Aalto University
School of Electrical
Engineering

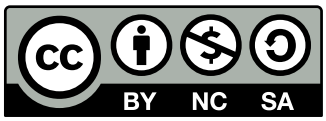
Master's programme in Computer, Communication and Information Sciences

Machine learning algorithms for 5G PUCCH Channel estimation

Yifan Wu

Master's Thesis

2023



Aalto University
School of Electrical
Engineering

Author Yifan Wu

Title Machine learning algorithms for 5G PUCCH Channel estimation

Degree programme Computer, Communication and Information Sciences

Major Communications Engineering

Supervisor Prof. Jyri Hämäläinen

Advisors Dr Alexis Dowhuszko, Dr Toufik Sadi

Collaborative partner Nokia Solutions and Networks Oy

Date 12.12.2023 **Number of pages** 64 **Language** English

Abstract

The arrival of 5G technology marks a new era in the field of mobile communications, which not only greatly improves data transmission speeds and network responsiveness, but also opens up a rich palette for emerging applications such as the Internet of Things, industrial automation, autonomous driving, and augmented reality. As these advanced applications continue to emerge, 5G networks face unprecedented technical challenges, especially in terms of more stringent requirements for maintaining efficient and stable mobile connectivity. Among these techniques, channel estimation is one of the most relevant procedures that should be continuously updated using novel approaches beyond current state-of-the-art solutions. Effective channel estimation algorithms can accurately capture and analyze the propagation characteristics of wireless signals in complex environments, which is crucial for the optimization of 5G networks to give an effective solution to the different kinds of services that have been identified. Traditional channel estimation algorithms based on Least Square optimization methods do not always provide sufficient performance when facing high-speed, dynamic and complex transmission environments in the context of 5G, especially when the wireless channel conditions are poor. In this context, the introduction of Convolutional Neural Networks (CNN) has been lately considered as a novel approach to revolutionize the way in which channel estimation is implemented.

Based on this, three CNN-based channel estimation algorithms have been studied in detail in this MSc thesis, with the aim to improve the channel estimation performance of PUCCH channels. For this purpose, a PUCCH link-level simulator based on the Matlab 5G Toolbox has been first built and its accuracy has been verified by comparing the obtained performance results in the form or BLER curves with the ones reported in reliable sources found in the literature. Based on this link simulator, channel estimation has been performed using three CNN based algorithms, namely "SRIR", "SR", "LI+SR". By testing the developed CNN-based algorithms for channel estimation under different working conditions, it can be concluded that the use of CNN for channel estimation is not recommended for Line-of-Sight channels, as its performance gain is not sufficient to justify the additional complexity required to implement CNN in 5G networks. On the contrary, for Non-Line-of-Sight channels, especially in channel conditions with high Doppler shift and high delay spread for high mobility users in Urban Macro Cell scenarios, the CNN-based channel estimation

algorithm is able to achieve a performance gain of 2-3 dB.

Keywords 5G, Physical Uplink Control Channel, Channel Estimation, Maching Learning, Demodulation References Signals, Convolutional Neural Network

Preface

First of all, I would like to give a special thanks to my advisor Alexis Dowhuszko. During the months of writing my thesis, we had more than 20 meetings and Alexis constantly provided me with guidance, and it was his help that enabled me to finalize my thesis. Secondly, I would like to thank my supervisor Prof. Jyri Hämäläinen for his willingness to guide me and for his valuable revisions.

At Nokia, first of all I would like to thank my advisor Toufik Sadi. Toufik has helped me in every way since my internship. He helped me finalize the topic of my thesis, and provided various technical or non-technical support as I worked on it. Also, thank you to my manager Farah Salah, and to everyone in the team, your help was like a guiding light to me, without you the thesis process would be painful.

Finally, I would like to thank my family for their support during my studies.

Lund, 21 November, 2023

Yifan Wu

Contents

Symbols and abbreviations

Abbreviations

3GPP	3rd Generation Partnership Project
5G	5th generation mobile communication
AI	Artificial Intelligence
BLER	Block error rate
CDL	Clustered Delay Line
CDMA	Code Division Multiple Access
CNN	Convolutional Neural Network
CP	Cyclic Prefix
CSI	channel state information
CSI-RS	Channel State Information Reference Signals
DL-SCH	downlink data channel
DNN	deep neural network
DMRS	Demodulation References Signals
eMMB	enhanced mobile broadband
EPC	Evolved Packet Core
FDD	Frequency Division Duplex
FDMA	Frequency Division Multiple Access
IDFT	inverse discrete Fourier transform
IoT	Internet of Things
IR	Image Restoration
ISI	Inter-Symbol Interference
LS	Least Squares
MIMO	Multiple-Input and Multiple-Output
MLD	Maximum Likelihood Detection
mMTC	massive machine-type communications
MMSE	Minimum Mean-Square Error
MSE	Mean Square Error

NFV	Network Function Virtualisation
NSA	Non-StandAlone
OSI	Open System Interconnection
PAPR	peak-to-average power ratio
PBCH	physical broadcast channel
PDCCH	physical downlink control channel
PDSCH	physical downlink shared channel
PRACH	physical random access channel
PTRS	Phase-tracking Reference Signals
PUCCH	physical uplink control channel
PUSCH	physical uplink shared channel
QoS	quality of service
ReLU	Rectified Linear Unit
SA	StandAlone
SC-FDMA	Single Carrier Frequency Division Multiplexing
SDMA	Space Division Multiple Access
SDN	Software Defined Networking
SNR	signal-to-noise ratio
SR	Scheduling Request
SR	Super Resolution
SRS	Sounding Reference Signals
Tanh	Hyperbolic tangent function
TDD	Time Division Duplex
TDMA	Time Division Multiple Access
TDL	Tapped Delay Line
UCI	uplink control information
UE	user equipment
URLLC	ultra-reliable low-latency communications
V-BLAST	Vertical Bell Laboratories Layered Space-Time
ZF	Zero Forcing

1 Introduction

In modern society, mobile communication has undoubtedly become an indispensable part of our productive life. It not only provides a tool for instant communication between people, but also plays a vital role in the development of business, socializing, entertainment, and other areas of our life. The 5th Generation (5G) mobile communication, has further increased the data transmission speed. In addition, 5G proposes three new major application scenarios: enhanced Mobile BroadBand (eMBB), Ultra-Reliable Low Latency Communications (URLLC), and massive Machine-Type Communications (mMTC). In different scenarios, 5G can enable communications that are not humancentric, such as autonomous driving, industrial automation, and smart cities. These applications will have a profound impact on society as a whole. In order to achieve these applications, a number of new technologies have been proposed, including the use of new spectrum range, large-scale multiple-input multiple-output technology, beamforming, as well as cloud and edge computing.

In mobile communication, signals are transmitted from the transmitter to the receiver through the air. Air is the medium or the channel which acts as the pathway for signals. However, the vast number of uncertainties makes it difficult to ensure a good enough channel gain to guarantee the target Key Performance Indicators (KPIs). For example, in open environments, signals from the transmitter often have multiple paths (e.g., direct, refracted, reflected) to the receiver, commonly referred to as the multipath effect. The signals from different paths overlap with each other at the receiving end, which changes the amplitude as well as the phase of the signals, resulting the fading of the the received signal power in frequency and time. In addition to the multipath effect, signal attenuation can be also caused by other phenomena, such as cloudy or rainy weather and obstacle blockage (especially, in millimeter wave bands and beyond). The channel in which the signal is attenuated is known as the fading channel. In a fading channel, the channel state changes continually. To eliminate this effect and decode accurately, the Channel State Information (CSI) needs to be estimated in a very short period of time, which is a process known as channel estimation. To enable the Ultra-low latency applications supported by 5G will therefore require more rapid and accurate channel estimation algorithms.

Currently, the use of pilot sequences is a popular technique for channel estimation. At the transmitting end, a set of known symbols is inserted into the transmitted message. Then, at the receiving end, combined knowledge of the transmitted and received signals is used to estimate the channel matrix. Since only the 5G transmitter-side processing flow has been defined in detail by the 3rd Generation Partnership Project (3GPP), and receiver-side channel estimation operation is not part of these standardized processes, this has allowed using the Reed Muller coding or Polar coding algorithms, different number of receiver antennas as well as MMSE and LS channel estimation algorithms. to design its own algorithms for achieving higher performance and thus take over the market. The estimation process usually uses Least Squares (LS) or Minimum Mean Square Error (MMSE) algorithms, which have been continuously optimized over the years [1] and tested at the 5G receiver [2], [3]. In [2], the authors demonstrated the application of the MMSE algorithm for channel estimation and its Block Error Rate

(BLER) performance in different Physical Uplink Control Channel (PUCCH) formats. In [3], the BLER curves of PUCCH Format 2 for each of these cases were verified by using the Reed Muller coding or Polar coding algorithms, different number of receiver antennas as well as MMSE and LS channel estimation algorithms.

Although LS algorithms require no prior channel statistics and have good generalization, their performance may be inadequate, especially when the channel conditions are very poor (e.g., on a high-speed train) [4]. In recent years, with the development of Machine Learning (ML) and deep learning, these techniques have been gradually applied to mobile communications [4–11]. For example, in [4] a Deep Neural Network (DNN) model was developed that implicitly estimates channel state information. The results show that deep learning has great potential for channel estimation, while achieving better performance than traditional methods under poor conditions. However, the authors replaced the entire communication link with a deep learning module, making it difficult to clearly identify the gain of deep learning on the specific module of channel estimation. Moreover, their model cannot be applied for applications requiring an entire channel time-frequency response. In [10], a Convolutional Neural Network (CNN) based algorithm (ChannelNet) was developed for channel estimation by considering the channel matrix as an image as well as using a cascade of Super-Resolution (SR) and Image-Restoration (IR) networks for interpolation and noise reduction of the image, respectively. Although some performance gains have been achieved using the ChannelNet algorithm, its complex network structure consumes a large number of resources for training and implementation of the neural network.

Therefore, the aim of this thesis is to develop an algorithm based on ML for channel estimation of PUCCH Format 2 in order to improve the channel estimation performance while minimizing the complexity of the algorithm. For this purpose, the Matlab NR toolbox will first be used to reproduce the performance curves reported in other papers [2], [3]. By setting the simulation parameters as identical as possible to those in these papers, purpose is to identify a baseline performance curve for comparisons of new ML-based solutions with state-of-the-art ones based on statistical channel estimation.. The channel estimation algorithm will then be designed based on CNN. Similar to [10], the channel matrix containing the pilot sequences will be regarded as an image, which is interpolated and noise reduced by the CNN to obtain the whole image. In this way, the time-frequency response of the entire channel is used to verify the accuracy of the channel estimation algorithm. Unlike [10], the network structure of the CNN will be optimized and adapted to PUCCH Format 2, thus enabling the final application of the algorithm. Finally, the performance of the proposed algorithm is evaluated by comparing the BLER and Mean Square Error (MSE) of the new CNN-based channel estimation algorithm against those of the conventional LS algorithm. The CNN algorithm will be designed, trained, and simulated in Matlab.

The rest of this thesis is organised as follows. Chapter 2 introduces the basics of 5G. Chapter 3 focuses on Physical Uplink Control Channel. Chapter 4 introduces the theory related to ML, focusing on neural networks and CNN. Chapter 5 presents the experimental procedure, simulation results and performance comparison. Chapter 6 draws conclusions and makes recommendations for future work.

2 5G fundamentals

This chapter focuses on the background knowledge about the 5th generation of mobile communications (5G) that is relevant to the scope of this MSc thesis. Combined with the standards defined by 3GPP, the three major use cases of 5G, spectrum resources, physical channels, and frame structure will be discussed to show the characteristics of 5G and the key differences between 5G and 4G.

2.1 5G use cases

5G is a very important generation in the history of mobile communication. 1G was designed using only analogue technologies in order to provide very limited voice transmission services. In the 2G era, digital processing of signals replaced the analogue one, making phone calls and SMS possible. In the 3G era, with the help of CDMA technology, people were able for the first time to surf the mobile Internet anytime and anywhere. Then, with the arrival of 4G, wideband multimedia services were added to mobile communications, enriching lives of people with a variety of audio and video services. By now, mobile communication terminals such as mobile phones, tablets and laptops have become an indispensable part of our lives.

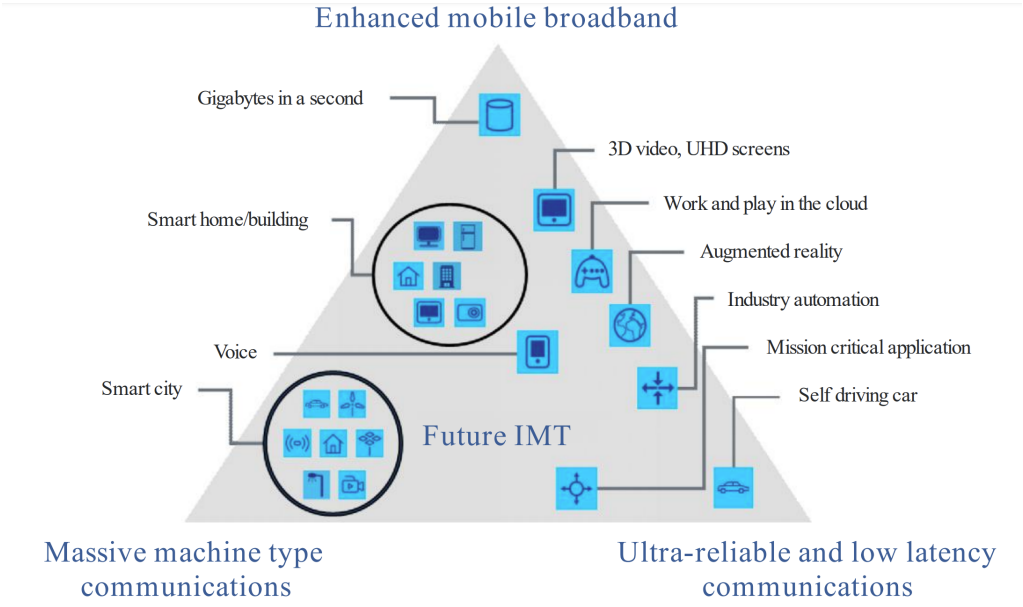


Figure 2.1: Usage scenarios For 5G based on the recommendation ITU-R M.2083 [12].

However, the focus of the 4G remained on human-centred communications, with limited help for the industrial sector. The arrival of 5G not only enabled to further optimise the Key Performance Indicators (KPIs) of concern for traditional mobile communications, such as data rate and capacity, but also opens up the possibility of new services, such as the Industrial Internet of Things (IoT). To this end, ITU-R M.2083 [12] sets the following three main use cases for 5G:

1. **Enhanced Mobile Broadband (eMBB):** This usage scenario remains focused on human-centred communications and further builds on the new requirements of beyond 4G in terms of improved performance and an increasing commitment to a seamless user experience [12]. This usage scenario will also cover a range of cases with individual requirements for different cases. For example, in user-intensive areas such as shopping centres, where capacity is often the key to enhancing the user experience and data rates need to be optimised, and the necessity to deal with the mobility of the users is not as important. In other wide-area coverage cases, mobility and seamless connectivity are the first priority, while data rate and capacity are less important. Optimising for different use cases, rather than using the same set of metrics, is the key point of enhanced mobile broadband.

2. **Ultra-reliable and low latency communications (URLLC):** Such application scenarios have very strict requirements on the parameters of throughput, latency, and availability [12]. In use cases such as remote surgery, autonomous driving, among others, both high latency and unreliable communication can create a notable impact on the end-user experience. And the technological innovation of 5G brings the possibility of application in these cases.

3. **Massive machine type communications (mMTC):** This kind of service mainly encompasses the Internet of Things (IoT). IoT deals with mobile devices, which are usually large in number and powered with battery, but the data that they transmit usually consists of small amounts of latency-insensitive data. In this case, the cost of the device needs to be kept as low as possible, while battery power determines the need to control the communication energy consumption. Typical examples are sensor networks for scenarios such as smart cities, environmental monitoring, smart homes and forest fire prevention.

We can see that different usage scenarios require very different network characteristics, and different networks do not need to implement all network characteristics. Therefore, modularity will be an important criterion in the design of future IMT-2020 (5G) systems. Once the key characteristics are given, the different networks can be designed to achieve specific performance metrics on demand.

Following are the key capabilities given by IMT-2020 [12], the following target KPIs are now summarized:

1. **Peak data rate**: Maximum data rate achievable per device under ideal conditions (measured in Gbps).

2. **User experienced data rate**: Available data rate for mobile devices everywhere in the coverage area (measured in Mbps or Gbps).

3. **Latency**: In mobile communication, the time that elapses between a signal being sent by a transmitter and being received by a receiver (measured in ms).

4. **Mobility**: Maximum speed at which the user terminal can move such that it can be supported by the seamless service (measured in km/h).

5. **Connection density**: Total number of connected devices per unit area (measured in devices per km2).

6. **Energy efficiency**: Number of information bits per unit of energy consumption (measured in bit/Joule).

7. **Spectrum efficiency**: Average data throughput per unit of spectrum resource (measured in bps/Hz).

8. **Area traffic capacity**: Total traffic throughput per geographical area (measured in Mbps/m2).

Figure 2.2 shows the importance of each key capability in different usage scenarios.

