Channel Estimation Optimization in 5G New Radio using Convolutional Neural Networks

Kanalestimeringsoptimering i 5G NR med konvolutionellt neuralt nätverk

David Adolfsson <adodavid@hotmail.com>

Faculty of Health, Science and Technology

Master thesis in Computer Science

Second Cycle, 30 hp (ECTS)

Supervisor: Karl-Johan Grinnemo, University of Karlstad, Karlstad, SWE <Karl-Johan.Grinnemo@kau.se>

Examiner: Giuseppe Caso, University of Karlstad, Karlstad, SWE <giuseppe.caso@kau.se>

Karlstad, June 13th, 2023

Abstract

Channel estimation is the process of understanding and analyzing the wireless communication channel's properties. It helps optimize data transmission by providing essential information for adjusting encoding and decoding parameters. This thesis explores using a Convolutional Neural Network (CNN) for channel estimation in the 5G Link Level Simulator, 5G-LLS, developed by Tietoevry. The objectives were to create a Python framework for channel estimation experimentation and to evaluate CNN's performance compared to the conventional algorithms Least Squares (LS), Minimum Mean Square Error (MMSE) and Linear Minimum Mean Square Error (LMMSE). Two distinct channel model scenarios were investigated in this study.

The results from the study suggest that CNN outperforms LMMSE, LS, and MMSE regarding Mean Squared Error (MSE) for both channel models, with LMMSE at second place. It managed to lower to the MSE by 85% compared to the LMMSE for the correlated channel and 78% for the flat fading channel. In terms of the overall system-level performance, as measured by Bit-Error Rate (BER), the CNN only managed to outperform LS and MMSE. The CNN and the LMMSE yielded similar results. This was due to that the LMMSE's MSE was still good enough to demodulate the symbols for the QPSK modulation scheme correctly.

The insights in this thesis work enables Tietoevry to implement more machine learning algorithms and further develop channel estimation in 5G telecommunications and wireless communication networks through experiments in 5G-LLS. Given that the CNN did not increase the performance of the communication system, future studies should test a broader range of channel models and consider more complex modulation schemes. Also, studying other and more advanced machine learning techniques than CNN is an avenue for future research.

Keywords

Channel Estimation, 5G, 5G NR, MIMO-OFDM, CNN, Machine Learning, LS, LMMSE, MMSE, DMRS, QPSK, Deep Learning, SNR, BER

Sammanfattning

Kanalestimering är en process i trådlösa kommunikationssystem som handlar om att analysera och förstå det trådlösa mediumets egenskaper. Genom effektiv kanalestimering kan dataöverföringen optimeras genom att anpassa signalen efter den trådlösa kanalen. Detta arbete utforskar användningen av ett konvolutionellt neuralt nätverk (CNN) för kanalestimering i Tietoevrys 5G-datalänkslagersimulator (5G-LLS). Målen är att (1) skapa ett Python-ramverk för kanalestimeringsexperiment samt att (2) utvärdera CNN:s prestanda jämfört med konventionella algoritmerna minsta kvadratmetoden (LS), minimalt medelkvadratsfel (MMSE) och linjärt minimalt medelkvadratsfel (LMMSE). Två olika kanalmodellsituationer undersöks i detta arbete.

Resultaten visar att CNN överträffar LMMSE, LS och MMSE i form av medelkvadratisk fel (MSE) för båda kanalmodellerna, med LMMSE på andra plats. CNN:n lyckades minska MSE:n med 85% jämfört med LMMSE för den korrelerade kanalen och med 78% för den snabbt dämpande kanalen. Vad gäller systemnivåprestanda, mätt med hjälp av bitfelsfrekvens (BER), lyckades CNN endast överträffa LS och MMSE. CNN och LMMSE gav liknande resultat. Detta beror på att LMMSE:s MSE fortfarande var tillräckligt låg för att korrekt demodulera symbolerna för QPSK-modulationsschemat.

Resultatet från detta examensarbete möjliggör för Tietoevry att implementera fler maskininlärningsalgoritmer och vidareutveckla kanalestimering inom 5G-telekommunikation och trådlösa kommunikationsnätverk genom experiment i 5G-LLS. Med tanke på att CNN inte överträffade samtliga kanalestimeringstekniker bör framtida studier testa ett bredare utbud av kanalmodeller och överväga mer komplexa moduleringsscheman. Framtida arbeten bör även utforska fler och mer avancerade maskininlärningsalgoritmer än CNN.

Nyckelord

Kanalestimering, MIMO-OFDM, CNN, Maskininlärning, 5g NR, LS, MMSE, LMMSE, DMRS, QPSK, Djupinlärning

Acknowledgments

Firstly, I would like to express my deepest gratitude to my thesis supervisor, Karl-Johan Grinnemo at Karlstad University, for his invaluable guidance, support, and expertise throughout this research. Their insightful feedback and constructive suggestions have greatly contributed to the success of this thesis.

I would also like to extend my sincere appreciation to my TietoEvry supervisor, Bengt Hallinger, for providing me with the opportunity to work on this project and for their education and guidance in the field of 5G telecommunications and the 5G Link Level Simulator. Their knowledge and expertise have been instrumental in shaping my understanding and enhancing the quality of this work. I am also grateful to the entire team at TietoEvry for their support and cooperation during the course of this research.

I am truly grateful for the guidance and education provided by my supervisors, as well as the support from my loved ones. Their contributions have been crucial for this thesis.

Acronyms

NR	New Radio
DL-SCH	Downlink Shared Channel
PD-SCH	Physical Downlink Shared Channel
PBCH	Physical Broadcast Channel
PDCCH	Physical Downlink Control Channel
CSI	Channel State Information
LS	Least Squares
MMSE	Minimum Mean Square Error
LMSSE	Linear Minimum Mean Square Error
AWGN	Additive White Gaussian Noise
5G-LLS	5G Link Layer Simulator
MIMO	Multiple-Input Multiple-Output
SISO	Single-Input Single-Output
SNR	Signal-to-Noise Ratio
DMRS	Demodulation Reference Signals
OFDM	Orthogonal Frequency Division Multiplexing
QPSK	Quadrature Phase-Shift Keying
CS	Compressed Sensing
SER	Symbol-Error Rate
QAM	Quadrature Amplitude Modulation
CB	Code Block
CRC	Cyclic Redundancy Check
AI	Artificial Intelligence
CNN	Convolutional Neural Networks
RNN	Recurrant Neural Network
ONNX	Open Neural Network Exchange
SGD	Stochastic Gradient Descent
RNN	Recurrent Neural Network
Bi-LSTM	Bidirectional Long Short-Term Memory
ELM	Extreme Learning Machine
BER	Bit-Error Rate
UE	User Equipment
BS	Base Station
MSE	Mean Square Error

Contents

Chapter 1

Introduction

5G New Radio (NR) is the fifth generation of wireless mobile technology. The demand for enhanced capabilities offered by 5G NR has been fueled by rapid technological advancements and the increasing need for high-speed connectivity. 5G NR is expected to provide faster data transfer speeds, lower latency, and increased capacity for connected devices compared to previous generations, enabling a wide range of new applications and use cases, such as remote surgery, autonomous vehicles, and smart cities. It is designed to accommodate the growing number of IoT devices, with predictions indicating that the number of such devices alone will reach 75 billion by 2025 [8]. 5G NR aims to serve both machine-to-machine communication and people, striving to deliver a seamless and responsive experience for all. These advancements are made possible through collaboration among various underlying technologies, one of the being *channel estimation*.

