1, \ldots, N_{sym}^{slot}. \tag{4.5}$$

On the opposite, the data symbols received on the second slot $\mathbf{Y}_2 = \mathbf{Y}[1, \ldots, N_{SC};\ N_{sym}^{slot}+1, \ldots, N_{sym}]$ can be equalized with a CFR $\hat{\mathbf{H}}_{pred} \in \mathbb{C}^{N_{SC}\times N_{sym}^{slot}}$ predicted using one of the algorithms reported in Section 4.2. During the tests, it was empirically observed that the simpler zero forcing algorithm was able to reduce the error frequency on equalized symbols carried on the second slot, $\hat{\mathbf{X}}_2$, with respect to using the MMSE equalizer:

$$\hat{\mathbf{X}}_2[m;n] = \frac{\mathbf{Y}_2[m;n]}{\hat{\mathbf{H}}_{pred}[m;n]}, \text{ with } m \in 1, \ldots, N_{SC},\ j \in 1, \ldots, N_{sym}^{slot}. \tag{4.6}$$

This behavior results from the non-uniform prediction accuracy over the resource grid, with symbols farther in time intuitively being equalized with less accurate CFRs (more details provided in Section 4.3.1), and, more in general, from artifacts in the predicted CFRs generated by the predictors. Finally, the equalized data symbols carried by $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$ are demapped and the corresponding bit LLRs are fed to the LDPC decoder on a slot basis (*i.e.*, each slot carries one codeword).

4.2 Channel prediction framework

The channel prediction algorithm takes as input the CFR estimated on one slot $\hat{\mathbf{H}}_{LS}$ and provides as output a prediction of the CFR on the subsequent slot. As a pre-processing step, a refined CFR $\hat{\mathbf{H}}_{ref} \in \mathbb{C}^{N_{SC}\times N_{sym}^{slot}}$ is obtained by means of LS channel estimation with interpolation using both the demapped symbols and the DM-RS on the first slot as ground truth. On the one hand, depending on the SNR, noisy initial channel estimates

Table 4.1: Layers parameters in the CED architecture.

Layer name	**Kernels**	**Kernels size**	**Stride**	**Padding/Cropping**
Conv2D_A1	8	$[9 \times 5]$	$[3 \times 3]$	$[0, 0, 0, 0]$
Conv2D_A2	2	$[7 \times 3]$	$[1 \times 1]$	$[0, 0, 0, 0]$
TConv2D_A1	8	$[3 \times 3]$	$[1 \times 1]$	*Same*
TConv2D_A2	2	$[7 \times 3]$	$[3 \times 3]$	$[0, 0, 0, 0]$
Conv2D_A3	2	$[9 \times 5]$	$[3 \times 3]$	$[0, 0, 0, 0]$

could introduce demodulation errors at the receiver, which result in quantized errors on the refined CFR matrix: in the case of QPSK symbols, for example, phase shifts of either $\pi/2$ or π may be added to the refined CFR depending on the severity of the initial CFR inaccuracy; with higher order modulations, amplitude errors can also be introduced. Nonetheless, such quantized errors can be identified and compensated by the channel predictor. On the other hand, this pre-processing step allows general CFR dynamics hidden by the scarce amount of DM-RS to emerge, providing additional information to the channel predictors. Once $\hat{\mathbf{H}}_{ref}$ is computed, its real and imaginary parts are stacked together, forming the input tensor $\hat{\mathbf{H}}_{in} \in \mathbb{R}^{N_{SC} \times N_{sym}^{slot} \times 2}$. Similarly, the output of each predictor $\hat{\mathbf{H}}_{out} \in \mathbb{R}^{N_{SC} \times N_{sym}^{slot} \times 2}$ contains the stacked real and imaginary parts of the predicted CFR, which can be reconstructed as $\hat{\mathbf{H}}_{pred} = \hat{\mathbf{H}}_{out}[1, \ldots, N_{SC};\ 1, \ldots, N_{sym}^{slot};\ 1] + j \cdot \hat{\mathbf{H}}_{out}[1, \ldots, N_{SC};\ 1, \ldots, N_{sym}^{slot};\ 2]$.

4.2.1 Prediction models

Three DL architectures for channel prediction are here considered, where each predictor takes as input a batch of N_{batch} input CFRs and computes the corresponding batch of predicted CFRs.

Convolution-based encoder-decoder

The Convolution-based Encoder-Decoder (CED) architecture, presented in [81], performs the CFR prediction by computing a compact representation of the input CFR (Figure 4.2). In particular, a first encoding section is tasked with the extraction of the low-level features, employing 2-Dimensional Convolutional (Conv2D) layers with no padding and different stride values. A complete list of the parameters of each layer is reported in Table 4.1. Given $N_{SC} = 48$ and $N_{sym}^{slot} = 14$, at the output of the encoding section

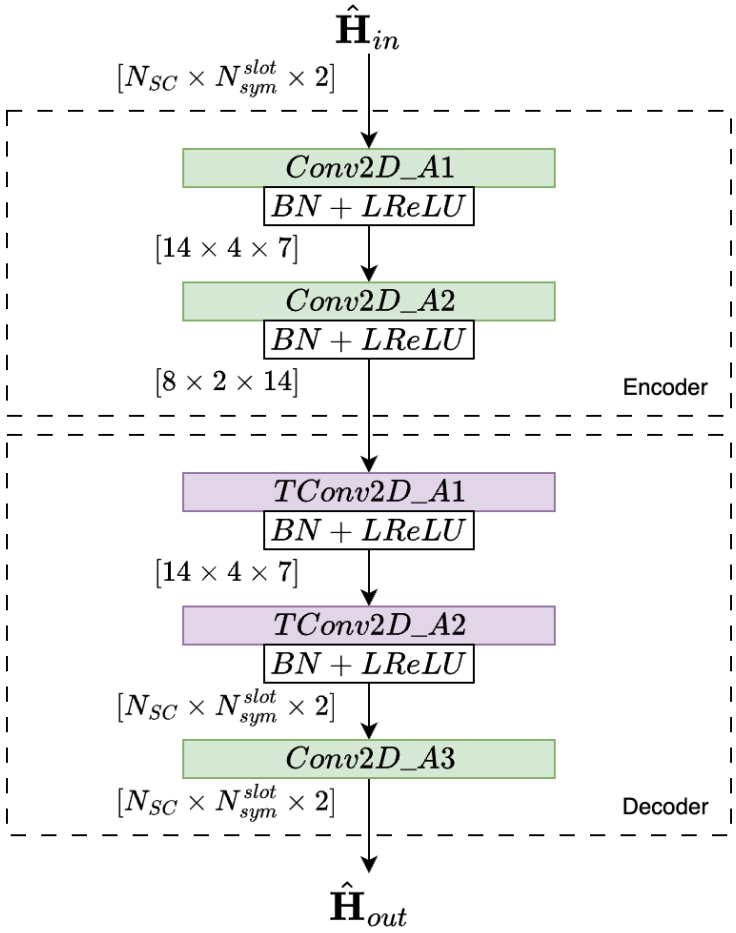


Figure 4.2: Convolution-based encoder-decoder architecture.

is a compact tensor of size $[8, 2, 14, N_{batch}]$ containing all the channel trends identified in the frequency and temporal domain in the input matrix. Based on the extracted features, the decoder builds the predicted CFR matrix, decompressing the compact low-level representation obtained by the channel encoder. For this task, Transposed Conv2D (TConv2D) layers are implemented: these layers apply rectangular filters to each element of the input tensor, effectively mirroring the compression carried out by Conv2D layers in the encoder. To reduce the checkerboard pattern that often affects the output of TConv2D layers, a smoothing filter is employed through a Conv2D layer with "same" padding (i.e., a Conv2D layer that preserves the tensor's size) using "replicate" pattern, i.e., the value of padding is equal to the value of the elements at the closest boundary of the tensor. Throughout the model, BN layers maintain the data distribution under control and LReLU activation functions introduce non-linearities in the model. All layers' parameters are reported in Table 4.1.

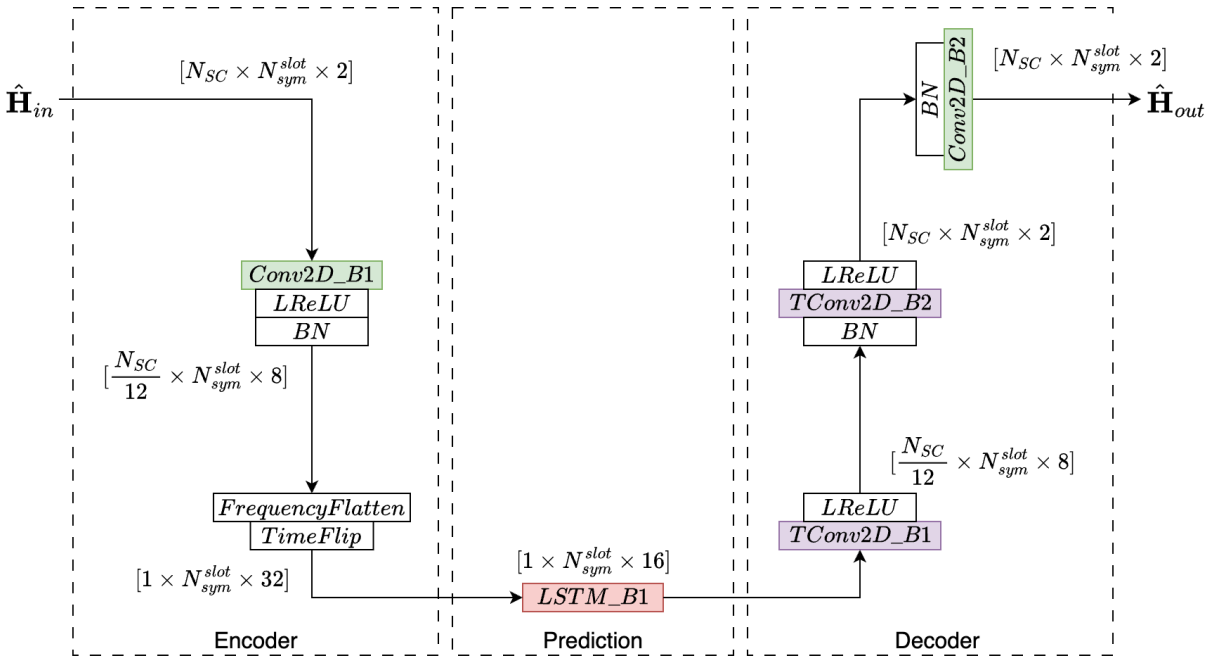


Figure 4.3: Hybrid CNN-LSTM architecture.

Hybrid CNN-LSTM

Similarly to CED, the Hybrid CNN-LSTM architecture maintains Conv2D layers and TConv2D layers in an encoding and decoding section, respectively, while also introducing an LSTM layer dedicated to the prediction (Figure 4.3) [82].In the encoder, a single Conv2D layer identifies complex patterns in time and frequency and adds them as features to the tensor. The frequency dimension is then flattened to the channel dimension, resulting in a tensor of size $[14 \times 32 \times N_{batch}]$. The tensor is mirrored on the temporal axis, *i.e.*, the last sample of the tensor becomes the first, and vice versa: this is required in light of the LSTM-based prediction, as the sequential nature of such layer would make the channel estimates predicted over the first OFDM symbol depend on the first input sample, *i.e.*, the most outdated one. On the opposite, the TimeFlip operation allows the LSTM to begin the prediction from the most recent sample to the most outdated one: effectively, the model learns and applies the inverse temporal trend of the CFR matrix; the impact of such design choice is reported in Section 4.3.1. The LSTM layer can then carry out the prediction task, leveraging its recurrent nature to identify patterns in the extracted low-level features. Similarly to CED, TConv2D layers are employed to restore the size of the output tensor $\hat{\mathbf{H}}_{out}$, while a Conv2D layer with $[3 \times 3]$ kernels acts as a smoothing filter for an improved CFR prediction accuracy. It should be noted that, differently from [82], skip connections are not implemented here as they were found to be detrimental to the prediction accuracy. One interpretation for such effect is that pilot boosting results in

Table 4.2: Layers parameters in the Hybrid CNN-LSTM architecture.

Layer name	**Kernels/Cells**	**Kernels size**	**Stride**	**Padding/Cropping**
Conv2D_B1	8	$[6 \times 5]$	$[12 \times 1]$	$[0, 0, 1, 1]$
Conv2D_B2	2	$[3 \times 3]$	$[1 \times 1]$	*Same*
LSTM_B1	16	N/A	N/A	N/A
TConv2D_B1	8	$[4 \times 3]$	$[1 \times 1]$	$[0, 0, 1, 1]$
TConv2D_B2	2	$[12 \times 3]$	$[12 \times 1]$	$[0, 0, 1, 1]$

more accurate channel estimates, which in turn can make the prediction simpler to carry out and the Conv2D layers on skip connections redundant from the prediction point of view. All layers' parameters are reported in Table 4.2.

Temporal convolutional network

Temporal Convolutional Networks (TCNs) have been introduced in [91] as a feed-forward alternative to RNNs. In TCNs, residual blocks with 1-dimensional convolutions, implemented with Conv2D layers with one-dimensional kernels, filter an input time series. To increase the receptive field of the output samples, the Conv2D layers employ a different dilation factor in each residual block; a typical choice for the generic n-th block is $d_n = 2^{n-1}$, $n \geq 1$. With CFRs being estimated slot-wise, the input time series is not streamed: thus, regular convolutions rather than causal convolutions are here implemented. As in the previous two models, encoding and decoding sections are necessary to maintain the computational complexity low. Thus, the overall structure of the model is similar to that of U-Net [18] (Figure 4.4). Differently from the previous two models, Average Pooling (AvgPool) and Unpooling (AvgUnpool) layers are employed at the encoder for compression and at the decoder for expansion, respectively. On the one hand, AvgPool layers act as averaging filters over the frequency-time dimensions, effectively reducing their input tensor size; on the other hand, AvgUnpool layers with size $[S_f \times S_t]$ repeat each frequency-time element by S_f times on the frequency axis and S_t on the temporal axis. Thus, Conv2D layers are the only trainable layers implemented, employing "same" padding to not affect the tensor size. The prediction is taken care of by the TCN, which comprises of two Conv2D-based residual blocks, and is supported by Conv2D layers on skip connections at different depths. Features on skip connections are concatenated channel-wise to the main section of the model by means of Depth Concatenation (DepthConc). The parameters for the layers employed in the TCN architecture are reported in Table 4.3.

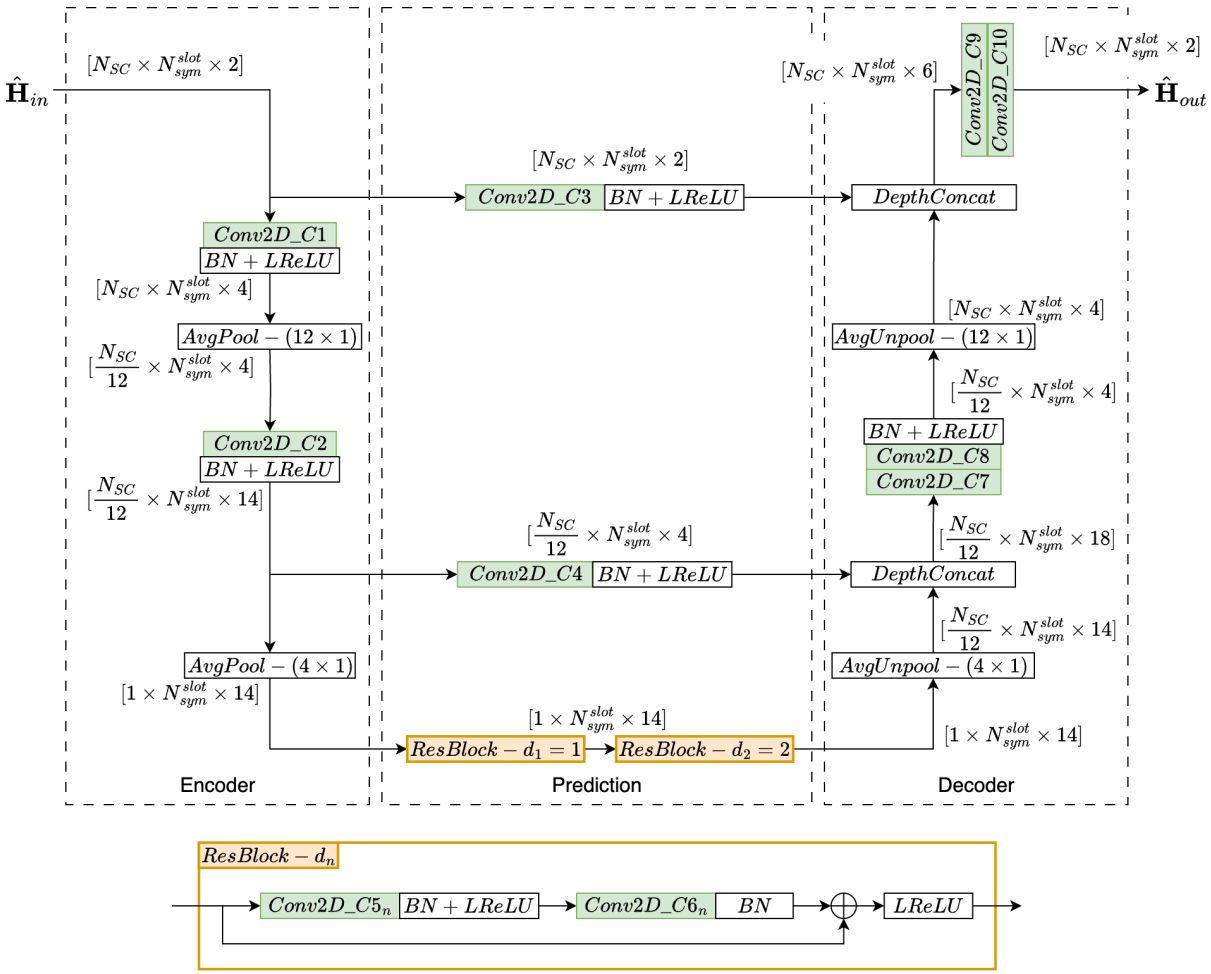


Figure 4.4: Temporal convolutional network architecture.

Table 4.3: Layers parameters in the TCN architecture.

Layer name	**Kernels**	**Kernels size**	**Stride**	**Dilation**	**Padding/Cropping**
Conv2D_C1	4	$[5 \times 1]$	$[1 \times 1]$	1	*Same*
Conv2D_C2	14	$[3 \times 1]$	$[1 \times 1]$	1	*Same*
Conv2D_C3	2	$[1 \times 3]$	$[1 \times 1]$	1	*Same*
Conv2D_C4	4	$[1 \times 3]$	$[1 \times 1]$	1	*Same*
Conv2D_C5$_1$	14	$[1 \times 3]$	$[1 \times 1]$	1	*Same*
Conv2D_C6$_1$	14	$[1 \times 3]$	$[1 \times 1]$	1	*Same*
Conv2D_C5$_2$	14	$[1 \times 3]$	$[1 \times 1]$	2	*Same*
Conv2D_C6$_2$	14	$[1 \times 3]$	$[1 \times 1]$	2	*Same*
Conv2D_C7	4	$[3 \times 1]$	$[1 \times 1]$	1	*Same*
Conv2D_C8	4	$[1 \times 3]$	$[1 \times 1]$	1	*Same*
Conv2D_C9	2	$[5 \times 1]$	$[1 \times 1]$	1	*Same*
Conv2D_C10	2	$[1 \times 5]$	$[1 \times 1]$	1	*Same*

4.2.2 Training

The training process involves generating on-demand training data for each model, *i.e.*, a batch of N_{batch} synthetic samples is generated at each epoch according to the system model described in Section 4.1; such data is then used to train the model for the current epoch (*e.g.*, as in the popular library Sionna [92]). With this strategy, the risk of overfitting on a dataset is avoided, making validation a *nice to have* rather than a *must have*. Clearly, overfitting to the training data assumptions, *e.g.*, the channel model considered, including the LoS condition and the Doppler spread, is still a risk: thus, the generalization capabilities of a predictor also need to be assessed (see Sections 4.3.2, 4.3.2, and 4.3.2). Considering a batch of N_{batch} input CFRs and labels, stacked on the last dimension, the models are trained to minimize the MSE loss function:

$$MSE = \frac{1}{N_{batch} N_{SC} N_{sym}^{slot}} \sum_{k=1}^{N_{batch}} \sum_{m=1}^{N_{SC}} \sum_{n=1}^{N_{sym}^{slot}} \left(\hat{\mathbf{H}}_{pred}[m; n; 1; k] - Re\{\mathbf{H}[m; N_{sym}^{slot} + n; k]\} \right)^2 + \left(\hat{\mathbf{H}}_{pred}[m; n; 2; k] - Im\{\mathbf{H}[m; N_{sym}^{slot} + n; k]\} \right)^2 \quad (4.7)$$

The Adam optimizer is employed to update the weights [93]. To ensure good performance at various SNR working points, the samples within each batch are generated at different E_b/N_0, with E_b representing the average energy per bit and N_0 the noise power spectral density of the receiver. In particular, to reduce the prediction accuracy error floor in non-noise-limited conditions, the E_b/N_0 of the n-th sample, γ_n, is sampled from the range $\Gamma = [0, 1, \ldots, 10]$ in dB according to the following probability mass function:

$$P_{\gamma_n}(i) = Prob\{\gamma_n = \Gamma(i)\} = \frac{i^2}{\sum_{k=1}^{|\Gamma|} k^2}. \tag{4.8}$$

The following training techniques are also implemented:

- **Learning rate warm-up**. While simple decay schedules are often used to monotonically decrease the learning rate λ, it has been observed that the training process sometimes benefits from setting the initial value of λ to a very low value λ_{min} and linearly ramping it up to a large value λ_{max} in the span of a few training epochs T_w [94]. This technique, called "learning rate warm-up", often helps in starting the gradient descent from the random weight initialization point towards the optimal direction.

- **Learning rate cosine annealing with warm restarts**. With this schedule, the learning rate follows a half-cosine trend, oscillating from λ_{max} to λ_{min} in T_c epochs:

$$\lambda(i) = \lambda_{min} + \frac{1}{2}(\lambda_{max} - \lambda_{min}) \left(1 + cos\left(\frac{mod(i, T_c)}{T_c}\pi\right)\right), \tag{4.9}$$

  where $\lambda(i)$ is the learning rate at the i-th training epoch from the start of the schedule (*e.g.*, after finishing the learning rate warm-up in this case). Notably, the $mod(\cdot)$ operator resets the learning rate to λ_{max} once T_c epochs have passed from the previous warm (*i.e.*, that does not re-initialize the model's weights) restart.

- **Early stopping**. The training process is typically stopped when the training loss has not improved for a few epochs. Given that cosine annealing tends to temporarily increase the loss after each warm restart, the implemented version of early stopping terminates the training process after T_s full cosine annealing cycles without a training loss improvement, restoring the weights to the values providing the smallest loss.

- **L2 regularization**. The loss function is penalized by a factor equal to the sum of the squared weights scaled down by a constant ϵ_R. While this technique is usually implemented to prevent overfitting, it was empirically observed that the models would benefit from a very small amount of L2 regularization rather than not implementing the technique at all.

Table 4.4: Simulation parameters (values in bold are to be considered the default value, except where stated otherwise).

Parameter	**Value**
Maximum Monte Carlo iterations	10^5 iterations
Modulation order	$M \in \{4, \mathbf{16}, 64\}$
Code rate	$R_c = 3/4$
Carrier frequency	$f_c = 2$GHz
Maximum UE speed	$v_{UE} \in \{\mathbf{5}, 30, 50\}$ km/h
Fading model	{NTN-TDL-C, NTN-TDL-A} [36]
Carrier frequency offset standard deviation	$\sigma_D = \frac{f_c \cdot 0.1 \cdot 10^{-6}}{3} \approx 66.67$ Hz [82]
Delay spread	30 ns
Number of subcarriers	$N_{SC} = 48$
Subcarrier spacing	$\Delta f = 15$ kHz
Satellite altitude	600 km

The optimal values for λ_{min} and λ_{max} can be obtained using the learning rate range test, where a DL model is trained for a few training steps and the learning rate is increased at each step [95]. λ_{min} and λ_{max} can then be chosen from the loss vs learning rate plot, selecting the range of learning rate values that provide a monotonic loss reduction.

4.3 Results and discussion

The performance of the proposed channel predictors was evaluated by means of end-to-end simulations on the MATLAB computing environment. Table 4.4 reports the main simulation parameters, while Table 4.5 reports the training parameters. The transceiver employs the M-QAM digital modulation and the low-density parity-check coding scheme; as in [82], the standard deviation of the carrier frequency offset is set according to the 3σ rule, such that ξ remains within the 5G requirement of 0.1 parts per million of the carrier frequency f_c 99.7% of the times. Each DL model is trained twice, one time for each considered fading model; Table 4.6 separately reports the minimum and maximum learning rate for each training instance.

Table 4.5: Training parameters.

Parameter	**Value**
Training batch size	$N_{batch} = 2048$
Maximum training epochs	10^4
L2 regularization factor	$\epsilon_R = 10^{-6}$
Learning rate warm-up period	$T_w = 40$ epochs
Learning rate cosine annealing period	$T_c = 100$ epochs
Early stopping patience	$T_s = 3$ full cosine annealing periods
Training E_b/N_0 values	$\{0, 1, \ldots, 10\}$ dB

Table 4.6: Minimum and maximum learning rate.

Architecture	**Fading model**	$\lambda_{\mathbf{min}}$	$\lambda_{\mathbf{max}}$
CED	NTN-TDL-C	10^{-4}	$8 \cdot 10^{-3}$
	NTN-TDL-A	10^{-4}	$4 \cdot 10^{-3}$
Hybrid CNN-LSTM	NTN-TDL-C	10^{-4}	$8 \cdot 10^{-3}$
	NTN-TDL-A	10^{-4}	$6 \cdot 10^{-3}$
TCN	NTN-TDL-C	10^{-4}	$5 \cdot 10^{-3}$
	NTN-TDL-A	10^{-4}	$3 \cdot 10^{-3}$

4.3.1 Models evaluation

The first assessment consists of an evaluation of DL-oriented metrics, *i.e.*, the computational complexity and the prediction accuracy achieved by the models presented in Section 4.2.1.

Computational complexity

With the considered framework, the channel predictors are assumed to be running on gNBs on board of satellites. While online learning frameworks are not considered here, two main constraints on the inference complexity should be taken into account:

1. The inference complexity should be low enough to ensure that one inference can run in real time (*i.e.*, the inference latency should not exceed the OFDM slot duration $T_{slot} = 1$ ms); and

2. The inference complexity should be low enough to ensure that the power consumption for inference does not impact the strict LEO satellites power budget.

The number of MACs required for inference are here estimated for each prediction model. The MACs represent the number of multiplications and additions that are carried out within a DL model, thus reflecting its computational complexity. Such value is here estimated by counting the number of multiplications carried out in each layer; contributions from layers without trainable parameters are considered negligible.

- **Conv2D**. The number of MACs for a Conv2D layer with C kernels of size $[W_f \times W_t]$ having input and output tensors $\mathbf{T}_{in} \in \mathbb{R}^{L_f^{(i)} \times L_t^{(i)} \times N^{(i)}}$ and $\mathbf{T}_{out} \in \mathbb{R}^{L_f^{(o)} \times L_t^{(o)} \times C}$, respectively, can be estimated as [81]:

$$MAC_{Conv2D} = L_f^{(o)} L_t^{(o)} N^{(i)} W_f W_t C. \tag{4.10}$$

- **TConv2D**. The number of MACs for a TConv2D layer with C kernels of size $[W_f \times W_t]$ having input and output tensors $\mathbf{T}_{in} \in \mathbb{R}^{L_f^{(i)} \times L_t^{(i)} \times N^{(i)}}$ and $\mathbf{T}_{out} \in \mathbb{R}^{L_f^{(o)} \times L_t^{(o)} \times C}$, respectively, can be estimated as [81]:

$$MAC_{TConv2D} = L_f^{(i)} L_t^{(i)} N^{(i)} W_f W_t C. \tag{4.11}$$

- **LSTM**. The number of MACs for an LSTM layer with U hidden units and as many cells as the input/output sequences length L_t, having input and output tensors $\mathbf{T}_{in} \in \mathbb{R}^{L_t^{(i)} \times N^{(i)}}$ and $\mathbf{T}_{out} \in \mathbb{R}^{L_t^{(o)} \times U}$, respectively, can be estimated as [96]:

$$MAC_{LSTM} = L_t U \left(4N^{(i)} + 4U + 3\right). \tag{4.12}$$

Table 4.7: Number of MACs, trainable parameters count, and inference latency estimate for each prediction architecture.

Architecture	**MACs**	**Trainable parameters**	**Inference latency**
CED	161k	5.5k	$\approx$ 1.05 ms
Hybrid CNN-LSTM	**138k**	5.6k	$\approx$ **0.90 ms**
TCN	155k	**3.3k**	$\approx$ 1.02 ms

Based on the MAC count and a selected hardware platform, not accounting for real world factors (*e.g.*, the impact of data quantization), a rough estimate for the inference latency can be obtained. In [67], an accelerator for DL inference consuming only 100 mW of power was embedded on board of a nano UAV, making it suitable for inference on board of LEO satellites. Considering this hardware platform, a convolution-based DL model with 1.1M MACs was shown to be able to run at an inference rate of 139 inferences/s, corresponding to an inference latency of approximately 7.19 ms/inference. Scaling down such figure based on the difference in MACs, an estimate for the prediction inference latency can be obtained for each considered architecture. Table 4.7 reports this figure, together with the number of MACs and the trainable parameters; the best values for each metric are shown in bold. While the TCN architecture is the most compact among the presented ones, requiring only 3.3k trainable parameters, the lowest computational complexity is achieved by the Hybric CNN-LSTM architecture, boasting only 138k MACs (*i.e.*, less than 35k MACs per PRB) and an inference latency of just 0.90 ms, less than the real-time threshold of $T_{slot} = 1$ ms. While it should be stressed that this figure is the result of several simplifications, it is a promising result in the direction of real-time channel prediction in NTNs. Furthermore, it should be noted that the remaining two architectures provide an inference latency only slightly larger than the 1 ms mark, making them potentially suitable for real-time inference through the implementation of complexity reduction techniques, *e.g.*, weight pruning [66], which are out of the scope of this thesis.