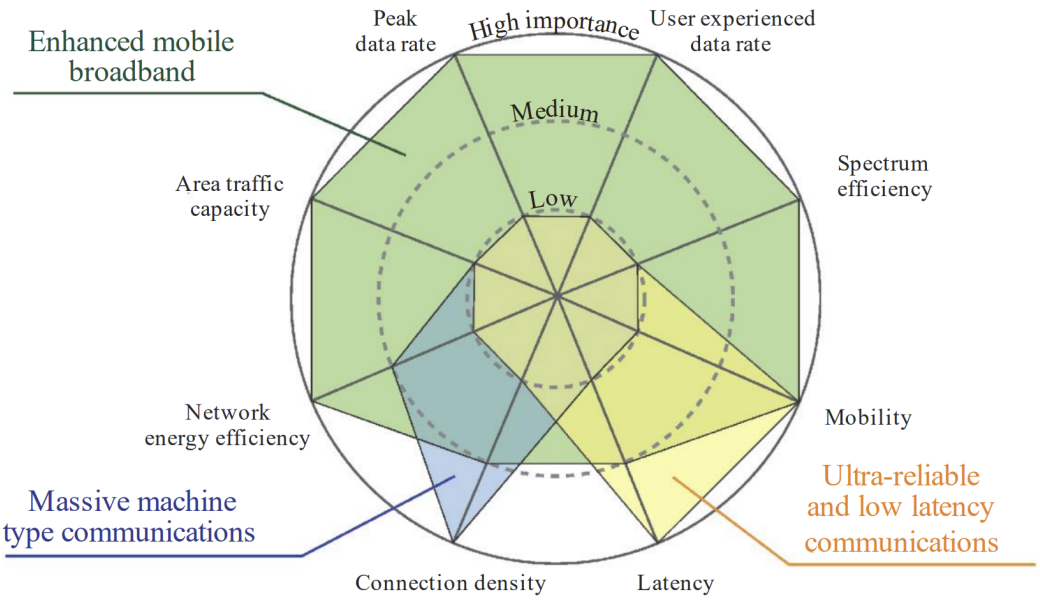


Figure 2.2: The importance of key capabilities in different usage scenarios [12].

2.2 Key Technology Components

In order to achieve the application scenarios listed in Section 2.1 and meet the desired performance metrics, 5G introduces a range of new technologies which include:

Spectrum Utilization: To address the problem of increasing shortage of spectrum resources, 5G introduces new spectrum. These new frequency bands, located mostly at very high frequencies (i.e., mid-band in 3.5GHz and high-band in millimeter wave range), can provide very wide channel bandwidths, which can dramatically improve

channel capacity as well as transmission rates. At the same time, 5G allows carriers to have smaller guard bands than 4G, resulting in improved spectrum utilisation.

Multi-antenna transceivers (MIMO and Beamforming): While the new spectrum solves the problem of insufficient resources, multi-antenna transceivers solves the problem of how to make the best use of the spectrum resources. This involves the issue of enabling space-division multiple access, identifying the way to enable that, multiple devices have access the mobile network at the same time. In practice, beamforming techniques send data streams in narrow beams to specific users, allowing different users to spatially multiplex spectrum resources. At the same time, the advantages of beamforming techniques are more easily achieved due to the use of massive Multiple-Input and Multiple-Output (MIMO) techniques in practice. The combination of the two technologies increases cell capacity while reducing the co-channel interference that is generated.

Flexible Physical Layer and Protocols: In order to achieve different application scenarios under the same architecture and meet different performance targets, the physical layer and protocol design of 5G is more flexible. For example, by allowing different bandwidth options, the use of mini-slots, adaptive reference signal spacing, asynchronous HARQ, the presence of these technologies allows 5G to dynamically adapt to the environment in which it is placed, and to the needs of the application, in order to optimise communication services.

Network Slicing: To deal with different application scenarios, 5G divides the physical network into different virtual networks, each of them optimized to provide service in a different scenario. This is a further development of the quality of service (QoS) technology of 4G. The key to network slicing is Network Function Virtualisation (NFV) and Software Defined Networking (SDN).

Dual Connectivity with LTE: In order to reduce costs, 5G is being designed in the early stages to consider the use of existing infrastructure from 4G. Both old LTE base stations (eNodeB) and newly built 5G base stations (gNodeB) will be connected to the Evolved Packet Core (EPC), which is a networking method called Non-Stand Alone (NSA). As 5G evolves with equipment turnover, Stand Alone (SA) will replace NSA for network services with lower latency.

Radio Cloud and Edge Computing: In previous communications technologies, the core network architecture was highly centralised with a very limited number of core sites, which resulted in high latency of services received at the edge of the core network. In the future, 5G will change this pattern, with core processing becoming more decentralised and cloud and edge servers being deployed at the edge of the network, increasing system flexibility while reducing mobile service latency.

With these technologies, 5G is significantly ahead of 4G in a number of key capabilities. 5G and 4G performance comparisons given by IMT-2020 are shown in Figure 2.3:

Several technologies used for 5G are explained in detail in the next subsections.

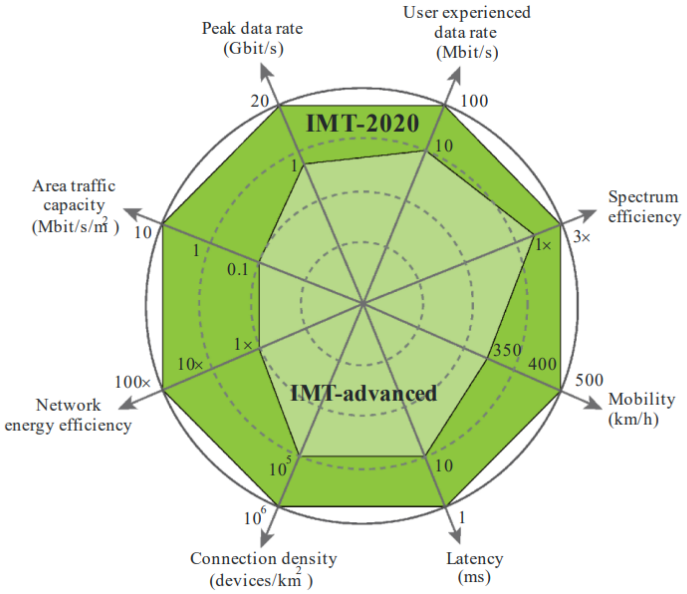


Figure 2.3: Enhancement of key capabilities from IMT-Advanced to IMT-2020 [12].

2.2.1 New Spectrum

Undoubtedly, spectrum resources are central to the development of mobile communications. While spectrum resources below 2 GHz have been almost exhausted in the previous developmental history, 5G significantly widens the available band range and is the first mobile radio system designed to utilise any spectrum between approximately 400 MHz and 90 GHz [13]. For ease of management, the entire frequency range is divided into two segments called FR1 and FR2, and the frequency ranges they contain are shown in Table 2.1.

Table 2.1: Definition of frequency ranges.

Frequency range designation	Corresponding frequency range
FR1	410 MHz – 7125 MHz
FR2	24250 MHz – 52600 MHz

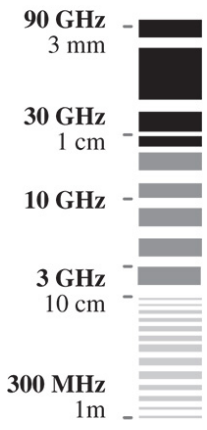
Among them, FR1 is the low frequency band of Sub 6 GHz, which is also the main frequency band for 5G. FR2, on the other hand, is the millimetre-wave band introduced for the first time in 5G, with abundant spectrum resources.

When choosing to use a specific frequency band, specific judgement should be made according to the application scenario in which it is used. According to the propagation characteristics of electromagnetic waves, when electromagnetic waves propagate in a uniform depletion medium, the existence of a constant path loss exponent makes the attenuation experienced during the propagation of waves of different frequencies basically equal with a distance. However, the higher the frequency, the shorter the

wavelength of the electromagnetic wave, and therefore the greater the loss for the same distance if we use the same kind of antenna in transmission and reception (e.g., a half-wavelength dipole).. Based on the above physical facts, 5G High Frequency millimetre wave coverage is poor. However, due to its considerably higher allocable bandwidth compared to traditional low frequency bands, its achievable communication capacity and data rate have increased significantly. Therefore millimetre wave bands are often used in scenarios such as fixed wireless access or to provide service in hotspot areas.

For the Low-Band spectrum included in FR1, it has a wide propagation range and is suitable to be selected for wide-area coverage. The 3.5 GHz band spectrum, known as the mid-band, effectively combines the advantages of high data rates and wide coverage. When using 100 MHz and 4×4 MIMO technology, it can provide a peak rate of 2 Gbps, and if combined with a high-gain base station antenna and beamforming technology, it can simultaneously achieve coverage close to that of the 2 GHz band [13]. Therefore, mid-band is the mainstream frequency band for 5G. The 1.5-2.6 GHz spectrum is already widely used by LTE technology and these bands will be likely good candidates for reallocation for 5G in the future, once migration from 4G to 5G starts reallocated for 5G. sub-1 GHz bands provide wide area coverage and deep indoor penetration. At the same time, due to the latency advantage of its use of Frequency Division Duplex (FDD) technology over Time Division Duplex (TDD), the sub-1GHz band is often used in URLLC scenarios in WANs.

For the band segmentation within FR1 and FR2 and the use cases of different bands are summarised in Figure 2.4.



Spectrum	Typical bandwidth	Use cases
Above 60 GHz	>1000 MHz TDD	• Local hot spot capacity • Indoor and outdoor • Extreme data rates 5 Gbps and beyond • Fixed wireless
24–39 GHz	800 MHz TDD	
2.5–4.9 GHz	100 MHz TDD	• Urban macrocell coverage • High data rate up to 2 Gbps • Capacity boost on LTE sites
FDD sub-3 GHz	20 MHz FDD	• Wide area and deep indoor coverage • Uplink coverage • Low latency and critical communication

Figure 2.4: Segmentation of the 5G spectrum and use cases [13].

While all of the above bands are licensed, 5G also allows the use of some unlicensed bands, which are typically used only indoors with a maximum transmit power that is limited to less than 250mW. Figure 2.5 shows the unlicensed bands allowed in different countries.

In addition to the introduction of new spectrum bands and the possibility to use unlicensed bands, 5G introduces a number of technologies to ensure service flexibility and improve spectrum utilisation. Firstly, 5G allows different carrier bandwidths

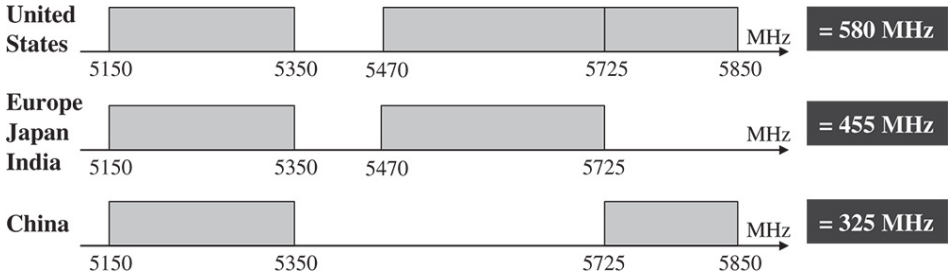


Figure 2.5: 5GHz unlicensed band in different countries [13].

for different frequency bands. Compared to LTE, which only supports a maximum carrier bandwidth of up to 20 MHz, 5G uses narrowband carrier (100 MHz) in the low frequency band and wideband carrier (400 MHz) in the high frequency band, with the wider bandwidth providing much higher communication capacity and data rates. Meanwhile, in LTE, 2 Mhz out of every 20 MHz is used as a guard band to avoid interference from neighbouring bands, resulting in a spectrum utilisation of only 90%. As 5G has more stringent RF requirements, only 2 MHz out of every 100 MHz is used as a guard band (at subcarrier spacing of 30 kHz and bandwidth of 100 Mhz), which increases the spectrum utilisation to 98%.Figure 2.6 shows the spectrum utilisation for 5G in different subcarrier spacing and bandwidth configurations.

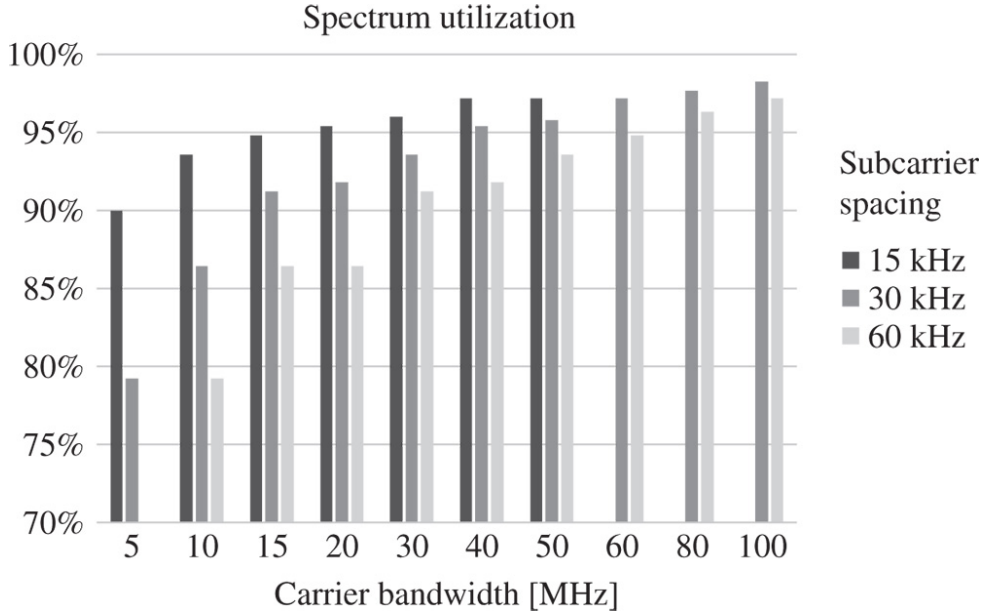


Figure 2.6: Spectrum utilization for 5G [13].

2.2.2 Multiple Access Schemes in the different generations of mobile communications

Multiple access refers to the technology that two or more devices use to establish a wireless channel link to share the available communication resources (also known as

degrees of freedom for communication). Generally speaking, it includes Frequency Division Multiple Access (FDMA), Orthogonal Frequency-Division Multiple Access (OFDMA), Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA), Space Division Multiple Access (SDMA) ,etc.. When the number of access users increases gradually, the technology that can maintain the orthogonality of signals between different users is called orthogonal multiple access technology.

In 4G, OFDMA plays a key role, after implementing OFDM to subdividing the wideband RF channel into orthogonal subcarriers, and loading, data from different users on the subcarriers for transmission. In order to reduce Inter-Symbol Interference (ISI) and to simplify the signal processing at the receiver, a Cyclic Prefix (CP) is added to the OFDM symbols, implementing the so-called CP-OFDM scheme.

However, the waveform of the CP-OFDM signal exhibits significant envelope fluctuations, resulting in a higher peak-to-average power ratio (PAPR). A high peak-to-average power ratio means that the peak power varies over a wider range compared to the average power. In order to to mitigate the impact of intra-band and out-of-band non-linear distortion in the transmission frontend [14], this requires higher demands on terms of the linear response of the High Power Amplifier (HPA). On the UE side of the uplink, this leads to higher power consumption and inefficient communication. For this reason, LTE uses CP-OFDMA only in the downlink and, in the uplink, use Single Carrier Frequency Division Multiplexing (SC-FDMA), also known as DFT-Spread OFDMA. By adding additional DFT/IDFT modules to the transceiver to transmit the subcarriers serially, it is possible to efficiently reduce the envelope undulation level and thus reduce the PAPR. The block diagrams of OFDMA and SC-FDMA transmission chains are shown in Figure 2.7.

However, SC-FDMA has implementation challenges as well. The newly introduced DFT/IDFT module adds complexity to the system, and ISI cannot be completely eliminated by CP. Still, the optimisation of PAPR makes SC-FDMA a multiple access solution for LTE uplink.

However, in 5G, the multi-antenna multi-stream transmission technique has been developed. Compared to SC-FDMA, although the higher PAPR reduces the transmission power of CP-OFDM by 1-2 dB, CP-OFDM still achieves better link performance when using multi-antenna multi-stream transmission [13]. Therefore, in 5G physical layer, CP-OFDM is used as uplink and downlink; only only at the edge of the cell, where the link budget is limited, SC-FDMA is used for uplink transmission.

2.2.3 Duplexing

In mobile communication, the user equipment (UE) and the base station are the two sides of the communication. The link from the UE to the gNB is called the uplink and conversely the link from the gNB to the UE is called the downlink. In order to make the uplink and downlink communication simultaneous, i.e., the transceivers of both the communicating sides can work at the same time, we need duplexing technique.