Channel estimation is a principal component for ensuring a reliable and efficient communication between base station antennas and user devices. The characteristics of the channel are estimated through the use of Demodulation Reference Signals (DMRS), a signaling sequence known both to transmitter and receiver. Using this, the receiver is able to demodulate the signals and establish the Channel State Information (CSI). Three widely used techniques for this are Least Squares (LS), Minimum Mean Square Error (MMSE) and Linear Minimum Mean Square Error (LMSSE). These techniques rely on assumptions about the channel which may not capture the complexity of real-world scenarios. While sufficient, research have showed promising results incorporating Deep Learning methods in other modules of the 5G physical layer [25, 28, 41]. Tietoevry therefore aims to investigate the potential of machine learning and Artificial Intelligence (AI) approaches in their 5G Link Layer Simulator (5G-LLS) to enhance the performance of channel estimation and further improve communication reliability and efficiency. This however depends on an accurate mathematical model of the channel for it to be applicable.

1.1 Background

Tietoevry has developed 5G-LLS to enable smooth and flexible experimentation with link layer and physical layer functionality. The simulator encompasses a wide range of techniques such as channel estimation, channel coding, modulation, MIMO settings, and more. It can be effortlessly extended and customized to optimize and evaluate various link layer and underlying functionalities.

Machine Learning and AI adaptions has recently seen a surge in academic and industrial wireless communication contexts (signal decoding, radio resource allocation and channel estimation, etc) [**2**]. The channel estimation module within the simulator currently only uses conventional algorithms. Tietoevry has offered this thesis to incorporate machine learning and AI methods into 5G-LLS to determine whether it is possible to make an improved channel estimator based on these techniques.

1.2 Problem Description

The current channel estimation module in Tietoevry's 5G Link Layer Simulator relies on conventional algorithms, which may not fully exploit the potential performance improvements offered by machine learning and AI techniques. As these new techniques have shown promising results in various wireless communication contexts, Tietoevry sees a need to investigate their applicability and effectiveness in enhancing channel estimation within the 5G-LLS. This thesis aims to address the following research question: Can machine learning and artificial intelligence methods be incorporated into the channel estimation module of the 5G Link Layer Simulator to improve its performance, and if so, what are the key factors influencing their success?

To answer this question, this thesis' purpose is to implement a framework for Tietoevry in which they can easily build, test and export models to the 5G-LLS. 5G-LLS is implemented in Matlab, though Tietoevry has requested that the framework leverages open source machine learning and AI frameworks for building the models. Furthermore, after implementation, the exported models are evaluated and tested against pre-implemented conventional channel estimation methods.

1.3 Thesis Objective

This thesis' purpose is to examine the potential benefits of using machine learning models to improve the accuracy over conventional channel estimation techniques. Evaluation and testing is done in Tietoevry's 5G-LLS.

1.4 Thesis Goals

The overall objective for this thesis is to implement machine learning models in Python to leverage open source machine learning frameworks and export them to Tietoevry's testbed running 5G-LLS for evaluation. In particular, the goals with this thesis work are to:

1. Understand the purpose and workings of conventional channel estimation techniques
2. Explore related work for current research in optimizing channel estimation
3. Build an Python program to train and export models to 5G-LLS simulator
4. Train, test and tune models
5. Evaluate the performance of the models in terms of BER over SNR and MSE over SNR.

1.5 Ethics and Sustainability

The workings of this thesis in no way affect privacy or security of user data, nor does it introduce biases to usage scenarios and/or user groups. In terms of sustainability, 5G NR networks improves on previous generations in terms of energy efficiency, environmental and societal impact. 5G NR has been designed with energy efficiency in mind. On top of that, accurate channel estimation reduces the need for excessive retransmissions and signal amplification, resulting in lower energy consumption and increased network capacity. Lastly, an accurate channel enables a more reliable network, essential for everything from remote surgery to smart cities[13].

1.6 Methodology

The methodology in this thesis project has been separated into different phases and uses mixed methods.

1. **Phase 1:** Researching existing algorithms and approaches for channel estimation and familiarizing with relevant 5G modules.
2. **Phase 2:** Finding related work to explore machine learning techniques and identify promising results.
3. **Phase 3:** Implementing suitable algorithms and integrating the models into 5G-LLS simulator.
4. **Phase 4:** Testing and tuning the models using Tietoevry's test-bed.

The overall methodology used in this study aims to provide a comprehensive and rigorous analysis of the potential benefits of using machine learning for channel

estimation in 5G systems.

1.7 Stakeholders

This thesis project was offered by and is conducted under Tietoevry's supervision. Tietoevry sought to determine the whether utilizing machine learning techniques for channel estimation could achieve an improved performance in terms of Bit-Error-Rate over SNR. A positive result would enable the use of more advanced modulation schemes, as the receiver will receive a more fine-grained signal to read from. All in all, it enables more robust signal and more data transmitted for a given bandwidth.

1.8 Delimitations

The main focus of this project is to provide Tietoevry with an open source alternative to test and simulate machine learning approaches to deal with channel estimation. The work then implements a machine learning model to be compared with existing algorithmic methods. The work should be compatible with Tietoevry's 5G-LLS simulator. While the simulator provides a valuable platform for conducting controlled experiments, it is important to acknowledge that the simulation environment may not fully replicate the complexities and nuances of a real-world communication system. The results and conclusions drawn from this study may not directly apply to other channel models with different characteristics. Furthermore, It should be noted that the effectiveness of the machine learning model and conventional algorithms for channel estimation may vary for different modulation schemes, and that the results obtained in this study are specific to the chosen modulation scheme.

1.9 Outline

This thesis is structured as follows. In Chapter 2, the necessary background information for 5G NR, channel estimation, and machine learning is presented, accompanied with related work in the field. Chapter 3 describes the design of the machine learning methodology that was used to address the problem. Chapter 4 describes how the simulations were carried out along with what metrics are being measured. Chapter 5 displays the result and presents the main findings of the work. Lastly, chapter 6 concludes the work by providing a summary and brief discussion of the work.

Chapter 2

Background

This chapter aims to establish the theoretical foundation for the thesis work. It begins with a description of 5G networks in 2.1, providing an overview of its functioning and key components. Subsequently, 2.2 delves into the topic of channel estimation, discussing its importance and various techniques employed in estimating the characteristics of wireless communication channels. Finally, 2.3 offers a background on different methodologies in Machine Learning and AI, highlighting their relevance and potential application in improving channel estimation.

2.1 5G NR and Mobile Networks

The rapid evolution of mobile wireless technology has witnessed significant advancements over the years, with each generation pushing the boundaries of connectivity and communication. In this context, 5G NR emerges as the latest milestone, surpassing its predecessors such as 4G LTE and 3G UMTS. Designed to meet the escalating demands for enhanced quality in wireless communication, 5G technology integrates various cutting-edge technologies and techniques. Multiple-Input Multiple-Output (MIMO), mmWave communication, and the Orthogonal Frequency Division Multiplexing (OFDM) technique describe a small part of a broader set of technologies cooperating to fulfill users' increasing requirements [1]. This section provides an overview of the physical layer in 5G, including the downlink physical channels and the 5G NR processing chain. It also highlights the significance of accurate channel estimation and resource mapping in enabling advanced modulation schemes and efficient resource allocation within the resource grid.

2.1.1 Physical Layer

The physical layer is responsible for transmitting and receiving data over the wireless medium. To do so reliably and efficiently, it is required to incorporate several functions and processes. These include modulation and demodulation, coding and decoding, scrambling and descrambling, resource mapping, MIMO processing, synchronization,

channel estimation, and power control. These processes are part of the 5G NR processing chain, which includes encoding operations at the transmitter Base Station (BS) and decoding operations at the receiver User Equipment (UE).

A Downlink Physical Channel is responsible for transmitting user data, control information, or system information from higher layers in the network protocol stack to the UE. Three downlink physical channels are defined and mapped to different resource elements, conveying information from higher layers. These are:

- Physical Downlink Shared Channel (PD-SCH)
- Physical Broadcast Channel (PBCH)
- Physical Downlink Control Channel (PDCCH)

User data and control information from higher layers are mapped to the physical layer's Downlink Shared Channel (DL-SCH), which through a series of operations and procedures, are mapped to the PD-SCH, as described in subsection 2.1.2. PBCH carries essential system information, such as cell identity, system frame number, and other parameters necessary for the UE to access and synchronize with the network. PDCCH carries control information, such as scheduling assignments, resource allocation, and power control commands.