Prediction accuracy

The channel prediction accuracy is evaluated in terms of MSE as a function of the E_b/N_0 of the communication link. The assessment was carried out for each of the six trained models on a dataset comprising 81920 samples per E_b/N_0 working point. Figures 4.5 and 4.6 report the results of the evaluation over the NTN-TDL-C and NTN-TDL-A fading models, respectively. In both cases, an estimation-limited region can be observed

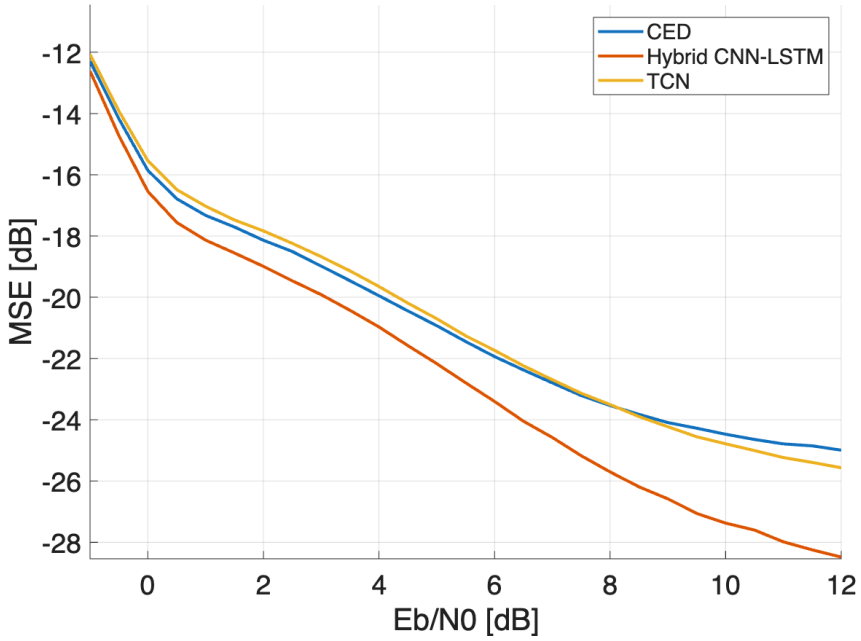


Figure 4.5: MSE vs E_b/N_0 with matching training and test conditions (test on $M = 16$, NTN-TDL-C, $v_{UE} = 5$ km/h).

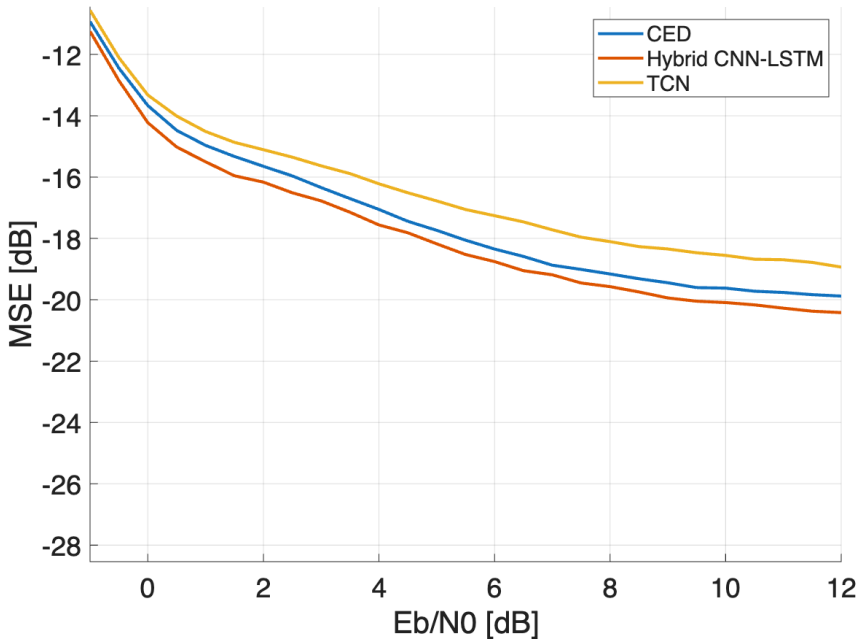


Figure 4.6: MSE vs E_b/N_0 with matching training and test conditions (test on $M = 16$, NTN-TDL-A, $v_{UE} = 5$ km/h).

until $E_b/N_0 = 0$ dB, where all considered predictors perform similarly due to the low accuracy of the input channel estimates under such high noise conditions. For higher $E_b/N_0 = 0$, the Hybrid CNN-LSTM is the architecture that consistently provides the smallest MSE at each E_b/N_0 working point, reaching an MSE of -23.40 dB at 6 dB of E_b/N_0 on the NTN-TDL-C fading model (a 1.5 dB and 1.7 dB better MSE than the CED and TCN architectures, respectively). Given the higher complexity of the NTN-TDL-A fading model with respect to NTN-TDL-C (no line of sight, more paths), the prediction accuracy is limited, with the Hybrid CNN-LSTM model being the only predictor able to cross the -20 dB MSE mark in the considered E_b/N_0 range.
Based on these results, the Hybrid CNN-LSTM architecture was selected for the remaining part of the performance assessment. Before proceeding with the evaluation of the user throughput, an additional analysis focused on the MSE evolution over time was carried out with the Hybrid CNN-LSTM architecture, separately evaluating the MSE of the predicted CFR at each OFDM symbol index. Two variants of the chosen architecture are also trained and included in this evaluation:

1. A variant with skip connections, originally presented in [82]; and
2. A variant without the TimeFlip layer.

Figure 4.7 reports the results of the assessment with NTN-TDL-C fading at an E_b/N_0 of 6 dB, including results for maximum UE speeds ranging from 5 km/h (the value used during training) up to 50 km/h. At 5 km/h, the MSE with the regular Hybrid CNN-LSTM architecture presented in Section 4.2.1 (blue continuous curve) grows from -25.2 dB to -21.4 dB over the span of one time slot. The inclusion of skip connections (yellow continuous curve) seems to have a negative impact on the MSE, leading to a 1 dB gap in MSE by the end of the OFDM slot. In contrast, the orange continuous curve proves that removing the TimeFLip layer from the base model has a detrimental impact on the MSE: while predictions at the end of the OFDM slot benefit from a rich memory in the LSTM cells, the output at the first portion of the slot is based on just a few outdated samples of the estimated CFR, leading to an MSE over the first symbol index of just -17 dB. A similar trend can be observed when increasing the maximum UE speed to 30 km/h (dash-dotted curves) and 50 km/h (dashed curves). As the maximum UE speed increases, the Doppler spread widens, and the coherence time of the propagation channel reduces: intuitively, this results in all prediction MSE curves being characterized by a steeper inclination as the maximum UE speed increases, with the MSE at the end of the OFDM slot on the 5 km/h case being 3 dB and 7 dB lower than the 30 km/h and 50 km/h cases, respectively.

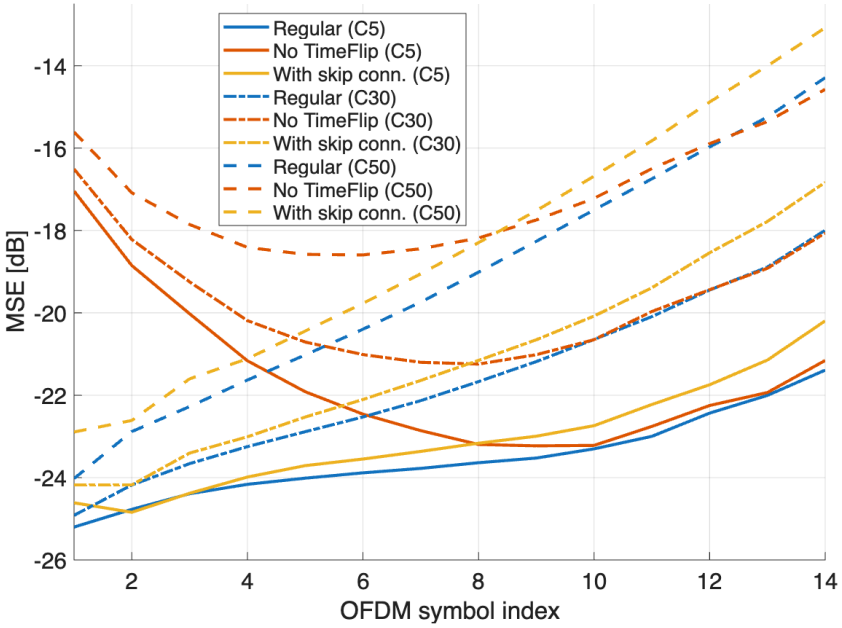


Figure 4.7: MSE vs OFDM symbol index with variants of the Hybrid CNN-LSTM architecture (test on $E_b/N_0 = 6$ dB, $M = 16$, NTN-TDL-C, $v_{UE} \in \{5, 30, 50\}$ km/h).

4.3.2 User throughput

This Section reports the results of the end-to-end assessment of the achieved user throughput with channel prediction. Recalling the system model reported in Section 4.1, the BLER on the transport blocks received on the first and second slot are referred to as $BLER_1$ and $BLER_2$, respectively. Based on this, the user throughput in the case of a regular estimation-based OFDM system can be computed as:

$$TP_{est} = \frac{m \cdot R_c \cdot N_{SC} \cdot (N_{sym}^{slot} - N_{\pi}^{slot})}{T_{slot}} \cdot (1 - BLER_1), \tag{4.13}$$

where R_c is the LDPC code rate and $m = log_2(M)$ is the number of bits per M-QAM symbol. Then, the average user throughput in the proposed system TP_{pred} can be evaluated as the average between TP_{est} and the throughput over the prediction-aided slot, resulting in the following formulation:

$$TP_{pred} = \frac{TP_{est}}{2} + \frac{1}{2}\frac{m \cdot R_c \cdot N_{SC} \cdot N_{sym}^{slot}}{T_{slot}}(1 - BLER_2). \tag{4.14}$$

Matching propagation conditions

Figure 4.8 reports the user throughput achieved under the NTN-TDL-C fading model with a UE speed of 5 km/h by the Hybrid CNN-LSTM architecture (continuous curves)

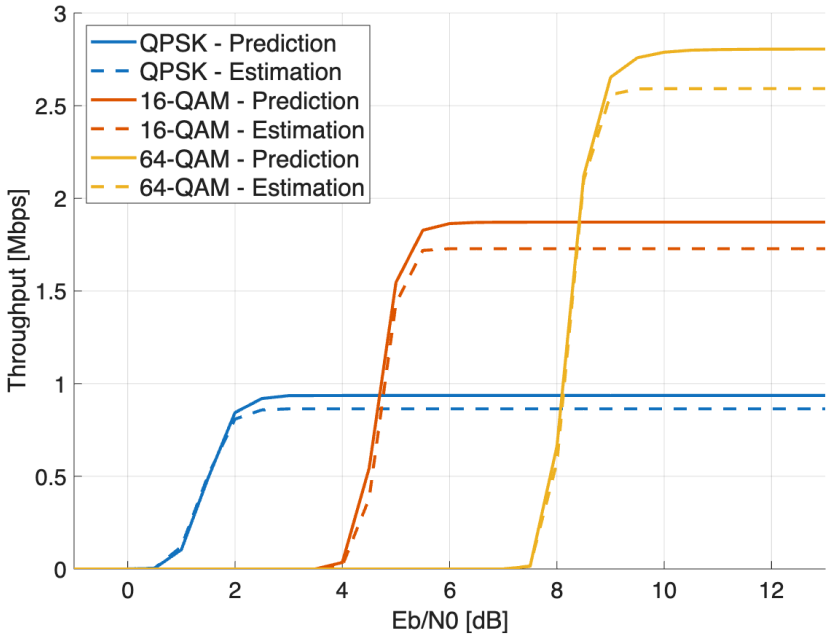


Figure 4.8: Throughput vs E_b/N_0 with matching training and test conditions (test on $M \in \{4, 16, 64\}$, NTN-TDL-C, $v_{UE} = 5$ km/h).

and by the estimation-based system (dashed curves). Regardless of the modulation order, the proposed model provides a consistent throughput gain with respect to the regular system model. With QPSK modulation (blue curves), significant gains are obtained from $E_b/N_0 = 2$ dB, with the theoretical maximum gain $\eta = 8.33\%$ being approached at 1 dB later. With 16-QAM-modulated data, the orange dashed curve shows that using channel estimation only results in a user throughput of 1.72 Mbps at 5.5 dB of E_b/N_0, while the prediction-based system (orange continuous curve) increases such value by 6.33%, while the 8.33% gain mark is approached at $E_b/N_0 = 6.5$ dB. Similarly, the curves representing 64-QAM data transmission (depicted in yellow) show that the Hybrid CNN-LSTM predictor is able to provide strong throughput gains even with high order modulations. Notably, while the prediction pre-processing step introduce quantized errors in the refined CFR that depend on the modulation order, the plots show that the Hybrid CNN-LSTM is able to generalize well with respect to this parameter. Figure 4.9 reports the results of the same assessment carried out on the Hybrid CNN-LSTM trained and tested under the NTN-TDL-A fading model. Here, similar trends can be observed as in Figure 4.8; however, the performance of the prediction-based system with 64-QAM-modulated data is reduced by 0.5 dB of E_b/N_0 with respect to the estimation-based system, with the prediction gains being observable only from $E_b/N_0 = 10$ dB. This is the result of a combination of factors: in general, as shown by the MSE assessment in Section 4.3.1,

channel prediction under the NTN-TDL-A channel model is less accurate; furthermore, the 64-QAM modulation on the one hand requires higher accuracy with respect to lower modulation orders because of the smaller distance between the constellation symbols, and on the other hand introduces a discrepancy between the quantized errors observed during training (related to 16-QAM) and those encountered during this test. Because of these effects, the peak throughput gain cannot be reached is not reached with 64-QAM data under these conditions.

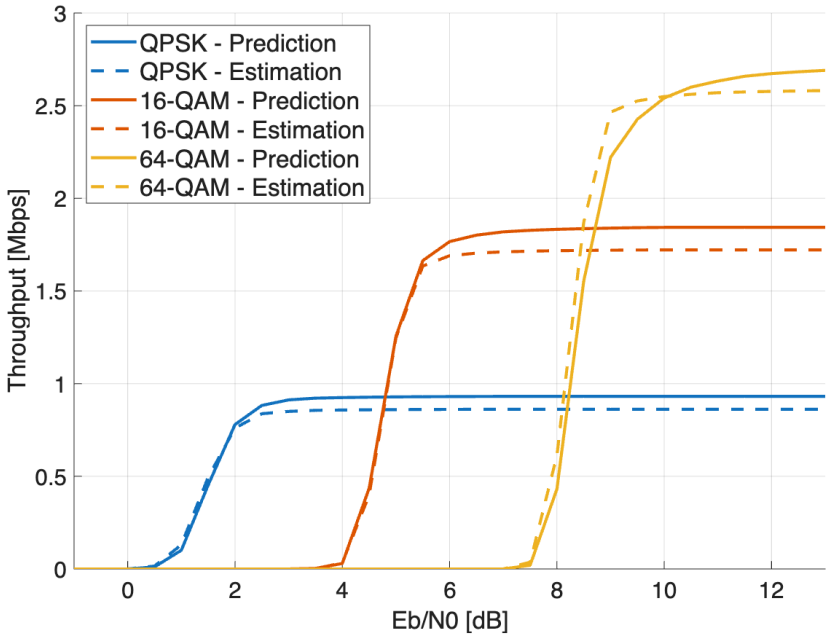


Figure 4.9: Throughput vs E_b/N_0 with matching training and test conditions (test on $M \in \{4, 16, 64\}$, NTN-TDL-A, $v_{UE} = 5$ km/h).

Mismatching UE speed

While Section 4.3.1 hinted at the possibility of running the proposed predictor in real time with low power on a LEO satellite, this does not take into account any online learning framework. For this reason, it should be ensured that a pre-trained model can be deployed in a gNB as is, *i.e.*, that it performs well under a wide range of propagation conditions. In Figure 4.10, the throughput achieved under mismatching UE speed under NTN-TDL-C channel is reported. Despite the more complex dynamics, the predictor is able to consistently increase the throughput even at a maximum speed of 50 km/h, although achieving lower gains with respect to lower maximum UE speeds and not being able to reach the theoretical maximum throughput gain (black dash-dotted line, corresponding to 1.87 Mbps). On the other hand, the performance on the NTN-TDL-A fading model

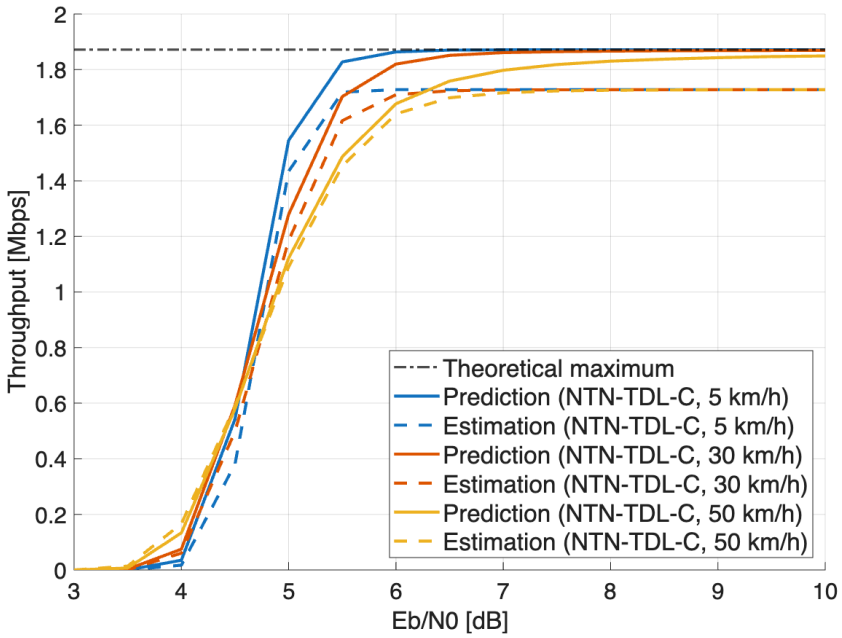


Figure 4.10: Throughput vs E_b/N_0 with mismatching training and test data UE speed (test on $M = 16$, NTN-TDL-C, $v_{UE} \in \{5, 30, 50\}$ km/h).

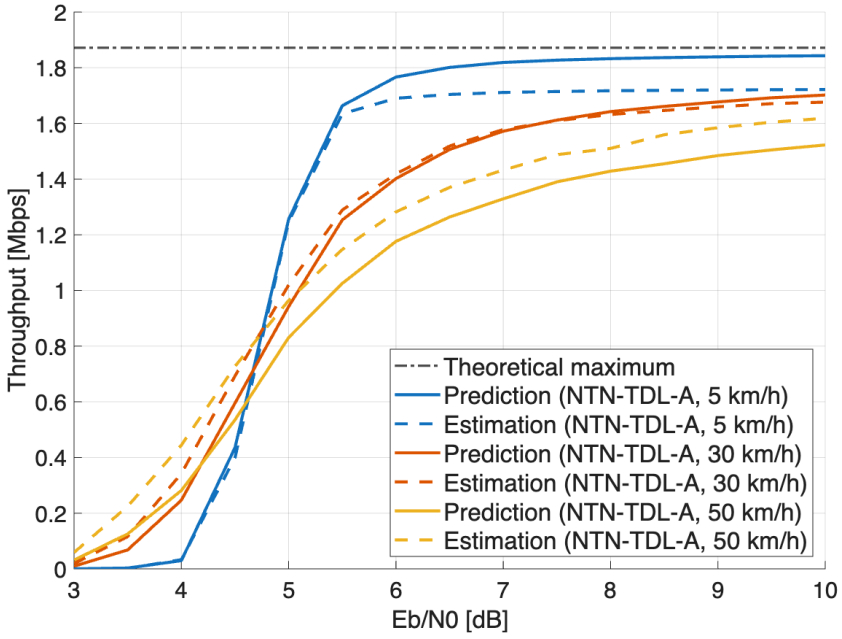


Figure 4.11: Throughput vs E_b/N_0 with mismatching training and test data UE speed (test on $M = 16$, NTN-TDL-A, $v_{UE} \in \{5, 30, 50\}$ km/h).

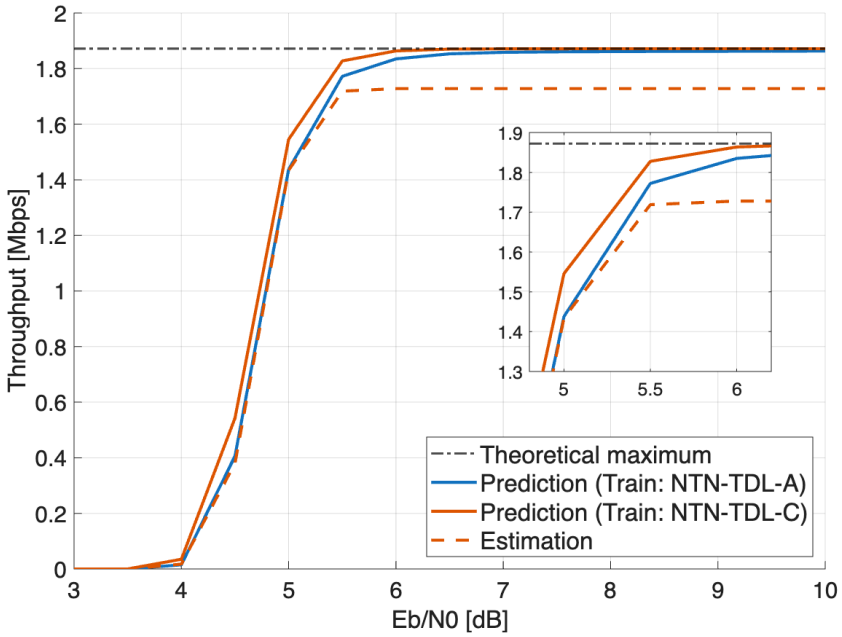


Figure 4.12: Throughput vs E_b/N_0 with mismatching training and test data fading model (test on $M = 16$, NTN-TDL-C, $v_{UE} = 5$ km/h).

(Figure 4.11) are significantly impacted by the increased channel dynamics, with the Hybrid CNN-LSTM model not being able to provide any throughput gain at 30 km/h of maximum UE speed and even resulting in a loss at 50 km/h. Nonetheless, it should be mentioned that NTNs are typically designed under the assumption of the LoS presence, while the NTN-TDL-A fading model only considers secondary paths.

Mismatching fading model

Due to the time-varying geometry that characterizes LEO-based NTNs, with moving UEs, gNBs on board of satellites, and, in general, scatterers, the line of sight may be temporarily lost. For this reason, it should be ensured that the models are able to generalize well to fading models with different characteristics than those utilized for training. Therefore, Figures 4.12 and 4.13 include the throughput achieved by the two trained Hybrid CNN-LSTM models (one on NTN-TDL-C data, the other on NTN-TDL-A data) on the NTN-TDL-C and the NTN-TDL-A fading models, respectively. The blue curve in Figure 4.12, corresponding to the mismatched prediction model (*i.e.*, trained on NTN-TDL-A data), shows that models trained on non-LoS (NLoS) data can generalize well to LoS fading models, resulting in a user throughput on par with that of the estimation-based system (orange dashed curve) until 5 dB of E_b/N_0 and consistent gains at higher SNRs. Due to the function approximation nature of neural networks, a BLER error floor emerges from

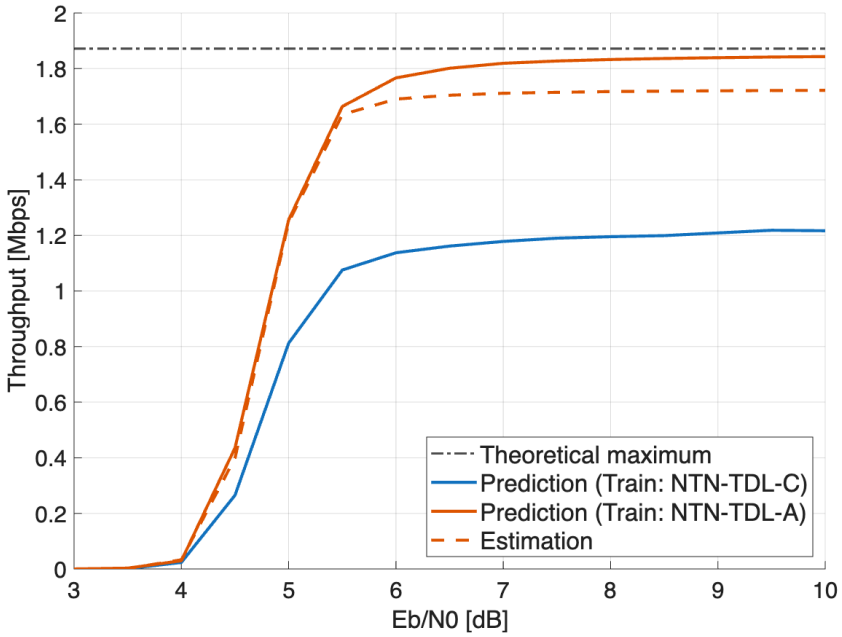


Figure 4.13: Throughput vs E_b/N_0 with mismatching training and test data fading model (test on $M = 16$, NTN-TDL-A, $v_{UE} = 5$ km/h).

the data mismatch, reducing the maximum throughput gain to 7.82% at $E_b/N_0 = 10$ dB. On the opposite, Figure 4.13 shows that prediction models trained on the NTN-TDL-C fading model rely on the LoS path to achieve a low MSE: in the absence of such primary component, the achieved throughput struggles to cross the 1.2 Mbps mark, a 30% loss with respect to the traditional estimation-based system.

Mismatching UE speed and fading model

The analysis carried out in the previous Section is here extended to a mismatch in both the fading model and the maximum UE speed. In this assessment, the focus is on the model trained under NTN-TDL-A fading and test it on NTN-TDL-C data considering a maximum UE speed of 30 km/h. The results reported in Figure 4.14 prove that the proposed predictor trained with a double dataset mismatch (blue curve) is effective at all E_b/N_0 working points, with consistent throughput gains from $E_b/N_0 = 5.5$ dB. The introduction of the training fading model mismatch over the maximum UE speed mismatch provides minimal losses, with a gap of less than 0.5 dB appearing between the continuous blue and orange curves until 6.5 dB of E_b/N_0 and the maximum throughput gain reached being 2 percentage points lower than the theoretical one (6.34% vs 8.33%).

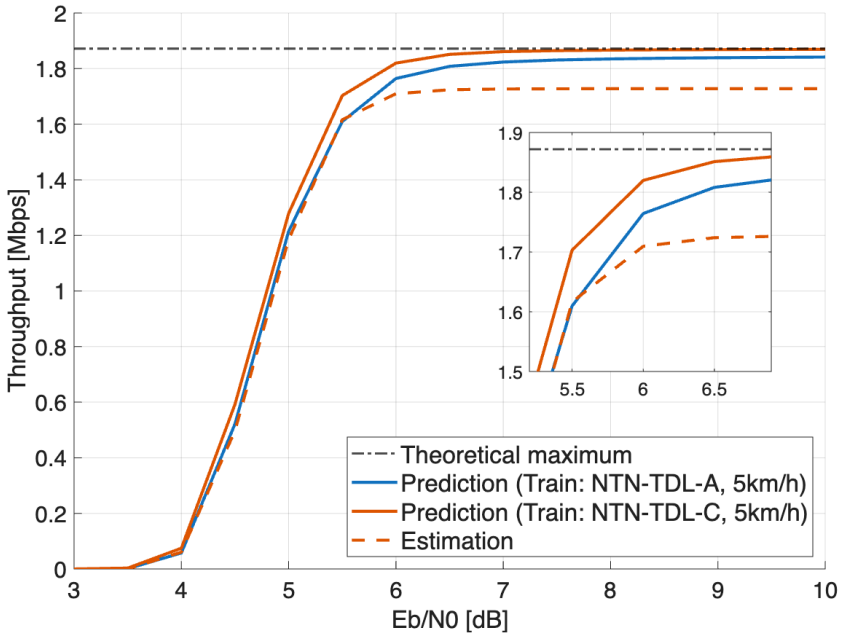


Figure 4.14: Throughput vs E_b/N_0 with mismatching training and test data fading model and UE speed (test on $M = 16$, NTN-TDL-C, $v_{UE} = 30$ km/h).

4.3.3 Iterative channel prediction

Finally, this Section extends the system model presented in Section 4.1 to the more general case of $N_{slot} > 1$ consecutive time slots, where the first includes DM-RS pilots and the remaining $N_{slot} - 1$ include only data symbols. In this scenario, the data symbols carried by the resource grid received at the i-th slot, $\mathbf{Y}_i \in \mathbb{C}^{N_{SC} \times N_{sym}^{slot}}$, are equalized using the CFR predicted during the previous slot, $\hat{\mathbf{H}}_{pred,i} \in \mathbb{C}^{N_{SC} \times N_{sym}^{slot}}$, resulting in the set of data symbols $\hat{\mathbf{X}}_i \in \mathbb{C}^{N_{SC} \times N_{sym}^{slot}}$, which can be processed by the demapper and the remaining stages of the receiver. At the same time, $\hat{\mathbf{X}}_i$ and $\mathbf{Y}_i$ are fed to the channel refinement block, which performs the preprocessing step reported in Section 4.2 to generate the refined CFR $\hat{\mathbf{H}}_{ref,i} \in \mathbb{C}^{N_{SC} \times N_{sym}^{slot}}$. Based on this, the channel prediction algorithm can produce a new predicted CFR $\hat{\mathbf{H}}_{pred,i+1} \in \mathbb{C}^{N_{SC} \times N_{sym}^{slot}}$, which will be used during the succeeding slot. The complete chain for the i-th slot is reported in Figure 4.15. Clearly, the channel prediction accuracy may be limited by the propagation of errors from a slot to another; nonetheless, the chain can theoretically continue indefinitely. Considering (4.1), the asymptotic theoretical throughput gain over an estimation-based system η_∞ can be obtained by considering an infinite number of slots $N_{slot} \to \infty$ through the following equation:

$$\eta_\infty = \lim_{N_{slot} \to \infty} \eta = \lim_{N_{slot} \to \infty} \frac{N_\pi^{slot}(N_{slot} - 1)}{N_{slot}(N_{sym}^{slot} - N_\pi^{slot})} = \frac{N_\pi^{slot}}{N_{sym}^{slot} - N_\pi^{slot}} \approx 16.67\%. \tag{4.15}$$

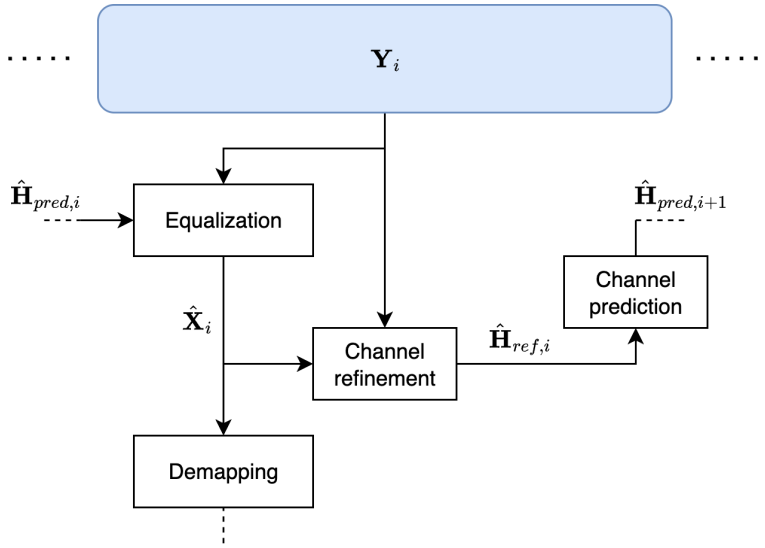


Figure 4.15: Iterative channel prediction strategy.