In 5G NR, duplexing techniques are divided into Time Division Duplex (TDD) and Frequency Division Duplex (FDD). In TDD technology, the uplink and downlink use the same frequency band, while different time slots are allocated. In contrast,

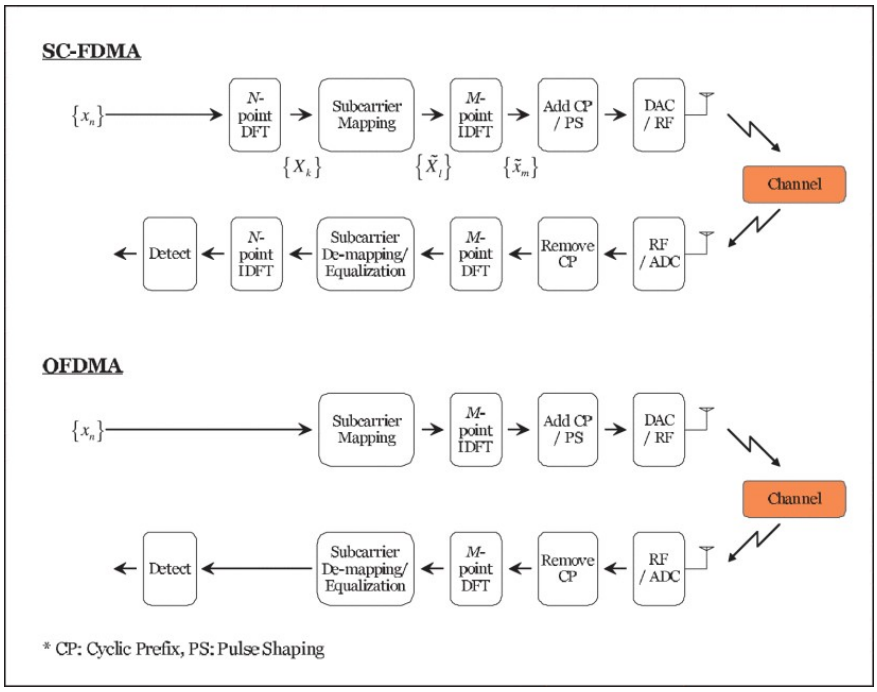


Figure 2.7: Transmitter and receiver structure of SC-FDMA and OFDMA systems [15].

in FDD technique, the uplink and downlink transmit in the same time slot and use different frequency band, properly separated by a duplexing gap, to prevent cross-link interference form the strong transmitted signal to the weak received signal, to achieve duplexing. Figure 2.8 describes the difference between these two duplexing techniques.

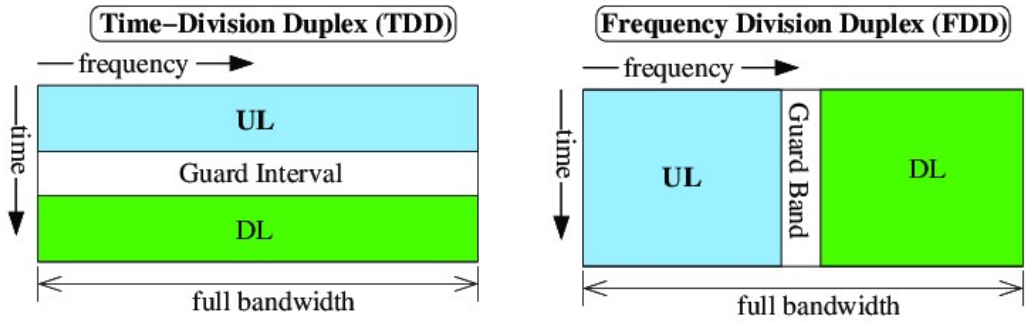


Figure 2.8: In FDD, downlink and uplink use different frequencies, while in TDD different time slot is used [16].

Both TDD and FDD technologies have their advantages and disadvantages. Comparatively speaking, the system using FDD technology is simpler, while the TDD system is more convenient for more flexible frequency allocation of new spectrum and suitable for asymmetric services. Meanwhile, the electromagnetic wave propagation characteristics of TDD are symmetric, and its equipment cost is lower. 5G mainstream

frequency band, mid-band, all use TDD as a duplexing scheme. And this thesis also uses TDD technology.

2.2.4 Frame structure

In mobile communication, data is transmitted in frames over a wireless network. Each frame occupies a relatively short time. For example, in LTE, the length of a frame is 10 ms, and the length of a subframe is only 1 ms. The advantage of this configuration is that in a relatively long time period (e.g., 1 second), the data of multiple users can be assigned to different subframes to transmit, and due to the switching time of subframes is very fast, the users "feel like" they are transmitting in real time.

The frame structure in the time domain in 5G NR is shown in Fig 2.9. A 10-ms-frame is divided into ten 1-ms-subframes, which are further divided into time slots, in which up to 14 OFDM symbols can fit. In this process, the number of time slots that each subframe contains is determined by the subcarrier spacing. When the subcarrier spacing is 15 kHz, each subframe contains only one time slot, which is the same as the LTE configuration. In addition to the time domain, 5G NR has a similar

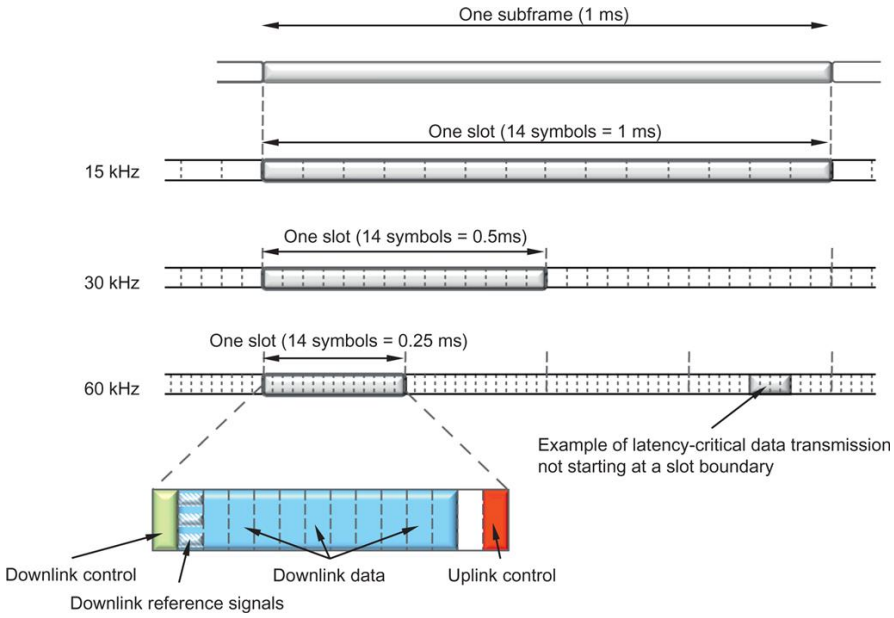


Figure 2.9: Frame Structure in the time domain in 5G NR[17].

structure in the frequency domain. In the frequency domain, the smallest unit is the subcarrier, each of which occupies a bandwidth of 15 kHz, 30 kHz, up to 960 KHz, depending on different subcarrier spacing configurations. In wireless communication, signals are transmitted through a two-dimensional Physical Time-Frequency Resource Block. Figure 2.10 illustrates a Physical Resource Block (PRB), consisting of 14 OFDM symbols and 12 subcarriers. The most basic unit is the Resource element, which consists of one OFDM symbol in the time domain and one subcarrier in the frequency domain. In the frequency domain, there are up to 3300 subcarriers forming 275 subcarrier groups. It is worth noting that the 12×14 PRB design is independent

of the frequency band in which it is located, the subcarrier spacing, and the number of time slots contained in the subframe.

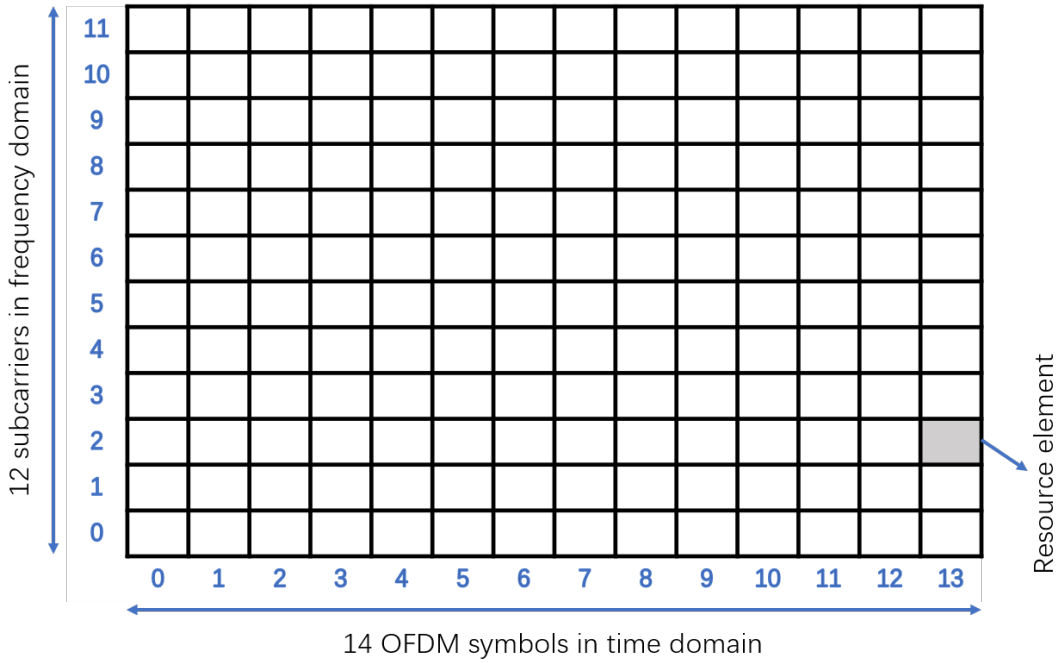


Figure 2.10: A physical resource block in 5G.

2.3 5G Physical layer

According to the definition of Open System Interconnection (OSI), the communication model has seven layers, each of which has its own tasks and functions. The work of this thesis is focused on the physical layer. The physical layer is located at the bottom of the seven-layer model and is responsible for transmitting and receiving raw data streams. The physical layer is also the only layer that provides physical connectivity.

3GPP [18] defines different types of physical channels, they are physical downlink control channel (PDCCH), physical downlink shared channel (PDSCH), physical uplink control channel (PUCCH), physical uplink shared channel (PUSCH), physical random access channel (PRACH) and physical broadcast channel (PBCH). The functions of each channel are briefly explained in the following list.

- **PDCCH**: It carries physical layer downlink control information. It is located before the PDSCH and carries the downlink scheduling information for the PDSCH and also transmits the allocation information for the uplink.
- **PDSCH**: It carries the information data of users information as well as high-level (above the physical layer) control information.
- **PUCCH**: It carries physical layer uplink control information. Note that the PUSCH can also be used to transmit control information, and the PUCCH is used only when the PUSCH is not allocated to transmit control information.

- **PUSCH**:It carries user data as well as high-level control information, and in some cases it is also capable of carrying physical layer uplink control information.
- **PRACH**: It is used to carry random access preamble from UE towards gNB and helps resource allocation and synchronization.
- **PBCH**: It is used for system information broadcasting.

In addition to the physical channels mentioned above, 3GPP defines a number of reference signals.

- **Demodulation References Signals (DMRS)**: These signals are used to aid in channel estimation and are widely used in PUSCH, PDSCH, PUCCH, PDCCH, PBCH. Details about how DMRS specifically works will be described in the next chapter.
- **Phase-tracking Reference Signals (PTRS)**: Such signals are used to help phase noise compensation, especially in the high frequency band. They are used in PUSCH and PDSCH.
- **Channel State Information Reference Signals (CSI-RS)**: These signals are sent from the base station and allow the UE to measure the channel quality and report it to the base station. This signal is only used for the downlink.
- **Sounding Reference Signals (SRSs)**: Such signals are used in the uplink direction to enable uplink frequency domain scheduling. This is necessary to estimate the channel bandwidth beyond the allocation for PUSCH or PUCCH. And since DMRS can only be transmitted with data, SRS is also useful for estimating the channel when no data is being transmitted.

In addition to the above four reference signals, there are also Remote Interference Management (RIM) reference signals and Positioning reference signals, which will not be discussed here.

The functions of the different channels and signals as well as the collaboration between them will not be covered further in this thesis. The focus of this thesis is on PUCCH, which will be explained in detail in the next chapter.

3 Physical Uplink Control Channel

In the previous chapter we presented the basic background knowledge about 5G and different physical channels. In this chapter we will focus on the PUCCH channel, and the channel estimation techniques on that channel.

3.1 PUCCH

As mentioned above, the PUCCH carries uplink control information (UCI), which consists of the following three types of information:

- **Hybrid-ARQ acknowledgments**: Provide feedback for the received downlink data channel (DL-SCH).
- **Scheduling Request (SR)**: The UE uses this information to request resources for uplink data channel (UL-SCH) transmission.
- **Channel-state information (CSI)**: It contains the condition of the downlink channel and assists downlink scheduling.

Note that each frame of PUCCH does not contain all three types of UCIs; the specific UCI assignments will be mentioned later.

In 5G NR, there are a total of five types of PUCCH depending on the number of OFDM symbols occupied and the number of UCI bits that can be carried, etc. The specific differences between the different PUCCH are shown in the Table 3.1.

Table 3.1: Differnet Formats of PUCCH.

PUCCH Format	UCI bits	Waveform	Modulation	Symbols occupied	PRBs occupied	Multiplexing capability
0	1-2	DFT-S-OFDM	Sequence selection with phase rotation	1–2	1	6 UEs
1				4–14		84 UEs
2	> 2	OFDM	QPSK	1–2	1-16	1 UE
3		DFT-S-OFDM	QPSK and Pi/2 BPSK	4-14		
4					1	4 UEs

Different PUCCHs are further classified into long PUCCH (Format 0 and 2) and short PUCCH (Format 1, 3, and 4) based on the length of the OFDM symbols that can be occupied. Figure 3.1 illustrates the difference between long PUCCH and short PUCCH when allocating physical resources.

Relatively speaking, the short PUCCH is more simple. So for the sake of simplicity, the short PUCCH was chosen for this thesis. Since there has been research on Machine Learning (ML) algorithms for PUCCH Format 0 [11], this thesis will focus mainly on PUCCH Format 2, which has not yet been studied. The physical layer processing of PUCCH Format 2 will be described in the following section.

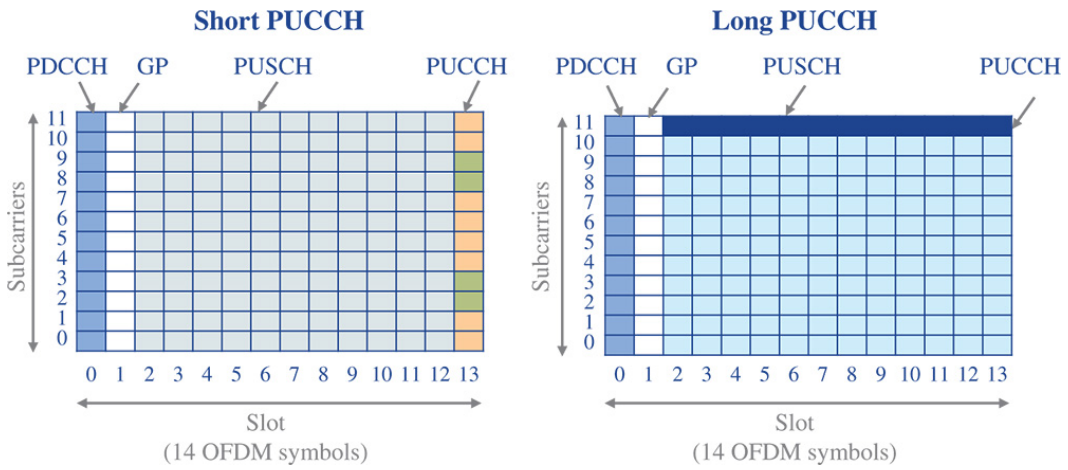


Figure 3.1: Physical Resource allocation for long and short PUCCH [13].

3.2 Transmitter side physical processing of PUCCH Format 2

The transmitter side workflow of PUCCH Format 2 is welly defined in 3GPP TS 38.211/38.212 [18][19]. The whole process flow diagram is shown in the Figure 3.2.

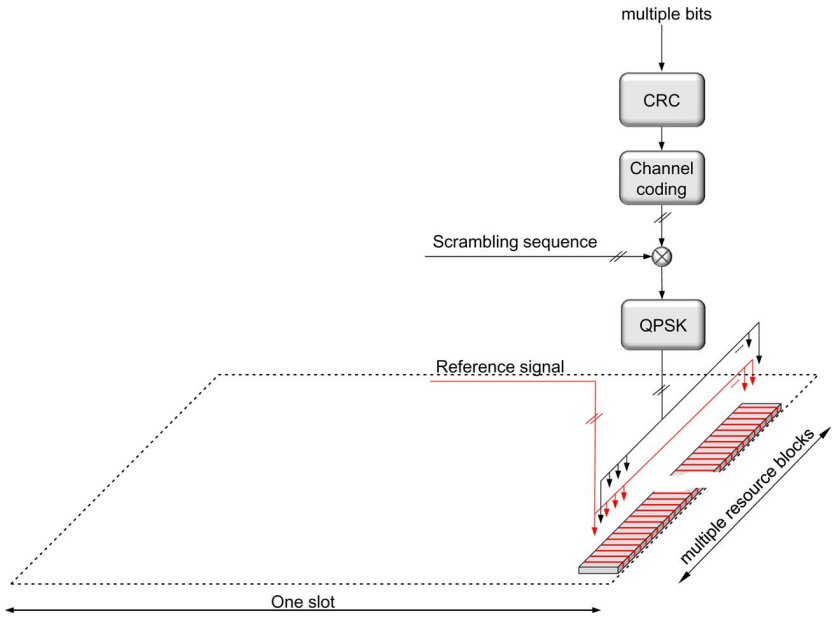


Figure 3.2: Transmitter work flow of PUCCH Format 2 [17].