2.1.2 Processing Chain

The transmission operations and procedures between the BS and the UE at the physical layer can be described by the 5G NR Processing Chain in the physical layer, displayed in Figure 2.1.1. It encompasses a series of important steps that ensure efficient and reliable data delivery. Starting with the Transport Block (TB), which represents the data exchanged between the 5G MAC layer, the processing chain involves several key operations, all explained in the list below [3].

- The Transport Block is the data delivered to and from the 5G MAC layer
- Cyclic Redundancy Check (CRC) creation
- Segmentation of Transport Blocks into Code Block (CB)s and CRC attachment
- Adds Error Correction Coding depending on CB length
- Rate Matching selects the coded bits that will be transmitted
- Interleaving alters the input sequence to distribute potential errors instead of getting bursts of them
- Concatenates all selected CBs and scrambles the data
- Converts bits to symbols by modulation
- Layer Mapping converts a stream of symbols destined for a single UE to parallel MIMO channels

- Performing beamforming and precoding by applying a precoding matrix to adjust amplitude and phase to each antenna
- Maps precoder data to each antenna's resource blocks
- Generate OFDM signals and adding a cyclic prefix

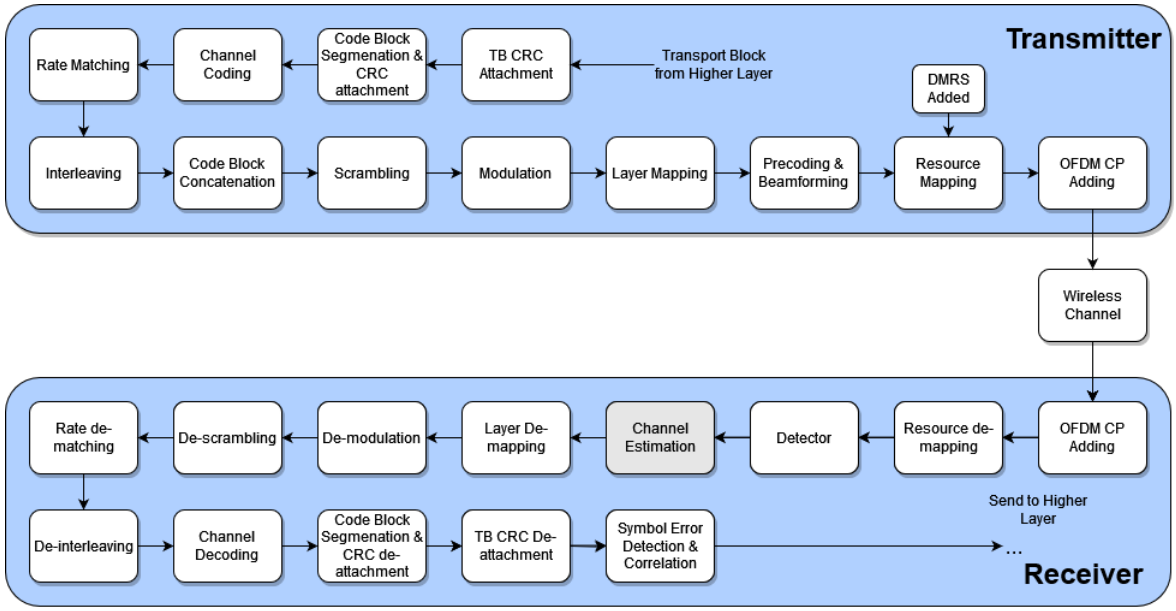

Figure 2.1.1: The 5G NR Process Chain

The process is reversed at the receiver.

The key points in the processing chain affecting channel estimation are the Modulation and Resource Mapping boxes in Figure 2.1.1. Modulation is the process of encoding the digital information to an analog signal. An advanced modulation scheme enables denser bit patterns within a constellation graph, which allows faster data rates. Figure 2.1.2 gives an example of three different modulation schemes, Quadrature Phase-Shift Keying (QPSK), 16-Quadrature Amplitude Modulation (QAM), and 64-QAM, where the integers represent states. As the number of points in the constellation increases, the modulation scheme can transmit more bits per symbol, resulting in higher data rates. The distance from the origin represents the amplitude whereas the angle represents the phase.

These schemes exemplify the varying levels of complexity and data capacity offered by different modulation techniques in 5G NR systems, and why accurate channel estimation is crucial for more advanced modulation schemes [1].

Resource mapping assigns data and control symbols to the physical resources. It fills the resource grid with information, including adding the DMRS symbols.

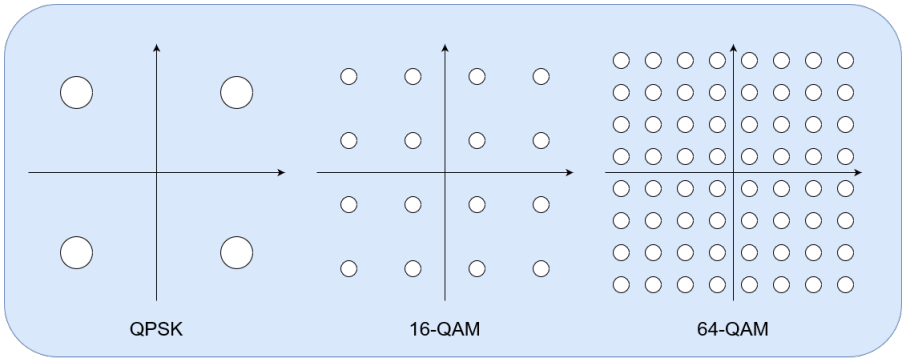

Figure 2.1.2: Constellation Graph with QPSK, 16-QAM, and 64-QAM.

2.1.3 Resource Grid

A resource grid in 5G NR is characterized by a subframe within a frame. A 10-ms frame contains ten subframes of 1 ms each for a defined subcarrier spacing - the duration of each OFDM slot is dependent on the subcarrier spacing. A resource grid in the subframe contains a horizontal axis representing the time domain and a vertical axis representing the frequency domain, each vertical tick being a different subcarrier. The time domain comprises 14 OFDM symbols, where each tick corresponds to one OFDM symbol. The resource grid can be seen in Figure 2.1.3.

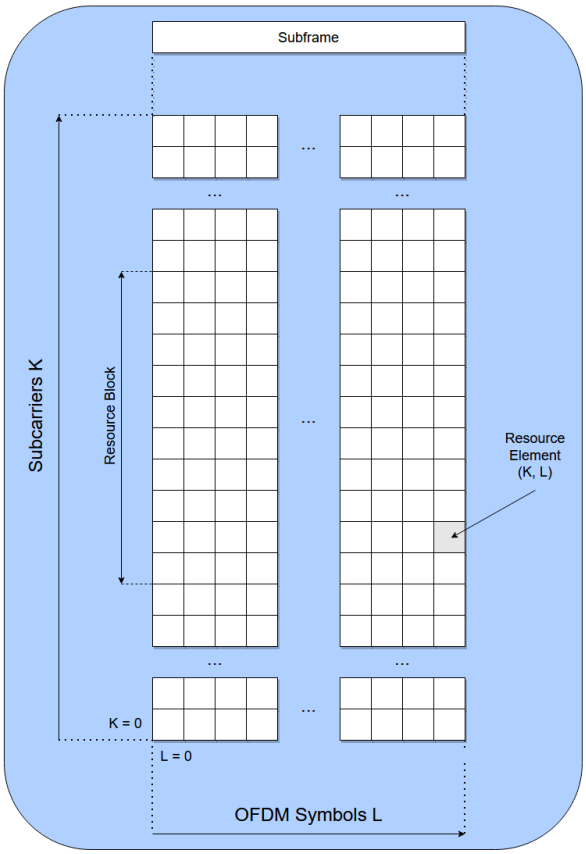

Figure 2.1.3: 5G NR Resource Grid

A resource grid consists of multiple resource blocks. Each block in the resource grid

consists of 12 consecutive resource elements in the frequency domain and one OFDM symbol in the time domain and represents the smallest resource granted to a single UE. One resource element is defined as one subcarrier during one OFDM symbol. It denotes the smallest unit of the available resources in the system that can be used to carry information [2].