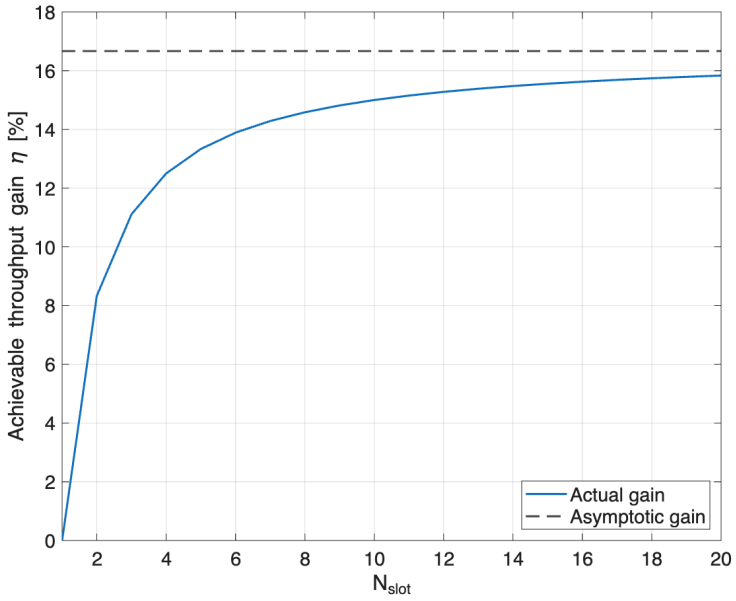


Figure 4.16: Achievable throughput gain vs N_{slot}.

Figure 4.16 reports the achievable gain as a function of N_{slot}. Notably, considering $N_{slot} = 10$, corresponding to the span of one 5G NR frame with the considered numerology, the throughput gain with respect to an estimation-based system reaches 15%, less than 2 percentage points from η_∞. The pretrained Hybrid CNN-LSTM model considered in the previous Sections is here employed without any change to its structure; the simulation parameters are the same reported in Table 4.4.

Prediction accuracy

As in Section 4.3.1, the accuracy is here evaluated through the MSE metric; the number of slots is limited to $N_{slot} = 4$. Figures 4.17 and 4.18 report the mesh plot of the MSE as a function of the OFDM symbol index (limited to the $N_{slot} - 1 = 3$ prediction-based slots) and the E_b/N_0 for the NTN-TDL-C and NTN-TDL-A fading models considering $M = 16$ and $v_{UE} = 5$ km/h in matching channel conditions. The mesh plot shows that, in LoS conditions, each iterative prediction stage (OFDM symbol indices 28 to 29 and from 42 to 43) strongly increases the MSE error floor. In particular, at high SNR ($E_b/N_0 = 10$ dB) the MSE degrades from -25.3 dB to -18.7 dB when predicting the third slot's CFR from the second slot's, and from -15.4 dB to -10.4 dB when predicting the fourth slot's CFR from the third slot's. Furthermore, the steepness of the MSE over time also reduces as the prediction covers slots further in time. In NLoS conditions (Figure 4.18), the MSE inter-slot discontinuities appear less evident: this is the result of the increased channel complexity, leading to far worse prediction accuracy than in LoS conditions. As an example, the MSE over slot 3 (OFDM symbol indices 29 to 42) degrades from -17.7 dB to -13.0 dB at 10 dB of E_b/N_0, a 4.7 dB gap over the span of one slot (compared to 3.3 dB over the same slot in LoS conditions). Intuitively, the throughput gains in NLoS conditions may be too challenging to provide the theoretical throughput gain reported in Figure 4.16.

User throughput

The throughput achieved with the considered iterative prediction structure can be evaluated by extending (4.14) to the case of $N_{slot} - 1$ pilot-less slots. In particular, denoting with $BLER_s$ the BLER achieved on the s-th slot, the average throughput considering a total of N_{slot} slots can be computed as:

$$TP_{pred}^{N_{slot}} = \frac{TP_{est}}{N_{slot}} + \frac{N_{slot} - 1}{N_{slot}} \frac{m \cdot R_c \cdot N_{SC} \cdot N_{sym}^{slot}}{T_{slot}} \sum_{s=2}^{N_{slot}} (1 - BLER_s). \tag{4.16}$$

Figures 4.19 and 4.20 report the throughput achieved in LoS and NLoS conditions, respectively, for N_{slot} ranging from 2 (coinciding with the case considered in Section 4.3) to

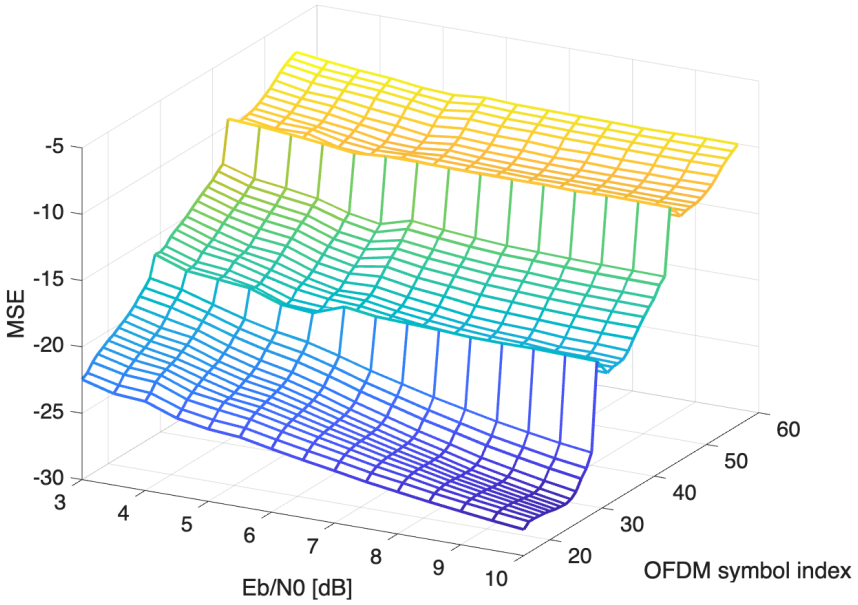


Figure 4.17: MSE vs OFDM symbol index vs E_b/N_0 with iterative prediction (test on $M = 16$, NTN-TDL-C, $v_{UE} = 5$ km/h).

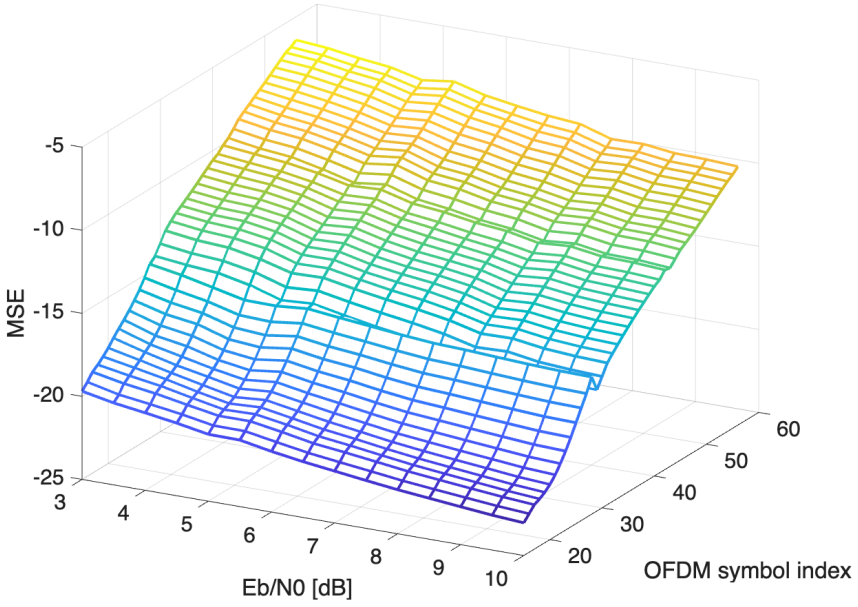


Figure 4.18: MSE vs OFDM symbol index vs E_b/N_0 with iterative prediction (test on $M = 16$, NTN-TDL-A, $v_{UE} = 5$ km/h).

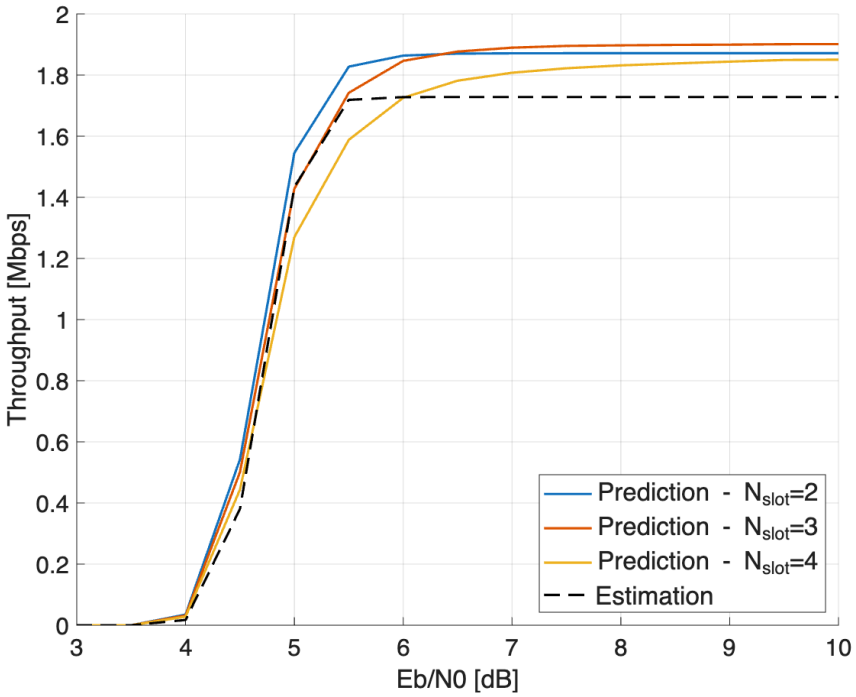


Figure 4.19: Throughput vs E_b/N_0 with matching training and test conditions and iterative prediction (test on $M = 16$, NTN-TDL-C, $v_{UE} = 5$ km/h).

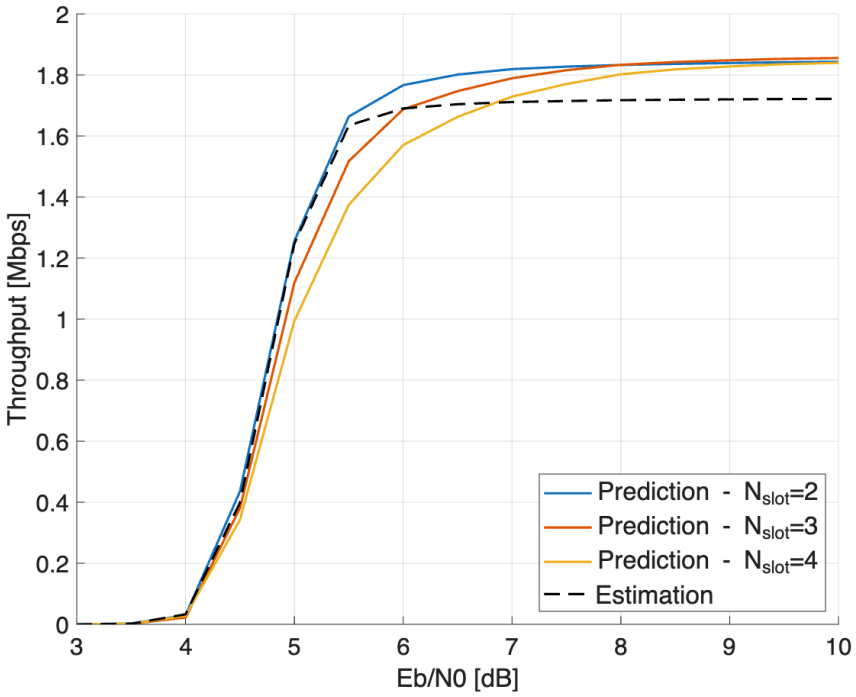


Figure 4.20: Throughput vs E_b/N_0 with matching training and test conditions and iterative prediction (test on $M = 16$, NTN-TDL-A, $v_{UE} = 5$ km/h).

4. In LoS conditions, the plot shows that for $N_{slot} = 3$ the prediction accuracy is still high enough to provide a throughput gain over the 2 slots case, *i.e.*, the case without iterative prediction. In particular, the average throughput is increased from 1.87 Mbps to 1.90 Mbps, representing an overall gain with respect to the estimation case of 9.8%: clearly, the reduced accuracy over the third slot results in a reduced $BLER_3$, leading to suboptimal performance (Figure 4.16 shows that the achievable throughput gain for $N_{slot} = 3$ would be 11.1%). Furthermore, while the system is still able to provide consistent gains with respect to the estimation-based system at any E_b/N_0 working point, it fails at improving the throughput against a non-iterative channel prediction at low SNR. This is exacerbated when the prediction range is expanded to the case $N_{slot} = 4$ (yellow curve), where the BLER over the last slot is so low that the average throughput falls under the estimation curve (dashed black curve) at low E_b/N_0 and is not able to reach the single iteration throughput (blue curve) in the considered E_b/N_0 range. In the NLoS case, the iterative prediction results in severe error propagation, resulting in the iterative prediction system failing to provide any meaningful throughput gain over the single iteration curve at any number of consecutive slots. Clearly, the Hybrid CNN-LSTM predictor requires a few adaptation to enable the iterative prediction; some ideas for future improvements will be discussed in the following Section.

4.4 Conclusions and future works

This study explored the application of DL to channel prediction as a means to reduce the pilot overhead in 6G NTNs. In the reported framework, DM-RS and null symbols are substituted with data in every other slot, thus making equalization rely on predicted channel coefficients when pilots are not transmitted. Three DL architectures were considered, namely the CED, the Hybrid CNN-LSTM, and the TCN, all of which process the CFR estimated over a slot to predict the CFR over the succeeding one. The computational complexity and the MSE achieved by the predictors under LoS and NLoS fading models were evaluated, showing that the Hybrid CNN-LSTM provides not only the lowest complexity out of the 3 models, but also the lowest MSE at any E_b/N_0 working point. It should also be noted that, taking simplifications of real-world conditions into account, the Hybrid CNN-LSTM model has the potential to run in real time on low-power hardware accelerators. With end-to-end simulations, it was shown that the chosen predictor provides consistent throughput gains in both LoS and NLoS conditions, reaching a maximum gain of 8.33% with respect to a regular OFDM system with various QAM modulation orders. Under the assumption of offline inference, *i.e.*, no online learning frameworks, it was also proven that the pre-trained model generalizes well to different

Doppler spreads when in LoS and is able to perform accurate predictions with minimal throughput losses when trained with NLoS samples and tested under LoS fading. As a result, channel-prediction-based systems should either 1) implement a predictor trained on NLoS data, which can also provide satisfying throughput gains in LoS with marginal losses with respect to the matched model, or 2) employ NLoS detection mechanisms to dynamically load the model trained on NLoS data when the LoS is temporarily lost, and vice versa. Furthermore, performance monitoring is required to ensure fail-safe mechanisms within the AI Lifecycle Management (LCM), *e.g.*, triggering fallbacks to orthogonal communications and requesting the delivery of updated models from dedicated entities [8]. Nonetheless, the capabilities of the Hybrid CNN-LSTM model in the real world should also be evaluated, taking into account further impairments, *e.g.*, imperfect temporal synchronization, non-linearities introduced by the power amplifier, and the Doppler drift due to the satellite's movement, employing either a channel emulator or, ideally, a measurement campaign with a LEO satellite. Furthermore, the model's inference latency should be assessed through a hardware implementation on a dedicated low power accelerator to ensure that the real-time constraint of 1 ms can be met. Finally, the iterative prediction system should be improved by reducing the error propagation from one slot to the succeeding one: this may be achievable by, in general, increasing the overall prediction accuracy (*e.g.*, by introducing a lightweight CNN dedicated to the CFR refinement step or by improving the predicted CFR's smoothness with the addition of a loss penalty), or by introducing curriculum learning techniques where, after converging to an adequate CFR prediction training accuracy on the i-th slot, the training focus is moved to the succeeding slot. With these techniques, the asymptotical throughput gain (*i.e.*, for $N_{slot} \to \infty$) of $\eta_\infty \approx 16.67\%$ may be approached.

Chapter 5

Channel Prediction for NOMA in NTNs

This study was carried out during the third year of PhD. The contents of this chapter are based on the paper "Channel Prediction with Temporal Convolutional Networks: A New Paradigm to Enable Non-orthogonal Multiple Access in 6G NTN" [97].

NOMA, introduced in 3GPP Rel.16, has shown promising gains in system throughput and user capacity in terrestrial networks [98]; however, existing designs may not be a good fit for NTNs due to unique challenges such as highly dynamic topologies and channel impairments. In LEO-based NTN systems, a twofold challenge emerges. First, the high mobility of LEO satellites complicates accurate CSI acquisition and tracking at the receiver [80]. Second, when satellite constellations are sparse, the visibility window, *i.e.*, the time frame for which the UE is able to communicate with the satellite, limits the time available for data transmission. On one hand, NOMA offers a promising solution by enabling multiple users to transmit simultaneously over shared resources, thereby increasing spectral efficiency and user access within the short visibility period, outperforming orthogonal systems under such constraints. On the other hand, NOMA systems are more sensitive to channel estimation errors than traditional systems, resulting in significant performance losses when accurate channel estimates are not available. Typically, preambles and/or DM-RSs are used for channel estimation. Their orthogonality is essential for estimation accuracy; however, it also reduces the NOMA throughput. Alternatively, low-correlation sequences have been proposed to support more users, at the expense of estimation performance. Thus, the design of the channel slot, *i.e.*, the allocation of data and pilot resources, remains an open research area in NOMA. An alternative approach to channel estimation relies on AI, which has the potential to learn from real data and perform accurate predictions, despite typically requiring larger computational capabili-

ties than channel estimators. As shown in the previous Chapter, channel prediction can be used to anticipate changes in channel conditions at an upcoming instant or over a time frame, offering the possibility of reducing the number of pilots. Nonetheless, its application to NTN systems can also be used to circumvent the design of non-orthogonal DM-RSs, thus enabling NOMA schemes. At the time of writing this thesis, the application of AI-based channel prediction to enable NOMA has received limited attention in the literature. Therefore, the objective of this study is to:

- design a channel prediction oriented channel structure to enable accurate NOMA transmissions with reduced orthogonal DM-RS and no non-orthogonal pilots;
- evaluate the capabilities of channel prediction in the context of NOMA. With channel prediction, non-orthogonal data transmissions can take place without the need for non-orthogonal pilots, enabling a system throughput increase compared to orthogonal systems;
- integrate and test the proposed solution with SCMA under realistic NTN channel models.

5.1 System model

In this work, an uplink transmission is considered where users on ground send their data to a gNB on-board of a LEO satellite. The transmitted signals are non-orthogonal and overlap in time and frequency according to the SCMA resource allocation. In the proposed system, users are assumed to be equipped with GNSS receivers to be able to pre-compensate the Doppler shift and keep track of the Doppler rate resulting from the mobility of the considered NTN node, and to operate in Radio Resource Control (RRC)-connected mode under a grant-based access paradigm. As such, NOMA resources are pre-assigned and scheduled. However, the channel coefficients between each UE and the satellite differently affect each transmitted signal. Due to the strong dependency of the SCMA receiver on the knowledge of the channel coefficients, AI-based channel prediction is employed to obtain accurate channel estimates and reduce the DM-RS overhead at the same time; to this aim, the slot structure proposed in the previous Chapter is adapted to the SCMA scheme, resulting in the slot structure reported in Figure 5.1. Each user is allocated N_{PRB} PRBs (each one consisting of 12 consecutive sub-carriers) according to the SCMA FG; *e.g.*, based on the signature matrix reported in 2.1, UE2 applies its spreading sequences on PRB1 and PRB3, while PRB1 carries the transmissions from UE2, UE3, and UE5. The first slot carries non-overlapping user-specific mini slots (each composed of four OFDM symbols) and DM-RS symbols in type-2 configuration, ensuring mutual

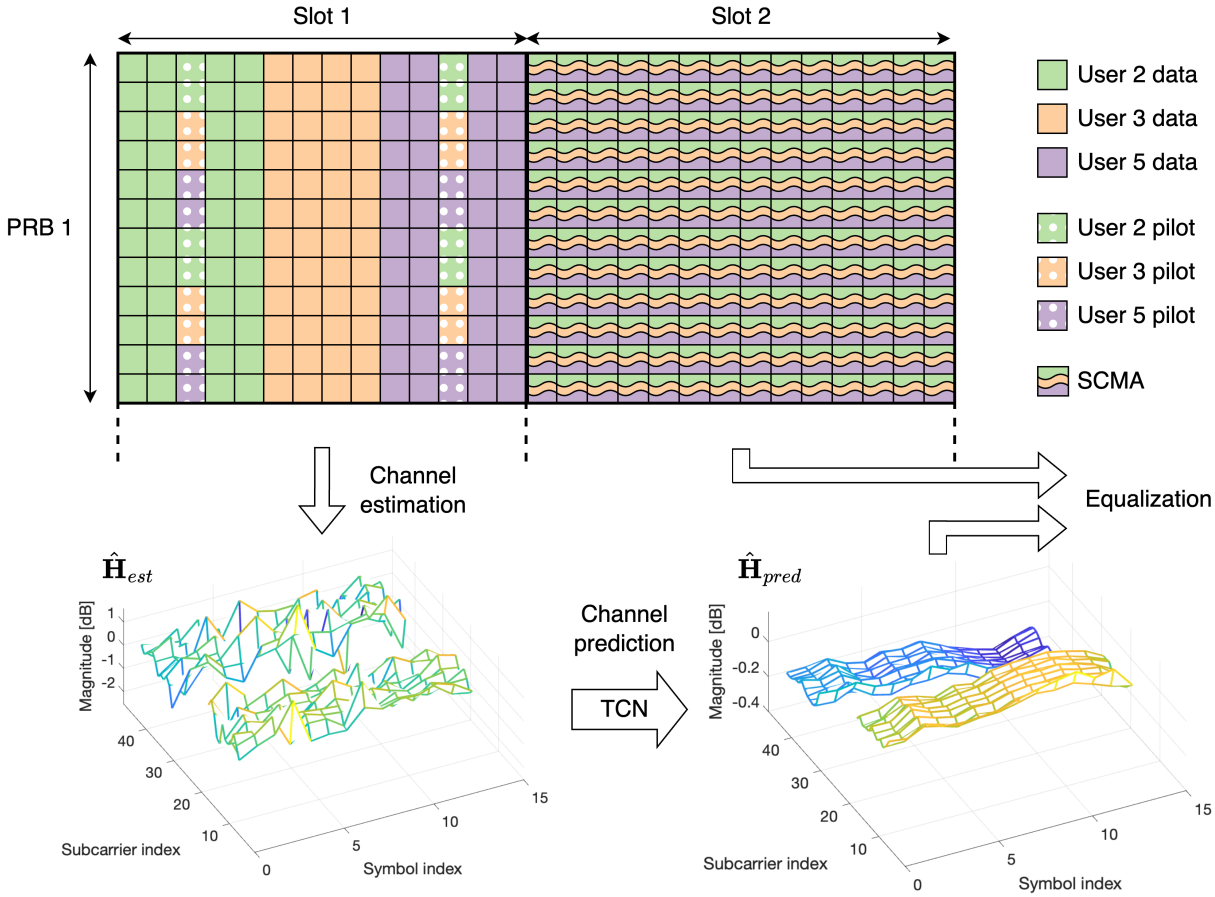


Figure 5.1: Proposed SCMA slot structure (focus on PRB1).

orthogonality between different users' resources; the second slot is exclusively allocated to SCMA-based data transmission without any pilot symbol, thus improving overall system throughput. As in Chapter 2, the UEs' transmitter employs an LDPC channel encoder and a bit interleaver; transmissions on the first slot employ a symbol mapping with a digital modulator, while non-orthogonal transmissions over the second slot are based on the SCMA encoder as in Chapter 2; finally, the j-th transmission chain concludes with a CP-OFDM modulator, including the RE mapping over the resource grid of user j, $\mathbf{X}_j \in \mathbb{C}^{N_{SC} \times N_{sym}}$. In particular, $\mathbf{X}_{1,j} = \mathbf{X}_j[1, \ldots, N_{SC}; 1, \ldots, N_{sym}^{slot}]$ represents the j-th user's resource grid over the first slot, setting to zero the orthogonal resource elements associated to OFDM symbols and/or subcarriers over which the user does not transmit; $\mathbf{X}_{2,j} = \mathbf{X}_j[1, \ldots, N_{SC}; N_{sym}^{slot} + 1, \ldots, N_{sym}]$ represents the j-th user's resource grid over the second slot, setting to zero the resource elements associated to PRBs that are not allocated to such user.

Each signal is propagated through a TDL channel represented by (4.2) considering a different channel realization for each user, including, in general, a different frequency synchronization error. At the receiver, the J received signals are superimposed and AWGN

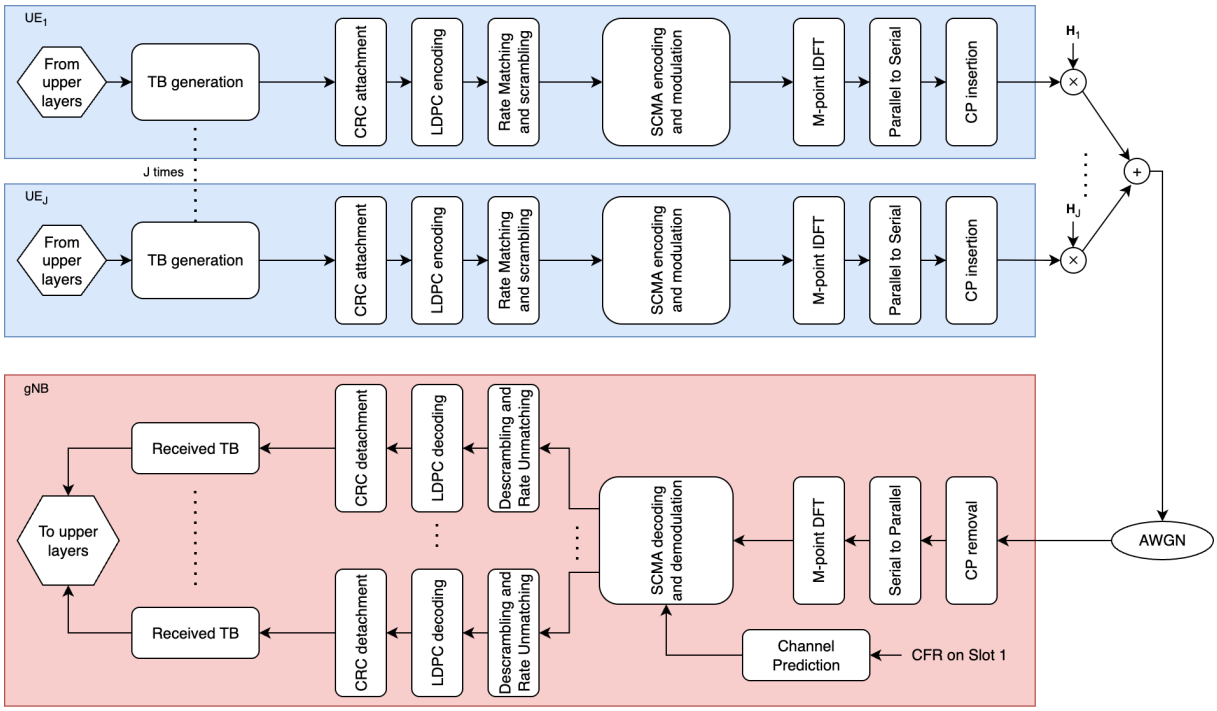


Figure 5.2: Block diagram for SCMA transmission.

corrupts the resulting waveform; then, the cyclic prefix is removed and the FFT algorithm is applied to obtain a resource grid $\mathbf{Y} \in \mathbb{C}^{N_{SC} \times N_{sym}}$. Denoting with $\mathbf{H}_j \in \mathbb{C}^{N_{SC} \times N_{sym}}$ the j-th user's CFR, the resource grid $\mathbf{Y}$ can be represented as:

$$\mathbf{Y} = \sum_{j=1}^{J} \mathbf{H}_j \odot \mathbf{X}_j + \mathbf{W}. \tag{5.1}$$

The processing chain for data equalization follows the same as in the previous Chapter with an extension to the multi-user case:

1. For each user $j \in 1, \ldots, J$, an initial CFR estimate $\hat{\mathbf{H}}_{LS,j}$ on the j-th user's allocated PRBs over the first slot is obtained by performing LS channel estimation on the corresponding pilot locations and interpolating in time and frequency as in (4.4).

2. Each user's data symbols on the first slot are equalized applying (4.5) over the corresponding user's allocated resource elements, resulting in the equalized symbols $\hat{\mathbf{X}}_{1,j}$ and, after demapping, the LLRs $\hat{\mathbf{LLR}}_{1,j}$.

3. For each user $j \in 1, \ldots, J$, the CFR on the second slot $\hat{\mathbf{H}}_{pred,j}$ is obtained by means of channel prediction based on the refined version of $\hat{\mathbf{H}}_{LS,j}$.

4. The superimposed SCMA symbols on slot 2 are demodulated by means of the Log-MPA algorithm using the predicted CFRs, obtaining the LLRs $\hat{\mathbf{LLR}}_{2,j}$ for user $j \in 1, \ldots, J$.