The whole process is realised in the following steps:

1. **UCI bits sequence generation**: The three UCI types described in the previous section cannot be carried for transmission at the same time. 5G NR defines three different scenarios which are HARQ-ACK/SR only, CSI only, HARQ-ACK/SR and CSI.

2. **Code block segmentation and CRC attachment**: This step depends on the result of the previous step. After the UCI bits sequence is generated, if the number of UCI bits (called payload) A is ≥ 12, then Code block segmentation and CRC attachment will be performed using the Polar code.If A is ≤ 11, then there will be no operation in this step.

3. **Channel coding for UCI**: The operation of this step is also related to the previous step. If CRC was attached in the previous step, then Polar coding will be used for channel coding, otherwise Reed-Muller coding will be used.

4. **Rate matching and code block concatenation**.

The above 4 steps are all Uplink control information related operations and the following operations are modulation related.

1. **Scrambling**: The scrambling sequence is based on the device identity (the C-RNTI) together with the physical-layer cell identity, ensuring interference randomization across cells and devices using the same set of time frequency resources.

2. **Modulation**: The block of scrambled bits shall be modulated using QPSK, resulting in a block of complex-valued modulation symbols.

3. **Spreading**: By expanding the signal in the frequency domain, it improves the signal's ability to counteract noise and interference.

4. **Mapping to Physical resources**: After all the above steps are completed, the PUCCH symbols are multiplied by a scaling factor to meet the transmit power requirements. These symbols are then mapped to physical resource blocks.

In the last step, it is necessary to ensure that the physical resource blocks being mapped are located in the resource blocks assigned for transmission and they are not occupied by DMRS symbols. This involves the generation and mapping process of DMRS symbols. This will be discussed in more detail in the next section.

3.3 Demodulation reference signals

As mentioned above, DMRS is the demodulation reference signal. In practice, the DMRS acts as a pilot signal, and the value of the DMRS symbol as well as its position in the time-frequency two-dimensional resource block are known at both ends of the transceiver. This allows the receiver to combine the value of the initial DMRS and the value of the channel-contaminated DMRS to estimate the channel response other than the DMRS position. This process is channel estimation and more technical details will be shown in the next section.

To achieve the low-latency feature of 5G, DMRS symbols are usually placed at the front of the frame, following a design approach that is known as front-loaded. This allows the location of the DMRS to be determined and channel estimation to

be performed at an early stage of reception. The earlier the channel estimation is completed, the sooner the received signal can be processed to achieve low processing latency. At the same time, the density of DMRS symbols can be flexibly adjusted according to different scenarios. For example, when the channel exhibits time or frequency selective fading, the number of DMRS symbols can be increased accordingly in the time or frequency domain direction to achieve more accurate channel estimation. However, more DMRS symbols means a reduction in the number of symbols able to carry the payload, leading to a reduction in throughput. Therefore a flexible configuration of DMRS is needed to achieve trade-off between throughput and channel estimation quality.

At the transmitter side, the physical-layer process for PUCCH Format 2 DMRS is much simpler than for PUCCH symbols and consists of the following two steps.

1. **Sequence generation**: The generation process is similar to the previous spreading code generation, but uses a different method to initialise the pseudo-random sequence [18].

2. **Mapping to physical resources**: When mapping to physical resources, PUCCH Format 2 adopts only a 1/3 overhead approach. For each PRB, 4 out of 12 subcarriers are allocated to DMRS. In the subcarrier count from zero, the DMRS occupies positions 1, 4, 7, and 9, as shown in Fig3.3.

Figure 3.3: 1/3 DMRS overhead.

Thus, for DMRS in PUCCH Format 2, there is no issue of flexible configuration. Its configuration is very fixed and therefore relatively simple. After processing at the physical layer, the signal passes through a fading channel, receives noise and finally reaches the receiver. Issues related to the channel will be described in Chapter Five. In the next subsection we describe in detail how DMRS is applied to channel estimation and introduce a common channel estimation algorithms, the LS algorithm.

3.4 Receiver side physical processing of PUCCH Format 2

Unlike the transmitter side, the processing chain at the receiver side is not harmonised and standardised. Therefore, different base station manufacturers can design their own algorithms to achieve higher performance as a way to dominate the market. In this section, we present a processing chain for the receiver side shown in Figure 3.4. Note that in real industrial applications, the algorithms are not the same as the ones presented in this section.

1. **Synchronisation**: Once the signal reaches the receiver, it is first synchronised. Synchronisation is done to remove the delay caused by the channel and is usually

Figure 3.4: Receiver chain example of PUCCH Format 2.

determined using the autocorrelation of the signal waveform. After obtaining the delay N, the first N points of the signal waveform are discarded and the remaining points are used as the final received signal.

2. **Channel estimation**: The signals that have undergone synchronisation are subsequently used for channel estimation. By estimating the characteristics of the channel using the various states exhibited by the received signal, a mathematical representation of the effect of the channel on the input signal is obtained. The channel time-frequency response of the entire resource grid can be estimated by using either the LS algorithm or the MMSE algorithm.

3. **Extract Resource Elements and Equalization**: Using the pre-known PUCCH symbol positions, the symbols at the corresponding positions in the receive grids and channel estimation matrices are extracted. Equalisation is subsequently performed using the MMSE algorithm. Channel equalisation is an anti-fading practice taken to improve the transmission performance of a communication system in a fading channel. Both it and channel estimation are two common methods used in order to overcome the multipath effect and mitigate the Inter-Symbol Interference (ISI) problem. However their principles are completely different.Channel estimation is used to calculate the channel response, while channel equalisation is used to compensate for channel attenuation and fading [20]. Depending on the algorithm, channel equalisation is classified as linear and non-linear equalisation. The commonly used linear equalisation algorithms are the Zero Forcing (ZF) algorithm and the Minimum Mean Square Error (MMSE) algorithm. The commonly used nonlinear equalisation algorithms are Vertical Bell Laboratories Layered Space-Time (V-BLAST) algorithm and Maximum Likelihood Detection (MLD) algorithm.

4. **Decoding**: After the above series of operations, we have obtained the symbol at the PUCCH position. Using the decode function allows us to get the information bit inside, i.e., the UCI. The decoding process is equivalent to the inverse process of encoding and also requires the use of Polar coding or Reed Muller coding depending on the situation. Since the signal is transmitted on a frame-by-frame basis and each frame is eventually divided into different time slots, we use the Block Error Rate when measuring the quality of the transmission. When we decode the UCI of each time slot, we compare it with the UCI of this time slot at the transmitter side, and if any bit of these two are not the same, we consider that a transmission error has occurred in this time slot. The final number of time slots with transmission errors divided by the total number of time slots is the Block Error Rate.

The most important and relevant part of this process to this thesis is the channel estimation part.

In a communication system, for the n-th time slot and the m-th subcarrier, the relationship between the received and transmitted signals of the whole system can be expressed as

$$Y_{m,n} = H_{m,n} \times X_{m,n} + N_{m,n}, \tag{1}$$

where $Y_{m,n}$ denotes the received signal, $X_{m,n}$ denotes the transmitted signal and $N_{m,n}$ denotes the noise. $H_{m,n}$ is the (m, n) element of the channel response H, which represents the channel matrix over all time slots and subcarriers.

Of these parameters, the received signal Y is known. Moreover, a part of the transmitted signal X is also known due to the use of a reference signal. Whereas, the channel H and the noise N are unknown. Therefore, the process of channel estimation is also the process of estimating H with known Y and part of X.

If there is no effect of noise in the system, then we can easily derive the channel response at the location of the reference signal as $h_r = y_r/x_r$, where h_r, y_r and x_r are the channel response at the location of the reference signal, the received reference signal, and the transmitted original reference signal, respectively.

In practice, however, it is impossible to ignore the effects of noise. In combination with the known information about the reference signals at both the sending and receiving sides, the channel h_r can be optimised so that after passing through the channel h_r, the transmitted reference signal x_r is as consistent as possible with the received y_r. The transmitted reference signal passing through the channel can be represented as $h_r x_r$, whose difference with y_r is $y_r - h_r x_r$. When optimising to minimize this difference, if the criterion chosen is the square of a two-norm, this algorithm is the LS algorithm. The channel estimation problem is transformed into an optimisation problem shown in the equation

$$\hat{H}_r^{LS} = arg \min_{h_r} \|y_r - h_r x_r\|_2^2, \tag{2}$$

where $\|.\|_2$ represents the two-norm. $\hat{H}_r^{LS}$ is the estimated diagonal matrix obtained using the LS algorithm. The optimisation of the above equation results in $\hat{h}_r^{LS} = diag(\hat{H}_r^{LS}) = y_r/x_r$. Coincidentally, the optimal channel estimate obtained using the LS algorithm is also the received reference signal divided by the transmitted original reference signal.

A better choice than LS is to use the MMSE algorithm. The results of the MMSE estimator are obtained from multiplying the estimate $\hat{h}_r^{LS}$ by a filter matrix F_{MMSE}. The filtering matrix Ammse requires the use of the channel correlation matrix between the received signal and the reference signal, as well as the autocorrelation matrix of the reference signal, so the MMSE algorithm is only useful if these information is known [10]. At the same time, the MMSE filtering matrix significantly increases the computational complexity as well as the computational time [21], and is therefore

difficult to be applied in practice. This thesis will focus on the LS algorithm, and the theory and practice regarding the MMSE algorithm will not be discussed.

It is worth noting that the LS algorithm can only obtain the channel estimate at the reference symbol positions, not the entire time-frequency plane. In order to further obtain the channel response at other locations, especially the PUCCH symbol location, we need to use interpolation. The operations after the LS channel estimation will be described in Chapter 5. In the next chapter, the ML background knowledge required for deep learning-based channel estimation algorithms will be presented.

4 Machine Learning Tools for Channel Estimation

This chapter will introduce the Machine Learning (ML) background knowledge required for this thesis. ML is a branch of Artificial Intelligence (AI) designed to allow computer systems to improve performance through empirical learning. The core idea behind ML is to extract patterns and regularities from large amounts of data, such that machines can make predictions or make decisions without explicit programming. In this chapter, the basic concepts of ML are first introduced, including the different sub-categories in which this learning paradigm can be implemented, as well as the key concepts of datasets, test sets, and validation sets. Subsequently, an introduction to Neural Networks (NNs) is given, including the concepts of neurons, layers and activation functions, as well as the overall explanation of the way in which feed-forward NNs work. Finally, Convolutional Neural Networks (CNN) are introduced. Since the newly developed algorithm uses CNNs, we put special attention to this category of NNs when presenting the content of this chapter.

4.1 Machine Learning Basics

ML centers on developing algorithms that learn from input data and, in a subsequent phase, use this learning to make predictions or decisions rather than following strict static program instructions. Early explorations of ML began in the 1950s. From simple models and algorithms, to the exploration of knowledge-intensive methods, and ultimately with the help of backpropagation algorithms and other key techniques, the foundations of modern deep learning have been laid. Since the beginning of the 21st century, the explosion of big data and increased computational power has driven the rapid development of deep learning, making ML a central force in modern technology and innovation. Nowadays, ML is widely used in healthcare, finance, retail and e-commerce, autonomous driving, and many other fields.

From a very general perspective, ML can be categorized into several main types based on the nature of the learning process [22]:

1. **Supervised Learning**: This is a major form of ML in which algorithms learn from training datasets with known answers. In this learning process, each training sample is composed of input features and expected outputs that are often called "labels". The task of a supervised learning algorithm is to learn a model by analyzing this data and being able to predict or determine the correct output when given a new input data that have not been seen before during the training process. Supervised learning has a wide range of applications, covering tasks ranging from simple email filtering (e.g., spam recognition) to complex image recognition and speech recognition. The two main types in which supervised learning can be sub-divided are classification and regression. In classification, the output is discrete labels such as the image recognition of cats or dogs, whereas in regression the output is a continuous values such as the prediction of predicting the price of a house. One of the key challenges in supervised learning

is to avoid overfitting, i.e., to ensure that the model not only performs well on the training data, but also generalizes to new datasets.

2. **Unsupervised Learning**: Unsupervised learning deals with data that is not labeled or categorized. In this learning approach, algorithms are used to recognize underlying structures and patterns in the data, rather than learning from training data with explicit answers. Since the input data is unlabeled, the goal of unsupervised learning is usually data clustering, association rule learning, or feature dimensionality reduction. This type of learning is particularly useful in scenarios where labels are difficult to be identified or costly to be obtained, such as in market segmentation, social network analysis, organization of complex datasets, or identification of unusual patterns in data such as in anomaly detection. One of the unsupervised learning challenges is evaluating the performance of the model, as there is no clear right answer or standard answer to validate the results.

3. **Semi-Supervised Learning**: Semi-supervised learning is a ML method that sits between supervised and unsupervised learning and deals with partially labeled datasets. In such datasets, a small portion of the data is usually labeled (i.e., the output is known), while the majority of the data is unlabeled. The core idea of semi-supervised learning is to use both labeled and unlabeled to build more effective learning models. This approach is particularly useful in situations where labeled data is scarce or the labeling process is costly, such as in large-scale image or text data processing. The advantage of semi-supervised learning is that it combines the accuracy of supervised learning with the effectiveness of unsupervised learning on large amounts of data, with the aim to improve learning efficiency and performance. However, the way in which labeled and unlabeled data should be combined, such that the misinformation that is introduced by the unlabeled data does not interfere with the model is one of the major challenges in semi-supervised learning.

4. **Reinforcement Learning**: Reinforcement learning focuses on procedure that should be followed to train algorithms to make decisions based on interaction with the environment. In reinforcement learning, an Agent learns during the process the best way to take an action in a given environment by trial and error. This process involves exploration by trying new or unknown actions to discover better solutions, and exploitation by using known information to maximize rewards. Whenever an agent receives feedback from the environment in the usual form of a reward or a punishment) based on the actions that have been taken, it adjusts its action strategy to try to increase the overall reward received. Reinforcement learning is widely used in various scenarios such as automated game players, self-driving cars, robot control, and other fields. The reason that makes reinforcement learning unique compared to other types of ML is that it focuses on long-term goals in uncertain and exploratory environments, not just immediate feedback.

4.2 Neural Networks

NNs are an algorithmic structure in ML that mimic the workings of the human brain and are used to recognize complex patterns and relationships in data. NN are networks of a large number of units, known as *neurons*, which communicate through weighted connections. Each neuron receives an input, processes it, and produces an output. The entire NN learns by adjusting the weights of the connections. A typical multilayer neural network topology is showen in Figure 4.1.

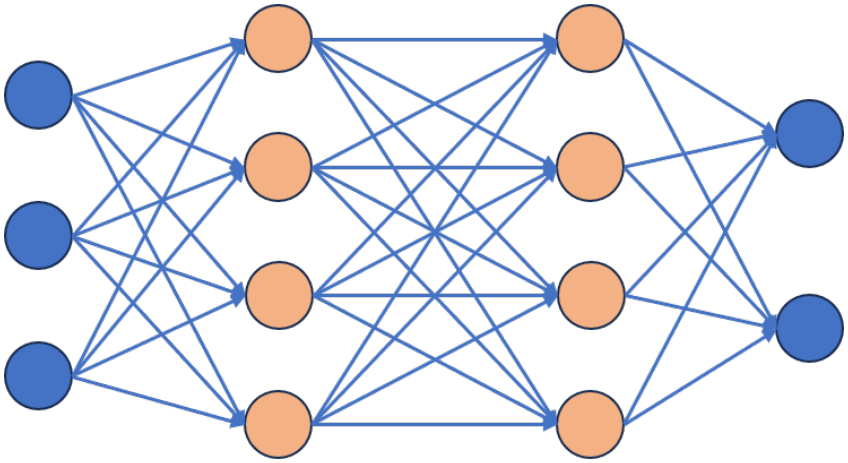


Figure 4.1: A typical multilayer neural network topology.

NNs are inspired by the structure and function of the nervous system in biology, especially the human brain. In the human brain, neurons communicate with each other through synaptic connections and transmit information through electrochemical signals. Each neuron is capable of processing and forwarding the signals it receives. This complex network structure allows the brain to process complex information such as perception, cognition and decision-making.

In computer science, this structure is abstracted and simplified into mathematical models. While modern NNs differ significantly in structure and function from true biological neural networks, the idea of borrowing inspiration from biology has provided a powerful impetus for the development of AI and ML. By mimicking the way in which a biological brain processes information, NNs are able to learn and perform a variety of complex tasks that range from basic pattern recognition to complex decision making.

4.2.1 The concept of a neuron in a NN

Neurons are the basic units that make up a NN, inspired by the nerve cells in the biological nervous system. In an artificial NN, each neuron receives input signals from the previous layer of neurons, which are passed through connections with weights. The neuron takes a weighted sum of all the input signals and then processes this sum through an activation function that determines the value of the signal at the output of the neuron. The activation function serves to introduce nonlinearities that allow the

neural network to learn and model complex patterns and functions. The output of the neurons is then passed on to the next layer of neurons. During training, the neural network learns a specific task by adjusting the connection weights, and this adjustment is based on the error value that is obtained when comparing actual observed data with the expected output. This structure and way of working of neurons allows NNs to perform a wide range of tasks that range from simple linear regression to complex image and speech recognition.