2.2 Channel Estimation

Channel estimation is a principal component for ensuring reliable and efficient communication between base station antennas and user devices. The characteristics of the wireless medium typically distort a user device's received signal. It requires channel estimation to compensate for the distortions and correctly interpret the data. For example, an incoming signal's phase depends on the receiver's position. A change of the location of the UE that shifts the phase 180 degrees would flip the bits 00 into 11 for a QPSK modulation [17]. By comparing the received symbols with the reference signal, all phase shifts can be compensated for [17].

The characterizing steps for retrieving the best estimate of the channel are as follows:

- Pilot signal DMRS transmission and reception
- Channel estimation at pilot locations
- Resource grid interpolation based on the channel estimation

2.2.1 Demodulation Reference Signals

The channels' characteristics are estimated using DMRS, also known as pilot subcarriers, a signaling sequence known to the transmitter and receiver. The receiver uses this reference to compensate for demodulating the signal accurately. The DMRS symbols estimate the frequency response at predefined locations in the resource grid. Establishing an estimate across all pilot subcarriers makes it possible to interpolate the channel's behavior at other time-frequency locations. The result is a channel matrix of the same size as the resource grid, which is used to equalize the effects of the channel [5].

In 5G NR systems, the positioning of DMRS symbols can vary based on different mapping types. These mapping types, such as type A, type B, and type C, offer distinct advantages in terms of flexibility, resource utilization, and channel estimation performance [5].

As channel estimation produces much overhead, pilot signals are typically not inserted into every subcarrier. Instead, they are interpolated from the computed estimates [9]. In 5G-LLS, type A single-symbol DMRS structure with four OFDM symbols are used,

as showcased in Figure 2.2.1, which provides a balance between resource allocation and channel estimation performance.

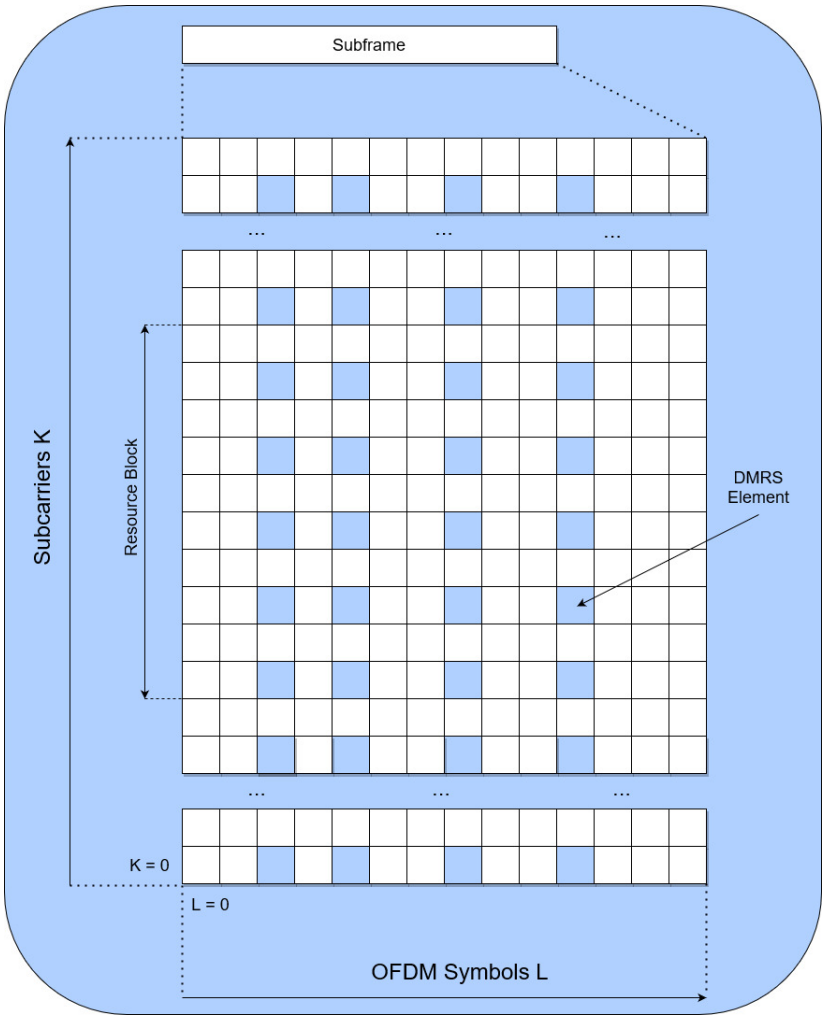

Figure 2.2.1: DMRS in the Resource Grid

2.2.2 The Channel Estimation Procedure

The channel estimation is computing the estimates based on the aforementioned pilot symbols and the received extracted values of the received grid's positions. This computation is commonly done using a method such as LS, MMSE or LMSSE [15][23].

The channel estimation process can be split into three steps:

- The transmitter and receiver determines DMRS setup
- The subframe is sent over the wireless medium
- The receiver extract the DMRS symbols and performs the calculation to estimate the channel

For a SISO environment, this could be described as the correlating mathematical

formula below, where y denotes the received signal, x represents the transmitted signal, and h is channel coefficients for the given frequency f, n denotes the noise. In this context, the noise is conforming to the Additive White Gaussian Noise (AWGN) model, meaning it is additive, the noise power is equally distributed across the frequency domain and Gaussian distributed. This noise model is commonly used in communication systems to approximate the statistical properties of real-world noise[30]. One common method to estimate the noise is to use the difference between the actual channel and the averaged channel estimate [9].

$$y(f) = h(f) \cdot x(f) + n \tag{2.1}$$

In a MIMO setting, each channel coefficient h_{ij} in the channel matrix represents each paring between transmitter i and receiver j. Equation 2.2 gives an example for 2×2 MIMO configuration.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \tag{2.2}$$

As the channel estimation only estimates over the pilot subcarriers, interpolation techniques are utilized to fill and compensate for the rest of the grid [9]. The received data is then acquired through *channel equalization*, which refers to the process of using the inverse of the computed channel matrix and applying it to the received signal, thereby compensating for the channel's distortion, attenuation, and delay [38].

In a MIMO$-$OFDM setting, the principle remains the same; however, the procedure must be applied to each subcarrier, OFDM-symbol and receiving antenna separately, where each receiving antenna experiences a superposed signal consisting of signals from all transmitting antennas, resulting in a higher complexity [1].

2.2.1.1 Traditional Channel Estimation Methods

The following subsections briefly explain the three traditional channel estimation algorithms that the Convolutional Neural Networks (CNN) is measured against in Chapter 5. As an in-depth understanding of the mathematical foundations behind these algorithms is not the central focus, they will not be thoroughly explained. They are presented but not derived. This is present in each reference.

Least Squares

The LS estimation is the ratio between the received and transmitted pilot signals. Computationally it is the element-wise division of each DMRS element. The LS method minimizes the squared error between the received signal and the estimated channel response.

$$h_{\text{LS},i,k} = \frac{y_i, k}{x_i, k}, \quad \forall i = 1, 2, \ldots, N \tag{2.3}$$

It has low computational complexity but does not perform well in noisy environments as it does not consider noise statistics. LS is suitable for scenarios with a good signal-to-noise ratio (SNR) and low system implementation complexity requirements [15].