5.2 TCN-based channel prediction

Based on empirical evaluations of the accuracy of the models presented in Section 4.2.1 in the NOMA framework, including the Hybrid CNN-LSTM variants, the TCN architecture was selected. Due to the PRB-wise resource allocation in SCMA, discontinuities in a user's CFR are to be expected between PRBs that are allocated to such user and PRBs that are reserved for other users: as a result, the PRB-agnostic convolutional layers in the CED architecture penalize such model's accuracy in the considered framework. The convolutional layers in the Hybrid CNN-LSTM and TCN models, on the opposite, compress and reconstruct CFRs PRB-wise and mainly differ from each other in the way the prediction task is carried out. The LSTM layer in the first model combines the extracted features together at each time step using dense layers, thus mixing together channel features regardless of them being on allocated PRBs or not. The convolutional filters in the TCN overcome this by running separate filters on each feature before mixing, making it simpler to maintain the features separation at the prediction stage. It should be noted that the same model is used for different users, *i.e.*, for different PRB allocations: for this reason, the training procedure involves training not only at different E_b/N_0 working points, but also considering the users' PRB allocations defined by the signature matrix $\mathbf{S}$ reported in (2.1). To this aim, the TCN is trained with a modified version of the MSE equation (4.7) to ensure that only the CFR estimates over the allocated resource elements affect the MSE. In particular, a user-specific binary mask $\mathbf{M}_j \in \{0,1\}^{N_{SC} \times N_{sym}^{slot}}$, $j = 1, \ldots, J$ is built based on the signature matrix $\mathbf{S}$:

$$\mathbf{M}_j[m;n] = \mathbf{S}\left[\lfloor (m-1)/N_{SC}^{PRB} \rfloor + 1; j\right] \;\; \forall \; m = 1, \ldots, N_{SC}, \;\; n = 1, \ldots, N_{sym}^{slot}. \tag{5.2}$$

Tensors containing one batch of input CFRs, label CFRs, and masks are obtained by considering one PRB allocation for each sample (*i.e.*, $\mathbf{M}[1, \ldots, N_{SC}; 1, \ldots, N_{sym}^{slot}; k] = \mathbf{M}_j[1, \ldots, N_{SC}; 1, \ldots, N_{sym}^{slot}]$ if the PRB allocation of user j was considered for the k-th batch sample). Then, the mask is combined in the regular MSE equation to mask out contributions over resource elements that are not allocated to a batch sample's corresponding user, leading to a masked MSE:

$$\begin{aligned} MSE = \frac{1}{\sum_{k=1}^{N_{batch}} \sum_{m=1}^{N_{SC}} \sum_{n=1}^{N_{sym}^{slot}} \mathbf{M}[m;n;k]} \sum_{k=1}^{N_{batch}} \sum_{m=1}^{N_{SC}} \sum_{n=1}^{N_{sym}^{slot}} \\ \mathbf{M}[m;n;k] \cdot \left(\hat{\mathbf{H}}_{pred}[m;n;1;k] - Re\{\mathbf{H}[m; N_{sym}^{slot} + n; k]\}\right)^2 + \\ \mathbf{M}[m;n;k] \cdot \left(\hat{\mathbf{H}}_{pred}[m;n;2;k] - Im\{\mathbf{H}[m; N_{sym}^{slot} + n; k]\}\right)^2. \end{aligned} \tag{5.3}$$

The same training techniques reported in 4.2.2 are also employed here using the same training parameters included in Table 4.5. Differently from the previous Chapter, an

Table 5.1: Simulation parameters (values in bold are to be considered the default value, except where stated otherwise).

Parameter	**Value**
Maximum Monte Carlo iterations	10^5 iterations
Modulation order	$M = 4$
Code rate	$R_c = 679/1024$
Carrier frequency	$f_c = 2$GHz
Maximum UE speed	$v_{UE} \in \{\mathbf{5}, 30\}$ km/h
Fading model	{NTN-TDL-C, NTN-TDL-A} [36]
Residual Doppler shift standard deviation	$\sigma_D = \frac{f_c \cdot 0.1 \cdot 10^{-6}}{3} \approx 66.67$ Hz [82]
Delay spread	30 ns
Number of subcarriers	$N_{SC} = 48$
Subcarrier spacing	$\Delta f = 15$ kHz
Satellite altitude	600 km

improved prediction performance was observed considering a mirrored exponential distribution of the training E_b/N_0, *i.e.*, generating samples at low SNR more often than at high SNR:

$$P_{\gamma_n}(i) = Prob\{\gamma_n = \Gamma(i)\} = \frac{(|\Gamma| - i + 1)^2}{\sum_{k=1}^{|\Gamma|} k^2}. \tag{5.4}$$

5.3 Numerical results

The results of the numerical evaluation of the proposed channel predictor are here presented. As in the previous Chapter, the TCN has been tested in an end-to-end simulation on the MATLAB computing environment. Each E_b/N_0 point was evaluated for a maximum of 2×10^4 Monte Carlo iterations, where each iteration consists in the transmission and reception of one transport block per UE per slot (*i.e.*, $J = 6$ transport blocks carried by orthogonal QPSK symbols in mini-slots, and $J = 6$ transport blocks carried by SCMA symbols in the pilot-free slot). Additionally, as in the previous study, a zero-mean Gaussian-distributed carrier frequency offset within 1 part per million of the carrier frequency 99.7% of the times is included as a further impairment. All parameters simulations are reported in 5.1.

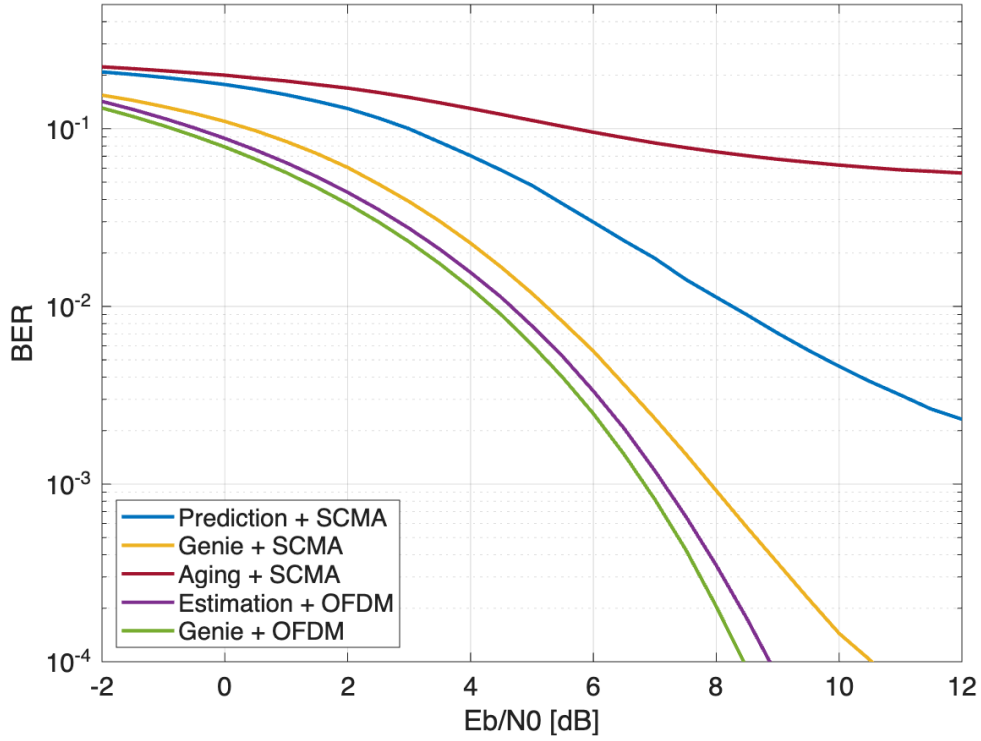


Figure 5.3: BER as a function of the E_b/N_0 (NTN-TDL-C channel model).

5.3.1 Bit error rate

Figure 5.3 compares the uncoded bit error rate (BER) achieved in the proposed system model and in a more typical OFDM system, maintaining the same number of allocated subcarriers $N_{SC} = 48$ and considering an allocation of one physical resource block of 12 subcarriers per UE (*i.e.*, 4 UEs on the considered bandwidth) carrying QPSK symbols. The analysis is carried out under the NTN-TDL-C channel model to highlight the ability of the predictor to adapt to different channel conditions. The proposed system model is evaluated considering genie-aided CFRs (*i.e.*, with perfect channel knowledge), aged CFRs (*i.e.*, the SCMA codewords are decoded using the most recent channel estimate obtained on the previous slot), and predicted CFRs (*i.e.*, using the channel predictor); the classical OFDM system includes genie-aided CFRs as well as LS-based CFRs. The plot reveals that the prediction-based system (blue curve) requires 8 dB of E_b/N_0 to cross the 1% BER threshold, a 3 dB loss with respect to the genie-aided SCMA system (yellow curve), while the aging-based system (red curve) reaches an error floor over the 5% mark. Such gaps result from the CFRs accuracy, which intuitively has a greater impact on SCMA systems than traditional OFDM systems.

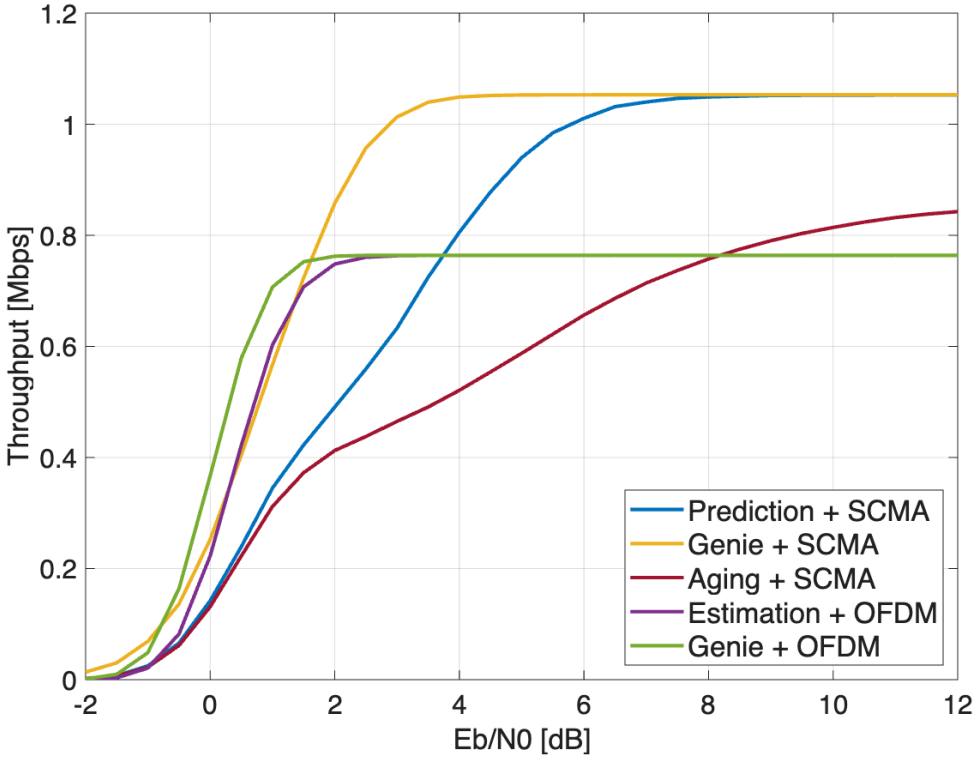


Figure 5.4: Throughput as a function of the E_b/N_0 (NTN-TDL-C channel model).

5.3.2 Aggregated throughput

Figure 5.4 reports the throughput achieved by the prediction-based NOMA system and the considered benchmarks. The proposed system achieves a peak aggregated throughput of 1.05 Mbps, a 37% increase over the traditional OFDM system (764 kbps). Perfect channel prediction always provides at least the same throughput with respect to a traditional OFDM-based system with channel estimation (purple curve), while being only 0.5 dB distant from genie-aided OFDM (green curve) until the peak OFDM throughput is reached at an E_b/N_0 of 1.5 dB. With the chosen channel predictor (blue curve), the aggregated throughput plot is characterized by a slower rise, highlighting poor prediction performance under low signal-to-noise ratio conditions. The benefits of the proposed system model appear evident after $E_b/N_0 = 3.75$ dB, with the aggregated throughput steadily increasing over the peak OFDM throughput, crossing the 1 Mbps mark (a 31% improvement over OFDM) at $E_b/N_0 = 5.75$ dB and reaching the peak aggregated throughput of 1.05 Mbps with 2 dB more. The plot also shows that a channel prediction framework is required for SCMA: indeed, the red curve shows that the aged channel coefficients introduce too many errors in the decoded blocks, resulting in a throughput gain over OFDM from $E_b/N_0 = 8$ dB and a ceiling (corresponding to a block error rate floor) far lower than the theoretical peak.

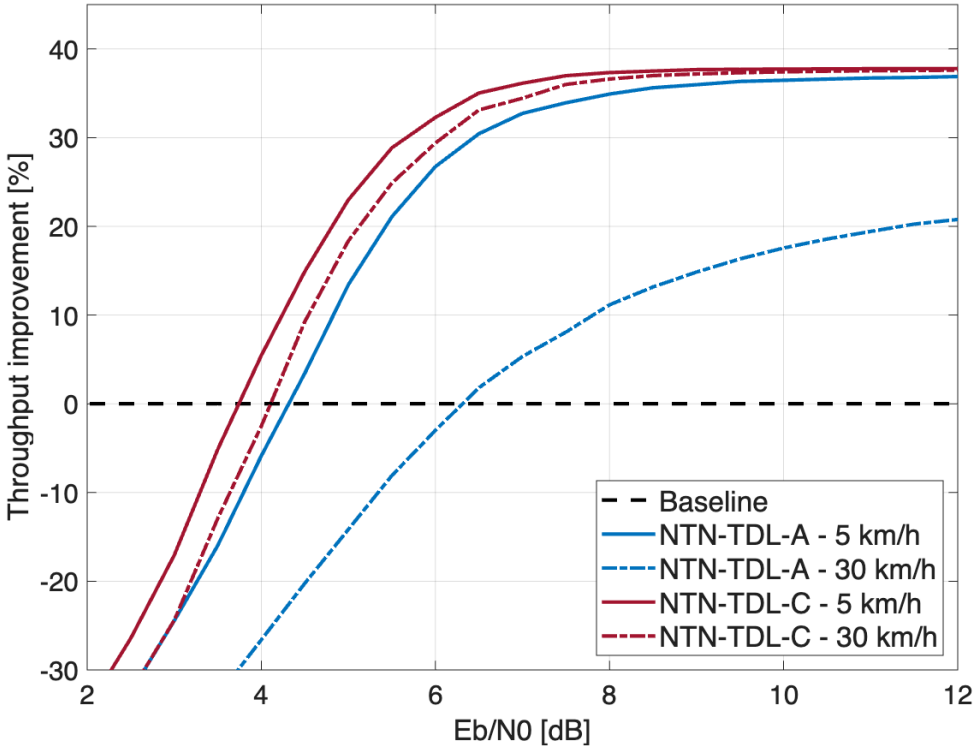


Figure 5.5: Throughput improvement as a function of the E_b/N_0.

5.3.3 Generalization to mismatching channel conditions

The aggregated throughput achieved using the proposed channel predictor is further assessed under mismatching channel conditions. The model has been trained on the NTN-TDL-A channel model (NLoS) with a maximum UE speed of 5 km/h. The pre-trained model is tested under: 1) training conditions; 2) NTN-TDL-C channel model (line of sight, LoS) [36]; 3) maximum UE speed of 30 km/h; and 4) both NTN-TDL-C and maximum UE speed of 30 km/h. Figure 5.5 reports the throughput improvement over the estimation-based OFDM system, expressed as a percentage of the aggregated throughput achieved in OFDM, as a function of the E_b/N_0. The plot shows that a mismatch in LoS condition does not significantly affect the performance of the model: indeed, regardless of the maximum UE speed, the prediction accuracy is large enough to ensure a throughput improvement at close to 4 dB of E_b/N_0, while the 20% threshold is passed at $E_b/N_0 = 4.75$ dB and $E_b/N_0 = 5.25$ dB for a maximum UE speed of 5 km/h (red line) and 30 km/h (red dash-dotted line), respectively. However, the complex dynamics of NLoS channels with large UE velocities significantly affect the capabilities of the proposed model, pushing the break-even point of the blue dash-dotted line at $E_b/N_0 = 6.25$ dB. Nonetheless, the TCN was able to generalize well to unseen typical satellite-based NTNs channel conditions, *i.e.*, LoS with pedestrian or medium mobility, even in the context of NOMA.

5.4 Conclusions and future works

This study extended the analyses reported in the previous Chapter to a NOMA-based NTN, where channel prediction is not only a means to achieve a lower pilot overhead, but also an enabler of NOMA techniques. To support this, a prediction-oriented slot structure was designed to ensure adequate channel prediction accuracy for all of the scheduled users. Through a set of numerical analyses, it was demonstrated that the proposed solution outperforms conventional orthogonal schemes and approaching the performance of genie-aided NOMA systems, with less than 3 dB of degradation in E_b/N_0. While future works could target additional improvements to the prediction model, the high sensitivity of the Log-MPA algorithm to CFR errors calls for AI-based SCMA demodulators that can better handle such inaccuracies; the solution proposed in Chapter 2 and the corresponding future developments should thus be trained considering these effects.

Chapter 6

Equalization of Faster-than-Nyquist signaling

This study was carried out during the second year of PhD. The contents of this chapter are based on the paper "Faster-than-Nyquist Equalization with Convolutional Neural Networks" [99].
Since the dawn of wireless communications, one of the greatest challenges has always been how to improve the radio resource utilization. As spectrum is a scarce and costly asset, both academia and industrial researchers have proposed new techniques over the years to increase the number of bits per second that can be transmitted over a fixed bandwidth, *e.g.*, CF-MIMO [80] and NOMA [55]. The appeal of FTN signaling lies in the straightforwardness of its approach: to achieve a higher data rate, one can simply transmit at a faster symbol rate using the same pulse. In 1975, J. E. Mazo investigated the effect of ISI on symbols transmitted at a rate R_s higher than that allowed by the famous Nyquist-Shannon sampling theorem, *i.e.*, twice the bandwidth of the modulating pulse. The author observed that, for ideal sinc pulses with bandwidth W and binary transmissions, no significant reduction in Minimum Euclidean Distance (MED) between received symbol sequences could be observed as long as $\frac{2W}{R_s} \geq \tau_{FTN,Mazo} = 0.802$, the so-called "Mazo limit" [27]. This result has motivated several authors to investigate FTN and push the technique over the Mazo limit, *e.g.*, [100, 101, 28, 29, 102]. The scientific community has focused on identifying and exploiting the intricate ISI patterns for equalizing the received FTN data streams. Today, 50 years after Mazo's original paper, DL techniques are pushing the boundaries of FTN transmissions [31, 32, 33]. Inspired by the great performance of CNNs with skip connections in image processing tasks, *e.g.*, [103], it is worth investigating whether such results apply to the equalization of ISI in FTN systems. Motivated by these considerations and analyses, this study:

- Proposes the application of CNNs with skip connections in FTN receivers to identify

and equalize ISI patterns;

- Evaluates and compares the BER, BLER, and Throughput (TP) achieved by the proposed model against several benchmarks, including a DL-based equalization scheme.

In addition, in an effort to advocate for reproducibility, taking once again inspiration from the DL scientific community, the code is open-source in a public repository [104].

6.1 System model

A typical transceiver is here considered, consisting of a bit source, a channel encoder, an interleaver, a digital modulator, and pulse shaping on the transmitter side, as well as a matched filter, a sampler at FTN rate R_s, a demodulator, a deinterleaver, and a channel decoder on the receiver side. Given the exploratory nature of this work, AWGN is the only considered channel impairment. The focus of this paper is the demodulator, which is here implemented with a CNN. The FTN baseband signal is modeled as follows. A sequence of energy-normalized QPSK symbols $\{x_n\}_{n\in\mathbb{Z}}$ is generated at the output of the digital modulator from LDPC-coded bits and fed to the FTN pulse shaper with symbol time $T_s = \frac{1}{R_s} = \tau_{FTN}\frac{1}{2W} = \tau_{FTN}T_N$, where $0 < \tau_{FTN} < 1$ is the FTN compression factor and T_N is the symbol time at Nyquist rate. The filter employs the unit-energy Square Root Raised Cosine (SRRC) pulse $h(t)$ with roll-off factor $\beta = 0.5$ and bandwidth W, producing the following signal [31]:

$$s(t) = \sum_{n=-\infty}^{+\infty} x_n h(t - n\tau_{FTN}T_N). \tag{6.1}$$

Clearly, the strict upper constraint on τ_{FTN} results in a violation of the Nyquist-Shannon sampling theorem; thus, as previously anticipated, ISI is introduced (Figure 6.1). After being corrupted by AWGN, represented by the zero-mean circularly symmetric Gaussian random process $n(t)$ with variance $\frac{N_0}{2}$, with N_0 being the noise power spectral density, the signal is processed by a filter matched to $h(t)$. Let $g(t) = \int h(\tau)h^*(\tau - t)d\tau$ denote the ISI channel autocorrelation function, and $\eta(t) = \int n(\tau)h^*(\tau - t)d\tau$ the matched filter response to AWGN [29], the output of the matched filter is:

$$y(t) = \sum_{n=-\infty}^{+\infty} x_n g(t - n\tau_{FTN}T_N) + \eta(t). \tag{6.2}$$

Sampling at FTN rate R_s yields a sequence $\{y_k\}_{k\in\mathbb{Z}}$, where each received symbol can be

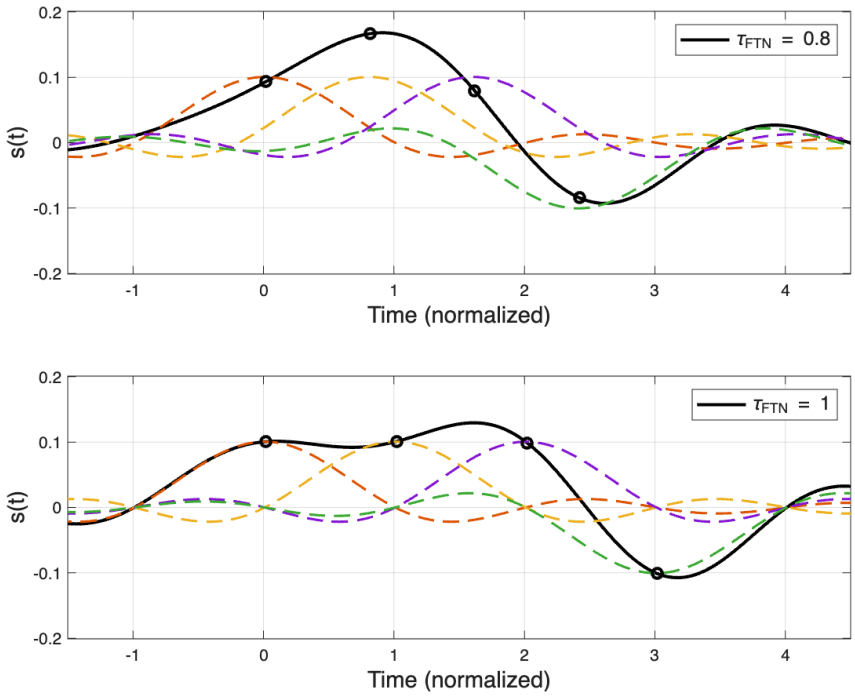


Figure 6.1: Example of FTN baseband signal (above) vs at-Nyquist baseband signal (below) corresponding to 4 binary symbols and sinc shaping pulse.

computed as

$$y_k = \sum_{n=-\infty}^{+\infty} x_n g((k-n)\tau_{FTN}T_N) + \eta(k\tau_{FTN}T_N), \tag{6.3}$$

or, considering a block of N_s symbols $\mathbf{y} = \{y_k\}_{1,\dots,N_s}$,

$$\mathbf{y} = \mathbf{G}\mathbf{x} + \boldsymbol{\eta}. \tag{6.4}$$

In (6.4), $\mathbf{G}$ represents the ISI channel matrix, *i.e.*, the contribution of preceding and succeeding symbols on the current one due to ISI. The channel autocorrelation function samples $\{g_n\}_{n\in\mathbb{Z}} = g(n\tau_{FTN}T_N)$ are truncated to the centermost $2L_I+1 < N_s$ taps, where L_I represents the one-sided ISI span, resulting in $\mathbf{G}$ being structured as a $N_s \times N_s$ Toeplitz matrix with $2 \times (N_s - L_I - 1)$ zero diagonals [28]:

$$\mathbf{G} = \begin{bmatrix} 1 & g_{-1} & \cdots & g_{-L_I} & 0 & 0 & \cdots \\ g_1 & 1 & \ddots & \ddots & g_{-L_I} & 0 & \ddots \\ \vdots & \ddots & \ddots & g_{-1} & \ddots & \ddots & \ddots \\ g_{L_I} & \ddots & g_1 & 1 & g_{-1} & \ddots & \ddots \\ 0 & g_{L_I} & \ddots & g_1 & 1 & \ddots & \ddots \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{6.5}$$

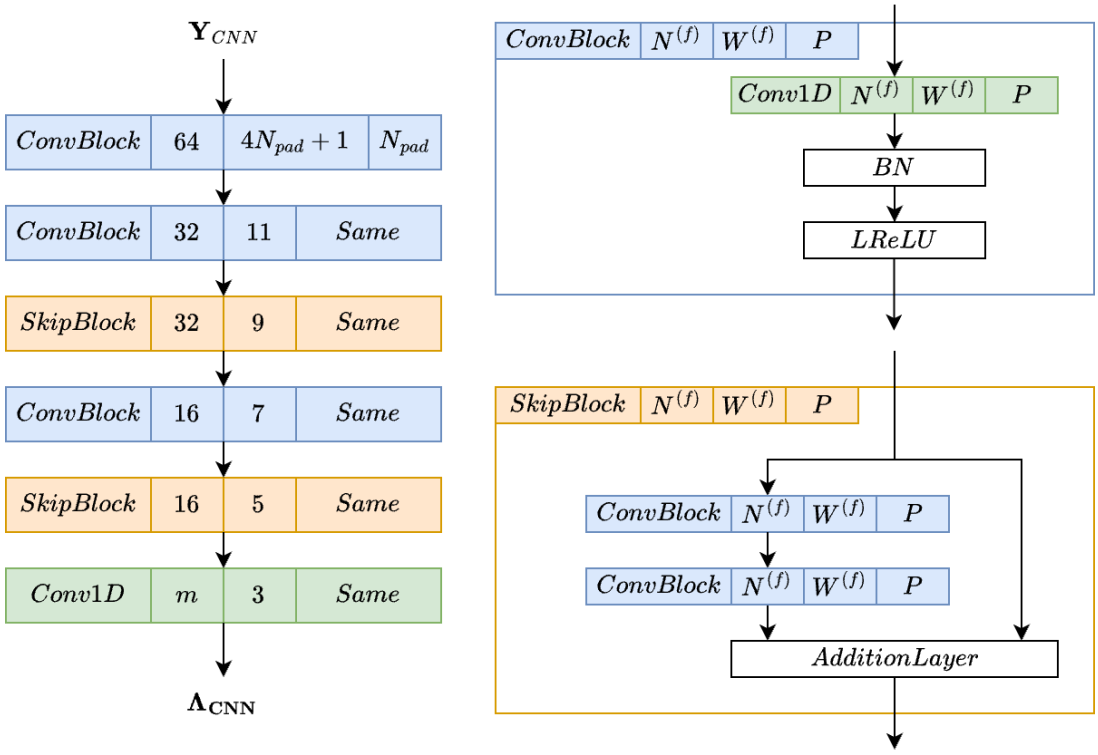


Figure 6.2: Diagram of the proposed CNN.

It should be noted that the structure of $\mathbf{G}$ does not fully capture ISI at the boundaries, *e.g.*, y_1 is affected only by the upcoming symbols in the block and not by previous transmissions; hence, this effect should be taken into account when simulating a continuous stream of data. The random vector $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \frac{N_0}{2}\mathbf{G})$ represents the effect of AWGN after the matched filter, resulting in colored noise. While some studies have applied methods to whiten the noise, *e.g.*, [28], it was here chosen not to further manipulate the samples. Thus, $\mathbf{y}$ is the input data of the proposed CNN.

6.2 CNN-based FTN demodulator

The task of the DL model is to convert a sequence of received symbols into the corresponding sequence of coded bit LLRs; thus, while the CNN must be tailored to a modulation order, it remains independent of the coding scheme. The receiver proposed in this work takes inspiration from pattern recognition models and employs a sequence of Conv1D layers, LReLU activation functions, and BN layers, together with skip connections; its structure is reported in Figure 6.2. Each Conv1D layer includes $N^{(f)}$ filters with $W^{(f)}$ taps, where each filter is applied to the $N^{(f-1)}$ channels of the input tensor to generate a new sequence of length L; thus, the number of channels of the output tensor coincides with

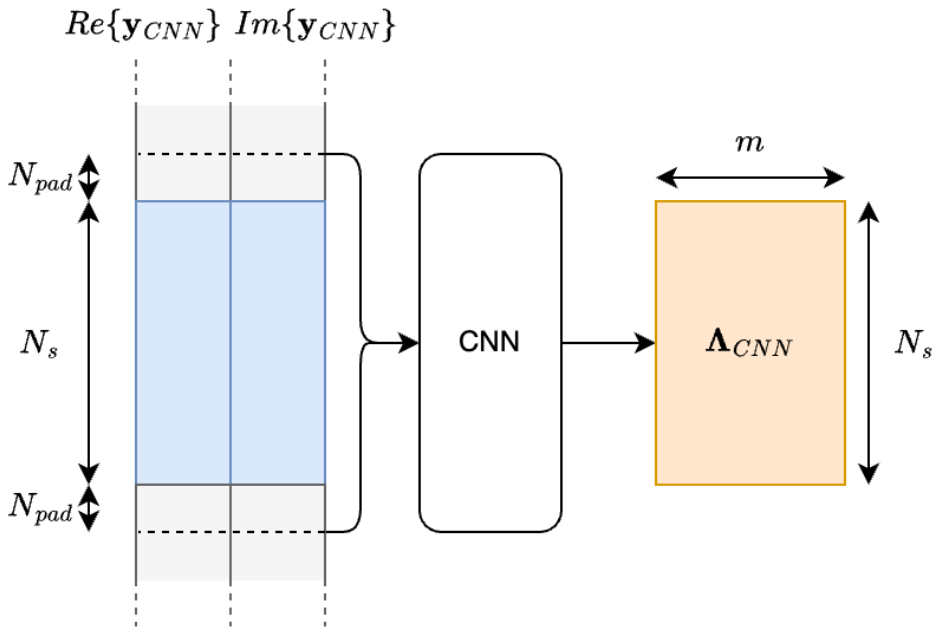


Figure 6.3: CNN input pre-processing.

$N^{(f)}$. Its length depends on the amount P of padding applied before filtering (P ="same" leaves the sequence length unchanged).

6.2.1 CNN input

The CNN pre-processing steps are depicted in Figure 6.3. A block of N_s received symbols $\mathbf{y}$ is padded on the left with the final N_{pad} symbols of the preceding block and on the right with the initial N_{pad} symbols of the succeeding block; such complex vector of length $L_{in} = N_s + 2N_{pad}$ is referred to as $\mathbf{y}_{CNN}$. While this method delays by $N_{pad}T_s$ the demodulation, it enhances the CNN's ability to extract ISI patterns. The vector is then split into its real and imaginary components, which are stacked together to form a $[L_{in} \times 2]$ matrix $\mathbf{Y}_{CNN}$.