4.2.2 Interconnection and weights of connections in a NN

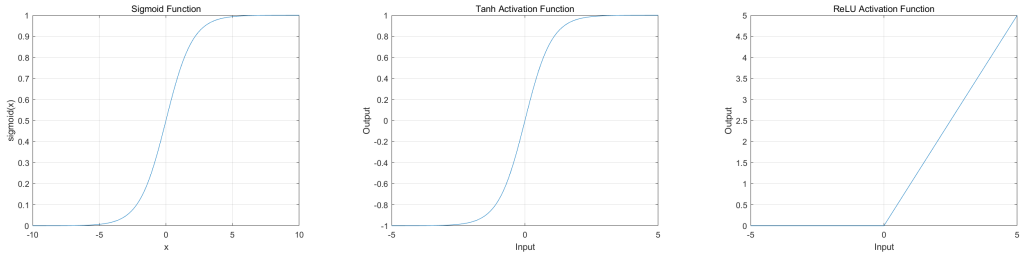
In NNs, connections and weights are the key elements that enable information transfer and learning. Connections represent pathways between neurons that allow signals to be transmitted from one neuron to another. Each connection is characterized by a coefficient known as weight, which is a numerical value that determines the strength and importance of the signal that is being transmitted. As the network undergoes a learning process, these weights are continuously adjusted based on the input data and the performance of the network. When a neural network receives inputs and produces outputs, each input of a given neuron is a weighted sum of the outputs of the neurons that were placed in the previous layers of the NN. The weights of the neurons determine how much each input contributes to the level of neuron activation. In this way, the weights control the way information flows through the network, allowing the NN to recognize complex patterns and relationships by learning and adjusting these weights.

4.2.3 The concept of an activation function in a NN

Activation functions play a vital role in a NN as they determine whether a neuron should be activated or not or, equivalently, whether a neuron should respond to the given input information or not. In a neural network, each neuron receives input signals from the previous layer, which are weighted and summed and passed to the activation function. The activation function processes this weighted input sum and then produces an output, which is then passed on to the next layer of the network. This process is repeated throughout the NN, allowing to perform complex transformations on the input data. There are many types of activation functions, each with its own specific mathematical expression and characteristics. Among the most common activation functions that have been extensively used in the literature are: [23]:

1. **Sigmoid activation function**: This function is a widely used activation function in NNs, especially in early network architectures. It is mainly characterized by its ability to map arbitrary values between 0 and 1 with smooth, continuous properties, and is often used to interpret outputs as probabilities or as an output layer in binary classification problems. Sigmoid function is shown in Figure 2(a) , and its mathematical expression is given by

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{3}$$



(a) Sigmoid (b) Tanh (c) ReLU

Figure 4.2: Three different activation functions in NN.

2. **Hyperbolic tangent function (Tanh)**: This activation function commonly used in NNs that map the input to a range from -1 to 1. This zero-centered output helps to increase the speed of convergence during the learning process of the NN. The Tanh function is superior to the Sigmoid function because it avoids the problem of non-zero centering of the output. However, it still suffers from the gradient vanishing problem[1], especially when dealing with extreme input values. Due to these properties, Tanh is commonly used in the hidden layer of neural networks. Tanh function is shown in Figure 2(b). Mathematical expression is given by

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{4}$$

3. **Rectified Linear Unit (ReLU)**: This activation function is one of the most popular activation functions in NNs, especially in deep learning models. The mathematical formula of this activation function is given by

$$\mathrm{ReLU}(x) = \max(0, x). \tag{5}$$

This means that if the input value is positive, the value is output directly; if it is negative, 0 is output. ReLU function is shown in Figure 2(c). This design keeps the ReLU function linear over positive intervals, which solves the gradient vanishing problem and allows the NN to train and converge faster. In addition, due to its computational simplicity since it only needs to determine whether the input is greater than 0, the ReLU activation function is very efficient in practical

[1]The gradient vanishing problem takes place during backpropagation of the network, when the gradients are passed through multiple levels, they get smaller and smaller until they are almost zero, which makes the weights of the lower levels of the network barely updated.

applications. However, ReLU has some limitations, such as the "dead ReLU" problem, where the derivative of the ReLU function is zero when the input is negative, resulting in the corresponding neurons failing to update and learn. Nevertheless, ReLU and its variants (e.g., Leaky ReLU and Parametric ReLU) remain one of the preferred activation functions for the construction of modern NNs due to their effectiveness in training deep networks.

4.2.4 The concept of loss function in a NN

In NNs, the loss function is a central component used to quantify the difference between the model output and the actual target value. The main purpose of this function is to measure the degree of error in the predictions of the model. During training, by minimizing the loss function, the neural network adjusts its weights and biases to improve the accuracy of its predictions. The loss function is not only used as a metric to optimize the parameters of the neural network, but also to evaluate the overall performance of the model. For different tasks, such as regression or classification, choosing the right loss function is critical because it directly affects the efficiency of model training and the accuracy of the final output. During backpropagation, the gradient of the loss function is used to guide the updating of the network parameters, ensuring that the model gradually reduces the prediction error during the iteration process. Thus, the loss function plays a decisive role in defining how the model learns and its effectiveness.

A common loss function is the Mean Squared Error (MSE) function. The MSE loss function is commonly used in NNs for regression tasks. The mean square error loss function is calculated by the formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{6}$$

where n is the number of samples, y_i is the actual value of the i th sample, and $\hat{y}_i$ is the corresponding model predicted value.

The core of the MSE loss function is to calculate the squares of the differences between the actual and predicted values, and then compute the mean value of these squared differences over all sample data inputs that are available. The advantage of MSE loss function approach is that it emphasizes the effect of larger errors, because the error input samples are squared during the computation of the mean value. Since the squared error is used as a loss, the model is incentivized to pay more attention to those samples with larger errors, thus reducing those errors during training.

The MSE loss function is very effective when dealing with regression problems, especially when predicting continuous values. The MSE loss function can also provide a simple and intuitive way to measure and optimize the predictive accuracy of a model. However, it is important to note that in some cases, the MSE loss function is very sensitive to outliers as it will double down on larger errors. This means that MSE may not be the best choice of loss function in situations where the influence of outliers or noisy data is large.

4.2.5 Learning process of Neural Networks

The learning process of a NN is an iterative process involving forward propagation and backpropagation designed with the purpose of optimizing the weights and biases of the network to reduce the discrepancy between the output and the actual target. Forward propagation usually has the following steps:

1. **Input data**: forward propagation starts at the input layer, where the input is usually preprocessed data such as normalized feature vectors, image pixel values, or text data.

2. **Layer-by-layer computation**: at each layer, each neuron receives input signals from neurons in the previous layer. Each input signal is scaled up or down by the corresponding weights, and then these weighted signals are summed up. Typically, this weighted sum is also augmented with a bias term, which is given by

$$z = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b, \tag{7}$$

   where w is the weight, x is the input, and b is the bias. The weighted sum is then passed to an activation function. The role of the activation function is to introduce nonlinearity, which allows the neural network to learn and model complex data patterns.

3. **Output data**: after processing in the last layer, which is usually the output layer, the network generates a final output. This output can be a continuous value (e.g., in a regression problem), a probability distribution of category labels (e.g., in a classification problem), or some other form of data.

In summary, forward propagation is a linear process in which the data starts at the input layer, goes through a series of hidden layers, and eventually reaches the output layer. The computations within each layer are independent, relying on the weights and biases of that layer, as well as the output of the previous layer. This process is executed independently on each training instance, with each input producing a new output. This output can later be used to calculate the loss function, which is the next critical step in training a NN. The design and implementation of forward propagation determines the way in which a NN processes the data and the complexity and type of learning it is capable of. The design of each layer (e.g., number of layers, number of neurons per layer, and choice of activation function) affects the performance and learning ability of the network.

Backward propagation in contrast, is the core algorithm used in NNs to train and optimize the network parameters (weights and biases). It is a gradient descent based method that adjusts the loss function by calculating the gradient of these parameters with respect to the network parameters in order to reduce the difference between the output and the target value. The process of backward propagation can be divided into the following four steps:

1. **Calculating Loss**: Before backward propagation begins, the output of the network is first obtained through forward propagation and then the loss function is calculated. The loss function measures the difference between the network output and the actual target value.

2. **Calculating the gradient**: the heart of backward propagation is to calculate the gradient of the loss function with respect to each parameter in the network. This involves a layer-by-layer backward traversal of the network from the output layer back to the input layer. For each layer, the gradient of the loss function relative to the output of that layer is first calculated. This is usually done via the chain rule, which is a basic trick in calculus for calculating the derivative of a composite function. This gradient is then used to compute the gradient with respect to the weights and biases of the layer. In short, this step determines how each weight and bias can be changed to minimize losses in the most efficient way.

3. **Updating the parameters**: once all the necessary gradients have been computed, an optimization algorithm (usually gradient descent or a variant thereof) is used to update the weights and biases of the network. This step involves reducing the values of the weights and biases by a small percentage based on the calculated gradients. The magnitude of the update is usually controlled by the learning rate, which is a hyperparameter that determines the step size of the parameter update in each iteration.

4. **Repeated Iterations**: this process is repeated over the entire training set, with each iteration consisting of a forward propagation and a backward propagation. Through these successive iterations, the network gradually learns the values of the parameters that reduce the prediction error.

In summary, backward propagation is a systematic process that involves not only computing gradients, but also how to effectively use these gradients to update network parameters. It is one of the most important training algorithms in deep learning and NNs, making it possible to train complex multilayer network structures. Through backward propagation, neural networks are able to learn from their mistakes and continually adjust their parameters to better accomplish a given task. This process is the cornerstone of realizing AI/ML applications.

4.3 Convolutional neural network

CNNs are deep learning models specialized for processing data with a grid structure, such as images. At their core lies the use of convolutional operations, which are special operations that capture localized features such as edges and corners by sliding small windows (convolution kernels) over the input data. The design principles of CNNs include local sensing fields, which allow the model to focus on small regions of the input data; parameter sharing, which reduces the complexity of the model by reusing the same weights across the entire input; and a hierarchical structure that

more complex and abstract feature representations are automatically learned through multi-layer convolutional operations.

A typical CNN contains several different types of layers. The first is a convolutional layer, which is responsible for performing convolutional operations that are used to extract different features of the input image. The convolutional layer is usually followed by an activation layer, such as the ReLU activation function, which introduces nonlinearities that allow the network to learn complex patterns. This is followed by a pooling layer, such as the Maximum Pooling Layer, which reduces computational complexity by reducing the spatial size of the feature map and provides a degree of translational invariance. After the convolution and pooling layers, there are usually fully connected layers that perform classification or regression based on the extracted features. The last layer is the output layer, which produces the final output based on the needs of the task, such as multi-class classification.

The ability of a CNN in feature extraction has made it a powerful tool in the field of computer vision, which is widely used in various fields such as image classification, facial recognition and object detection. In addition, the applications of CNNs extend to other fields such as speech recognition, video analysis and text processing. Through its unique structural design, CNNs are able to effectively process and interpret complex input data to achieve efficient and accurate pattern recognition.

4.3.1 Cnvolutional Layer in CNN

The convolutional layer is a crucial part of a CNN and is responsible for extracting features from the input data. The convolutional layer is implemented through convolutional operations in which small arrays of weights called convolution kernels are used to slide over the input data. These convolution kernels perform element-level multiplication operations with localized regions of the input data and sum the results to generate a feature map. The size of the feature map is determined by the step size of the sliding of the convolution kernel, while the edges of the input data can be controlled by adding additional zeros (zero padding) to control the spatial dimensions of the feature map.

Within the convolutional layer, each convolutional kernel focuses on capturing specific features of the input data, such as edges or textures. This is because the convolutional layer is locally connected, making each convolutional kernel connected to only a small portion of the input data. In addition, all convolutional kernels share the same weights over the entire input data, which not only reduces the number of parameters in the model, but also improves training efficiency. Convolutional layers usually contain multiple convolutional kernels, each responsible for extracting different types of features, such that the multikernel structure enables the network to extract a rich feature set from the input data.

In order to enable the network to learn more complex data patterns, the convolution operation is usually followed by the application of a nonlinear activation function, such as ReLU. This step is necessary because the convolution operation itself is linear. By introducing a nonlinear activation function, the convolutional layer is not only able to capture the basic features of the input data, but also able to construct more

complex and abstract feature representations, which enables CNNs to exhibit excellent performance when processing high-dimensional data such as images and videos.

4.3.2 Pooling Layer in CNN

The pooling layer is an important component in a CNN, usually located after the convolutional layer. Its main role is to reduce the spatial size of the feature map, thereby reducing the number of parameters and computational complexity in the network, while improving the robustness of the model to small positional changes [24]. By reducing the parameters, the pooling layer also helps to mitigate overfitting and enhance the generalization of the model. Operationally, the pooling layer slides over the feature map and applies the pooling operation by defining a window, which is usually 2×2 or 3×3 in size, and a step size that determines how far the window slides.

There are two main types of pooling layers: maximum pooling and average pooling. Maximum pooling is the most common type, which takes the maximum value from each window region and helps capture the most significant features. Average pooling, on the other hand, takes the average value within the region and helps smooth the feature map. Despite the relative simplicity of the operation, the pooling layer plays a key role in building efficient convolutional neural networks by reducing the size of the feature map, which not only reduces the computational burden on the subsequent layers, but also makes the network more insensitive to small displacements and changes in the input data. This insensitivity to small positional changes allows the CNN to better extract key features from images or other similar data, thus improving the overall performance and robustness of the model.

4.3.3 Fully Connected Layer in CNN

The fully connected layer is an important part of a CNN and is usually located at the end of the network. The neurons in this layer are connected to all the activations in the previous layer and are responsible for synthesizing the features extracted in the previous layers to make the final decision. In a fully connected layer, if the previous layer was a convolutional or pooling layer, its output is spread into a one-dimensional vector and fed into the fully connected layer. Here, each neuron performs a weighted summation of these inputs with bias, and finally processes them through an activation function. The role of this layer in a neural network can be understood as a "decision layer", making predictions such as classification or regression based on the extracted features.

The fully-connected layer is characterized as parameter-dense, since each input is connected to all outputs. This makes it the most computationally intensive and overfitting prone part of the network. The outputs of the fully connected layer can be real-valued, such as in regression problems, or they can be probability distributions, which are often implemented via softmax activation functions in classification problems. In complex CNN architectures, there may be multiple fully-connected layers arranged in succession to progressively integrate the features extracted from the inputs,

ultimately forming a prediction for a given task. Thus, fully connected layers play a crucial role in integrating features and making task-specific predictions.

5 Numerical Results and Performance Analysis

The background knowledge required for this thesis was presented in the previous chapters of this Thesis. This chapter introduces the most crucial practical part of the thesis. Firstly the link-level simulation of PUCCH Format 2 with the aid of the Matlab 5G Toolbox is introduced. In the link level simulation section, the channel model and the LS algorithms implemented in Matlab are presented and a comparison of the obtained results with those reported in other literature are given. Finally, the Convolutional Neural Network (CNN) based channel estimation algorithm is presented in detail. This section includes the structure of the CNN model, the details of the dataset that is used for training and validation, the description of the training and learning process of the NN, and the final performance comparison.

5.1 Link-level simulation

Matlab 5G Toolbox is a software developed by Mathworks to simulate, analyse and test 5G communication systems. Its functions cover waveform generation, link-level simulation, test and measurement, and system-level simulation, and so on. This thesis mainly uses its link-level simulation function because the purpose is to analyze the performance of the PUCCH, which is a Physical Channel of 5G NR.

5.1.1 Implementation of the whole OFDM Frame Transmitted in Uplink

As described in Chapter 3, 3GPP has defined in detail the processing flow at the transmitter side of 5G communications. Therefore, from the generation of Uplink Control Information (UCI) bits until the mapping to the physical resource block, the 5G toolbox adopts the same operation as the 3GPP protocol. After the mapping to the physical resource block shown in Figure 3.2, the toolbox takes the operations shown in the Figure 5.1.

Figure 5.1: Block diagram of the PUCCH transmission.

OFDM modulation After mapping to the physical resource blocks, the signal is presented in a time-frequency two-dimensional matrix format. The allocation of resources in the time and frequency domains is determined by specific parameter settings. In this step, we need to transform the two-dimensional data matrix (containing IQ samples in the time-frequency grid before IFFT) into a one-dimensional signal (time domain signal samples after IFFT) with the help of the *nrOFDMModulate* function in Matlab. It is worth noting that here we need to set parameters such as sample rate and carrier frequency. After modulation, the signal is a one-dimensional complex vector, which is the time domain signal that is transmitted over the wireless channel after up-conversion into the corresponding passband portion of the spectrum.

5.1.2 Channel

After the signal modulation, the effect introduced during the propagation through the wireless channel needs to be simulated. The way to model the effects of various factors in a real channel on the signal is the focus of this section. 3GPP [25] defines in detail how the effects of the various factors in a fading channel can be modelled, giving the generic channel generation steps shown in the Figure 5.2.