Minimum Mean Square Error

The MMSE estimate of the channel matrix H is given by:

$$\hat{H}_{\text{MMSE}} = (H^H) \left(H(H^H) + \frac{\sigma^2}{P} I \right)^{-1} H_{\text{LS}}$$

Where P is the power and I is the identity matrix.

Which can be derived from the $e = \hat{X} - X = WY - X$ - where W is the Weight Matrix, an essential parameter needed to determine the MMSE and σ is the Gaussian additive channel noise - given that there is no correlation between neither the transmitted data, received data and the noise. H is the channel matrix as determined by the reference signals. The full derivation is showed in detail in [39].

The MMSE method is an effective approach for channel estimation in noisy environments as it minimizes the mean square error between the received signal and the estimated channel response. Compared to the LS method, MMSE generally provides better performance. However, it relies on knowledge of the noise variance and channel statistics, which can be difficult to obtain in practical scenarios. Therefore, the MMSE method is most suitable for situations with moderate to high noise levels and when reliable estimates of channel statistics are available or can be obtained [15].

Linear Minimum Mean Square Error

The LMMSE estimate is given by:

$$H_{\text{LMMSE}} = R_{hh}(R_{hh} + \frac{\beta}{\text{SNR}} \cdot I)^{-1} \cdot H_{\text{LS}}$$

where β is constant dependent on the transmitted data.

$$\beta = \frac{E(|x^2|)}{E(|1/x^2|^2)}$$

The algorithm is simplified Linear Minimum Mean Squared Error (LMMSE) estimator for channel coefficient estimation. This estimator assumes that two parameters, R_{hh} (auto-correlation matrix of channel coefficients) and σ (noise standard deviation), are known at the receiver's side[42].

In LMMSE estimation, the assumption of linearity simplifies the algorithm compared to the general MMSE approach. This is allowing for simplifications in the estimation algorithm, and that LMMSE can provide good performance in scenarios where the channel - the relationship between the transmitted signal and the received signal - can be reasonably approximated as linear. It is the best-performing channel estimation algorithm within 5G-LLS for the channels concerned in this thesis.

2.2.3 Channel Models

A channel model describes the environment the wireless signal has to traverse and the obstacles it has to overcome. It is a mathematical representation of the environmental impact on the signal. Multi-path signals, fading and shadowing, and relative movement of the receiving device are a few examples that can be determined using different channel models [10]. In Figure 2.1.2, the channel model is depending on the wireless medium, separating the transmitter and receiver.

2.3 Machine Learning

This subsection provides an introduction to the concept of machine learning, how it differs from AI, and offers a brief explanation of its fundamental principles. It further delves into the field of deep learning, with a particular focus on CNN which serves as the central theme of this thesis.

2.3.1 ML and AI Overview

AI refers to the field of study and development of machines that can learn, reason, and solve problems by mimicking human cognitive abilities. It is a broad subject incorporating many disciplines, including computer science, statistics, mathematics, engineering, linguistics, neuroscience, and philosophy [16, 21]. machine learning is a subset of AI that refers to the ability to learn and improve based on experience in the shape of data. In contrast to an algorithm developed to address a particular problem, a machine learning model is a mathematical function that has learned the patterns of specific data to construct a result. The model is a function that takes input and creates output, i.e., predictions. The model type depends on the problem and the data being used. Figure 2.3.1 showcases the broader context of Deep Learning within machine learning, and machine learning within the realm of AI.

The learning process of machine learning is the process of creating a model and can be broadly categorized into three kinds of learning models, namely:

- Supervised learning,
- Unsupervised learning,
- Reinforcement learning.

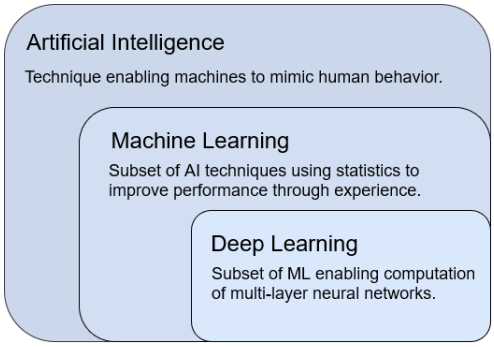

Figure 2.3.1: Relationship between AI, ML and DL. Remodeled from [4]

The model is trained using labeled data in *Supervised Learning*, which means each input has a pre-determined output in the dataset. Some common algorithms are Linear Regression, Decision Trees, Support Vector Machines [24] and Extreme Learning Machine (ELM) [22].

In *Unsupervised Learning*, there are no labels on the input data; it does not belong to a predefined category and is not known upon initialization. While Supervised learning uses the data to construct a model using the labels as guidelines, unsupervised builds the model from the data directly. Some common algorithms are K-means clustering, Principal component analysis (PCA)[18].

Reinforcement Learning is a machine learning model where the model is built through trial and error. An ”agent” is defined which is to perform a task. A reward function gives the agent positive or negative reinforcement according to its performance. The agent accounts for the reinforcement upon retaking the task and is placed in a loop while trying to satisfy the reward function. Some common algorithms are Q-learning and Deep Q-networks [16][12].

Although all methods share the goal of learning from data to make predictions, the learning process differs across categories, employing distinct approaches and underlying algorithms. A versatile approach that can be employed within each learning method is the utilization of Neural Networks.

2.3.2 Neural Networks

Neural networks are machine learning models inspired by the structure and function of biological neural networks in the human brain. A simple neural network consists of three layers: an input layer, a hidden layer, and an output layer. The input layer takes the data forward to the first hidden layer. The hidden layers take input from previous layers based on the weight and bias and perform activation computations on the data to enable non-linear transformations before passing it on to another hidden layer or the output layer. The output layer produces the prediction. All nodes, or *neurons*, within a neural network are connected to each neuron in the next layer. Each neuron receives inputs from multiple neurons and calculates an output value using an activation function on a weighted sum of its inputs. Each neuron-to-neuron connection

has a weight and bias attached to it, and the goal of the learning process is to adjust the weight of every neuron pair to minimize a cost function. When the cost function has converged, the model is deemed complete.

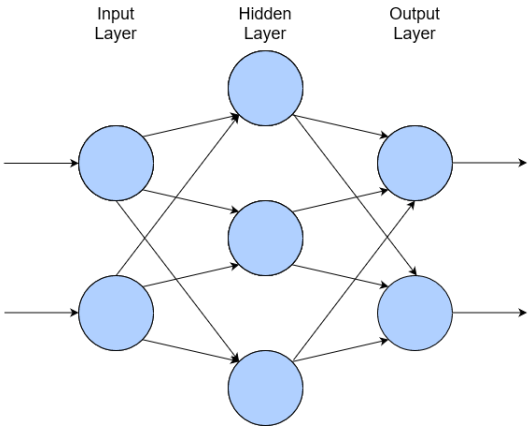

Figure 2.3.2: Simple Neural Network

Each neuron past the input layer has an activation function applied to the input data as it passes. It is this activation function that introduces the non-linearity. Without it, the output of a neuron would only be a linear combination of the input data.

Once data has passed through the neural networks, it outputs a result used to calculate the cost function, which measures the difference between the predicted output and the actual output. The measure calculates how well the model performs and is updated with each training iteration.

Backpropagation is the process of calculating the gradients of the cost function concerning the weights and biases of the Neural Network. Once the cost function has been calculated, the backpropagation begins. The backpropagation uses an optimization algorithm, such as Stochastic Gradient Descent (SGD) or the *Adam* algorithm, to update the weights in favor of the cost function [14, 33].

2.3.3 Deep Learning

Deep learning is a subset of machine learning that uses layered neural networks to learn complex patterns and relationships in data. The ”deep” refers to the multiple layers. It has been successfully applied to various wireless communication problems, including channel estimation, signal detection, modulation classification, and beamforming, among others [31].