6.2.2 CNN output

To fully exploit the capabilities of the LDPC decoder, the CNN outputs the bit LLRs related to the data block of interest. Such $[N_s \times m]$ matrix is referred to as $\mathbf{\Lambda}_{CNN}$, where the i-th row contains the vector of m LLRs associated with the i-th data symbol in the block. The corresponding $[N_s \times m]$ matrix of transmitted bits, used as labels during the training process, is named $\mathbf{B}$.

6.2.3 Loss function

While LLRs assume values in $\mathbb{R}$, it is still possible to approach the problem as a classification task. Indeed, as for the CNN-based NOMA demodulator presented in Chapter 2, the model can be trained to produce LLRs by incorporating the sigmoid activation into the loss function, *i.e.*, using the L-BCE loss function reported in 2.7.

6.2.4 Dataset, training, and inference

As for the previous Chapters, a batch of N_B input matrices and their corresponding labels is generated during each training epoch according to the system model presented in Section 6.1. The E_s/N_0 ratio, where $E_s = \mathbb{E}\left[|x_n|^2\right]$ is the average energy per symbol, is randomly set for each example within a batch based on an exponential distribution (*i.e.*, as reported for channel prediction in Section 4.2.2). The loss function gradients with respect to the CNN parameters are computed with backpropagation; then, the Adam optimizer updates such parameters using L2 regularization. The learning rate is reduced at loss function plateaus and training is terminated via early stopping after long plateaus. A CNN is separately trained for each considered value of τ_{FTN}. Once training is completed, the models can then be deployed for inference within the receiving chain; online training is beyond the scope of this study.

6.3 Results

In this section, the link level performance of the proposed receiver are analyzed. Simulations were conducted in the MATLAB computing environment with the aid of the CVX package [105]; the source code is available at [104]. Due to space constraints, only the main simulation parameters are reported in Table 6.1. To ensure that boundary conditions in the ISI channel matrix $\mathbf{G}$ do not affect the validity of the results, further padding is added to each generated block of symbols, simulating a continuous stream; thus, each symbol fed to the CNN is equally affected by ISI.

6.3.1 Benchmark receivers

For a complete evaluation, a set of benchmark receivers is selected from the literature to implement alongside the proposed receiver:

- MED [106]: the optimal symbol-by-symbol QAM demodulator without any FTN-dedicated pre-processing.

Table 6.1: Simulation parameters

Parameter	**Value**
Training E_b/N_0 range	$E_b/N_0 = [-10, 10][dB]$
Maximum Monte Carlo Iterations	$N_{MC} = 10^5$
Modulation	QPSK ($m = 2$)
Code rate	$R_c = \{1/2, 3/4\}$
FTN compression	$\tau_{FTN} = \{0.6, 0.7\}$
SRRC roll-off	$\beta = 0.5$
One-sided ISI span	$L_I = \{33, 28\}$ symbols
Nyquist symbol time	$T_N = 1\mu s$
Block size	$N_s = 50$ symbols
Block padding	$N_{pad} = 12$ symbols
Batch size	$N_B = 4096$
Initial learning rate	0.01
Learning rate decay factor	10
Learning rate decay patience	50 epochs
Early stopping patience	150 epochs
L2 regularization factor	10^{-4}

- Frequency-Domain Equalization (FDE) [29]: a CP of length $2\nu_{FDE}$ is added to the symbols block, transforming $\mathbf{G}$ in a circulant matrix. The MMSE algorithm is then applied on the Fourier transform of the received symbols.

- Set Theory Semi-Definite-Relaxation-based Sequence Estimation (STSDRSE) [102]: the FTN sequence detection task is modeled as a non-convex optimization problem, of which a relaxed version with quadratic constraint is solved. Gaussian randomization ensures the original constraints are satisfied. To achieve accurate BER plots with reasonable processing time, the number of Monte Carlo iterations with this receiver was limited to 10^4.

- DNN [31]: a DNN with 4 hidden layers for FTN equalization that outputs an ideally ISI-free sequence of symbols to be fed to the MED receiver. In a similar fashion to the proposed receiver, a sliding window mechanism extracts a sequence of symbols with padding. It is important to note that the original paper does not specify all parameters required to ensure full reproducibility. *E.g.*, the input and output length of the DNN are not mentioned in [31]; using a solver, plausible values for these parameters were extrapolated based on the reported computational complexity and hidden layers size. Additional details on this and similar assumptions are reported in [104].

6.3.2 Bit Error Rate

The uncoded BER is first evaluated as a function of the E_b/N_0 ratio, where $E_b = \frac{E_s}{m}$ is the energy per bit in the FTN signal (Figure 6.4). The black dashed curve reports the theoretical BER with QPSK in AWGN channel (*i.e.*, $\tau_{FTN} = 1$). The plot shows that the proposed receiver outperforms all considered benchmarks. At $\tau_{FTN} = 0.7$, the CNN achieves a BER of 10^{-3} at an E_b/N_0 of 8.5 dB, with a gain of 2 dB against the DNN benchmark and remaining within 2 dB of the ISI-free performance. STSDRSE performs similarly, although taking much longer to converge on the optimization problem solution. Moving to $\tau_{FTN} = 0.6$, the stronger ISI widens the performance gap, with the CNN reaching a BER of 1.5×10^{-3} at 10 dB of E_b/N_0; on the opposite, the DNN and STSDRSE benchmarks do not reach the 10^{-2} threshold. Both compression factors are too strong for FDE, resulting in significantly degraded performance. Without any form of equalization, the ISI-agnostic MED receiver reaches a plateau at $BER = 10^{-1}$ for $\tau_{FTN} = 0.6$. It is worth noting that the DNN benchmark BER curves differ from those reported in [31]. It is believed that the discrepancy is due to different truncation of the channel autocorrelation function samples $\{g_n\}_{n\in\mathbb{Z}}$, leading to different ISI spans L_I, as well as the consideration of the boundary conditions in $\mathbf{G}$, which may not have been

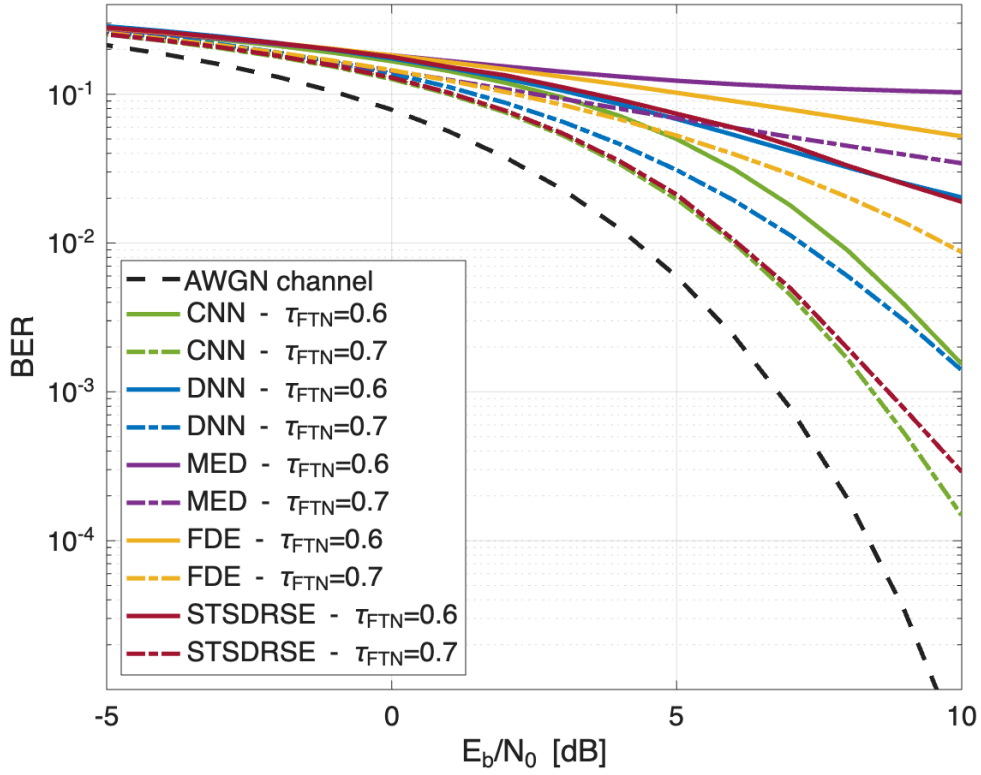


Figure 6.4: BER as a function of E_b/N_0.

accounted for in the original work. Nonetheless, the open repository includes the code used to train the DNN benchmark.

6.3.3 Block Error Rate

The focus is now on the BLER, reported in Figures 6.5 and 6.6 for code rates 3/4 and 1/2, respectively. A theoretical BLER curve is obtained by randomly introducing independent bit errors following the theoretical BER in AWGN. At the mildest compression factor, the LDPC decoder is able to correct most of the errors that the CNN was not able to mitigate, leading to a BLER close to ISI-free error rates; the performance degradation at $\tau_{FTN} = 0.6$ remains limited, with $BER = 10^{-3}$ being achieved at 6.5 dB of E_b/N_0 (3dB from AWGN). On the opposite, the DNN requires a 4dB larger E_b/N_0 to reach the same threshold, *i.e.*, it achieves a 0.1% BLER at more than 10dB of E_b/N_0. When the code rate is increased to 3/4, the decoder is limited by the DNN's equalization capabilities, and an error floor is revealed over $BER = 10^{-2}$. As in the BER plots, FDE and MED are the least performing receivers, with MED failing to achieve a BLER below 10%. As previously mentioned, STSDRSE plots are omitted due to the excessive simulation time required to achieve accurate results at high E_b/N_0; nonetheless, it was empirically observed that the algorithm tended to be outperformed by the DNN benchmark for both values of τ_{FTN}.

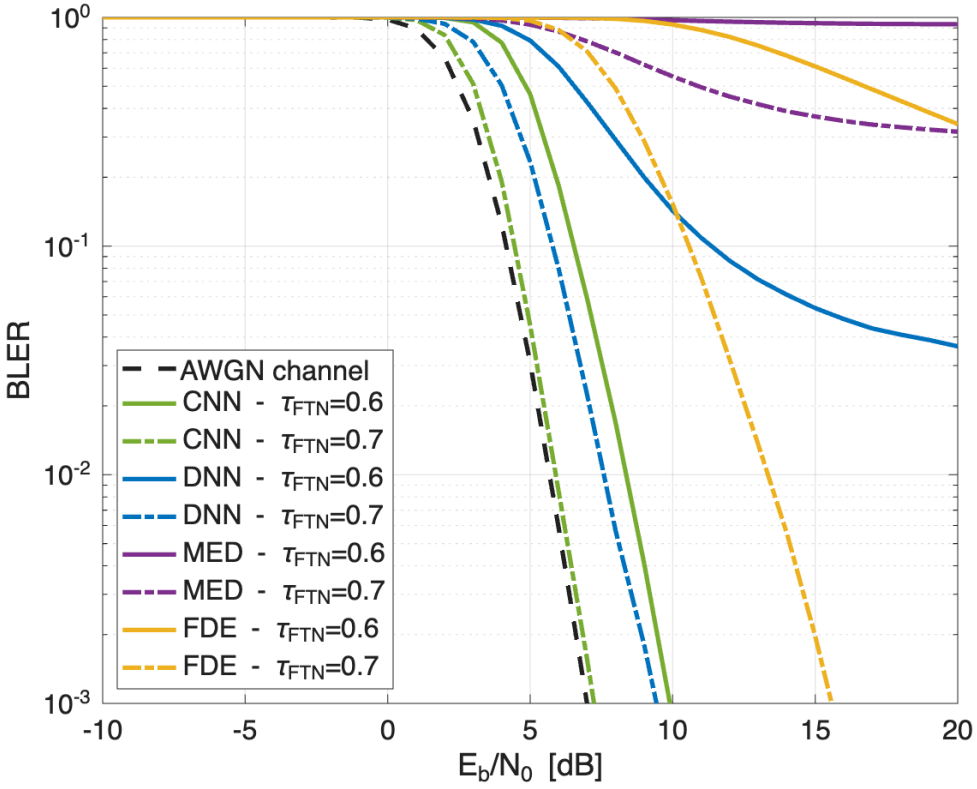


Figure 6.5: BLER as a function of E_b/N_0 for code rate $R_c = 3/4$.

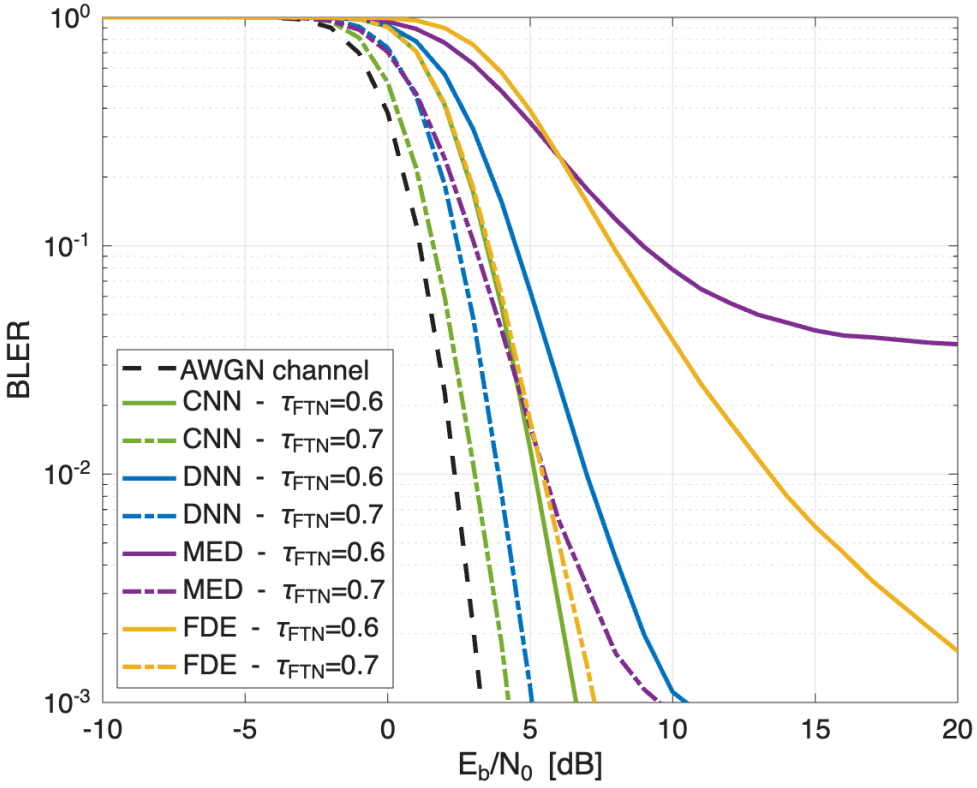


Figure 6.6: BLER as a function of E_b/N_0 for code rate $R_c = 1/2$.

6.3.4 Throughput

The appeal of FTN lies in its potential TP gains; however, this metric depends not only on the increased symbol rate at the transmitter, but also on the capabilities of the receiver:

$$TP = \frac{m \cdot R_c}{\tau_{FTN} \cdot T_N} \cdot \gamma \cdot (1 - BLER), \tag{6.6}$$

where $\gamma = \frac{N_s}{N_s + 2\nu_{FDE}}$ for FDE, accounting for the CP overhead, and $\gamma = 1$ for all other receivers. Clearly, $\tau_{FTN} = 1$ in the AWGN channel. The TP gains achieved by the CNN with respect to transmissions at Nyquist rate are thus investigated in the following, comparing such results with the considered FTN benchmarks. Regardless of the code rate, Figures 6.7 and 6.8 reveal that FTN does not improve the TP at low E_b/N_0; this is due to the increased error rate in blocks transmitted at FTN rate. However, the CNN-based receiver surpasses the AWGN TP baseline at lower E_b/N_0 values than any other FTN receiver benchmark (2.5dB and -0.5dB for code rate 3/4 and 1/2, respectively). At $R_c = 1/2$, the proposed receiver reaches its peak TP of 1.47 Mbps and 2.14 Mbps ($\tau_{FTN} = 0.6$ and 0.7, respectively) with a gain of 1-2 dB of E_b/N_0 with respect to the DNN benchmark; however, the DNN fails at reaching its peak throughput with high code rate and strong FTN compression due to the BLER floor, while the CNN achieves 2.5 Mbps at an E_b/N_0 of 10dB. Notably, the optimal TP curve is achieved by the proposed receiver by initially setting $\tau_{FTN} = 0.7$ and $R_c = 1/2$, then increasing the coding rate to 3/4 at 3.5dB of E_b/N_0, and eventually switching to $\tau_{FTN} = 0.6$ at 6dB of E_b/N_0. Indeed, the compression factor τ_{FTN} can be added as a third dimension to adaptive code and modulation, leading to better TP at any E_b/N_0 working point if a powerful FTN receiver, such as the proposed CNN, is implemented.

6.4 Computational complexity

In the previous Section, it was shown that CNNs can provide substantial throughput gains in FTN. However, computational complexity is also a key parameter to take into consideration. The number of MAC units in a Conv1D layer can be evaluated with (2.10), while the additional complexity of BN and LReLU is regarded as negligible. Considering the model reported in Figure 6.2, the total amount of MAC operations amounts to 2.67M, or 53.5k MACs per received symbol. In contrast, the DNN benchmark requires only 8196 multiplications per symbol [31], limiting the number of required MAC units per symbol to such value. Despite the higher computational cost, as also reported for the NOMA demodulator in Chapter 2, three key considerations should be taken into account. 1) Due to the vast amount of tunable hyperparameters in CNNs, it is possible that lower-complexity configurations yielding similar link-level performance were not fully explored.

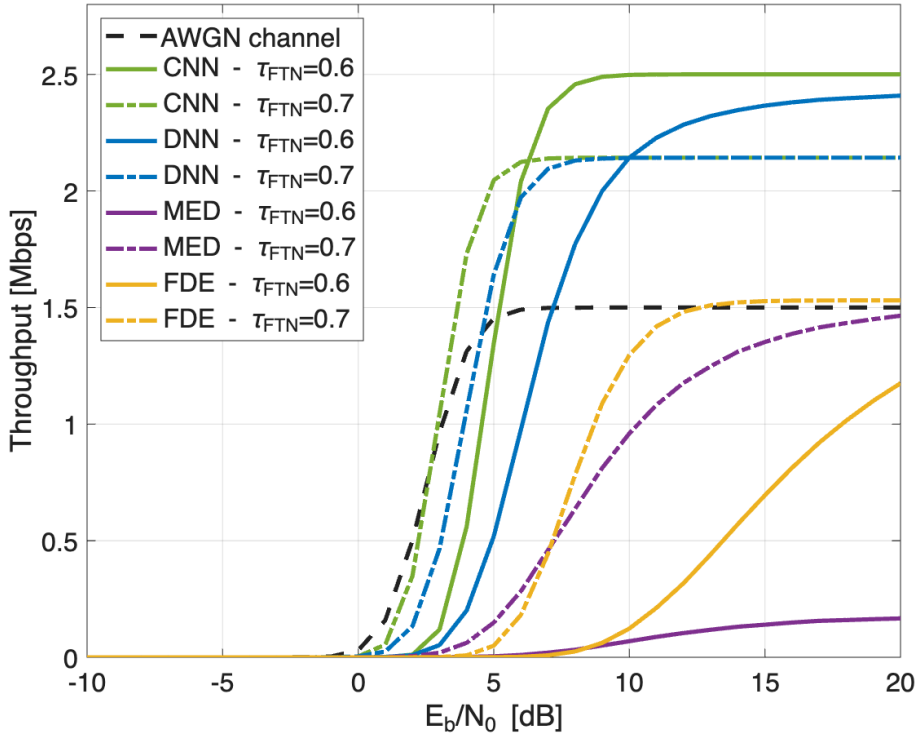


Figure 6.7: Throughput as a function of E_b/N_0 for code rate $R_c = 3/4$.

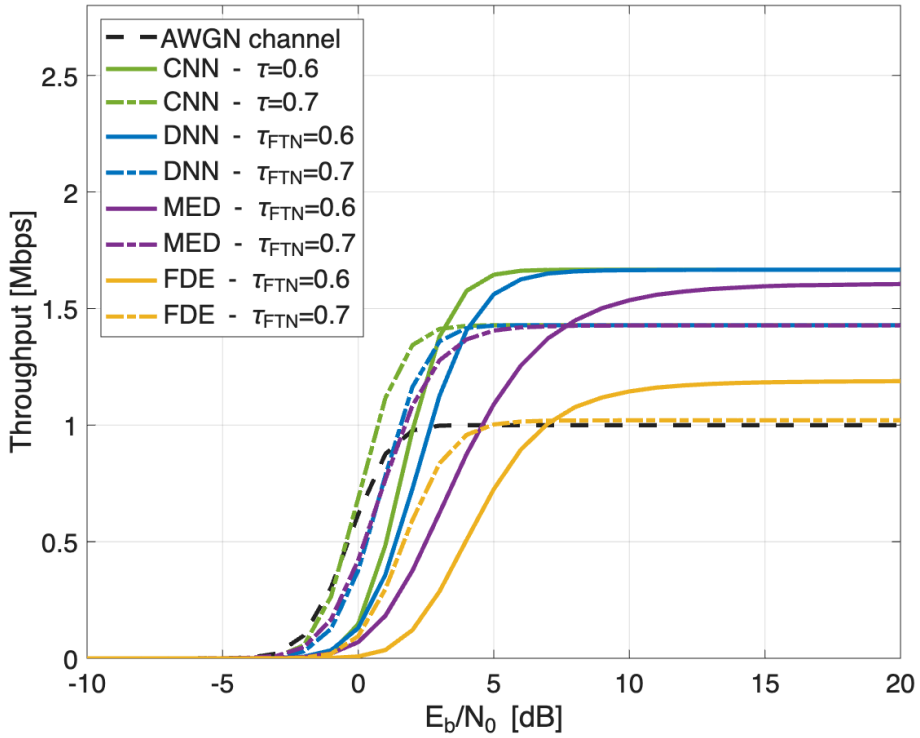


Figure 6.8: Throughput as a function of E_b/N_0 for code rate $R_c = 1/2$.

2) Model compression techniques, *e.g.*, quantization and pruning, though outside the scope of this work, can significantly reduce MAC counts and inference time with negligible performance losses or, in some cases, even marginal gains [66]. 3) Specialized hardware accelerators enable efficient inference with large DL models with small processing time as in [67]. Hence, the adoption of model compression techniques and hardware accelerators makes CNN-based FTN receivers a feasible solution from the computational complexity standpoint.

6.5 Conclusions and future works

Motivated by recent advances in DL for pattern recognition, this work proposed the application of CNNs to FTN receivers. The model was trained and its performance was evaluated in terms of BER, BLER, and TP against several benchmarks. In an effort to advocate for reproducibility, the code is open-sourced in a public repository. The proposed model achieves BLER levels at less than 1dB and 3.5dB of E_b/N_0 from AWGN channel performance for $\tau_{FTN} = 0.7$ and $\tau_{FTN} = 0.6$, respectively, outperforming all reported benchmarks. These gains directly translate to TP improvements, enabling the full exploitation of FTN's spectral efficiency advantages. Indeed, with $\tau_{FTN} = 0.6$ the TP reaches 2.5 Mbps at $E_b/N_0 = 10$dB, while benchmarks are limited by the strong ISI. As current cellular systems employ multi-carrier transmissions, FTN transceivers have recently been proposed for OFDM, *e.g.*, [107]. With the introduction of inter-carrier interference in addition to ISI, the CNN presented in this activity could be extended to a 2-dimensional case for FTN OFDM receivers in NTN user links and to realistic NTN channel models. Nonetheless, with the NTN service link air interface not being standardized by 3GPP, FTN can also be integrated in legacy NTN standards based on single carrier transmissions, *e.g.*, Digital Video Broadcasting - Second Generation Satellite Extensions (DVB-S2X) and Digital Video Broadcasting - Second Generation Return Channel Satellite (DVB-RCS2). In both cases, due to an increase in the PAPR of the generated waveform, the impact of power amplifier non-linearities on the link level performance should also be considered. Finally, future works should also explore different DL architectures for FTN signaling equalization, *e.g.*, RNNs and Transformers, and techniques to make DL models independent of the modulation order, e.g., modular approaches or output LLR masking.

Chapter 7

Conclusions

The main objective of this thesis was to investigate the application of AI to the PHY in the context of NTNs, proposing DL-based solutions to tackle key NTN challenges, *e.g.*, short data transmission window and channel aging, while exploiting the characteristics of satellite-based wireless communications. In parallel with the development of this thesis, both AI and NTNs have assumed a central role in the standardization activities of multiple 3GPP WGs. On the one hand, Release 18 also included a study item on applications of AI to the NR air interface, identifying not only use cases, but also the interactions between gNBs and UEs (*e.g.*, in the case of dual-sided models) and the LCM of DL models, including training and fallback mechanisms to traditional algorithms. On the other hand, NTNs have been introduced in 3GPP specifications in Release 17 (2022) and have since been the focus of multiple studies and advancements through Release 18 and, at the time of writing this thesis, Release 19 and 20. During this thesis, three use cases for the application of AI to the PHY of NTNs were identified: 1) Code-domain NOMA demodulation; 2) Channel prediction for channel aging, pilot overhead reduction, and as a NOMA enabler; and 3) FTN signaling equalization. In one of the earlier studies included in this thesis, a CNN-based demodulator was considered for SCMA demodulation in the context of NTN-based MEC systems. The DL model takes as input the received superimposed symbols and the users' channel estimates: such values are jointly processed by convolutional layer to remove most of the inter-user interference due to the non-orthogonal scheme; then, the resulting symbols are collected and separately demodulated on UE-specific branches in a MTL fashion. Through this study, it was shown that the proposed CNN greatly improves the BLER with respect to the MPA-based benchmark, thus providing higher throughput at low SNRs; however, this comes at the cost of an increase in complexity, a challenging aspect in the context of on board processing. Thus, either model optimization techniques (*e.g.*, through pruning and weight quantization) or different DL architectures should be considered to improve the feasibility of on board inference. In particular, the graph-like

structure of SCMA codebooks makes GNNs an ideal candidate for an SCMA receiver; with the appropriate masking, the Transformer architecture is also worth investigating. In the context of channel prediction, a lightweight DNN was proposed in an initial study as a means of predicting the channel gain in a CF-MIMO system based on the previous sample and geometrical factors (*i.e.*, the slant range and the elevation angle). The outcomes of the study highlighted an improvement of the per-user capacity 10- and 90%-iles by up to 15% and a reduction of the outage probability with a minimal increase in complexity. Further activities on channel prediction focused on the prediction to of the CFR to reduce the DM-RS overhead, proposing three prediction architectures based on convolution. The best performing model in terms of prediction accuracy and complexity, a Hybrid CNN-LSTM, was shown to be able to guarantee consistent throughput gains in several cases, including those where the testing propagation conditions (LoS availability, UE speed) were not the same as those seen during training, reaching a maximum gain of 8.33% with respect to a benchmark pilot-based OFDM system. The low complexity of the considered model suggests that real time inference under stringent SWaP constraints may be feasible thanks to dedicated hardware accelerators, although the impact of real-world effects should be assessed through a hardware implementation. With the proper adaptation to the training pipeline, further pilot signals may be removed by iteratively performing channel prediction based on the previous slot's refined CFR, asymptotically achieving a throughput gain of $\eta_\infty = N_\pi^{slot}/(N_{sym}^{slot} - N_\pi^{slot}) \approx 16.67\%$. One of the proposed CFR prediction architecture was also employed in a CP-OFDM-based system with SCMA transmissions to provide channel estimates for the demodulation of SCMA symbols. In this context, the model was shown to operate at less than 3 dB of SNR from the perfect channel estimation conditions, providing a throughput improvement over orthogonal systems by more than 30%. Clearly, the transmissions' non-orthogonality calls for more accurate channel estimates, thus requiring more complex prediction models; a different strategy may also involve training a DL model for SCMA demodulation in presence of channel estimation errors, *e.g.*, using the GNN architecture. Finally, a CNN was proposed as an ISI equalizer and detector in FTN signaling single carrier systems, where a spectral efficiency gain is achieved by pushing the data rate past the Nyquist-Shannon sampling theorem bound. The DL model proved effective under strong FTN compression factors, leading to consistent throughput gains with respect to at-Nyquist transmissions and achieving state-of-the-art performance against several benchmark receivers. Nonetheless, the performance may be further increased by considering different architectures, *e.g.*, RNNs and Transformer. Furthermore, the technique should be extended to multi-carrier systems, where a loss in orthogonality introduces ICI on top of ISI, and to fading models, thus including channel estimation in the CNN-based receiver design.

It goes without saying that this thesis covered only few of the possible areas of application of AI for PHY in NTNs. Considering the current NTN framework and the envisioned standardization activities towards Release 20 and 6G, the following short-term research directions are of particular interest:

- **GNSS-free operation**. Currently, NTN systems operate under the assumption of GNSS availability at the UE. Thanks to the introduction of ancillary information in the System Information Block (SIB) 19 in Release 17, including the satellite ephemeris information, a UE can maintain synchronization to an NTN gNB by pre-compensating the Doppler effect and the propagation delay if it is able to estimate its own position. In the context of GNSS-free operation, the initial synchronization phase represents a challenging task due to the large Doppler shift and delay ranges within a LEO-based NTN cell. Here, the signal processing capabilities of CNNs may prove useful to enable GNSS-free operation in NTNs; however, the scientific literature in this context is scarce and dedicated studies should be carried out to ensure that adequate synchronization accuracy can be achieved with reasonable complexity.

- **Channel estimation**. Channel estimation in NTN systems poses unique challenges compared to their terrestrial counterparts, primarily due to the high mobility of satellites, the long propagation delays, and the pronounced Doppler dynamics that lead to rapid channel variations. Conventional estimation techniques based on pilot-assisted schemes or linear filters may struggle to track these fast-varying and highly non-stationary channels efficiently. In this context, data-driven approaches based on AI, with a particular focus on ResNets and CNNs, can learn to exploit the temporal and spatial correlations in the NTN channel, enabling more accurate and robust estimation under limited pilot overhead. Moreover, DL-based estimators can adapt to diverse propagation conditions, such as varying elevation angles or differential Doppler across beams, which are difficult to model analytically.

- **CSI feedback compression**. In NTN systems, the feedback of CSI from the UE to an orbiting gNB is challenging due to long propagation delays and limited uplink bandwidth, particularly in LEO and GEO scenarios. Conventional codebook-based feedback schemes may become inefficient, leading to outdated or coarse channel information. AI-based approaches, such as AEs, can learn compact representations of the CSI tailored to the NTN channel characteristics, including sparsity and temporal correlation, enabling more efficient and accurate feedback. However, additional studies are needed to quantify the trade-off between compression efficiency,

reconstruction accuracy, and complexity before these techniques can be adopted in practical NTN deployments.