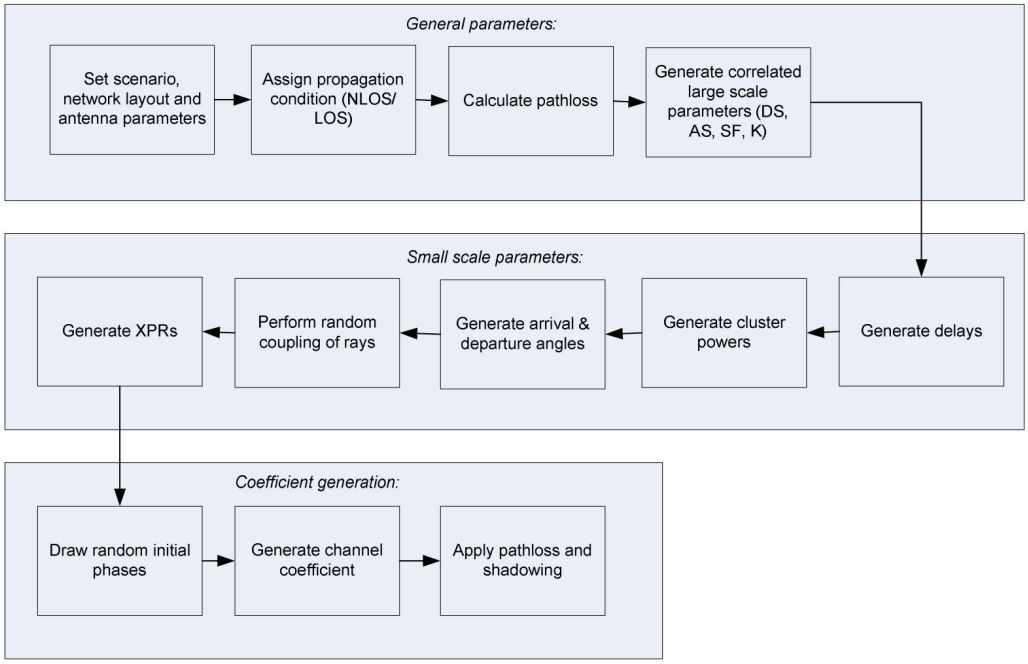


Figure 5.2: Channel coefficient generation procedure [25].

To enable link-level simulation, 3GPP [25] defines Clustered Delay Line (CDL) models. These models are suitable for the full frequency range from 0.5 GHz to 100 GHz with a maximum bandwidth of 2 GHz. To simplify the evaluation, e.g. when modelling non-MIMO scenarios, Tapped Delay Line (TDL) models can be simplified from CDL models by assuming that both the transmit and receive antennas are ideally isotropic. Since the system in this paper is a 1×1 SISO system, all the channel models involved are TDL models.

When modelling a TDL model, there are two key parameters to be defined to characterize the propagation of a radio signal over a wireless channel, which are the Doppler shift and the delay spread. These two parameters will directly affect the scenario represented by the channel. Doppler shift is the change in frequency due to relative motion between the user equipment (UE) and the base station (BS). When the UE moves towards the receiver, the Doppler shift causes the transmission frequency to increase; conversely, when the UE moves away from the BS, the transmission frequency will decrease. The relationship between the Doppler shift and the velocity

of motion is given by

$$f_d = f_c \times v/c, \tag{8}$$

where f_c represents the carrier frequency, v represents the relative velocity of the UE and the base station, and $c \approx 3 \times 10^8$ m/s represents the speed of light. It can be intuitively seen that the frequency spread caused by Doppler shift is more severe when the relative velocity of motion is higher. If we know the maximum value of the velocity, we can obtain the maximum Doppler shift f_m. In communications, the coherence time is the statistical average of the time intervals during which the channel impulse response can be assumed to remain unchanged, and this value is inversely proportional to the maximum Doppler shift. If the symbol time that uses the wireless communication system is higher than the coherence time of the channel, there is a high probability that the channel will change during the transmission time slot of a symbol thus leading to signal distortion, in a wireless communication configuration that is known as time-selective fading. In engineering, the relationship between the coherence time and the maximum Doppler shift is determined by [26].

$$T_c = 0.423/f_m. \tag{9}$$

Another way to determine the delay spread is based on measuring the difference between the arrival time of the earliest multipath component and the latest multipath component in the multipath transmission. This parameter describes the multipath richness of the channel. Similar to coherence time, coherence bandwidth describes the strong amplitude correlation between any two frequency components over a specific frequency range; that is, the coherence bandwidth defines the range in which the multipath channel maintains constant gain and linear phase over the coherence bandwidth range. When the signal bandwidth is greater than the coherence bandwidth, the different frequency components of the signal are subjected to different fading and frequency selective fading occurs.

Depending on whether the transmission is line-of-sight or not, as well as on delay spread, 3GPP [27][28] assigns different scenarios to the different TDL channel models, as shown in the Table 5.1.

In Matlab 5G Toolbox, the *nrTDLChannel* function can be used to generate a specified TDL channel model, and parameters such as the maximum Doppler shift and delay spread of the channel can be specified at the same time. The time-frequency 2D grid plots along with the characteristics along the time-domain direction and frequency-domain direction for different channels are shown in Figure 5.3.

In the link-level simulation, the signal after modulation is filtered by the generated channel model, thus simulating the effect of the channel on the signal. After passing through the channel, Gaussian white noise is added to the signal using the awgn function depending on the different SNR values. At this point, the signal has experienced all the influences of a fading channel in a real environment, and the operation of the signal at the receiver will be described in the next section.

Table 5.1: Different Channel Models that have been defined to characterize the performance of different PUCCH format.

TDL Channel	Scenario	Delay Spread/ns
TDL-A	Indoor NLoS	117.3
TDL-B	Urban Microcell NLoS	81.2
TDL-C	Urban Macrocell NLoS	1295.2
TDL-D	Indoor Hotspot LoS	22.1749
TDL-E	Urban Macro LoS	1324.7

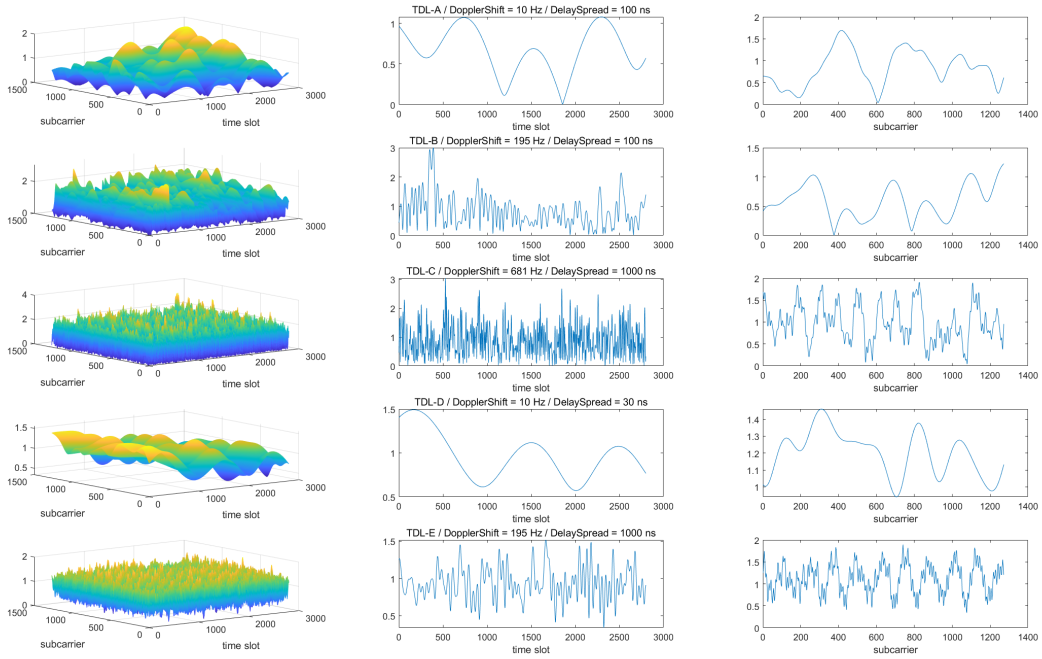


Figure 5.3: Three-dimensional channel images, and two-dimensional images of the channel along the time and subcarrier directions.

5.1.3 Receiver chain in Matlab 5G Toolbox

The operation of the receiver side has already been mentioned in Chapter 3, and this section will specifically address the operation in practical applications. In the Matlab 5G Toolbox link-level simulation, the processing chain at the receiver side is shown in Figure 5.4.

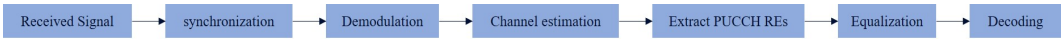


Figure 5.4: Receiver chain in Matlab 5G Toolbox.

In order to obtain accurate information about the channel for comparison with the estimated channel, Matlab has a number of functions that are able to obtain channel configuration information, which will be presented in different modules. It should be noted that obtaining accurate channel information is not possible in a real wireless communication environment.

Synchronisation: In the 5G toolbox, synchronisation is done by the *nrTimingEstimate* function. The delay of the signal is obtained by calculating the cross-correlation between the reference signal and the received signal. After obtaining the delay N, the first N points of the received signal are discarded. The perfect delay estimation is done with the function *nrPerfectTimingEstimate*, by comparing the *pathGains* and *pathFilters* of the channel.

Demodulation: By using the function *nrOFDMDemodulate*, the time-domain received signal is demodulated and the time-frequency I-Q samples in the form of a two-dimensional matrix is reconstructed. Zero padding is used if the N points discarded during synchronization cause the reconstructed resource matrix to be incomplete at this step.

Channel estimation: Matlab 5G Toolbox uses the LS algorithm mentioned in Chapter 3 for channel estimation and the function used is *nrChannelEstimate*. First, the LS channel estimate $\hat{h}_r^{LS}$ is obtained by dividing the received reference signal by the original reference signal at the transmitter. Subsequently, spline interpolation is used in the frequency domain direction to obtain channel estimates for all the subcarriers in which there were no reference symbols to estimate the channel gain, and the channel impulse response in the time domain can be obtained using inverse discrete Fourier transform (IDFT). In the time domain, denoising can be performed by windowing function; at the same time, the noise power is estimated by measuring the variance between the original LS estimates and the same locations in the denoised estimate.

In order to implement perfect channel estimation, the function to be used is *nrPerfectChannelEstimate*. The channel impulse response is reconstructed by using information such as *pathGains* and *pathFilters* of the channel and demodulated based on the carrier information to finally obtain a perfect channel estimate.

Extract REs and Equalization: This is done using the functions *nrExtractResources* and *nrEqualizeMMSE*. In this step matlab uses MMSE as channel equalization algorithm.

Decoding: Finally, *nrPUCCHDecode* is used to get the PUCCH symbols and *nrUCIDecode* is used to get the UCI bits. The entire link-level simulation is complete.

After decoding, Block Error Rate (BLER) can be calculated to measure PUCCH Format 2 link-level performance. When running the simulation, the SNR value can be specified to vary over a range to obtain a curve of BLER with SNR. In the next subsection, the PUCCH Format 2 link-level performance curve in Matlab 5G Toolbox will be shown.

5.1.4 Link-level performance

To validate the Matlab 5G Toolbox, results in [2] will be used as a reference. When running the simulation, the parameter settings are kept consistent with those in [2]. The specific parameter settings are shown in the Table 5.2.

Table 5.2: Parameters for link-level simulation.

Parameters	Configuration
Carrier frequency	4 GHz
System bandwidth	20 MHz
Subcarrier spacing	15 kHz
Number of transmit antennas	1
Number of receive antennas	2
Channel model	TDL-C
Delay spread	1000 ns
Max Doppler shift	11 Hz (3 km/h)
Payload size (without CRC)	20 bits
Number of PRBs	4
Number of OFDM symbols	1
Number of UEs	1

The comparison of the results is shown in Figure 5.5. It can be observed that there is a roughly 2 dB gap between the results obtained by Matlab 5G toolbox compared to results reported in in [2]. The main reason for this gap is the difference in channel estimation algorithms, [2] uses the Minimum Mean-Square Error (MMSE) algorithm whereas the algorithm in Matlab is Least Square (LS).

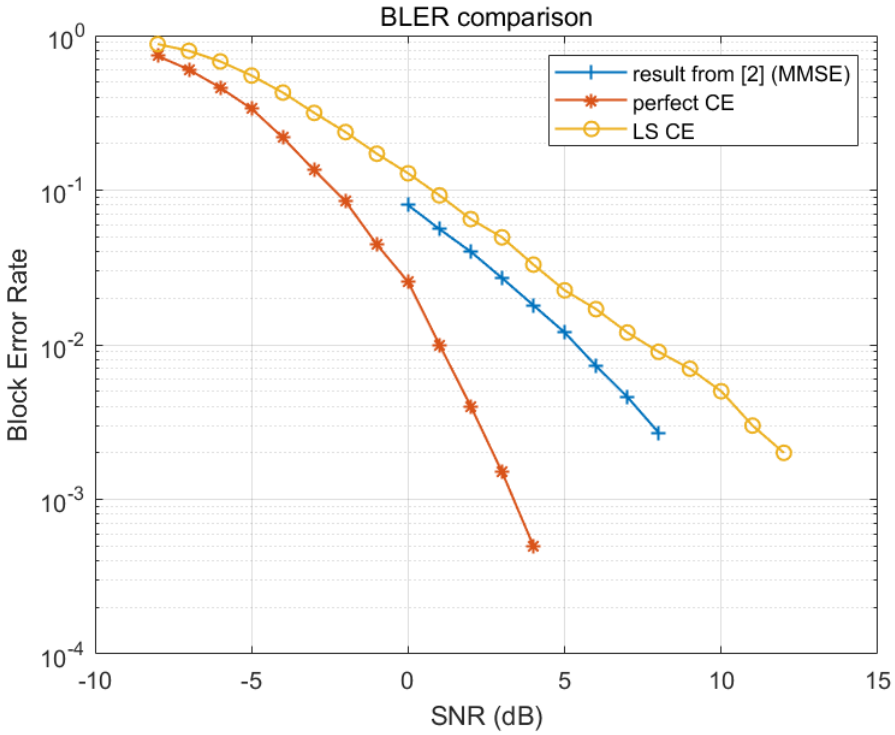


Figure 5.5: BLER performance comparison.

5.2 Convolutional Neural Network based channel estimation

From the previous background information, we can realize that the time-frequency two-dimensional matrix of a channel is like a picture, and the channel values at different time-frequency locations are like pixel points. Therefore, the processing of the channel time-frequency matrix can be transformed into an image processing problem. The channel estimation process mentioned in the previous section actually uses the known information on the Demodulation References Signals (DMRS) locations to derive information on the entire channel matrix. This is like restoring a high resolution image from a low resolution image. In fact, this is the idea behind the use of Convolutional Neural Network (CNN) to deal with the channel estimation problem in this thesis.

This idea has been mentioned and verified in other papers. In [10], a Super Resolution (SR) and Image Restoration (IR) network are created to implement Channel Estimation based on the network structure in [29] and [30]. [10] They used a cascade of two convolutional networks and named it *ChannelNet*. First, the SR network is used to process the $\hat{h}_r^{LS}$ obtained through the LS algorithm, and the information of the entire channel matrix is obtained through interpolation. The second network IR is then used to denoise the matrix obtained in the previous step. The denoised matrix is the channel response we want.

Due to the use of a cascade of two networks, the entire ChannelNet training process is also divided into two parts. In the first step, the actual channel response in channel matrix H is used as the training label to train the SR network by minimizing the error

between the interpolated output and H. Subsequently, the optimized SR network parameters are saved, and the real channel H is used as the training label again to minimize the error between the result after denoising through the IR network and H. Finally, after two-step training, the SR and IR networks are obtained and used to achieve channel estimation.

The results of [10] show that *ChannelNet* is highly competitive compared to the MMSE algorithm. This brings hope to the application of CNN-based algorithms in specific situations. Therefore, the model used in this thesis will be based on ChannelNet, and two improved models are proposed on this basis. In the next section, the three models used in the thesis will be introduced.

5.2.1 Structures of Neural Network models

We have three models:

1. **SRIR model**: The first model uses the same network structure as *ChannelNet*, that is, a cascade of SR network and IR network. For the SR network, its structure is shown in the Figure 5.6. In the structure diagram, the input layer receives the area of the PUCCH in the channel grid. In the convolutional layer, N represents the number of filters.

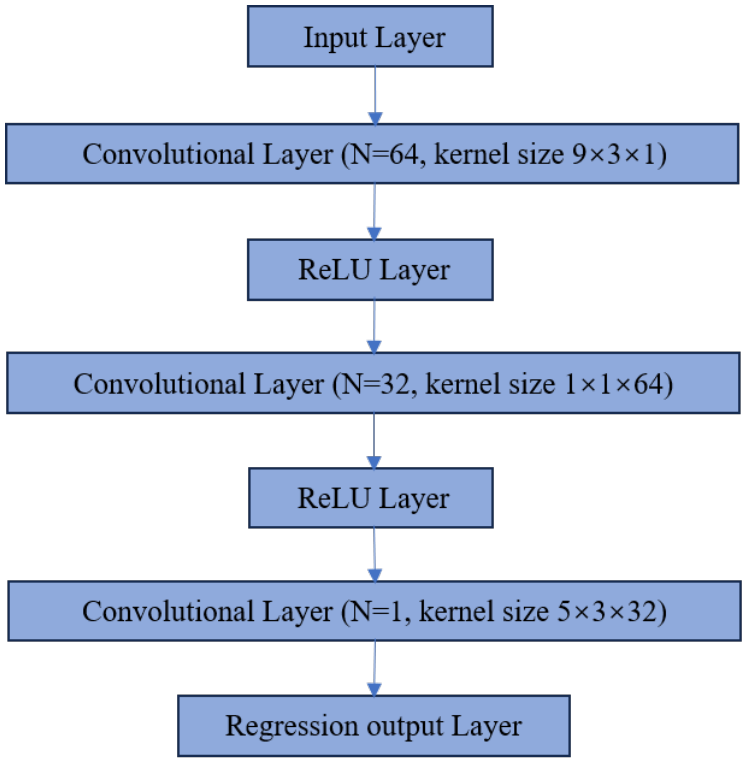


Figure 5.6: The structure of SR. Contains three convolutional layers followed by ReLU layer, and finally uses regression for output.