Several types of deep learning algorithms are designed to solve specific problems. Some common deep learning algorithms include CNN, Recurrent Neural Network (RNN), Generative Adversarial Networks (GANs), Autoencoders, and Deep Belief Networks (DBNs) [11]. What differentiates the different algorithms is their architecture - the number of layers, the type of layers, the activation functions in the different layers, and how they are connected.

2.3.3.1 Convolutional Neural Network Overview

CNN is a type of Neural Network commonly used for grid-like data topologies, such as images, which can be considered a 2-D grid of pixels. The input is often a $m \times n \times r$ matrix, where m and n correspond to the width and height, and r is the depth. For instance, the depth in an image is the number of color channels - one for grayscale and three for an RGB color model.

A CNN typically consists of three specific layers. These are the convolutional layer, pooling layer, and fully connected layers. In short, the convolutional layer applies filters (kernels) to extract features on top of the grid. The pooling layer provides a method of downsampling the grid to reduce computations. The network then uses backpropagation and a predefined optimizer method, such as gradient descent, to optimize the filters so that they can strive to match the labels, like in a conventional machine learning method. Lastly, the fully connected layers are responsible for predicting the output.

Convolutional Layer

The convolutional layer is one of the defining layers of a CNN. In this layer, filters, or *kernels*, are applied to the input data grid to perform the convolutional operation to generate an output feature map. The kernel is a grid consisting of weights adjusted throughout the training process. This grid is applied over all elements in the input grid by sliding vertically and horizontally according to a predefined stride. The kernel grid size is a hyperparameter that is defined when configuring the model. Each kernel creates a separate feature map with the dot product between the kernel weights and the covered input values. This sliding procedure is visualized in Figure 2.3.3, where the integers represent arbitrary kernel weights over the input, and the different colored kernels represents the kernel position at different times. The feature map creation is displayed in Figure 3.1.4.

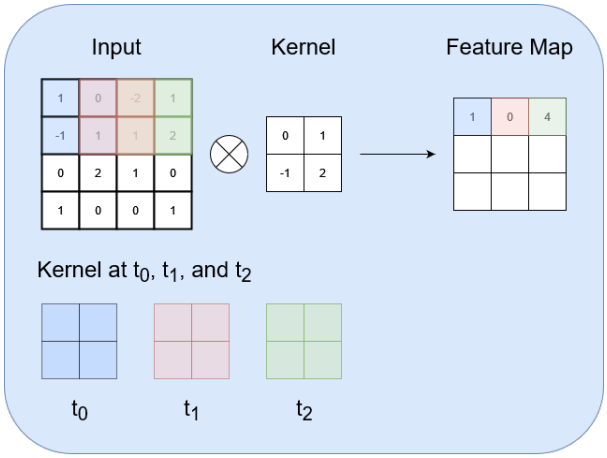

Figure 2.3.3: Kernel, inspired by Figure 8 in [7]

When producing a feature map, it is also common to include padding, increasing the size of both the input and output feature maps. Padding helps preserve the data's

spatial dimension while giving more significance to data near the edges, as the kernel will pass over a central resource element more often than an element contained in a corner [7]. Without padding, the feature map will shrink with each layer.

The number of filters, or kernels, is also known as the *output channels* of a layer and is together with kernel size, padding, and stride hyperparameters set when configuring the model. The number of filters determines the capacity of the feature extraction of the model. With more filters, the convolutional layer can extract more features from the layers, meaning it can model more complex relationships.

Pooling Layer

The pooling layer is responsible for downsampling the data, which means shrinking a large feature map to a smaller one using a predefined method. For example, a 4×4 grid can be split into four sub-grids, where a pooling method can be used to establish the most dominant feature within each sub-grid and construct a 2x2 grid as a new feature map, including only the most significant values [7].

Fully Connected Layer

A fully connected layer is used as an output layer in classification tasks, a common use case for a CNN. The fully connected layer's output is then the complete CNN [7].

2.4 Related Work

The field of wireless communication has witnessed significant advancements in recent years, with the integration of deep learning techniques showing promising results. This text discusses three research works that leverage deep learning approaches to enhance the performance of communication systems in different scenarios.

Soltani *et al.* [41] combines two CNN to perform a two-step image processing on the time-frequency grid of the channel response in a Single-Input Single-Output (SISO)-OFDM system. The received grid is considered to have two channels - one for each I/Q domain. Initially, the received grid with its pilot symbols is considered a low-resolution image on which a super-resolution algorithm is used to enhance the resolution. Afterward, an image restoration algorithm is applied to reduce the effect of noise. Its performance in vehicular channel models outperforms MMSE. However, the paper fails to provide the effect of the noise reduction the second phase offers. The paper uses two versions of the same model, each trained on data gathered at different Signal-to-Noise Ratio (SNR) levels, which becomes apparent in the results as one performs better at lower SNR ratios and the other at higher. Their work also includes the effect of pilot symbols and how the Mean Square Error (MSE) differs for each algorithm. This balance between overhead and performance is often overseen but is just as crucial for the overall communication system performance. Pradhan .A *et al.* [26] uses a similar approach but also models the DMRS grid as a low resolution image which is enhanced using a separate CNN, from where the output is fed into the main Super-Resolution CNN. This way fewer pilots can be used, creating less overhead. Ru, Xin *et al.* [37] extends the work of Soltani [41] by combining it with the approach proposed by Dong, Chao *et al.* [19]. This combined model offers the advantage of adaptability to systems with different input sizes, making it applicable in a wide range of scenarios.

Li, Xuhong *et al.* [25] proposes a special purpose CNN architecture optimized to work in a mine environment. This work showcases the abilities of custom-made solutions in scenarios where traditional approaches are lackluster. In their work, the authors have identified the fading characteristics of a mine in terms of Doppler shifts and multipath effect and used a Nakagami fading channel model to represent the environment. Despite employing a relatively simple architecture, their approach outperformed conventional algorithms in this specific scenario.

Aswathy, K. Nair and Vivek M. [28] highlight the overhead created by excessive pilot symbols and the interdependencies of channel estimation and channel decoding and propose a joint deep learning approach that simultaneously addresses channel estimation and symbol detection. They proposed Bidirectional Long Short-Term Memory (Bi-LSTM), which goes beyond channel estimation and handles decoding the received symbols into bit groups. Bi-LSTM is a variant of RNN, a type of deep learning architecture that is typically used to handle sequence models or time-series data well aligned with communication systems. First, they utilize Compressed Sensing

(CS), exploiting the sparsity between OFDM ticks in the time domain to delimit pilot symbols within a resource grid required to provide adequate performance. Second, by treating the OFDM subcarriers as timestamps, the LSTM can effectively capture and utilize the temporal dependencies present in the data. The model outperforms conventional algorithms using fewer pilot symbols, thus eliminating overhead and increasing performance further.

Deep learning techniques can enhance the performance of communication systems by addressing specific challenges in different environments. Moreover, leveraging dependencies between different modules in the processing chain may yield better performance and increase efficiency. This research highlights the potential of deep learning techniques in enhancing various aspects of communication systems, including resolution enhancement, noise reduction, specialized architectures, joint channel estimation, and symbol decoding. Meenalakshmi, M. *et al.* [27] provides an overview of current deep learning approaches to channel estimation[27].

This work builds upon the foundations of Soltani *et al.*'s studies by applying 3D Convolution on the MIMO-OFDM grid in a 5G setting. In addition, it incorporates two distinct channel models to demonstrate the capability of creating a generalized machine learning model.

Chapter 3

Design of an Machine Learning-Based Channel Estimation Algorithm

This chapter focuses on designing and implementing an machine learning channel estimation algorithm using a CNN. Section 3.1 describes the CNN architecture, including its layers, hyperparameters, and the training process. In Section 3.2, the process of generating the necessary training data and labels is presented.