Further long-term research directions, *i.e.*, beyond Release 20, can also be envisioned:

- **Orthogonal Time Frequency Space (OTFS) multiplexing**. Although only CP-OFDM and Direct Fourier Transform (DFT)-spread OFDM have been confirmed for 6G, various candidate waveforms have been considered in 3GPP standardization activities, including OTFS. Thanks to the delay-Doppler domain processing, the Doppler shift due to the relative speed of a LEO satellite can be better managed with respect to OFDM-based waveforms; however, data equalization typically require iterative algorithms, introducing not only higher complexity, but also a higher latency at the receiver. In this context, DL models can approximate optimal algorithms and accelerate the equalization of data symbols in OTFS-based NTN systems.

- **Grant-free transmission**. Grant-free access can significantly reduce signaling overhead and latency in NTN systems by allowing UEs to transmit short packets without entering connected mode. However, the wide range of Doppler shifts and propagation delays, also considering the potential collisions among asynchronous users, make reliable detection and decoding particularly challenging. AI-based receivers can jointly perform user activity detection, preamble identification, and data decoding by learning complex correlations in the received signal, even under severe timing and frequency offsets. In the context of multi-user detection, convolution-based ResNets may prove useful to iteratively identify and subtract different users' signals.

- **End-to-end transceiver optimization**. End-to-end learning offers a promising alternative to traditional transceiver design by jointly optimizing the transmitter and receiver through data-driven training, allowing the system to learn signal representations that are inherently robust to NTN impairments. Using autoencoding architectures, modulation, coding, and detection can be co-optimized to maximize communication reliability and spectral efficiency under realistic channel dynamics. Such approaches can adapt to non-stationary environments and hardware nonlinearities without relying on explicit analytical channel models.

Appendix A

SINR Estimation in User-centric NTNs

This study was carried out during the third year of PhD under the framework of the SNS-JU 5G-STARDUST project. The contents of this chapter are based on the project deliverable "D4.7 - Final report on AI-based radio resource management, RAN softwarisation and onboard processing" [108] and the paper "Attention-based SINR estimation in User-centric NTNs" [109].

The SINR achieved on each communication link plays a major role in the effectiveness of the data exchange. On the one hand, the achievable rate in communication systems is limited by the SINR through the Shannon formula, effectively limiting the amount of bits that can be sent per time and bandwidth unit. On the other hand, procedures in communication systems rely on SINR-related metrics to take decisions, *e.g.*, handovers from one cell to another can be carried out once the measured Reference Signal Received Quality (RSRQ) overcome a certain threshold, [110]. Such proxy for link quality can typically be measured at the terminal using reference signals; however, NTNs introduce complications during the reporting phase: Geostationary-class satellites are characterized by a large propagation delay which can result in outdated reports, *e.g.*, in the case of user mobility, while Non-Geostationary-class satellites at lower orbit claim lower latency but introduce aging effects due to their inherent orbital movement. Furthermore, reporting occupies communication resources that can otherwise be dedicated to data transmission: *e.g.*, scheduling reports in location-based beamforming NTN systems only contain the users' geographical coordinates, which can be estimated at the terminal by means of GNSS and thus do not require the transmission of dedicated pilot data from the NTN's side. Focusing on user-centric beamforming NTN systems, the assessment of the SINR strongly relies on the set of users scheduled during the same time slot. Indeed, once a group of users is scheduled and the beamforming matrix is computed, either by means of MMSE, Location-based MMSE (LB-MMSE), or Conventional Beamforming (CBF), the SINR achieved by each user can be evaluated using equation (3.10) (considering the

theoretical clear sky channel formula based on the user's location in case of LB-MMSE). However, such evaluation is based on the MMSE equation (3.12), which has a computational complexity in the order of $\mathcal{O}(N_{UE,sched}^2 N_R)$, where $N_{UE,sched}$ is the number of users scheduled during the same time slot and N_R represents the number of radiating elements on board of the satellite payload. Thanks to the function approximation capabilities of neural networks, the SINR achieved by a group of scheduled users in a user-centric beamforming NTN system may be evaluated based on the scheduling report with decreased computational complexity. For this purpose, the popular attention mechanism can be employed to identify relationships between users' scheduling reports, thus extracting information related to potential inter-user interference. Such algorithm can be pre-trained and deployed in the near-real time RAN Intelligent Controller (RIC), providing a useful metric for tasks such as user scheduling (*e.g.*, to select a group of users from a candidate set based on the group's mean achievable SINR).

A.1 Dual multi-headed self-attention for SINR estimation

A.1.1 Introduction to Multi-headed self-attention

The attention mechanism was first introduced in the context of natural language processing to let algorithms selectively process sentences based on the relationships between words [111]. Attention maps a triplet of vectors, namely the query $\mathbf{q}$, key $\mathbf{k}$, and value $\mathbf{v}$, to an output vector $\mathbf{a}$. Such output vector is, in general, computed as a weighted sum of $\mathbf{v}$, where the weights represent the compatibility of the query with the corresponding key. Intuitively, a mask is extracted from the query and the key and is then applied to the value vector, thus including only the relevant values in the output. The most popular version, named Scaled Dot-Product Attention (SDPA), was presented in [112] and is the key function of the famous transformer architecture at the basis of modern LLMs. Considering a set of m queries and n pairs of unordered keys and values simultaneously, represented by the matrices $\mathbf{Q} \in \mathbb{R}^{m \times d_k}$, $\mathbf{K} \in \mathbb{R}^{n \times d_k}$, and $\mathbf{V} \in \mathbb{R}^{n \times d_v}$, respectively, the SDPA can be computed as:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} \odot \mathbf{M}\right)\mathbf{V}, \tag{A.1}$$

where d_k is the dimension of the queries and keys vectors, d_v is the size of the values vectors, and $\mathbf{M} \in \mathbb{R}^{m \times n}$ is an input matrix that masks out values that should not be considered for the computation of the attention scores. Clearly, the first term of the

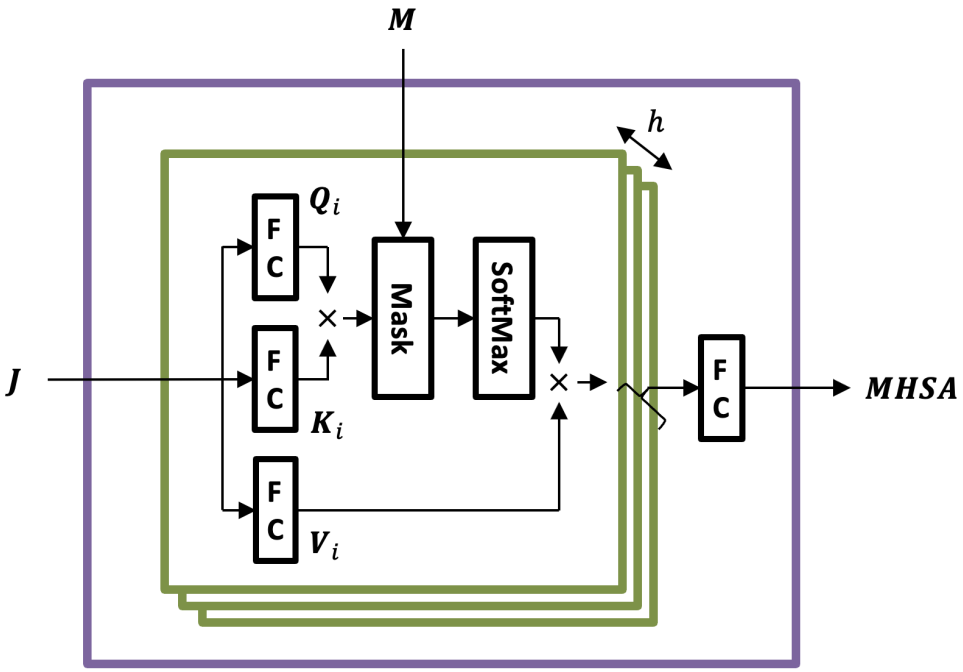


Figure A.1.1: Multi-Headed Self-Attention diagram.

matrix multiplication represents the weights, which are normalized to 1 for each query through the softmax function:

$$Softmax(\mathbf{h})_i = \frac{e^{h_i}}{\sum_{j=1}^{n} e^{h_j}} \tag{A.2}$$

In the case of self-attention, $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are computed from the same input matrix $\mathbf{J} \in \mathbb{R}^{n \times d}$, typically representing a sequence of $n = m$ tokens with embedding size d, by means of three FC layers with d_k or d_v neurons applied to each token separately:

$$\mathbf{Q} = \mathbf{J}\mathbf{W}^Q,\ \mathbf{W}^Q \in \mathbb{R}^{d \times d_k} \tag{A.3}$$

$$\mathbf{K} = \mathbf{J}\mathbf{W}^K,\ \mathbf{W}^K \in \mathbb{R}^{d \times d_k} \tag{A.4}$$

$$\mathbf{V} = \mathbf{J}\mathbf{W}^V,\ \mathbf{W}^V \in \mathbb{R}^{d \times d_v}. \tag{A.5}$$

Furthermore, the authors in [112] observed that the SDPA mechanism is able to extract better attention scores if separately applied to different sets of queries, keys, and values computed on the same input matrix. Thus, J can be linearly projected h times in parallel using a different set of weights for each head, represented by the matrices $\{\mathbf{W}_i^Q\}_{i=1}^h$, $\{\mathbf{W}_i^K\}_{i=1}^h$, and $\{\mathbf{W}_i^V\}_{i=1}^h$, to obtain a set of h queries matrices $\{\mathbf{Q}_i\}_{i=1}^h = \mathbf{J}\{\mathbf{W}_i^Q\}_{i=1}^h$ and a set of h pairs of keys and values matrices $\{\mathbf{K}_i\}_{i=1}^h = \mathbf{J}\{\mathbf{W}_i^K\}_{i=1}^h$ and $\{\mathbf{V}_i\}_{i=1}^h = \mathbf{J}\{\mathbf{W}_i^V\}_{i=1}^h$, with $d = h \cdot d_k$ holding true. h SDPA modules are implemented to obtain h separate attention scores, which can then be concatenated and processed with a last FC layer with

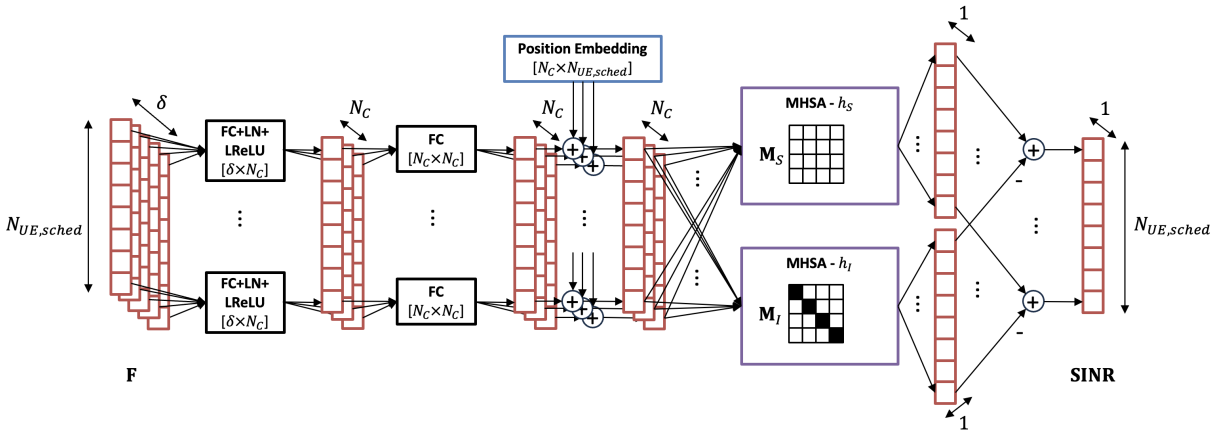


Figure A.1.2: DMHSA SINR estimation model.

weights matrix $\mathbf{W}^O \in \mathbb{R}^{d_v \times d}$ to obtain the MHSA matrix (Figure A.1.1):

$$MHSA = Concatenate(head_1, head_2, \ldots, head_h)\mathbf{W}^O, \tag{A.6}$$

$$head_i = Attention(Q_i, K_i, V_i). \tag{A.7}$$

A.1.2 DMHSA model

The considered deep learning approach to SINR estimation is a Dual MHSA (DMHSA) model (Figure A.1.2). The neural network takes as input a matrix $\mathbf{F} \in \mathbb{R}^{N_{UE,sched} \times \delta}$ containing a set of δ features for each of the $N_{UE,sched}$ scheduled users (with $N_{UE,sched} \leq N_B$, where $N_B = 24$ is the maximum number of beams that the considered antenna array can generate). It should be noted that a different number of users may be scheduled in different samples within a batch: to ensure that computation in the DMHSA model can be efficiently parallelized, a binary mask $\mathbf{m}^{(DMHSA)} = [\mathbf{1}^{1 \times N_{UE,sched}}, \mathbf{0}^{1 \times (N_B - N_{UE,sched})}] \in \mathbb{R}^{1 \times N_B}$ (where $\mathbf{1}^{a \times b}$ and $\mathbf{0}^{a \times b}$ represent the $a \times b$ matrix containing only ones and zeros, respectively) is also given as input to the model and applied to each operation for each input sample. Thus, the model applies a function $f^{(DMHSA)}(\cdot)$ that maps the features matrix $\mathbf{F}$ to the corresponding vector of estimated SINR $\hat{\mathbf{Y}}^{(DMHSA)} = f^{(DMHSA)}(\mathbf{F}, \mathbf{m}^{(DMHSA)})$ based on the binary mask. For a generic user k, the corresponding features vector $\mathbf{F}_{k,:} \in \mathbb{R}^{\delta}$ is generated as follows:

- In the case of CSI-based beamforming, the polar representation is selected. To reduce the dimensionality of the data, three types of information are concatenated to form the features vector: 1) the standardized argument of the user's CSI vector (*i.e.*, the phase of the CSI divided by its standard deviation $\sigma_\varphi = \pi/\sqrt{3}$, under the assumption $\angle \hat{h}_{k,n}^{(t)} \sim U(-\pi, \pi)\ \forall k, n \in 1, \ldots, N_R$), $\left[\frac{\angle \hat{\mathbf{h}}_{k,1}^{(t)}}{\sigma_\varphi}, \ldots, \frac{\angle \hat{\mathbf{h}}_{k,n}^{(t)}}{\sigma_\varphi}, \ldots, \frac{\angle \hat{\mathbf{h}}_{k,N_R}^{(t)}}{\sigma_\varphi}\right]$;

2) the standardized mean squared absolute value of the user's CSI vector, $\left|\tilde{\mathbf{h}}_k^{(t)}\right|^2 = \left(\sum_{n=1}^{N_R}\left|\hat{\mathbf{h}}_{k,n}^{(t)}\right|^2 - N_R\mu_h\right)/\left(N_R\sigma_h\right)$, with μ_h and σ_h being standardization parameters to be determined statistically; and 3) the ratio $N_{UE,sched}/N_B$. It should be noted that, given the considered channel model, $\left|\tilde{\mathbf{h}}_k^{(t)}\right|^2 = \left|\hat{\mathbf{h}}_{k,n}^{(t)}\right|^2 \ \forall k, n \in 1, \ldots, N_R$ as the path between a user and each radiating element is assumed to be subject to the same attenuation phenomena. The features vector with CSI reporting has size $\delta^{(CSI)} = N_R + 2$ and can be expressed as:

$$\mathbf{F}^{(\mathrm{CSI})}[k; 1, \ldots, \delta^{(CSI)}] = \left[\frac{\angle\hat{\mathbf{h}}_k^{(t)}[1]}{\sigma_\varphi}, \ldots, \frac{\angle\hat{\mathbf{h}}_k^{(t)}[n]}{\sigma_\varphi}, \ldots, \frac{\angle\hat{\mathbf{h}}_k^{(t)}[N_R]}{\sigma_\varphi}, \left|\tilde{\mathbf{h}}_k^{(t)}\right|^2, \frac{N_{\mathrm{UE,sched}}}{N_B}\right]. \tag{A.8}$$

- In the case of location-based beamforming, the user's location is computed in the u-v reference system based on the location report. Then, the pair of coordinates $[u_k, v_k]$ is concatenated with the ratio $N_{UE,sched}/N_B$. Thus, the features vector with location reporting has size $\delta^{(GEO)} = 3$ and can be expressed as:

$$\mathbf{F}^{(\mathrm{GEO})}[k; 1, \ldots, \delta^{(GEO)}] = \left[u_k, v_k, \frac{N_{UE,sched}}{N_B}\right]. \tag{A.9}$$

It is worth mentioning that the addition of the $N_{UE,sched}/N_B$ ratio provides information related to the amount of interference to be expected, thus improving the SINR estimation accuracy.

The deep learning model employed for SINR estimation is depicted in Figure A.1.2. In the first section, a cascade of layers extracts embeddings from the features matrix $\mathbf{F}$, acting on each user's features vector independently. The sequence includes:

- FC layers with $N_C = 8$ neurons.
- Layer Normalization (LN) layers, which normalize their input vector $\mathbf{x}^{(LN)}$ with the mean $\mu^{(LN)}$ and standard deviation $\sigma^{(LN)}$ of $\mathbf{x}^{(LN)}$ and apply two vectors of learnable parameters $\boldsymbol{\gamma}^{(LN)}$ and $\boldsymbol{\beta}^{(LN)}$:

$$\mathbf{y}^{(\mathrm{LN})} = \boldsymbol{\gamma}^{(\mathrm{LN})} \cdot \frac{\mathbf{x}^{(\mathrm{LN})} - \mu^{(\mathrm{LN})}}{\sigma^{(\mathrm{LN})} + \epsilon} + \boldsymbol{\beta}^{(\mathrm{LN})}, \tag{A.10}$$

$$\mu^{(\mathrm{LN})} = \frac{1}{N_c}\sum_{n=1}^{N_c}\mathbf{x}_n^{(\mathrm{LN})}, \tag{A.11}$$

$$\sigma^{(\mathrm{LN})} = \sqrt{\frac{1}{N_c}\sum_{n=1}^{N_c}\left(\mathbf{x}_n^{(\mathrm{LN})} - \mu^{(\mathrm{LN})}\right)^2}, \tag{A.12}$$

where ϵ is an arbitrary small value for numerical stability.

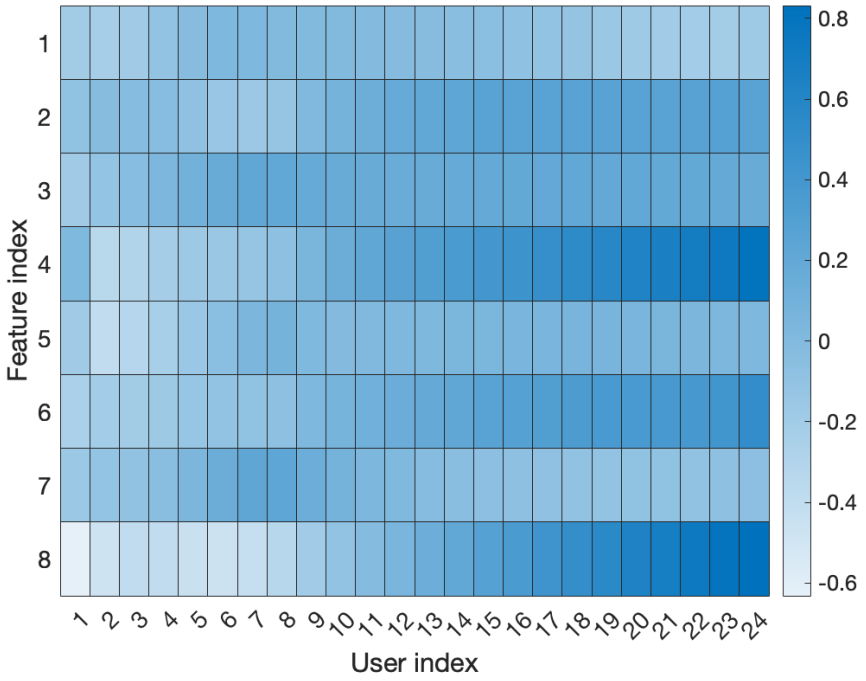


Figure A.1.3: Example of learnt PE vectors ($N_C = 8$).

- LReLU activation function.
- Position Embedding (PE), which creates a trainable vector of N_C weights for each user index to be summed to the corresponding user's generated features. In LLMs, PE can be used to inherently embed information regarding the distance between tokens (*i.e.*, word embeddings) in a sequence (*i.e.*, in a sentence), which provides aid to the MHSA in better understanding relationships between tokens. In SINR estimation, PE provides additional information on the number of scheduled users, thus improving the neural network's understanding of the amount of interference to be expected. Figure A.1.3 reports the heatmap of the learnt PE vectors in the case of location-based SINR estimation, highlighting more extreme changes in PE for users with high index. Indeed, only the first $N_{UE,sched}$ columns are used when $N_{UE,sched}$ users are scheduled.

The obtained vectors are then fed to two parallel MHSA + FC modules that operate on the set of features of $N_{UE,sched}$ users. Based on equation (3.10), the rationale behind such structure is to assess for each user the SNR in dB with the first module and $10log_{10}(1 + INR_k)$ with the second module, where INR_k represents the interference-to-noise ratio (INR) experienced by the k-th user. The first module employs $h_S = 4$ heads ($d_k = d_v = N_C/h = 2$) and does not include an attention mask (*i.e.*, $\mathbf{M}_S = \mathbf{1}^{N_{UE,sched} \times N_{UE,sched}}$) to extract all of the interactions between users; the second one includes $h_I = 4$ heads ($d_k =$

$d_v = N_C/h = 2$) and an attention mask $\mathbf{M}_I = \mathbf{1}^{N_{UE,sched} \times N_{UE,sched}} - \mathbf{I}^{N_{UE,sched} \times N_{UE,sched}}$, where $\mathbf{I}^{a \times a}$ is the a×a identity matrix, in order to exclude the interactions between a user and itself and including only interference-related attention scores. The output of each MHSA step is fed to a dedicated FC layer with a single output neuron, which is applied to each user's features separately to compute the SNR or INR-related quantities. The standardized SINR in dB is computed by subtracting the second module's output from the first's, and the final output can be obtained by means of application of properly estimated standardization parameters.

A.1.3 Computational complexity

Regardless of the choice between CSI-based beamforming and location-based beamforming, the SINR can evaluated through equation (3.10) (with the theoretical CSI being computed based on the users' geographical location in case of location-based beamforming). When assessing the MMSE beamforming algorithm's complexity from (3.12), which is at the base of the SINR evaluation, two main sources of complexity can be identified:

- The matrix multiplication between the CSI matrix $\hat{\mathbf{H}}^{(t)} \in \mathbb{C}^{N_{UE,sched} \times N_R}$ and its conjugate transpose with complexity $\mathcal{O}\left(N_{UE,sched}^2 N_R\right)$; and
- The inverse operation on the $N_{UE,sched} \times N_{UE,sched}$ complex matrix $\hat{\mathbf{H}}^{(t)} \left(\hat{\mathbf{H}}^{(t)}\right)^H + \alpha_r \mathbf{I}_{N_{UE,sched}}$, with complexity $\mathcal{O}\left(N_{UE,sched}^3\right)$.

Thus, the overall complexity of the SINR estimation through the computation of the MMSE beamforming equation is in the order of $\mathcal{O}\left(N_{UE,sched}^2 N_R\right) + \mathcal{O}\left(N_{UE,sched}^3\right)$. However, the number of scheduled users $N_{UE,sched}$ is typically limited with respect to the number of radiating elements N_R, *i.e.*, $N_{UE,sched} \ll N_R$. With this consideration, the MMSE complexity reduces to:

$$\mathcal{O}_{MMSE} = \mathcal{O}\left(N_{UE,sched}^2 N_R\right). \tag{A.13}$$

The computational complexity of the DMHSA model can be mainly attributed to FC and MHSA layers:

- The first FC layer embeds δ features into a vector of size N_C for each user, resulting in a computational complexity $\mathcal{O}\left(\delta N_{UE,sched} N_C\right)$.
- The second FC layers further refines the N_C features for each user, resulting in a computational complexity $\mathcal{O}\left(N_{UE,sched} N_C^2\right)$.

- For each head, the MHSA layers firstly apply linear projections to compute the queries, keys, and values matrices, achieving complexity $\mathcal{O}\left(N_{UE,sched}N_C d_k\right)$; the subsequent computations in the SDPA involve a matrix multiplication between queries and keys ($\mathcal{O}\left(N_{UE,sched}^2 d_k\right)$) and between the softmax outputs and the values ($\mathcal{O}\left(N_{UE,sched}^2 d_v\right)$); finally, the heads outputs are concatenated to form a vector of $N_C = h \cdot d_k$ features and passed through one final FC layer ($\mathcal{O}\left(N_{UE,sched}N_C^2\right)$). Thus, recalling $d_k = d_v = N_C/h$, the computational complexity of the MHSA layers is:

$$\begin{aligned}\mathcal{O}_{MHSA} &= h \cdot \left[2\mathcal{O}\left(N_{UE,sched}N_C d_k\right) + \mathcal{O}\left(N_{UE,sched}^2 N_C d_v\right)\right] + \mathcal{O}\left(N_{UE,sched}N_C^2\right) = \\ &= h \cdot \left[2\mathcal{O}\left(N_{UE,sched}(N_C^2)/h\right) + \mathcal{O}\left(N_{UE,sched}^2 N_C/h\right)\right] + \mathcal{O}\left(N_{UE,sched}N_C^2\right) = \\ &= \mathcal{O}\left(N_{UE,sched}^2 N_C + N_{UE,sched}N_C^2\right). \end{aligned} \tag{A.14}$$

  Notably, the MHSA complexity does not depend on the number of attention heads h.

- The last pair of FC layers generates one feature from a vector of length N_C for each user, resulting in a computational complexity $\mathcal{O}\left(N_{UE,sched}N_C\right)$.

Thus, the overall computational complexity with the DMHSA is:

$$\mathcal{O}_{DMHSA} = \mathcal{O}\left(\delta N_{UE,sched}N_C\right) + \mathcal{O}\left(N_{UE,sched}^2 N_C + N_{UE,sched}N_C^2\right). \tag{A.15}$$

In case of CSI-based beamforming, $\delta^{(CSI)} = N_R + 2$; thus, the computational complexity becomes $\mathcal{O}\left(N_{UE,sched}N_C N_R\right) + \mathcal{O}\left(N_{UE,sched}^2 N_C + N_{UE,sched}N_C^2\right)$. Considering that $N_{UE,sched} \ll N_R$, the complexity reduces to $\mathcal{O}\left(N_{UE,sched}N_C N_R + N_{UE,sched}N_C^2\right)$. Furthermore, the choice of $N_C = 8$ makes so that $N_C \ll N_R$; thus, the overall computational complexity for SINR estimation with the CSI-DMHSA model is:

$$\mathcal{O}_{DMHSA}^{(CSI)} = \mathcal{O}\left(N_{UE,sched}N_C N_R\right), \tag{A.16}$$

which, given $N_C < N_{UE,sched}$ (typically $N_{UE,sched} = N_B = 24$), implies that the complexity is effectively reduced with respect to the evaluation through the computation of the MMSE beamforming equation. In case of location-based beamforming, $\delta^{(GEO)} = 3$; thus, the computational complexity becomes $\mathcal{O}\left(N_{UE,sched}^2 N_C + N_{UE,sched}N_C^2\right)$. With the same consideration on N_C as above, the complexity of the GEO-DMHSA model reduces to:

$$\mathcal{O}_{DMHSA}^{(GEO)} = \mathcal{O}\left(N_{UE,sched}^2 N_C\right), \tag{A.17}$$

providing a great computational complexity reduction for SINR estimation. The results of the computational complexity assessment are summarized in Table A.1.
Figure A.1.4 reports the Big-O computational complexity achieved by the two DMHSA models (CSI-based and location-based in orange and yellow, respectively) and the MMSE

Table A.1: Computational complexity for SINR estimation.

SINR estimation method	**Computational complexity**
MMSE-based	$\mathcal{O}_{MMSE} = \mathcal{O}\left(N_{UE,sched}^2 N_R\right)$
CSI-DMHSA	$\mathcal{O}_{DMHSA}^{(CSI)} = \mathcal{O}\left(N_{UE,sched} N_C N_R\right)$
GEO-DMHSA	$\mathcal{O}_{DMHSA}^{(GEO)} = \mathcal{O}\left(N_{UE,sched}^2 N_C\right)$

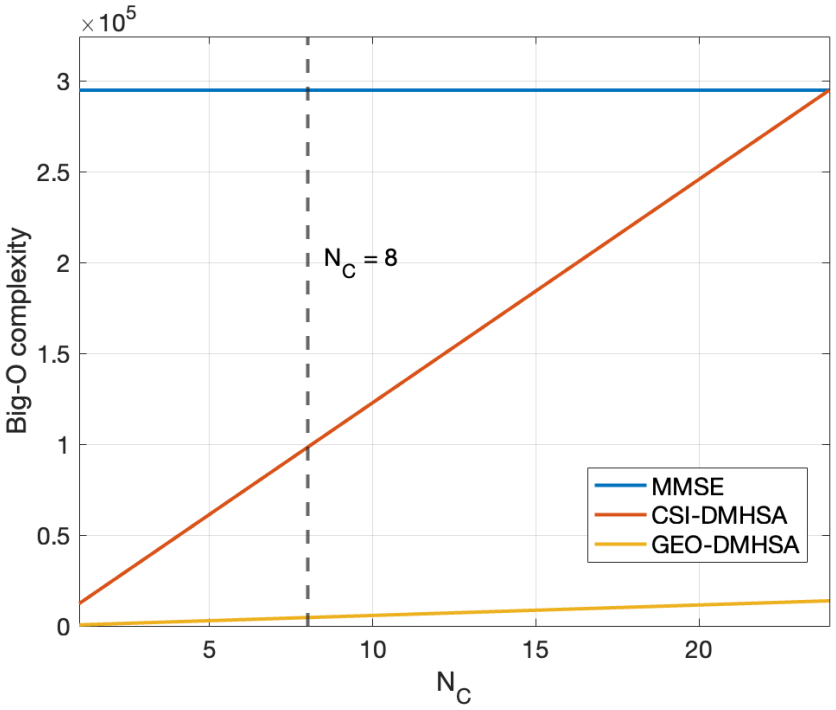


Figure A.1.4: Computational complexity as a function of N_C.

baseline (blue line) as a function of the number of channels N_C, *i.e.*, the size of the embeddings of each user's features vector; the grey dotted line represents the value set in the trained models, *i.e.*, $N_C = 8$. The plot shows that the reduced features vector in the GEO-DMHSA model ensures that the complexity remains orders of magnitude lower than that of MMSE-based SINR estimation: at $N_C = 8$, the location-based algorithm achieves a complexity of $\approx 4.6 \cdot 10^3$ against the $\approx 2.9 \cdot 10^5$ required by the benchmark. The CSI-DMHSA model also ensures a reduction in complexity to $\approx 9.8 \cdot 10^4$; however, the large features vector makes so the complexity in the CSI-based case scales much faster than in the location-based case with respect to N_C: while longer embeddings could improve the SINR estimation accuracy, the increased complexity may disfavour the application of the CSI-DMHSA method. This is not the case with GEO-DMHSA, where even with

embeddings of 24 elements the complexity is at one order of magnitude under the MMSE-based SINR estimation. It should also be mentioned that the CSI-DMHSA and GEO-DMHSA models require 4.8k and 762 learnable parameters, respectively, making them extremely compact.