The structure of the second network IR is slightly more complex. It contains 20 convolutional layers, each layer contains 64 filters, and the size of each filter

is 3×3. The first 19 convolutional layers are followed by a ReLU layer, and the last convolutional layer is followed by the same regression layer in the SR network as the output. Due to the large number of layers, its network structure diagram is not explicitly shown in this thesis for the sake of brevity. From the overall structure, SR and IR are somewhat similar, but the number of layers of the IR network is much more than that of the SR network. In practice, more layers means that both training and prediction will take more time, which will most likely result in higher processing latency.

2. **SR model**: In [10], although the training of the network is divided into two steps, the actual channel H is used as the training label in both steps. This brings the possibility of using only one network. Letting a single network process two things at the same time may cause a loss of performance, but it will increase the processing speed. The key to the problem is how much performance cost is required to obtain the time gain. In order to explore this issue, the second model removes the IR network and only retains the SR network for interpolation and noise reduction. The network structure of the SR part is consistent with model 1.

3. **LI+SR model**: In the first two models, the input of the SR network is the channel $\hat{h}_r^{LS}$ estimated by LS. Since the LS algorithm can only obtain the information on the DMRS positions, other positions are padded with 0. Actually, in the non-DMRS region of PUCCH, there are PPUCCH symbols transmitted through the channel, which contain UCI information bits. Due to the inability to ascertain the effect that creates the propagation over the wireless channel on them, we cannot use the information from these PUCCH symbols during channel estimation. Only the information from DMRS symbols is known and reliable. If the interpolation algorithm is used to obtain rough information of PUCCH symbols after LS channel estimation, we can use the information of the entire PUCCH area and use it as the input of the SR network. The third model adopts this idea. Since the PUCCH Format 2 area is very narrow, this paper chooses the linear interpolation method. The details of data processing will be mentioned in the following sections. The structure of the SR network remains the same as before.

5.2.2 Implementation of Neural Network models

When using CNN for channel estimation, this thesis uses a different parameter configuration from the table5.2. Specifically, the carrier frequency is set to 3.5 GHz to align with the practical scenario of 5G mid-band. At the same time, for the sake of simplicity, the system adopts the 1×1 SISO system. In addition, a PUCCH configuration of 16 PRBs and 2 OFDM symbols is used to maximize the PUCCH area. The specific configuration parameters are shown in the Table 5.3.

The entire process of using CNN for channel estimation is shown in the Figure 5.7. The whole process is divided into two parts, reflected in the upper and lower layers of

Table 5.3: Parameters for CNN-based Channel estimation.

Parameters	Configuration
Carrier frequency	3.5 GHz
System bandwidth	20 MHz
Subcarrier spacing	15 kHz
Number of transmit antennas	1
Number of receive antennas	1
Channel model	TDL-A/B/C/D/E
Delay spread	30, 100, 300, 1000 ns
Max Doppler shift	10 Hz (3 km/h), 195 Hz (60 Km/h), 681 Hz (210 Km/h)
Payload size (without CRC)	80 bits
Number of PRBs	16
Number of OFDM symbols	2
Number of UEs	1

the figure. The upper layer of the figure describes the training process of the model, and the lower layer describes the process of application of the model after training. The process of the application is the same as that in Section 5.1.3.

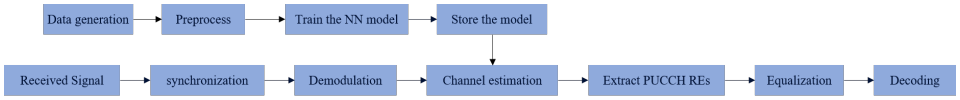


Figure 5.7: The entire process of using CNN for channel estimation.

The training process of the model is:

1. **Data generation**: Data generation is a very important step in the process of training a model. Its purpose is to generate DMRS data for model training. First, the original DMRS data is generated according to the DMRS processing flow mentioned in Section 3.3, and then the PUCCH symbols at other positions are generated according to the PUCCH Format 2 processing flow mentioned in Section 3.2. After modulating the channel matrix that contains all symbols into an OFDM signal, it is passed through a TDL channel and Gaussian white noise is added to simulate the process of signal transmission. It is worth noting that different data sets need to be generated for different channel models, and the CNN model trained after this will also be applied to that specific channel type. When

adding noise, the signal-to-noise ratio (SNR) is randomly selected between -10 dB and 20 dB to give the model some degree of generalization capability. The transmitted signal is then synchronized and demodulated according to the steps described in Section 5.1.3. Then the LS algorithm is used to get the preliminary channel estimate $\hat{h}_r^{LS}$, which is the training data for model 1 and model 2. For model 3, we need to perform frequency domain linear interpolation on the LS channel estimate $\hat{h}_r^{LS}$ to obtain $\hat{h}_r^{LILS}$, which serves as the training data. At the same time, *nrPerfectChannelEstimate* is used to obtain the real channel H as the training labels for all 3 models. The training data and training labels are the data we need. In the configuration specified in Table 5.3, the PUCCH occupies 16 PRBs, with each occupying 8 PRBs at the OFDM symbols in position 13 and 14. Therefore, the size of the PUCCH area is $(8 \times 12) \times 2 = 96 \times 2$. Where '96' represents PUCCH occupying 96 subcarriers in the frequency domain, and '2' signifies it occupies 2 OFDM symbols in the time domain. Since the DMRS of PUCCH Format 2 uses 1/3 overhead, for the training data $\hat{h}_r^{LS}$ in model 1 and 2, 32 of the 96 subcarriers in the frequency domain are DMRS complex-valued symbols, and the remaining positions are 0. For the training data $\hat{h}_r^{LILS}$ in model 3 and training label H, all 96 subcarriers in the frequency domain are filled with the complex values of the channel.

2. **Preprocess**: After obtaining the training data and training labels, it is necessary to preprocess them to meet the input requirements of the CNN. Since CNN can only handle real numbers, complex numbers in the training data and training labels need to be split. Taking the training label as an example, the original data is a complex matrix of size 96×2. The real and imaginary parts of the data are split and concatenate in the fourth dimension, forming $96 \times 2 \times 1 \times 2$. This is because CNN requires the input data format to be $96 \times 2 \times 1$. When the sample size of the training data is N, the data size after splitting is $96 \times 2 \times 1 \times 2N$, and in the last dimension, the first N are the real values of the channel data, the last N are the imaginary values. After the complex values are split, 80% of the entire training data set is used as training data and 20% is used as validation data.

3. **Train the Models**: The training of the model is completed in Matlab, and the function used is trainNetwork. Some parameters used during training are shown in the Table 5.4.

4. **Use the trained model for channel estimation**: After the model training is completed, it will be saved in a .mat type file. When applying the CNN model for channel estimation operations, by calling the mat type file and using the predict function, the estimation matrix of the PUCCH area channel can be obtained. Subsequently, extract PUCCH REs and perform equalization and decoding operations. Therefore, the CNN model replaces only the channel estimation part as shown in Figure 5.4. The performance results of using the CNN model for channel estimation will be presented in the next section.

Table 5.4: Parameters for training CNN Model.

Parameters	Configuration
Optimizer	Adam
Input shape	$96 \times 2 \times 1$
Initial Learn Rate	3×10^{-4}
Epochs	30
Batch size	16

5.3 Performance results

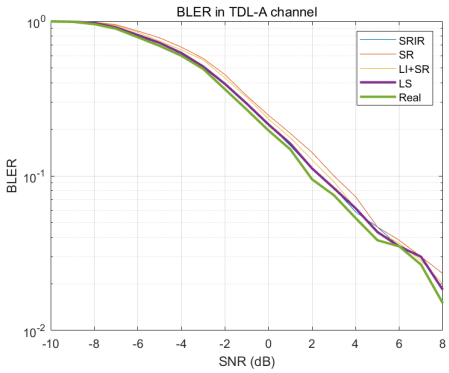
This section will show the difference in results between channel estimation using three CNN models and channel estimation using *nrChannelEstimate* for different channel conditions. The real channel information obtained by *nrPerfectChannelEstimate* will be used as a reference.

5.3.1 BLER performance

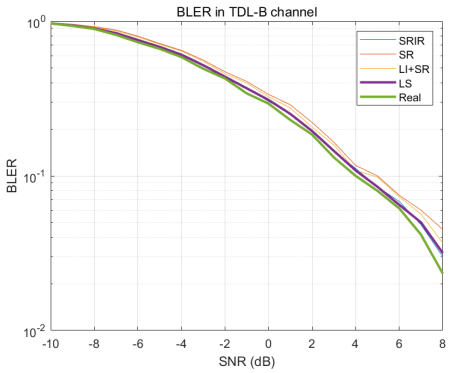
Among all the performance comparison images, BLER is the most intuitive image to observe the performance difference between different channel estimation algorithms. Typically, as the SNR increases, the BLER decreases. When the SNR is increased to a certain extent, it is likely that the presence of BLER will not be observed for a limited frame length. In order to obtain more accurate BLER values as much as possible at higher SNR values, a frame length of 60 frames was used for the simulation. Even so, the BLER value becomes inaccurate when the SNR value is above 8 dB. However, if the frame length is further increased, this again results in a significant increase in simulation time. Therefore the SNR range of the BLER performance plot is limited to -10 to 8 dB. For the non-line-of-sight channels TDL-A/B/C, the BLER performance curves obtained using different channel estimation algorithms are shown in Figure 5.8.

It can be observed that none of the three newly proposed algorithms show significant gain for both TDL-A and TDL-B channels. In both channel configurations, the LS algorithm already achieves very similar results to perfect channel estimation, so there is no space for CNN-based channel estimation algorithms to gain. As for the TDL-C channel, the channel conditions are very poor due to its high delay spread and high Doppler shift, so we can observe a gain of about 3 dB for the CNN-based algorithm compared to the LS algorithm. Overall, the performance of the three CNN-based models is very similar, and their training and prediction times will be shown later.

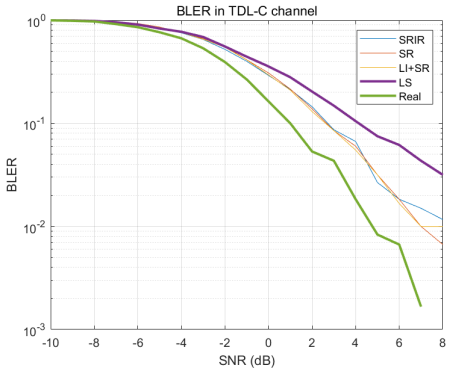
The line-of-sight channels are shown in Figure 5.8. Unfortunately, CNN-based algorithms perform poorly in both cases. In the TDL-D channel condition, Model 1 does not appear to have learned useful information from the data, resulting in its BLER curve failing to converge. And for models 2 and 3, their performance is also inferior to the LS algorithm. In the TDL-E case, all three models show no convergence.



(a) TDL-A

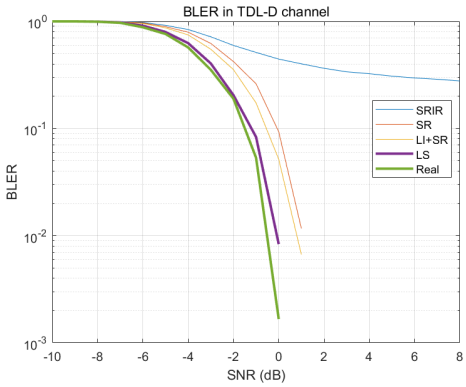


(b) TDL-B

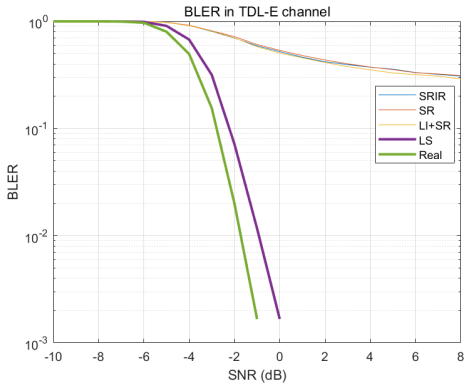


(c) TDL-C

Figure 5.8: BLER performance for NLOS channels.



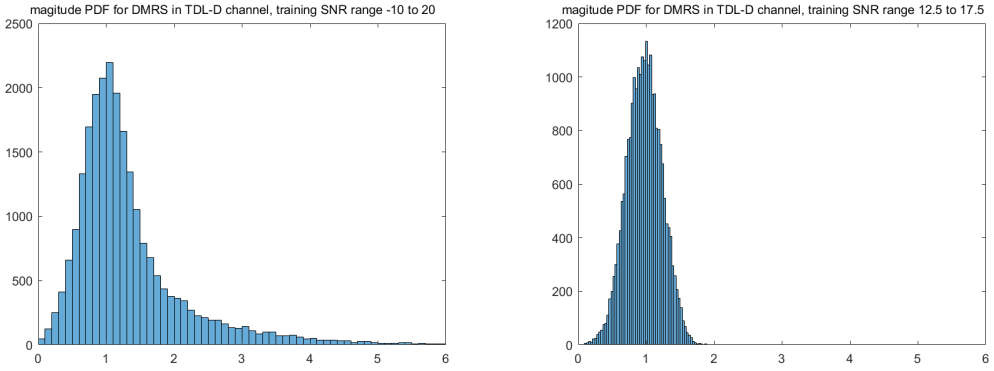
(a) TDL-D



(b) TDL-E

Figure 5.9: BLER performance for LOS channels.

To explore the reasons for the poor performance of the CNN algorithm in line-of-sight channel conditions, Figure 5.10 shows the distribution of DMRS magnitude values in the training data. Taking the TDL-D channel as an example, we can see that when the SNR of the training data ranges from -10 to 20, the data does not show the Rician distribution that should be expected for a line-of-sight channel, but instead, it is more characteristic of a Rayleigh channel. We used features from the Rayleigh channel to train the model and eventually applied it to the Rician channel, which is the main reason for the poor performance of the model. And when the SNR value of the training data is increased to between 12.5 and 17.5, we can observe that the data presents the Rician channel as it should be for the line-of-sight channel. It can be inferred that higher noise power causes changes in the channel distribution, which prevents the CNN from learning critical features in the data.



(a) PDF with training data SNR -10 to 20 (b) PDF with training data SNR 12.5 to 17.5

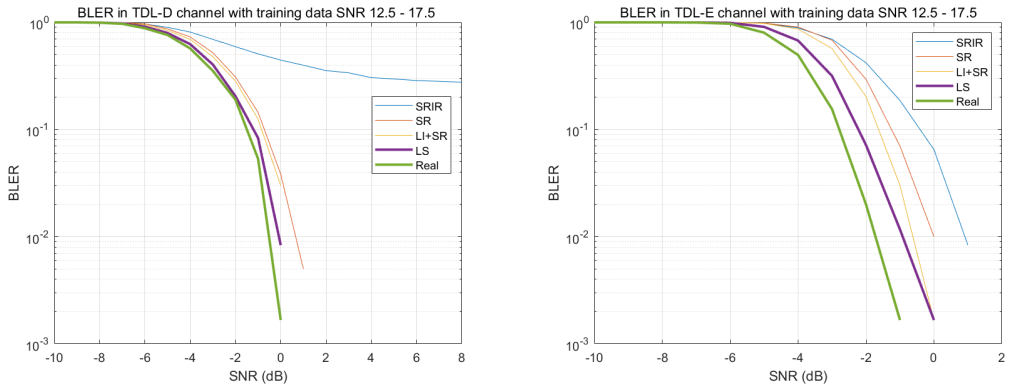
Figure 5.10: PDF of training data in LOS channel.

When the SNR of the training data ranges from 12.5 to 17.5, the BLER performance of the three models in the line-of-sight channel is shown in Figure 5.11. It can be observed that, for the TDL-D channel, model 1 still fails to converge, but the performance of models 2 and 3 is slightly improved. The improvement is more pronounced for the TDL-E channel, where all three models converge, yet they still do not outperform the traditional LS algorithm.

5.3.2 MSE on PUCCH positions

Although BLER is the most intuitive way to compare performance, due to the limitation of the number of simulation frames, we are unable to get the performance difference between different algorithms at higher SNR ranges. Therefore, we can directly compare the matrices after channel estimation, using the real channel matrix as a reference. The formula for measuring MAE is

$$\text{MAE} = \frac{|\hat{H} - H_{real}|}{|H_{real}|}. \tag{10}$$



(a) TDL-D (b) TDL-E

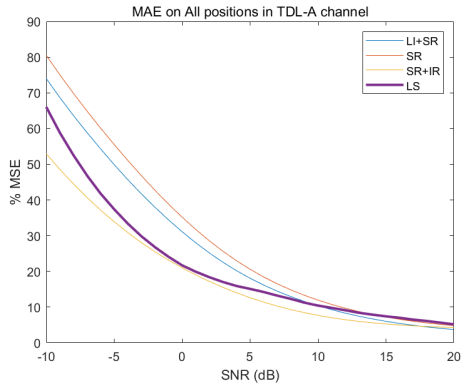
Figure 5.11: BLER performance for LOS channels with training data SNR 12.5 to 17.5.