3.1 CNN Architecture

This section discusses and motivates the choices that laid the foundation for using the CNN. It discusses the input and output data pre-processing and formatting, layers, hyperparameters, and the training process.

3.1.1 Motivation for CNN-based Channel Estimation

In this research, a CNN has been selected as the primary algorithm for channel estimation due to its prevalence in related work and its applicability to the subject at hand. The use of CNNs in various fields, including computer vision and signal processing, has demonstrated their effectiveness in extracting and learning meaningful features from complex data. The choice of a CNN is motivated by the resource grids and the nature of channel estimation in 5G networks, which involves processing vast amounts of data from multiple antennas and across different frequency and time dimensions. CNNs are well-suited for handling multidimensional data, making them an ideal candidate for this task.

Furthermore, the utilization of CNNs in recent research pertaining to channel estimation provides a solid foundation for this research. Building upon the insights gained from prior work, this study seeks to enhance the performance of channel estimation in 5G networks by leveraging the strengths of CNN architectures.

3.1.2 Machine Learning Framework

Tietoevry has requested that the thesis project use an open-source library to perform all machine learning. To conform with their desires, PyTorch [32] was chosen. It is widely used in the field of data science and machine learning and provides a lot of online educational resources for users. It also has built-in support for GPU acceleration, leveraging the computational power of graphics processing units (GPUs) to accelerate model training by utilizing parallel processing.

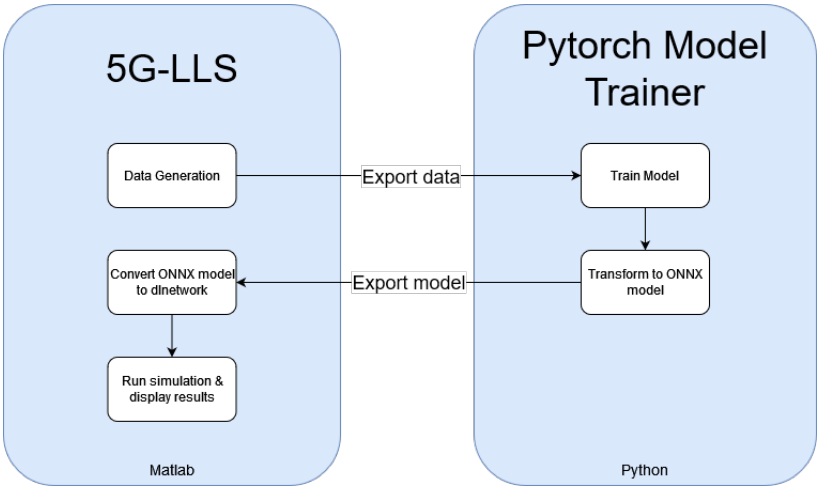

Figure 3.1.1: Connection between 5G-LLS and the Python program using PyTorch for machine learning tasks

Additionally, PyTorch can export trained models into the Open Neural Network Exchange (ONNX) format [29]. ONNX serves as a framework for converting models between different machine learning frameworks[29]. This functionality allows for the export of PyTorch models, which can then be imported and transformed into MATLAB's `dlnetwork` object for use within the 5G-LLS environment. Figure 3.1.1 display the connection between 5G-LLS and the python program used for the machine learning tasks described in the following sections.

3.1.3 Channel Matrix Data Pre-Processing and Formatting

This subsection describes the input and output data of the CNN, and why it is formatted as it is.

The input of a neural network is called a *tensor*. A multi-dimensional matrix that holds elements of a single data type [36]. Here, each element in the resource grid represents a phase component from the received signal, defined by complex numbers. The input tensor consists of 4 dimensions: Subcarriers, OFDM Symbols, Receive Antennas and lastly, the real and imaginary parts of the phase.

The resource grid consists of 300×14 resource elements and is described in Section 2.1.3. As the system is using a 4x4 MIMO configuration, the actual channel

matrix is a 300 × 14 × 4 × 4 matrix. Below is how each dimension can be interpreted for more clarity.

- 300 represents the number of subcarriers in the resource grid. Each subcarrier carries a portion of the overall transmitted signal
- 14 represents the number of OFDM symbols within a transmission time interval. These symbols are used to transmit data across time and frequency
- 4 × 4 represents the MIMO configuration with 4 antennas at both the transmitter and receiver. Each element in the matrix corresponds to a specific channel response between a transmit-receive antenna pair for a given subcarrier and OFDM symbol

As described above, the 4 × 4 represent each transmit-receive antenna pair. However, the receiver is unaware of each transmitter's effect. It can subsequently be transformed into a 300 × 14 × 4 matrix where the last dimension represents each receiver's superposed signal (the sum of the responses), as can be observed in the left-most antenna mesh-grid in Figure 3.1.2.

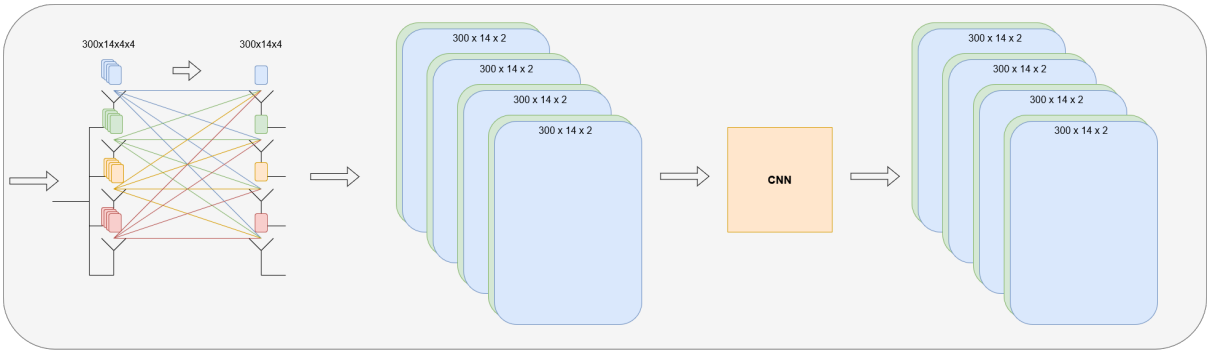

Figure 3.1.2: Data Transformation Chain

As indicated by the the 4 resource grids to the left of the CNN, each resource grid has been separated into its real and imaginary parts independently. This is because the CNN does not handle complex numbers, and instead treats the real and imaginary parts as two additional channels in an extra dimension.

In conclusion, the input and output of the CNN is a 300 × 14 × 2 ×-4 shaped tensor.

3.1.4 Layer Configuration

The CNN architecture consists of 7 convolution layers including input and output layers. The objective is to approximate values based on interpolation and not to perform any categorization, thereby using only convolutional layers and no pooling layers. As prior mentioned, the output is the same shape as the input, as to why downsampling with pooling is superfluous. Figure 3.1.3 gives an overview of the layers, whereas Figure 3.1.4 displays the convolutional function.