A.1.4 Model training

Two separate models with the reported DMHSA architecture have been trained using CSI reports as input data for one of them and location reports for the other. The models have been trained by simulating scheduling instances at random samples of a satellite pass, using the same parameters reported in Table A.2. At each instance i, $N_{UE,sched,\,i} \sim U(8, N_B)$ users are randomly selected for scheduling, their CSI vectors (or location vectors) are pre-processed and collected as training input data sample $\mathbf{F}$ (Section A.1.2), and their SINR is assessed by means of the MMSE equation and stored as label (*i.e.*, true SINR value) for the corresponding input data sample. Once the batch is filled with $N_{batch} = 8192$ samples, the input data $\mathbf{X}^{(DMHSA)} \in \mathbb{R}^{N_B \times \delta \times N_{batch}}$ and the corresponding padding masks $\mathbf{M}^{(DMHSA)} \in \mathbb{R}^{N_B \times N_{batch}}$ are fed to the model, and the corresponding SINR estimates $\hat{\mathbf{Y}}^{(DMHSA)} \in \mathbb{R}^{N_B \times N_{batch}}$ are obtained; then, the MSE loss function is evaluated between the SINR estimates at the output of the model and the corresponding ground truth labels $\mathbf{SINR}^{(DMHSA)} \in \mathbb{R}^{N_B \times N_{batch}}$:

$$MSE = \frac{\sum_{i=1}^{N_{batch}} \sum_{u=1}^{N_B} \mathbf{M}^{(DMHSA)}[u;i] \odot \left(\hat{\mathbf{Y}}^{(DMHSA)}[u;i] - \mathbf{SINR}[u;i]\right)^2}{\sum_{i=1}^{N_{batch}} \sum_{u=1}^{N_B} \mathbf{M}^{(DMHSA)}[u;i]}. \tag{A.18}$$

Through the backpropagation algorithm and the Adam optimizer [93], the trainable weights of the model are updated, and a new batch of data starts being generated. The training process employed the same techniques reported in Section 4.2.2, with Table A.3 reporting the parameters employed to train both of the DMHSA models. Similarly, a new batch of data is generated at each epoch and samples contained in previous batches are not reused in the following epochs: for this reason, a validation set is not generated.

A.2 Performance evaluation

A.2.1 DMHSA with random scheduler and variable number of users

The objective of the first evaluation is to determine the SINR estimation error under general conditions: *e.g.*, training on randomly scheduled users ensures that the models

Parameter	**Value**	**Comments**
Subcarrier spacing	$\Delta f = 120$ kHz	132 PRBs available
Carrier frequency	$f_c = 20$ GHz	
User bandwidth	$B = 190.08$ MHz	
Scenario	Rural	See [113]
Number of radiating elements	$N_R = 512$	
Power per radiating element	$P_{av,el} = 65$ mW	
Total power on-board	$P_{av} = 15.2218$ dBW	$10log_{10}(P_{av,el} \cdot N_R)$
Minimum user elevation angle	30°	
Service area elevation angle	30°	
Channel model	Clear-sky	Based on 3.4 with $L_i^0 = 1$
NTN node altitude	1000 km	
Service area center	(45°N, 5°E)	

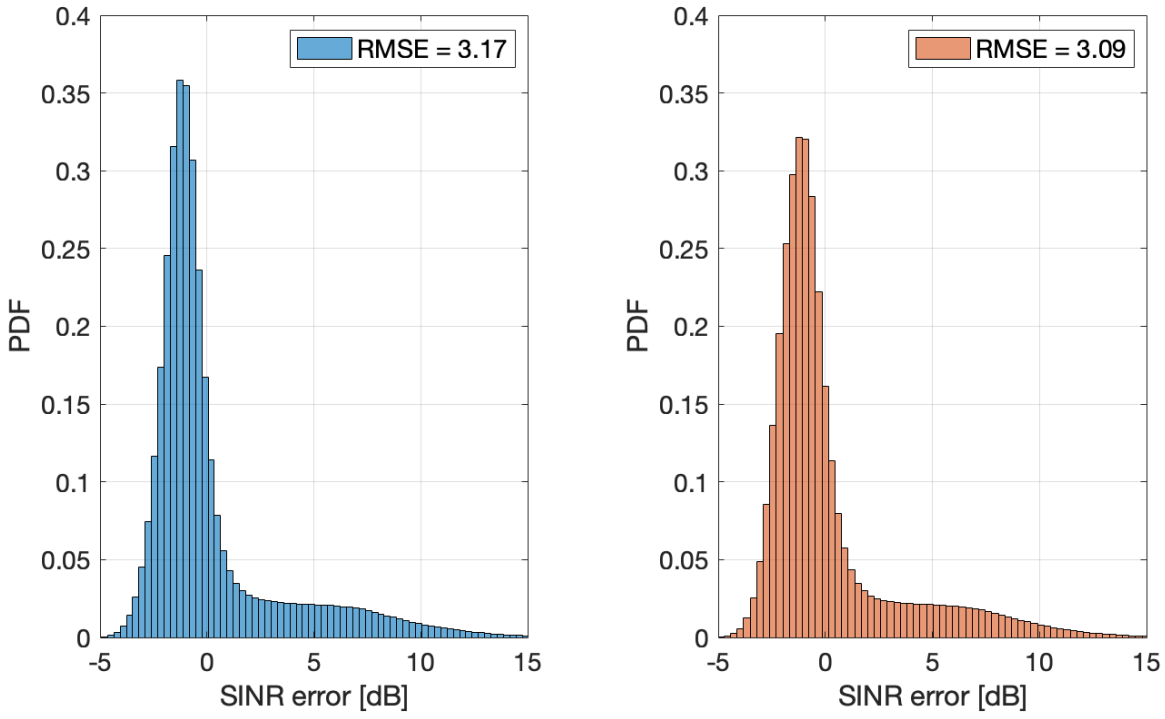
Table A.2: Simulation parameters for DMHSA training.

observe both cases of optimal scheduling and ill-conditioned beamforming matrices. For this reason, the models were first evaluated on a test set of $2.56 \cdot 10^6$ test samples under the same conditions of the training set, *i.e.*, each sample contains a random number of randomly scheduled users' reports obtained in random instants of the satellite pass, for a total of $\approx 4 \cdot 10^7$ SINR estimates. As for the training data, the simulation parameters used to generate test data are the same reported in Table A.2.

Figure A.2.1 reports the histogram of the error distribution for both CSI-DMHSA and GEO-DMHSA, computed as $\mathbf{E} = \hat{\mathbf{Y}}^{(DMHSA)} - \mathbf{SINR}$, with Probability Density Function (PDF) normalization. The distribution is skewed in both cases, suggesting that the models cannot provide accurate estimates in certain conditions: this is the case of ill-conditioned beamforming matrices resulting from poor user scheduling (*i.e.*, the random scheduler often selects users that are geographically close together, resulting in highly correlated CSI vectors). This consideration is made clearer with Figure A.2.2, where the same assessment is carried out by separating the test datasets based on the number of scheduled users. As expected, for both models, the Root MSE (RMSE) increases with the number of users scheduled, and so does the frequency of high-error estimations. Nonetheless,

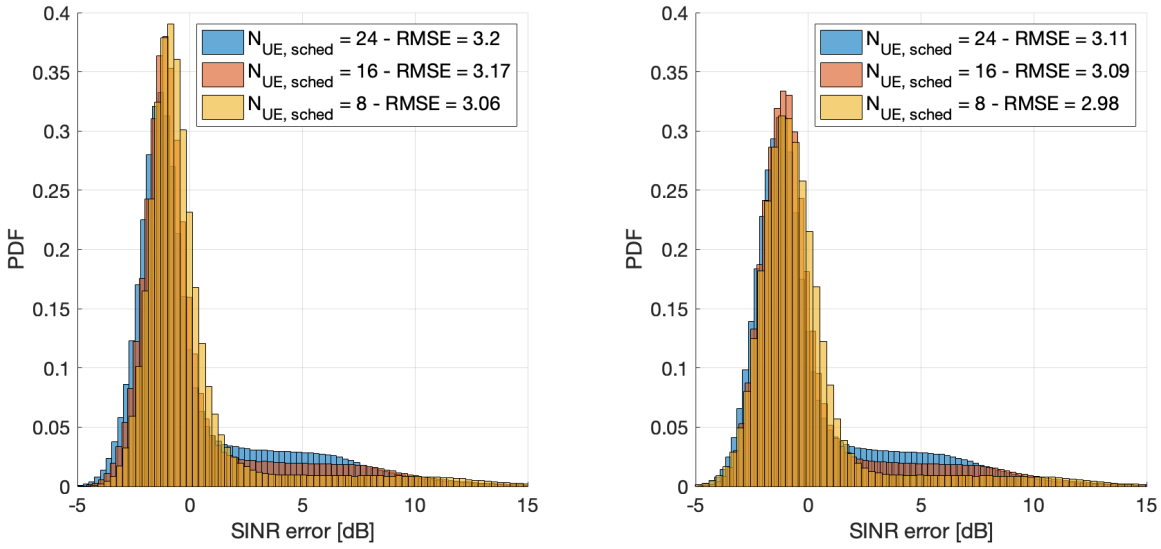
Training parameter	**Value**
Training batch size	$N_{batch} = 8192$
Maximum number of training epochs	15000 epochs
L2 regularization factor	$\epsilon_R = 10^{-6}$
Learning rate warm-up period	$T_w = 40$ epochs
Learning rate cosine annealing period	$T_c = 100$ epochs
Minimum learning rate	$\lambda_{min} = 10^{-4}$
Maximum learning rate	$\lambda_{min} = 5 \cdot 10^{-3}$
Early stopping patience	$T_s = 4$ full cosine annealing cycles

Table A.3: Training parameters for DMHSA.



(a) CSI-DMHSA (b) GEO-DMHSA

Figure A.2.1: SINR estimation error distribution with random scheduler.



(a) CSI-DMHSA (b) GEO-DMHSA

Figure A.2.2: SINR estimation error distribution with random scheduler for different $N_{UE,sched}$ values.

the lowest-complexity GEO-DMHSA model also achieves a lower SINR estimation RMSE with respect to the CSI-DMHSA model. While the error tails reach high SINR values, it should be noted that they are caused by poor scheduling choices made by the random scheduler; thus, the accuracy of the models should be evaluated considering a proper scheduling technique.

A.2.2 DMHSA with PQS scheduler

In the second evaluation step, the Priority Queue Scheduler (PQS) presented in [113] is implemented. In this context, a set of N_{UE} users is considered to be present in the service area, where the k-th user has a traffic request denoted as $\mathcal{C}_k^{(REQ)}$, $k = 1, \ldots, N_{UE}$; the objective of the scheduler is to select groups of, in the most general case, $N_{UE,sched} \leq N_B$ users to be served during the same time slot with user-centric beamforming. The PQS algorithm achieves this by 1) classifying the packets to be transmitted into $N_{PC} \geq 2$ priority classes based on the unmet capacity request and the remaining visibility of the corresponding user; 2) filling N_{PC} priority-based queues with the corresponding packets; and 3) for each time slot in a scheduling period, randomly selecting packets from the non-empty queue with highest priority until the maximum number of schedulable users per slot (corresponding to N_B) is reached, ensuring that a geographic or CSI scheduling constraint is satisfied. Such constraints, reported in [113], limit the selected users' CSI vectors' correlation, either using the actual CSI vectors in case of CSI-based scheduling,

Parameter	**Value**
Scheduling slot duration	$T_{slot}^{(sched)} = 10$ ms
Scheduling periodicity	$T_{sched} = 2$ s
Maximum residual visibility	$\sigma_{vis} = 50$ slots
Unmet capacity factor	$\sigma_{cap} = 2$
Number of priority classes	$N_{PC} = 2$
Minimum capacity request	$\mathcal{C}_{min} = \{5, 20\}$ Mbps
Maximum capacity request	$\mathcal{C}_{min} = \{100, 500\}$ Mbps

Table A.4: Configuration parameters for PQS.

or the inter-user distance as a proxy in case of geographic scheduling. By implementing the PQS algorithm, a more realistic case is considered for SINR estimation, in which a scheduling algorithm selects groups of users to be served during the same time slot for which the SINR should be assessed (*e.g.*, to determine the expected rate for each user, or to evaluate the sum-rate achievable by different groups in order to select which group to schedule). The PQS scheduler parameters and the additional simulation parameters with respect to Table A.2 are reported in Table A.4; the CSI and GEO scheduling constraint threshold was selected from the values reported in [113] based on the minimum and maximum requested capacity $\mathcal{C}_{min}$ and $\mathcal{C}_{max}$. It should be noted that the models are assumed to be deployed as is, *i.e.*, without retraining on user reports with PQS scheduling. For this reason, due to the different SINR statistics with respect to the training dataset, the obtained estimator is biased; such bias is estimated and subtracted from the models' output.

Figure A.2.3 reports the CDFs of the absolute error $\mathbf{E}_{abs} = \left|\hat{\mathbf{Y}}^{(DMHSA)} - \mathbf{SINR}\right|$ achieved by the CSI-DMHSA and GEO-DMHSA models under the minimum and maximum capacity requests $\mathcal{C}_{min}$ and $\mathcal{C}_{max}$, respectively, reported in the legend as $\mathcal{C}_{min}/\mathcal{C}_{max}$. The plot shows that both models are able to achieve a median absolute error (*i.e.*, corresponding to the 50% CDF) under 0.7 dB for all cases. Overall, the performance of the DMHSA models is similar, with the location-based model providing slightly improved high percentiles and the CSI-based model excelling in the lower ones (except in the 5/100 case, where the opposite is true). The plot also shows that the cases with the largest $\mathcal{C}_{min}$ are the ones providing the best absolute error: as reported in [113], a low $\mathcal{C}_{min}$ value leads the PQS scheduler tends to schedule less users than the number of beams, thus resulting

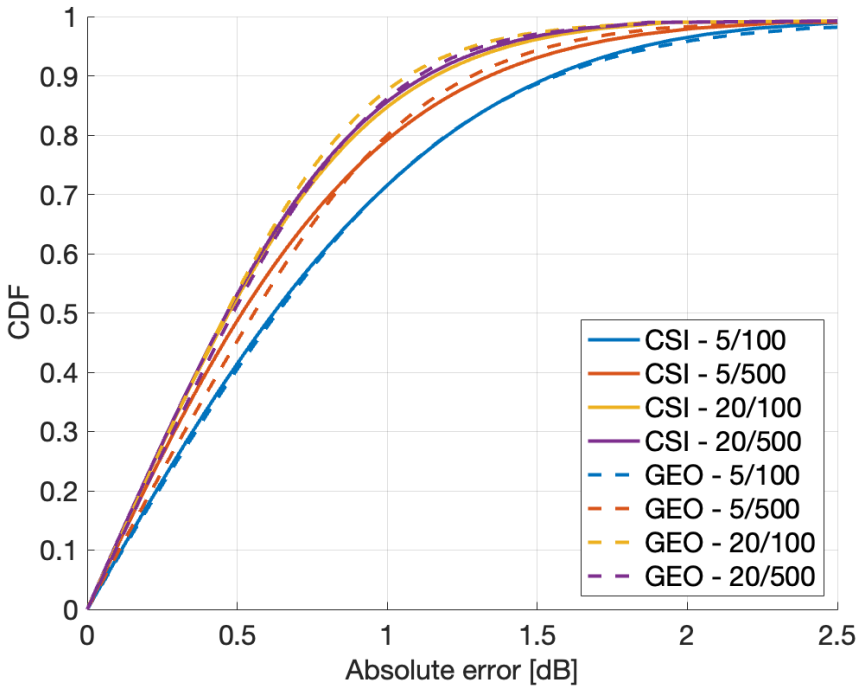


Figure A.2.3: CDF of the SINR estimation absolute error with PQS scheduling.

in consistently high SINR values, a case that was not well represented in the training data considering the implementation of a random scheduler; on the opposite, $\mathcal{C}_{min} = 20$ Mbps results in a better aligned SINR distribution. Similarly, an increase of $\mathcal{C}_{max}$ from 100 Mbps to 500 Mbps with $\mathcal{C}_{min} = 5$ Mbps leads to a significant improvement of the absolute error due to the more stringent constraint on the CSI vectors correlation and the inter-user distance.

Table A.5 reports the SINR estimation RMSE achieved by the DMHSA models under the considered capacity requests. Notably, the RMSE remains under the 1dB threshold in most cases (except for the 5/100 case with GEO-DMHSA). The two models perform similarly at high minimum capacity request; on the opposite, the CSI-DMHSA model has an advantage over its location-based counterpart at low minimum capacity request. Clearly, the observations reported for Figure A.2.3 are also reflected in the RMSE.

A.3 Conclusions and future enhancements

The objective of this study was that of exploiting AI/ML solutions to estimate the SINR of a group of users scheduled for transmission in a user-centric beamforming system, reducing the computational complexity with respect to the assessment of the SINR through the beamforming matrix while maintaining satisfying estimation accuracy. The considered ar-

		CSI-DMHSA		**GEO-DMHSA**	
		$\mathcal{C}_{max}$			
		100 Mbps	**500 Mbps**	**100 Mbps**	**500 Mbps**
$\mathcal{C}_{min}$	**5 Mbps**	0.96 dB	0.84 dB	1.18 dB	0.90 dB
	20 Mbps	0.71 dB	0.70 dB	0.68 dB	0.70 dB

Table A.5: SINR estimation RMSE with PQS.

chitecture, based on the MHSA mechanism, achieves a reduction of the Big-O complexity of a factor 3 in the CSI-based case and of two orders of magnitude in the location-based case. This last result is a consequence of the lower dimensionality of the user reports for location-based user-centric beamforming systems, which avoid scaling the computational complexity by the number of antenna elements N_R. Furthermore, both models achieve satisfying SINR estimation performance considering users with various capacity requirements being scheduled with the PQS algorithm, with the RMSE not passing the 1dB threshold in most of the cases. With the estimation performance being evaluated, the DMHSA models can be deployed for inference in different procedures. Scheduling algorithms typically provide a single group of users to be served during the same time slot. While mechanisms, *e.g.*, the priority queue and the CSI correlation / inter-user distance check of the PQS algorithm, are in place to improve the capacity offered in a time slot, the decision is often sub-optimal in this regard. Thus, the capacity to a group of potentially scheduled users may be evaluated through the SINR by means of DMHSA-based estimation. This opens the way to schedulers that propose multiple candidate groups and prioritize that providing the highest average SINR, *i.e.*, the highest offered capacity.

Appendix B

A Primer on Artificial Intelligence

The purpose of this chapter is to provide a brief overview of AI and the techniques implemented in the preceding chapters of this thesis. For a more detailed presentation of AI-, ML-, and DL-related aspects, one can refer to [114, 115, 116].

B.1 Artificial Intelligence, Machine Learning, and Deep Learning

AI broadly refers to the capability of machines to perform tasks that would typically require human intelligence, such as perception, reasoning, learning, and decision-making. Rather than being a single technique or algorithm, AI represents an umbrella term encompassing a wide spectrum of methods aimed at enabling systems to act intelligently in complex and often uncertain environments. Historically, early AI systems relied heavily on symbolic reasoning, expert systems, and hand-crafted rules, which explicitly encoded human knowledge. While effective in some scenarios, such approaches often struggled to scale or adapt to dynamic conditions, motivating a shift toward data-driven paradigms.

Within this broader AI landscape, Machine Learning (ML) emerged as a paradigm in which systems improve their performance through experience, typically in the form of data. Instead of being explicitly programmed with rigid rules, ML models infer patterns, relationships, and representations directly from observations. This data-centric approach has proven particularly powerful in domains characterized by high dimensionality, noise, and uncertainty, all of which are common features of modern communication systems. In general terms, ML algorithms aim to optimize a performance criterion by adjusting model parameters based on training data, allowing the learned behavior to generalize to previously unseen situations. ML encompasses a variety of learning settings, which differ in how supervision and feedback are provided during training:

- **Supervised learning**, where models are trained using labeled data pairs, enabling tasks such as classification or regression.
- **Unsupervised learning**, which focuses on discovering latent structures or statistical regularities in unlabeled data.
- **Reinforcement learning**, where an agent interacts with an environment and learns a policy through trial-and-error based on reward signals.

Although these categories differ in formulation, they share the fundamental objective of enabling systems to learn from data and improve autonomously over time.

DL can be viewed as a specialized subfield of machine learning that is distinguished by its use of multi-layered models, commonly referred to as Deep Neural Networks. The defining characteristic of DL is its ability to automatically learn hierarchical representations of data, where higher-level features are constructed from lower-level ones. This representation learning capability reduces the need for manual feature engineering, which has traditionally been a bottleneck in many signal processing and optimization tasks. Advances in computational power, the availability of large-scale datasets, and algorithmic innovations have collectively driven the rapid adoption of DL across numerous application domains.

It is important to emphasize that DL is not a replacement for ML, but rather an extension of it. All DL methods are ML methods, yet not all ML techniques rely on deep architectures. Thus, from a conceptual standpoint, the relationship between AI, ML, and DL can be summarized as a nested hierarchy (Figure B.1.1):

- Artificial intelligence defines the overarching goal of creating intelligent systems.
- Machine learning provides a data-driven means to achieve AI by enabling systems to learn from experience.
- Deep learning represents a class of machine learning techniques based on deep neural architectures capable of learning complex representations.

B.2 Deep Neural Networks

DNNs constitute the core computational models underlying most modern deep learning approaches. Their conceptual foundation can be traced back to the perceptron, one of the earliest mathematical models of an artificial neuron. In its simplest form, a perceptron maps an input vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T \in \mathbb{R}^N$ to a single output through a

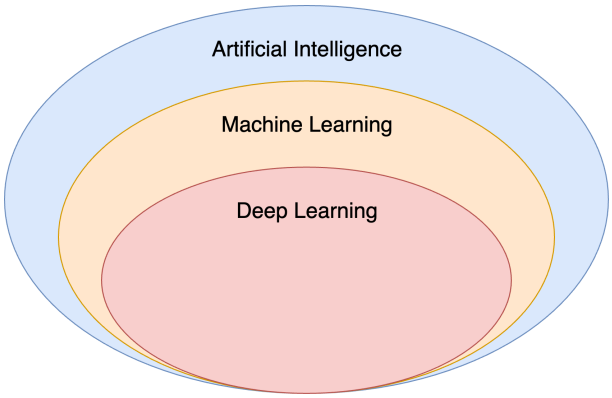


Figure B.1.1: Nested hierarchy of AI, ML, and DL.

weighted linear combination followed by a nonlinear activation function. Mathematically, this operation can be expressed as:

$$y = \phi\left(\mathbf{w}^T\mathbf{x} + b\right), \tag{B.1}$$

where $\mathbf{w} = [w_1, w_2, \ldots, w_N]^T \in \mathbb{R}^N$ denotes the vector of trainable weights, $b \in \mathbb{R}$ is a bias term, and $\phi(\cdot)$ is a nonlinear function that enables the model to represent complex input-output relationships. Despite its simplicity, this formulation captures the essential idea of learning a parametric mapping from data.

A DNN is obtained by interconnecting multiple perceptron-like units into a structured composition of successive transformations. By stacking such transformations in a layered structure, DNNs are able to model complex nonlinear functions through a sequence of simpler operations, effectively building intermediate representations that progressively abstract the input data (Figure B.2.1). From a functional perspective, a DNN implements a parameterized mapping between input and output spaces, where the parameters consist of the weights and biases associated with all constituent units. The depth of the network reflects the number of successive transformations applied, *i.e.*, the layers, which in turn governs the expressive power of the model.

The behavior of a DNN is determined by its parameters, which are learned from data through a training process that aims to minimize a predefined objective or loss function, as detailed in Section B.2.2. Once trained, a DNN can be viewed as a deterministic nonlinear system that processes new inputs according to the representations it has learned. This abstraction is particularly useful when interpreting DNNs as flexible function approximators, a perspective that underpins their growing adoption in signal processing and communication system design.

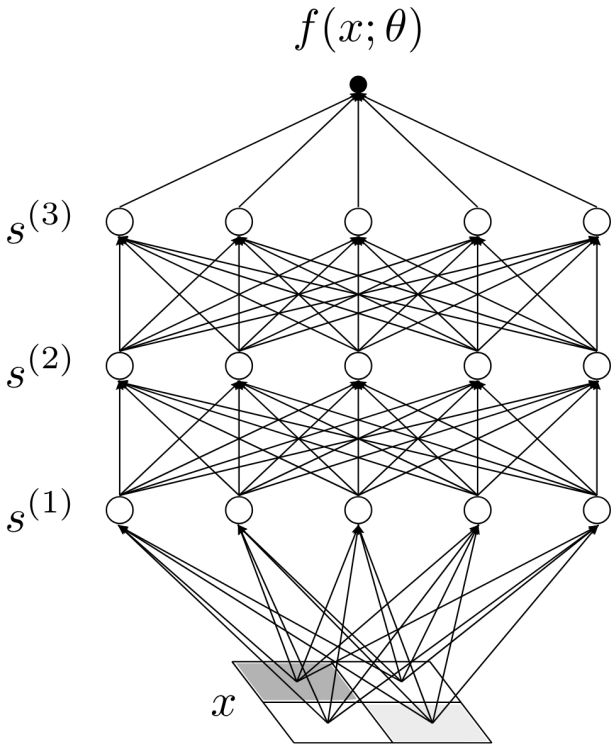


Figure B.2.1: Example of a multi-layer neural network: the input x is progressively transformed into intermediate representations $s^{(i)}$ at layer i to obtain an output $f(x;\theta)$ based on the parameters θ [116].

B.2.1 Typical layer architectures

While the perceptron is the basic unit of a neural network, not all layers employ such element in the same way. To extract different features, different layer architectures can be employed, even in the same neural network: for instance, a design choice for a model employed for time series classification may foresee layers specialized on temporal features extraction first, which can then be combined through general purpose layers. In the following paragraphs, a few of the most popular layers are presented.

Fully-Connected layers

Fully-connected (FC) layers, also referred to as dense layers, represent the most direct extension of the perceptron model to vector-valued inputs and outputs. In an FC layer, each output unit is connected to all input units, resulting in a global linear transformation followed by an element-wise nonlinearity. Given an input vector $\mathbf{x} \in \mathbb{R}^{N\times 1}$, an FC layer produces an output vector $\mathbf{y} \in \mathbb{R}^{M\times 1}$ according to:

$$\mathbf{y} = \phi\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right), \tag{B.2}$$

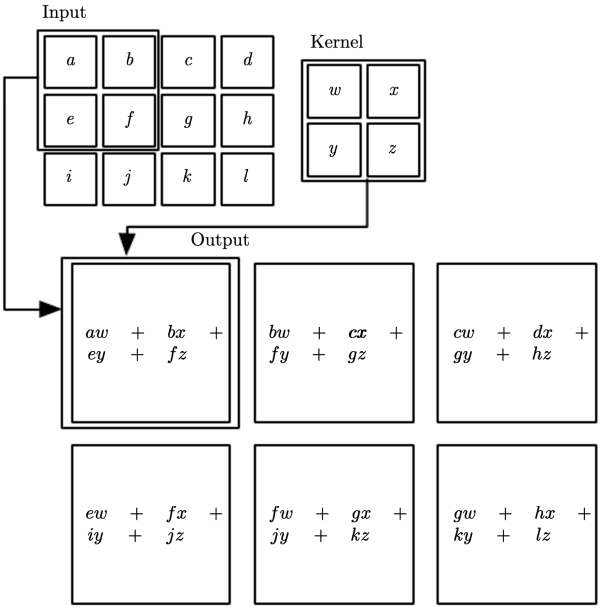


Figure B.2.2: Example of 2D convolution of a single input channel with a single kernel [114].

where $\mathbf{W} \in \mathbb{R}^{M \times N}$ is the weight matrix, $\mathbf{b} \in \mathbb{R}^{M \times 1}$ is the bias vector, and $\phi(\cdot)$ is the nonlinear activation function (assumed to be applied component-wise). This formulation highlights that an FC layer implements an affine mapping of the input space followed by a nonlinearity, thereby enabling the modeling of complex input-output relationships when combined with other layers. However, this comes at the cost of a large number of trainable parameters, which scales with the product of the input and output dimensions. As a result, FC layers tend to be computationally intensive and, thus, may not be the best choice for large-sized structured input data (*e.g.*, exhibiting spatial or temporal correlations based on domain expertise).

Convolutional layers

Two-dimensional (2D) convolutional layers are designed to efficiently process data that exhibit a spatial or grid-like structure. Unlike fully-connected layers, which apply a global transformation to the input, 2D convolutional layers exploit local connectivity and parameter sharing to capture spatially localized patterns. Let the input to a convolutional layer be represented as a set of feature maps collected in a tensor $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, where H and W denote the spatial dimensions and C the number of input channels. A 2D convolutional layer produces an output tensor $\mathbf{Y} \in \mathbb{R}^{H' \times W' \times K}$ by convolving the input

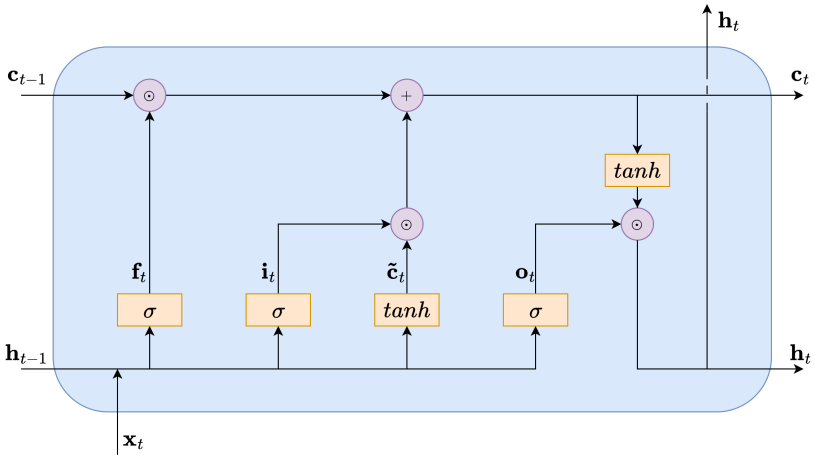


Figure B.2.3: Flow of information in an LSTM layer.

with a set of learnable kernels. Formally, the k-th output feature map is given by:

$$y_k(i,j) = \phi\left(\sum_{c=1}^{C}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} w_{k,c}\,[m;n]\,x_c\,[i+m;j+n] + b_k\right), \tag{B.3}$$

where $w_{k,c}\,[m;n]$ denotes the (m,n)-th coefficient of the kernel associated with the c-th input channel and the k-th output feature map, b_k is a bias term, and $\phi(\cdot)$ is the nonlinear activation function; the convolution process is exemplified in Figure B.2.2.