For non-line-of-sight channels, the MAE performance comparison is shown in Figure 5.13. It can be seen that Model 1 (SR+IR) has the best performance under all three channel conditions and outperforms the conventional LS algorithm over the entire SNR range. For channels TDL-A and TDL-B, Models 2 and 3 exhibit a slight advantage over the LS algorithm at 10 dB and above. For the TDL-C channel, which has the worst channel conditions, all three models show a gain relative to the LS algorithm at around -4 dB, and the gain increases with increasing SNR. And after the SNR is greater than 0, models 2 and 3 perform better than model 1.

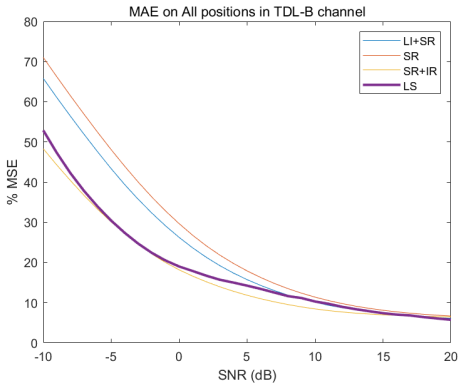
For line-of-sight channels, when SNR-10 to 20 dB are used as training data, the MAE performance of the model is compared in the Figure 5.13. It can be seen that the results are in line with previous BLER performance comparisons, where the three CNN-based models consistently perform worse than the LS algorithm. However, under TDL-D channel conditions, the gap between Models 2 as well as 3 and the LS algorithm is relatively small when the SNR reaches 10 dB and higher. In contrast, when SNR 12.5 to 17.5 dB is used for the training data, the results are shown in Figure 5.14. For the TDL-D channel, Models 2 and 3 show a very slight advantage after about 7 dB. For the TDL-E channel, only Model 3 shows a small advantage over the LS algorithm after about 6 dB.

5.3.3 Channel Magnitude Curve in Subcarrier Direction

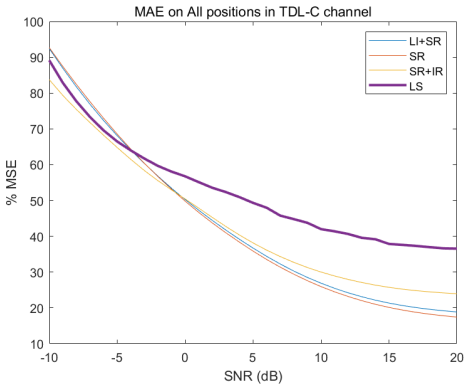
This subsection will show diagrams of the magnitude of the estimated channel obtained by different algorithms in the subcarrier direction compared to the real channel. In the time direction, the PUCCH region is not continuous from frame to frame. Therefore it is difficult to perform a comparison of the channel magnitude in the time domain over a period of time. For the NLOS channel, the resultant comparison graph is shown in Figure 5.15. For TDL-A and TDL-B channels, Model 1 achieves the best results, and the estimated channels obtained using Model 1 are able to maintain the same trend as the real channel over a long range. It also outperforms the LS algorithm even though



(a) TDL-A

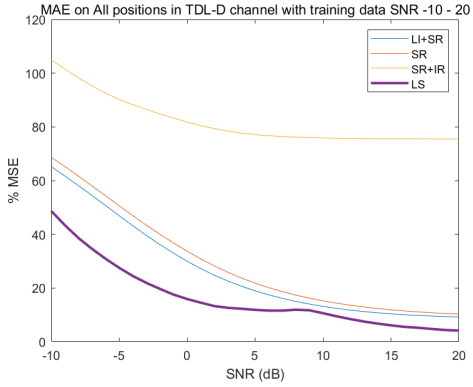


(b) TDL-B

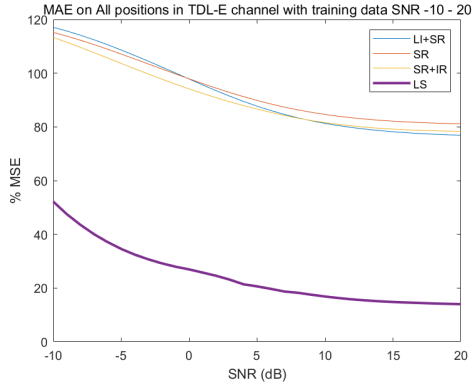


(c) TDL-C

Figure 5.12: MAE performance for NLOS channels.

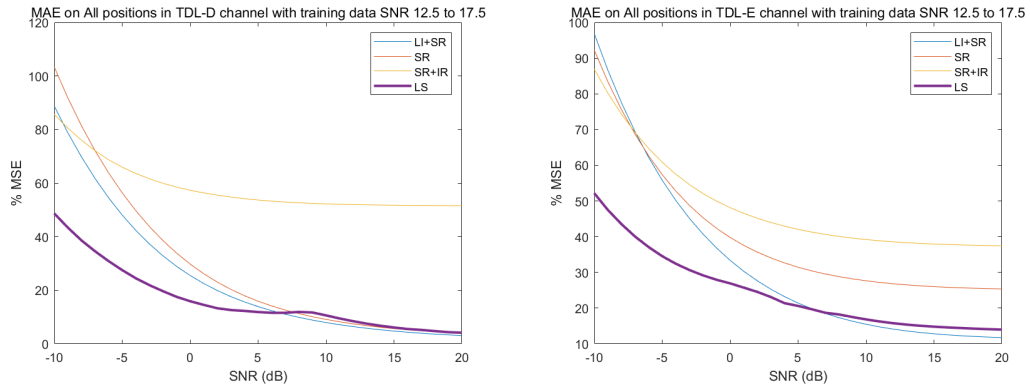


(a) TDL-D



(b) TDL-E

Figure 5.13: MAE performance for LOS channels with training data SNR -10 to 20 dB.



(a) TDL-D (b) TDL-E

Figure 5.14: MAE performance for LOS channels with training data SNR 12.5 to 17.5 dB.

Table 5.5: Running time for CNN Models (Model 2 as a baseline)

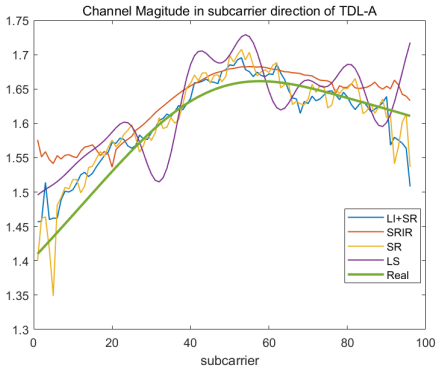
Model	Preprocess time	Prediction time
Model 1	27.2	1.9
Model 2	1	1
Model 3	254.6	1.2

it does not achieve a perfect match with the real channel. For the TDL-C channel, the three models have similar results, all of them give very similar results to the real channel and all of them outperform the results of the LS algorithm. For line-of-sight channels, the three models perform poorly.

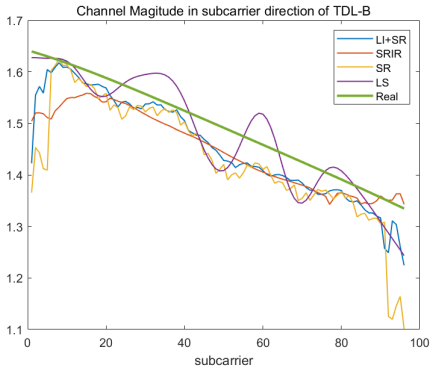
5.3.4 computational processing time comparison

The previous subsections presented the performance differences between the different algorithms and this subsection will present the differences in the running time. The differences will be provided by relative values. The running time of model 2 will be used as a benchmark. For model 3, performing linear interpolation in the preprocess stage will take a lot of time.

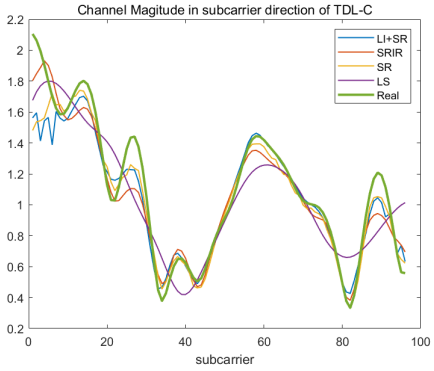
In this chapter, we show the Matlab 5G toolbox used in this thesis and the link-level Simulator built based on it. Subsequently, the three CNN-based channel estimation models proposed in this thesis are presented. Finally, the performance of the three models is compared with the traditional LS algorithm. In the next chapter, we will draw the conclusions of this thesis.



(a) TDL-A

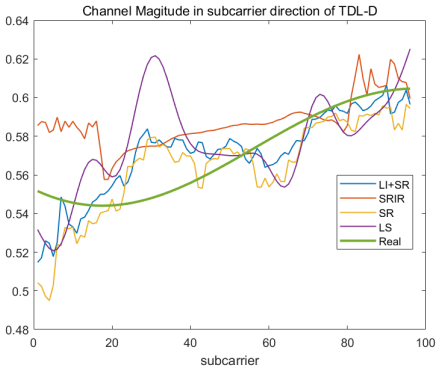


(b) TDL-B

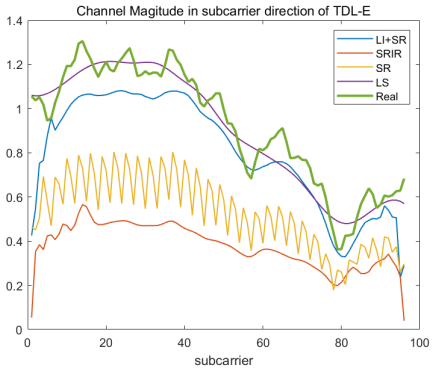


(c) TDL-C

Figure 5.15: Channel Magnitude Curve in Subcarrier Direction for NLOS channels.



(a) TDL-D



(b) TDL-E

Figure 5.16: Channel Magnitude Curve in Subcarrier Direction for LOS channels.

6 Conclusions

6.1 Summary

Without loss of generality, PUCCH Format 2 was studied in this thesis, including its DMRS allocation as well as the physical layer operations on the transceiver side. Based on this, we designed three different CNN based channel estimation algorithm. They use SR and IR networks, single SR network, linear interpolation with SR network respectively. The performance was compared with respect to state-of-the-art solutions based on LS algorithms. From the results, for the LoS channel (TDL-D/TDL-E), none of the three models are able to obtain gains. For NLoS channels, the three algorithms achieve at least the same performance as LS when the SNR is greater than 5 dB. Specifically, the three CNN-based algorithms can obtain a gain of 2-3 dB when the UE moves at 210 km/h, the delay spread is 1000 ns, and the channel type is a TDL-C channel. In conclusion, CNN-based channel estimation could give some notable gain when implemented in outdoor scenarios with high-mobility, such as high-speed trains. In the other kind of situations where mobility is minor and LoS propagation with limited delay spread exists, the performance gain with respect to LS-based state-of-the-art solutions does not justify the additional implementation cost that such an ML-approach approach would have. For the three models, the use of the SR network alone (model 2) not only yields similar performance to the other two algorithms, but is also far superior to the other two algorithms in terms of the time required for preprocessing and prediction, and is therefore recommended to be used with this model for channel estimation.

6.2 Future improvements

Since this study only considered the link performance in the 1×1 SISO system, in future work, the performance of CNN-based channel estimation algorithms in MIMO scenarios can be first considered for testing. It is also meaningful to consider in how to apply CNN-based algorithms in hardware and compare the complexity.

References

[1] Y. G. Li, L. J. Cimini and N. R. Sollenberger, "Robust channel estimation for OFDM systems with rapid dispersive fading channels", IEEE Trans. Commun., vol. 46, no. 7, pp. 902-915, July 1998

[2] L. Kundu, G. Xiong and J. Cho, "Physical Uplink Control Channel Design for 5G New Radio," in Proc. *IEEE 5G World Forum*, July 2018, pp. 233-238.

[3] T. H. Phuoc Nguyen, H. Nguyen and B. Khuc, "Performance Evaluation of Channel Estimation Methods for 5G NR Uplink Control Channel in the Scenario of Low Signal-to-Noise Ratios," in Proc. *International Conference on Advanced Technologies for Communications*, 2022, pp. 18-22.

[4] H. Ye, G. Y. Li and B. -H. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," in IEEE Wireless Communications Letters, vol. 7, no. 1, pp. 114-117, Feb. 2018.

[5] Neev Samuel, Tzvi Diskin and Ami Wiesel, "Deep mimo detection", in Proc. *IEEE International Workshop on Signal Processing Advances in Wireless Communications*, July 2017, pp. 1-5.

[6] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer", IEEE Trans. Cogn. Commun. Netw., vol. 3, no. 4, pp. 563-575, Dec. 2017.

[7] D. Erdogmus, D. Rende, J. C. Principe and T. F. Wong, "Nonlinear channel equalization using multilayer perceptrons with information-theoretic criterion", in Proc. *Neural Networks for Signal Processing XI: Proceedings of the 2001 IEEE Signal Processing Society Workshop*, Sept. 2001, pp. 443-451.

[8] C.-K. Wen, W.-T. Shih and S. Jin, "Deep learning for massive MIMO csi feedback", in IEEE Wireless Communications Letters, vol. 7, no. 5, pp. 748-751, Oct. 2018.

[9] H. He, C.-K. Wen, S. Jin and G. Y. Li, "Deep learning-based channel estimation for beamspace mmwave massive MIMO systems", in IEEE Wireless Communications Letters, vol. 7, no. 5, pp. 852-855, Oct. 2018.

[10] M. Soltani, V. Pourahmadi, A. Mirzaei and H. Sheikhzadeh, "Deep Learning-Based Channel Estimation," in IEEE Communications Letters, vol. 23, no. 4, pp. 652-655, Apr. 2019.

[11] A. K. Yerrapragada, J. K. S, A. Gautam and R. K. Ganti, "Machine Learning Decoder for 5G NR PUCCH Format 0," in Proc.*National Conference on Communications*, Feb. 2023, pp. 1-6.

[12] ITU-R, “IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond,” Recommendation ITU-R M.2083-0, International Telecommunication Union - Radiocommunication Sector (ITU-R),09 2015.

[13] H. Holma, A. Toskala, and T. Nakamura, 5G Technology: 3GPP New Radio. John Wiley & Sons Ltd., 2020. Editors: Harri Holma and Antti Toskala and Takehiro Nakamura.

[14] A. Bateman, D. M. Haines, and R. J. Wilkinson, “Linear transceiver architectures,” in 38th IEEE Vehicular Technology Conference, pp. 478–484, Jun 1988.

[15] H. G. Myung, J. Lim and D. J. Goodman, "Single carrier FDMA for uplink wireless transmission," in IEEE Vehicular Technology Magazine, vol. 1, no. 3, pp. 30-38, Sept. 2006.

[16] Xu, Chao, Naoki Ishikawa, Rakshith Rajashekar, Shinya Sugiura, Robert Maunder, Zhaocheng Wang, Lie-Liang Yang, and L. Hanzo. "Sixty Years of Coherent Versus Non-Coherent Tradeoffs and the Road From 5G to Wireless Futures". IEEE Access, vol. 7, pp. 178246-178299, Dec. 2019.

[17] Dahlman, E., Parkvall, S., and Skold, J. 5G NR: The next generation wireless access technology. Academic Press, 2018.

[18] 3GPP, “Technical Specification Group Radio Access Network;NR;Physical channels and modulation (Release 16),” Technical Specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), 10 2021. Version V16.7.0

[19] 3GPP, “Technical Specification Group Radio Access Network;NR;Multiplexing and channel coding (Release 16),” Technical Specification (TS) 38.212, 3rd Generation Partnership Project (3GPP), 07 2020. Version V16.2.0

[20] Li, Bo & Yang, Hongjuan & Liu, Gong-Liang & Peng, Xiyuan. (2017). "A new joint channel equalization and estimation algorithm for underwater acoustic channels." EURASIP Journal on Wireless Communications and Networking. pp. 315-322, Jan. 2019.

[21] D. Neumann, T. Wiese and W. Utschick, "Learning the MMSE Channel Estimator," in IEEE Transactions on Signal Processing, vol. 66, no. 11, pp. 2905-2917, June 2018.

[22] A. Jung,“Machine Learning: The Basics,” Springer, Singapore, 2022

[23] J. Patterson and A. Gibson, Deep Learning. O’Reilly Media, Inc., 2017. ISBN: 9781491914250.

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

[25] 3GPP, “Technical Report Group Radio Access Network;NR;Study on channel model for frequencies from 0.5 to 100 GHz (Release 16),” Technical Report (TR) 38.901, 3rd Generation Partnership Project (3GPP), 11 2020. Version V16.1.0

[26] S. Ahmadi, 5G NR: Architecture, Technology, Implementation, and Operation of 3GPP New Radio Standards. Academic Press, 2019. Academic Press is an imprint of Elsevier.

[27] 3GPP, “3GPP TSG RAN WG1 Ad Hoc meeting for Channel Model R1-161748” Technical Specification Groups(TSG) RAN, 3rd Generation Partnership Project (3GPP), 03 2016.

[28] 3GPP, “3GPP TSG RAN WG1 MeetingR1-162959” Technical Specification Groups(TSG) RAN, 3rd Generation Partnership Project (3GPP), 04 2016.

[29] C. Dong, C. C. Loy, K. He and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks," *arXiv preprint arXiv:1501.00092* (2015)

[30] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising,” *arXiv preprint arXiv:1608.03981* (2017)