The defining advantage of 2D convolutional layers lies in their ability to learn spatially invariant features using a limited number of parameters. By reusing the same kernels across different spatial locations, convolutional layers significantly reduce model complexity compared to fully-connected alternatives, while preserving sensitivity to local correlations. This inductive bias makes them particularly effective when the relevant information is encoded in local patterns whose absolute position may vary across samples, as in the case of SCMA decoding (Chapter 2), channel prediction (Chapters 4 and 5), and FTN signaling equalization (Chapter 6).

Long Short-Term Memory layers

LSTM layers belong to the class of recurrent neural network architectures designed to model sequential data by explicitly accounting for temporal dependencies. Unlike feed-forward layers, which process inputs independently, LSTM layers operate on ordered sequences and maintain an internal state that evolves over time. This capability allows them to capture both short-term dynamics and long-range dependencies, which are difficult to model using conventional recurrent formulations due to issues such as vanishing or exploding gradients. At each discrete time step t, an LSTM layer receives an input vector $\mathbf{x}_t \in \mathbb{R}^{N\times 1}$, a hidden state $\mathbf{h}_{t-1}$, and a cell state $\mathbf{c}_{t-1}$ from the previous time step.

The evolution of the cell state is governed by a set of gating mechanisms that regulate the flow of information (Figure B.2.3). A common formulation is given by [117]:

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f\right), \tag{B.4}$$

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i\right), \tag{B.5}$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o\right), \tag{B.6}$$

$$\tilde{\mathbf{c}}_t = \tanh\left(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c\right), \tag{B.7}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t, \tag{B.8}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right), \tag{B.9}$$

where $\sigma(\cdot)$ denotes the logistic sigmoid function ($\sigma(x) = (1 + e^{-x})^{-1}$), $\odot$ represents element-wise multiplication, and $\mathbf{W}(\cdot)$, $\mathbf{U}(\cdot)$, and $\mathbf{b}(\cdot)$ are trainable parameters. The gating structure enables the network to selectively retain, update, or discard information over time, thereby stabilizing learning across long sequences. LSTM layers are primarily applied in scenarios where the temporal ordering of data carries critical information. Typical applications include time-series prediction (as in the case of channel prediction in Chapter 4), sequence modeling, and state tracking problems.

B.2.2 Training process

The effectiveness of deep neural networks critically depends on the training and optimization process, through which the parameters of the model are adjusted so as to achieve the desired input–output behavior. Training can be broadly interpreted as the solution of a high-dimensional optimization problem, where the objective is to minimize a loss function that quantifies the discrepancy between the network predictions and the target outputs. This process is inherently data-driven and forms the bridge between the abstract network architecture and its practical performance on a given task.

Let a neural network be represented as a parameterized function $f(\mathbf{x}; \boldsymbol{\theta})$, where $\mathbf{x}$ denotes the input, $\boldsymbol{\theta}$ collects all trainable parameters (*e.g.*, weights and biases across all layers), and the output $f(\mathbf{x}; \boldsymbol{\theta})$ represents the network prediction. Given a training dataset $\mathcal{D}$ consisting of N input-output pairs $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}), i = 1, \ldots, N$, the training objective is typically formulated as:

$$\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \ell\left(f(\mathbf{x}^{(i)}; \boldsymbol{\theta}), \mathbf{y}^{(i)}\right), \tag{B.10}$$

where $\ell(\cdot, \cdot)$ is a task-dependent loss function. Common choices include mean squared error for regression problems (as for channel prediction in Chapter 4) and cross-entropy-based losses for classification tasks (as for SCMA demodulation in Chapter 2). Although

the explicit form of the loss varies across applications, its role is always to provide a scalar measure of performance that guides the optimization of the network parameters.

Training proceeds iteratively by computing gradients of the loss function with respect to the parameters and updating those parameters in a direction that reduces the loss. This is typically accomplished using variants of gradient-based optimization algorithms. The key enabler of this process is the backpropagation algorithm, which efficiently computes the required gradients by exploiting the compositional structure of the network.

Backpropagation relies on the chain rule of calculus to propagate error signals from the network output backward through successive layers. Consider a network composed of L layers, where the l-th layer implements a transformation:

$$\mathbf{h}^{(l)} = \phi^{(l)}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right), \tag{B.11}$$

with $\mathbf{h}^{(0)} = \mathbf{x}$. The gradient of the loss with respect to the parameters of a given layer depends on both the local derivatives of that layer and the gradients propagated from higher layers. By recursively applying the chain rule, backpropagation computes these gradients with a computational cost that scales linearly with the number of parameters, making training of deep architectures tractable.

Once gradients are available, parameter updates are performed according to a chosen optimization rule. The simplest and most widely used approach is gradient descent, where parameters are updated as:

$$\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}), \tag{B.12}$$

with $\eta > 0$ denoting the learning rate. In practice, the loss and gradients are often estimated using subsets of the training data, leading to stochastic or mini-batch gradient descent methods that trade off computational efficiency and gradient variance. Numerous refinements of gradient descent exist, incorporating ideas such as adaptive learning rates, momentum, and normalization of updates, all aimed at improving convergence speed and stability. Among these methods, the Adam algorithm is widely adopted in practice, as it combines momentum-based updates with adaptive per-parameter learning rates computed from first- and second-order moment estimates of the gradients, offering robust performance across a broad range of deep learning applications [93].

The optimization landscape associated with deep neural networks is typically nonconvex, characterized by a large number of local minima, saddle points, and flat regions. As a result, training outcomes can depend on factors such as parameter initialization, learning rate schedules, and data ordering. Despite this apparent complexity, empirical evidence has shown that gradient-based methods are remarkably effective in finding parameter configurations that generalize well to unseen data, even in highly overparameterized regimes.

An important aspect of training is the ability of the learned model to generalize beyond the training dataset. Overfitting occurs when a network captures noise or spurious correlations specific to the training data, resulting in degraded performance on new inputs. To mitigate this effect, various regularization strategies are commonly employed, including explicit penalties on parameter norms, constraints on model capacity, and stochastic mechanisms that introduce robustness during training. Although these techniques differ in implementation, their shared goal is to encourage solutions that balance fitting accuracy with model simplicity.

From a system-level perspective, training can be performed offline, online, or in a hybrid manner, depending on the application constraints. Offline training leverages large datasets and substantial computational resources to learn a model that is later deployed for inference. Online or adaptive training, on the other hand, updates the model incrementally as new data become available, allowing the network to track time-varying environments. This distinction is particularly relevant in communication systems, where channel conditions, interference patterns, and network configurations may evolve over time.

Bibliography

[1] ITU-R, "Recommendation M.2160: Framework and overall objectives of the future development of IMT for 2030 and beyond," 2023.

[2] E. Hossain and A. Vera-Rivera, "6G Cellular Networks: Mapping the Landscape for the IMT-2030 Framework," *IEEE Transactions on Technology and Society*, pp. 1–16, 2025.

[3] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani, and P. Pérez, "Deep Reinforcement Learning for Autonomous Driving: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2022.

[4] Z. Li, J. Zhang, T. Tan, X. Teng, X. Sun, H. Zhao, L. Liu, Y. Xiao, B. Lee, Y. Li, Q. Zhang, S. Sun, Y. Zheng, J. Yan, N. Li, Y. Hong, J. Ko, H. Jung, Y. Liu, Y.-c. Chen, C.-w. Wang, V. Yurovskiy, P. Maevskikh, V. Khanagha, Y. Jiang, L. Yu, Z. Liu, D. Li, P. J. Schüffler, Q. Yu, H. Chen, Y. Tang, and G. Litjens, "Deep Learning Methods for Lung Cancer Segmentation in Whole-Slide Histopathology Images—The ACDC@LungHP Challenge 2019," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 2, pp. 429–440, 2021.

[5] 3GPP, "TR 37.817 - Study on enhancement for data collection for NR and EN-DC," 2022.

[6] ——, "TS 28.105 - Management and orchestration; Artificial Intelligence/ Machine Learning (AI/ML) management," 2025.

[7] ——, "TR 38.843 - Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR air interface," 2024.

[8] X. Lin, "An Overview of AI in 3GPP's RAN Release 18: Enhancing Next-Generation Connectivity?" *Global Communications*, 2024.

[9] 3GPP, "R1-2506595 - Feature Lead summary #2 on 6G waveform," 2025.

[10] M. Honkala, D. Korpi, and J. M. J. Huttunen, "DeepRx: Fully Convolutional Deep Learning Receiver," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3925–3940, 2021.

[11] D. Korpi, M. Honkala, J. M. J. Huttunen, and V. Starck, "DeepRx MIMO: Convolutional MIMO Detection with Learned Multiplicative Transformations," in *ICC 2021 - 2021 IEEE International Conference on Communications (ICC)*, 2021, pp. 1–6.

[12] S. Cammerer, F. A. Aoudia, J. Hoydis, A. Oeldemann, A. Roessler, T. Mayer, and A. Keller, "A Neural Receiver for 5G NR Multi-User MIMO," in *2023 IEEE Globecom Workshops (GC Wkshps)*, 2023, pp. 329–334.

[13] R. Wiesmayr, S. Cammerer, F. A. Aoudia, J. Hoydis, J. Zakrzewski, and A. Keller, "Design of a Standard-Compliant Real-Time Neural Receiver for 5G NR," in *2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2025, pp. 1–6.

[14] F. A. Aoudia and J. Hoydis, "End-to-End Learning for OFDM: From Neural Receivers to Pilotless Communication," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1049–1063, 2021.

[15] S. Cammerer, J. Hoydis, F. Ait Aoudia, and A. Keller, "Graph Neural Networks for Channel Decoding," in *2022 IEEE Globecom Workshops (GC Wkshps)*, 2022, pp. 1–6.

[16] Y. Choukroun and L. Wolf, "Error Correction Code Transformer," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022 (NeurIPS 2022)*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., 2022.

[17] N. D. Trac and S. Kim, "U-Shaped Error Correction Code Transformers," *IEEE Trans. Cogn. Commun. Netw.*, vol. 11, no. 3, pp. 1466–1481, 2025.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, ser. Lecture Notes in Computer Science, N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi, Eds., vol. 9351. Springer, 2015, pp. 234–241.

[19] S. Park, H. Kwak, S. Kim, S. Kim, Y. Kim, and J. No, "Multiple-Masks Error Correction Code Transformer for Short Block Codes," *IEEE J. Sel. Areas Commun.*, vol. 43, no. 7, pp. 2518–2529, 2025.

[20] C. Han, H. Zhao, Z. Chen, and F. Wang, "Sparse Neural Network for Detection and Decoding of Non-Binary Polar-Coded SCMA," *IEEE Transactions on Wireless Communications*, vol. 22, no. 7, pp. 4475–4488, 2023.

[21] H. Cheng, Y. Xia, Y. Huang, Z. Lu, and L. Yang, "Deep Neural Network Aided Low-Complexity MPA Receivers for Uplink SCMA Systems," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9050–9062, 2021.

[22] M. Kim, N.-I. Kim, W. Lee, and D.-H. Cho, "Deep Learning-Aided SCMA," *IEEE Communications Letters*, vol. 22, no. 4, pp. 720–723, 2018.

[23] L. Miuccio, D. Panno, and S. Riolo, "A Flexible Encoding/Decoding Procedure for 6G SCMA Wireless Networks via Adversarial Machine Learning Techniques," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, pp. 3288–3303, 2023.

[24] F. Jiang, D.-W. Chang, S. Ma, Y.-J. Hu, and Y.-H. Xu, "A Residual Learning-Aided Convolutional Autoencoder for SCMA," *IEEE Communications Letters*, vol. 27, no. 5, pp. 1337–1341, 2023.

[25] M. Sun, Q. Zhang, H. Yao, R. Gao, Y. Zhao, and M. Guizani, "Resource Allocation and Deep Learning-Based Joint Detection Scheme in Satellite NOMA Systems," *IEEE Trans. Wirel. Commun.*, vol. 24, no. 1, pp. 526–539, 2025.

[26] J. Yang and A. Mohajer, "Multi objective constellation optimization and dynamic link utilization for sustainable information delivery using PD-NOMA deep reinforcement learning," *Wirel. Networks*, vol. 31, no. 2, pp. 1839–1859, 2025.

[27] J. E. Mazo, "Faster-than-nyquist signaling," *The Bell System Technical Journal*, vol. 54, no. 8, pp. 1451–1462, 1975.

[28] S. Li, B. Bai, J. Zhou, P. Chen, and Z. Yu, "Reduced-Complexity Equalization for Faster-Than-Nyquist Signaling: New Methods Based on Ungerboeck Observation Model," *IEEE Transactions on Communications*, vol. 66, no. 3, pp. 1190–1204, 2018.

[29] S. Sugiura, "Frequency-Domain Equalization of Faster-than-Nyquist Signaling," *IEEE Wireless Communications Letters*, vol. 2, no. 5, pp. 555–558, 2013.

[30] A. Ibrahim, E. Bedeer, and H. Yanikomeroglu, "A Novel Low Complexity Faster-than-Nyquist (FTN) Signaling Detector for Ultra High-Order QAM," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 2566–2580, 2021.

[31] P. Song, F. Gong, Q. Li, G. Li, and H. Ding, "Receiver Design for Faster-Than-Nyquist Signaling: Deep-Learning-Based Architectures," *IEEE Access*, vol. 8, pp. 68 866–68 873, 2020.

[32] S. Lai and M. Li, "Recurrent Neural Network Assisted Equalization for FTN Signaling," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[33] S. Paul, N. Seshadri, and R. D. Koilpillai, "Deep-Learning based Equalization of Highly Compressed Faster Than Nyquist Signals," in *2023 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2023, pp. 396–401.

[34] A. Vanelli-Coralli, N. Chuberre, G. Masini, A. Guidotti, and M. El Jaafari, *5G Non-Terrestrial Networks: Technologies, Standards, and System Design.* John Wiley & Sons, Inc., 2024.

[35] M. A. Jamshed, A. Kaushik, M. Dajer, A. Guidotti, F. Parzysz, E. Lagunas, M. Di Renzo, S. Chatzinotas, and O. A. Dobre, "Non-Terrestrial Networks for 6G: Integrated, Intelligent, and Ubiquitous Connectivity," *IEEE Communications Standards Magazine*, vol. 9, no. 3, pp. 86–93, 2025.

[36] 3GPP, "TR 38.811 - Study on New Radio (NR) to support non-terrestrial networks (Release 15)," 2020.

[37] B. Al Homssi, K. Dakic, K. Wang, T. Alpcan, B. Allen, R. Boyce, S. Kandeepan, A. Al-Hourani, and W. Saad, "Artificial Intelligence Techniques for Next-Generation Massive Satellite Networks," *IEEE Communications Magazine*, vol. 62, no. 4, pp. 66–72, 2024.

[38] Y. Zhang, Y. Wu, A. Liu, X. Xia, T. Pan, and X. Liu, "Deep Learning-Based Channel Prediction for LEO Satellite Massive MIMO Communication System," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1835–1839, 2021.

[39] D. Yan, M. Jia, Q. Guo, and D. Niyato, "Temporal-Frequency Domain Channel Prediction for LEO Satellite Communication System: A Novel TFformer Structure," *IEEE Transactions on Wireless Communications*, vol. 24, no. 10, pp. 8208–8220, 2025.

[40] Y. Zhang, A. Liu, P. Li, and S. Jiang, "Deep Learning (DL)-Based Channel Prediction and Hybrid Beamforming for LEO Satellite Massive MIMO System," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23 705–23 715, 2022.

[41] G.-Y. Chang, C.-K. Hung, and C.-H. Chen, "A CSI Prediction Scheme for Satellite-Terrestrial Networks," *IEEE Internet of Things Journal*, vol. 10, no. 9, pp. 7774–7785, 2023.

[42] M. Ying, X. Chen, Q. Qi, and W. H. Gerstacker, "Deep Learning-Based Joint Channel Prediction and Multibeam Precoding for LEO Satellite Internet of Things," *IEEE Trans. Wirel. Commun.*, vol. 23, no. 10, pp. 13 946–13 960, 2024.

[43] M. Ying and X. Chen, "Channel Prediction-based Robust Multibeam Precoding Design for LEO Satellite Internet of Things," in *2023 IEEE/CIC International Conference on Communications in China (ICCC)*, 2023, pp. 1–6.

[44] S. Kim, J. Park, and C. Lee, "CNN-based Doppler Shift Estimation for Low Earth Orbit Satellites," in *2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, 2022, pp. 1–3.

[45] J. Gáñez-Fernández, J. M. Escalante, and J. A. López-Salcedo, "Deep Learning Coarse Doppler Estimation for LEO Signals of Opportunity," in *2024 11th Workshop on Satellite Navigation Technology (NAVITEC)*, 2024, pp. 1–4.

[46] A. Siriwanitpong, K. Sanada, H. Hatano, K. Mori, and P. Boonsrimuang, "Deep Learning-Based Channel Estimation With 1D CNN for OFDM Systems Under High-Speed Railway Environments," *IEEE Access*, vol. 13, pp. 13 128–13 142, 2025.

[47] X. Ding, T. Ni, Y. Zou, and G. Zhang, "Deep Learning for Satellites Based Spectrum Sensing Systems: A Low Computational Complexity Perspective," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 1, pp. 1366–1371, 2023.

[48] Z. Ren, J. Jin, W. Li, and Y. Zhan, "Intelligent Action Selection for NGSO Networks With Interference Constraints: A Modified Q-Learning Approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 3, pp. 2231–2242, 2022.

[49] A. Saifaldawla, F. G. Ortiz-Gomez, E. Lagunas, S. Daoud, and S. Chatzinotas, "NGSO-To-GSO Satellite Interference Detection Based on Autoencoder," in *2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2023, pp. 1–7.

[50] A. Saifaldawla, F. Ortiz, E. Lagunas, A. B. M. Adam, and S. Chatzinotas, "GenAI-Based Models for NGSO Satellites Interference Detection," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 2, pp. 904–924, 2024.

[51] X. Sui, Z. Jiang, Y. Lyu, R. Fan, H. Hu, and Z. Liu, "Integrating Convex Optimization and Deep Learning for Downlink Resource Allocation in LEO Satellites Networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 10, no. 3, pp. 1104–1118, 2024.

[52] L. Liang, P. Duan, G. Cui, and W. Wang, "Multiagent DRL for Two-Timescale Bandwidth Allocation in Multibeam Satellite Networks," *IEEE Internet of Things Journal*, vol. 12, no. 8, pp. 10 613–10 626, 2025.

[53] Q. T. Ngo, Y. He, B. Jayawickrama, and E. Dutkiewicz, "Multi-Agent DDPG-Based Joint Beam Hopping and Resource Allocation for Cognitive GEO-LEO Satellite Networks," *IEEE Networking Letters*, pp. 1–1, 2025.

[54] T. Lyu, Y. Xu, F. Liu, H. Xu, and Z. Han, "Task Offloading and Resource Allocation for Satellite-Terrestrial Integrated Networks," *IEEE Internet of Things Journal*, vol. 12, no. 1, pp. 262–275, 2025.

[55] B. De Filippo, C. Amatetti, R. Campana, A. Guidotti, and A. Vanelli-Coralli, "An SCMA Receiver for 6G NTN based on Multi-Task Learning," in *2024 IEEE Globecom Workshops (GC Wkshps)*, 2024, pp. 1–6.

[56] A. Guidotti, A. Vanelli-Coralli, M. El Jaafari, N. Chuberre, J. Puttonen, V. Schena, G. Rinelli, and S. Cioni, "Role and Evolution of Non-Terrestrial Networks Toward 6G Systems," *IEEE Access*, vol. 12, pp. 55 945–55 963, 2024.

[57] Y. Jing, J. Wang, C. Jiang, and Y. Zhan, "Satellite MEC with Federated Learning: Architectures, Technologies and Challenges," *IEEE Network*, vol. 36, no. 5, pp. 106–112, 2022.

[58] J. Zhang, Q. He, Z. Yu, B. Bai, and M. Zhu, "Low-Complexity Coherent Iterative Receiver for SCMA-Based LEO Satellite Communications," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[59] Y. Hao, Z. Song, Z. Zheng, Q. Zhang, and Z. Miao, "Joint Communication, Computing, and Caching Resource Allocation in LEO Satellite MEC Networks," *IEEE Access*, vol. 11, pp. 6708–6716, 2023.

[60] 3GPP, "TR 38.812 - Study on Non-Orthogonal Multiple Access (NOMA) for NR," 2016.

[61] A. T. Abebe and C. G. Kang, "Enabling Grant-Free Access with Non-orthogonal Preambles and Multi-codebook SCMA," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, 2019, pp. 883–888.

[62] H. Nikopour and H. Baligh, "Sparse code multiple access," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2013, pp. 332–336.

[63] J. Liu, G. Wu, S. Li, and O. Tirkkonen, "On Fixed-Point Implementation of Log-MPA for SCMA Signals," *IEEE Wireless Communications Letters*, vol. 5, no. 3, pp. 324–327, 2016.

[64] InnovateAsia, "1st 5G Algorithm Innovation Competition - SCMA," available at: http://www.innovateasia.com/5g/en/gp2.html.

[65] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[66] J. Guo, J. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Compression and Acceleration of Neural Networks for Communications," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 110–117, 2020.

[67] L. Lamberti, L. Bellone, L. Macan, E. Natalizio, F. Conti, D. Palossi, and L. Benini, "Distilling Tiny and Ultrafast Deep Neural Networks for Autonomous Navigation on Nano-UAVs," *IEEE Internet of Things Journal*, vol. 11, no. 20, pp. 33 269–33 281, 2024.

[68] Q. Luo, J. Zhu, Q. Peng, G. Chen, Y. Xu, P. Xiao, and H. W. Kim, "Advanced Nonlinear SCMA Codebook Design Based on Lattice Constellation," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2026.

[69] B. Fontana da Silva, D. Silva, B. F. Uchôa-Filho, and D. Le Ruyet, "A Multistage Method for SCMA Codebook Design Based on MDS Codes," *IEEE Wireless Communications Letters*, vol. 8, no. 6, pp. 1524–1527, 2019.

[70] J. Yang, Y. Zeng, W. Wan, and X. Ye, "Design of Uplink SCMA Codebooks Using Evolution Strategies," *IEEE Communications Letters*, vol. 28, no. 7, pp. 1673–1677, 2024.

[71] Q. Wang, T. Li, R. Feng, and C. Yang, "An Efficient Large Resource-User Scale SCMA Codebook Design Method," *IEEE Communications Letters*, vol. 23, no. 10, pp. 1787–1790, 2019.

[72] L. Zhao, C. Wang, M. Pang, W. Wang, W. Wang, F. Jiang, and L. Xu, "A Novel Spherical Codebook Design for Uplink SCMA in Satellite Communications," in *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, 2024, pp. 1–5.

[73] B. De Filippo, R. Campana, A. Guidotti, C. Amatetti, and A. Vanelli-Coralli, "Cell-Free MIMO in 6G NTN with AI-predicted CSI," in *2024 IEEE 25th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2024, pp. 631–635.

[74] F. Riera-Palou, G. Femenias, M. Caus, M. Shaat, and A. I. Pérez-Neira, "Scalable Cell-Free Massive MIMO Networks With LEO Satellite Support," *IEEE Access*, vol. 10, pp. 37 557–37 571, 2022.

[75] ITU-R, "Recommendation M.2101: Modelling and simulation of IMT networks and systems for use in sharing and compatibility studies," 2017.

[76] H. Kasasbeh, R. Viswanathan, and L. Cao, "Noise Correlation Effect on Detection: Signals in Equicorrelated or Autoregressive(1) Gaussian," *IEEE Signal Processing Letters*, vol. 24, no. 7, pp. 1078–1082, 2017.

[77] B. Ahmad, D. G. Riviello, B. De Filippo, A. Guidotti, and A. Vanelli-Coralli, "Analysis of graph-based user scheduling for ka-band leo ntn systems," in *28th Ka and Broadband Communications Conference (Ka) and the 40th International Communications Satellite Systems Conference (ICSSC)*, 2023.

[78] A. Guidotti, C. Amatetti, F. Arnal, B. Chamaillard, and A. Vanelli-Coralli, "Location-assisted precoding in 5G LEO systems: architectures and performances," in *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2022, pp. 154–159.

[79] 3GPP, "TR 38.821 - Solutions for NR to support non-terrestrial networks (NTN)," 2021.

[80] A. Guidotti, A. Vanelli-Coralli, and C. Amatetti, "Federated Cell-Free MIMO in Nonterrestrial Networks: Architectures and Performance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 60, no. 3, pp. 3319–3347, 2024.

[81] B. De Filippo, C. Amatetti, and A. Vanelli-Coralli, "Channel Prediction in 6G Non-Terrestrial Networks With Deep Learning: a Physical Layer Analysis," in *2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2025, pp. 1–6.

[82] B. De Filippo, C. Amatetti, and A. Vanelli-Coralli, "Uplink OFDM Channel Prediction with Hybrid CNN-LSTM for 6G Non-Terrestrial Networks," in *2025 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2025, pp. 7–12.

[83] B. De Filippo, C. Amatetti, and A. Vanelli-Coralli, "Low-complexity Deep Learning techniques for Channel Prediction in 6G Non-Terrestrial Networks," *EURASIP Journal on Wireless Communications and Networking*, 2026, Submitted for publication.

[84] B. De Filippo, C. Amatetti, A. Guidotti, and A. Vanelli-Coralli, "Deep learning techniques for channel prediction in non-terrestrial networks," in *Artificial Intelligence for Integrated Terrestrial and Non-Terrestrial Networks*, M. A. Jamshed, Ed. Springer Nature Switzerland AG, 2026.

[85] 6G-NTN, "D4.4 - Report on unified and data driven air-interface for 6G-NTN," 2025.

[86] O. Abbasi and G. Kaddoum, "Channel Aging-Aware LSTM-Based Channel Prediction for Satellite Communications," *IEEE Networking Letters*, vol. 6, no. 3, pp. 183–187, 2024.

[87] C. Cui, W. Jing, Z. Lu, and X. Wen, "Attention Aided Channel Prediction Scheme For Satellite-Terrestrial Networks," in *2024 IEEE 35th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2024, pp. 1–5.

[88] M. Ying, X. Chen, Q. Qi, and W. Gerstacker, "Deep Learning-Based Joint Channel Prediction and Multibeam Precoding for LEO Satellite Internet of Things," *IEEE Transactions on Wireless Communications*, vol. 23, no. 10, pp. 13 946–13 960, 2024.

[89] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Cambridge University Press, 2024.

[90] J.-J. van de Beek, O. Edfors, M. Sandell, S. Wilson, and P. Borjesson, "On channel estimation in OFDM systems," in *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, vol. 2, 1995, pp. 815–819 vol.2.

[91] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," 2018, preprint at https://arxiv.org/abs/1803.01271.

[92] J. Hoydis, S. Cammerer, F. Ait Aoudia, M. Nimier-David, L. Maggi, G. Marcus, A. Vem, and A. Keller, "Sionna," 2022, https://nvlabs.github.io/sionna/. Accessed 15 September 2025.

[93] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[94] D. S. Kalra and M. Barkeshli, "Why Warmup the Learning Rate? Underlying Mechanisms and Improvements," in *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, Eds., 2024.

[95] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 464–472.

[96] P. Freire, S. Srivallapanondh, A. Napoli, J. E. Prilepsky, and S. K. Turitsyn, "Computational Complexity Evaluation of Neural Network Applications in Signal Processing," 2024, preprint at https://arxiv.org/abs/2206.12191v2.

[97] C. Amatetti, B. De Filippo, and A. Vanelli-Coralli, "Channel Prediction with Temporal Convolutional Networks: A New Paradigm to Enable Non-orthogonal Multiple Access in 6G NTN," *IEEE Communication Standards Magazine.*, 2026, Accepted for publication.

[98] Y. Yuan, Z. Yuan, and L. Tian, "5G Non-Orthogonal Multiple Access Study in 3GPP," *IEEE Communications Magazine*, vol. 58, no. 7, pp. 90–96, 2020.

[99] B. De Filippo, C. Amatetti, and A. Vanelli-Coralli, "Faster-than-Nyquist Equalization with Convolutional Neural Networks," in *2025 IEEE 36th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2025, pp. 1–6.

[100] J. B. Anderson, A. Prlja, and F. Rusek, "New reduced state space BCJR algorithms for the ISI channel," in *2009 IEEE International Symposium on Information Theory*, 2009, pp. 889–893.

[101] A. Prlja and J. B. Anderson, "Reduced-Complexity Receivers for Strongly Narrowband Intersymbol Interference Introduced by Faster-than-Nyquist Signaling," *IEEE Transactions on Communications*, vol. 60, no. 9, pp. 2591–2601, 2012.

[102] E. Bedeer, M. H. Ahmed, and H. Yanikomeroglu, "Low-Complexity Detection of High-Order QAM Faster-Than-Nyquist Signaling," *IEEE Access*, vol. 5, pp. 14 579–14 588, 2017.

[103] W. Quan, R. Zhang, Y. Zhang, Z. Li, J. Wang, and D.-M. Yan, "Image Inpainting With Local and Global Refinement," *IEEE Transactions on Image Processing*, vol. 31, pp. 2405–2420, 2022.

[104] B. De Filippo, C. Amatetti, and A. Vanelli-Coralli, "DeepFTN," 2024. [Online]. Available: https://github.com/Defil1998/DeepFTN/

[105] CVX Research, Inc, "CVX: Matlab software for disciplined convex programming," 2011. [Online]. Available: https://cvxr.com/cvx

[106] S. Benedetto and E. Biglieri, *Principles of digital transmission: with wireless applications*. Springer, 1999.

[107] K. Wang, A. Liu, X. Liang, S. Peng, and Q. Zhang, "A Faster-Than-Nyquist (FTN)-Based Multicarrier System," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 947–951, 2019.

[108] 5G-STARDUST, "D4.7 - Final report on AI-based radio resource management, RAN softwarisation and onboard processing," 2025.

[109] B. De Filippo, A. Guidotti, and A. Vanelli-Coralli, "Attention-based SINR estimation in User-centric NTNs," in *2026 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2026, Accepted for publication.

[110] Y. I. Demir, M. S. J. Solaija, and H. Arslan, "On the Performance of Handover Mechanisms for Non-Terrestrial Networks," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, 2022, pp. 1–5.

[111] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[112] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[113] 5G-STARDUST, "D4.6 - Final report on the unified air interface, user-centric and beamforming solutions," 2025.

[114] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[115] S. J. Prince, *Understanding Deep Learning*. MIT Press, 2023. [Online]. Available: http://udlbook.com

[116] D. A. Roberts, S. Yaida, and B. Hanin, *The Principles of Deep Learning Theory*. Cambridge University Press, 2022. [Online]. Available: https://deeplearningtheory.com

[117] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.