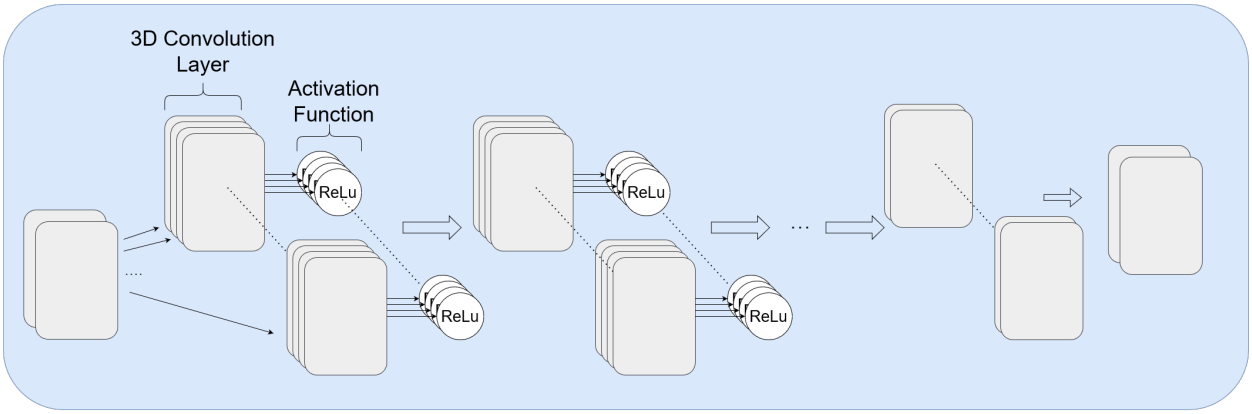

Figure 3.1.3: Convolutional Neural Network

The number of layers has been determined through trial and error in combination with related work to find a good balance for the complexity of the problem. More straightforward problems may be effectively addressed with fewer layers. More complex problems may require deeper networks with multiple layers to capture intricate patterns and relationships in the data.

The model consists of 3D-convolutional layers to capture the dimension of the MIMO-OFDM system. It includes subcarriers, OFDM symbols, antennas, and complex values, where the complex values are treated as separate input channels. Each convolutional layer has a specific kernel size, padding configurations, and output channels, creating equal feature maps as input to the next layer.

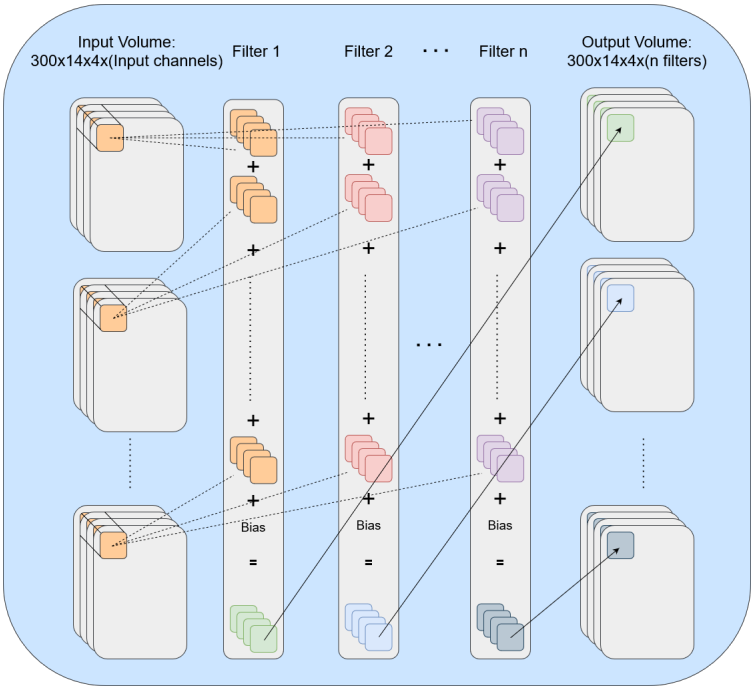

Figure 3.1.4: Convolution Function

Inspired by the work of Soltani, the layer configuration of the CNN has been adapted

to address the more complex nature introduced by MIMO. The initial layer employs a padding of 4, ensuring that elements on the edge of the grid receive the same relevance as those within the grid when kernels operate. To capture broader structures in the data, a 9×9 kernel with an appropriate stride is employed in the first convolutional layer.

Subsequent middle layers are designed to capture more localized differences in the grid using smaller kernels, while maintaining the same number of output channels. Five additional layers perform similar convolutional operations, allowing the network to extract and learn features from the channel data. It was observed that increasing the number of middle layers up to 5 yielded improved validation scores, indicating the significance of capturing these complex features. The last layer inputs 16 channels, and the kernel shrinks further to provide more fine-grained tuning and decreases the number of output channels to two as the initial input to the CNN to produce the final output of the model.

3.1.5 Training Process

The Python program uses nested loops for training and validating the model. By optimizing the model's parameters through gradient-based optimization algorithms, such as gradient descent, the training process aims to minimize the loss function [35]. The nested loop structure employed in the training process ensures that the model undergoes multiple iterations, known as epochs, to update its weights and biases based on the calculated gradients at each training iteration.

Procedure 1 Training Loop

```
for each epoch do
    for each batch sample do
        Run data through the model
        Zero the gradients for every batch
        Compute the loss of the model
        Perform backpropagation
        Adjust weights according to the optimizer algorithm
        Validate data
    end for
end for
```

Through this iterative optimization process, the model gradually improves its ability to make accurate predictions on unseen data, enhancing its generalization capability. The number of epochs runs until the MSE has stagnated and cannot optimize the cost function output further, and this is validated through monitoring the validation loss. The model has been trained with a batch size of 32 to match to create a balance between convergence speed and quality of gradient estimates. The batch size determines the number of training data samples propagated through the model at each interval. In a too-small batch, individual samples have a more substantial impact on the weights and

can sometimes lead to overfitting the model. A larger batch provides more accurate gradient estimates by averaging over more data each iteration which can help the model generalize better to unseen data, the impact of which can be observed during validation and testing.

The learning rate is set to 0.001, which guides how much the weights are updated each iteration. The learning rate is also set to balance convergence speed and the ability to find an optimal solution.

The optimizer uses an MSE loss function. This method strives to adjust the weights to minimize the Mean Square Error. The method is suitable for establishing the difference between the data and the label, as all resource elements contain a value.

Lastly, the result is validated throughout the testing process using unseen data. It is done several times during each epoch, each traverse through the whole training dataset, to keep track of the improvement of the model.

3.2 Data Generation

The training dataset is gathered from the 5G-LLS. The simulation is run over an SNR range between -20 dB and 20 dB, incrementing by 0.25 dB at each step. This range is considered reasonable, as it reflects the variation of the Signal-to-Nose Ratio (SNR) commonly encountered in real-life scenarios. It runs over all channel models, including them in the same file to create a generalized model. Creating special-purpose models that operate in special conditions is also possible. Likewise, it would have been possible to create two separate models - one for the simple MIMO channel and one for the sparse MIMO channel; however, this CNN is designed to work for both channels.

The complete generated dataset is called the *training dataset*. The training dataset consists of both *training data* and the corresponding *training labels*. A validation dataset is also gathered in the same way as the training dataset. The validation, like the testing data, is unseen data. The datasets must be separated so that validation is not done on the same data used to train the model. Performing validation or testing with training data can lead to overfitting, creating a less generalized model that would not deliver good results for data other than the one used in training. The training and validation data are exported to the Python program for training purposes. Testing is performed by exporting the models built using Python back to 5G-LLS and running through the simulator.

While the impact of dataset size on performance was not explicitly studied in this work, it is a topic worthy of further investigation. The relationship between dataset size and performance is complex, as increasing the dataset size can potentially improve the model's generalization capabilities.

3.2.1 Training Data

Training data is gathered at the receiver end of the communication system. It is gathered similarly that DMRS is used within 5G-LLS. It does so by first estimating the channel by performing element-wise multiplication of the received DMRS grid with the complex conjugate of the known values that the transmitter and receiver have agreed upon - this is to remove the effect of the noise. This described multiplication is LS, which is a fundamental concept that is often used as a building block within more advanced algorithms like the MMSE. Once the channel has been estimated over the DMRS symbols, the rest of the resource grid is estimated using linear interpolation through Matlab's `ScatteredInterpolant` function. The function takes the coordinates of the DMRS points and their corresponding values as input and creates an `interpolant` object, which is then used to create the final estimated channel matrix for a resource grid. This procedure is repeated for each receiving antenna.

A dataset size that was sufficiently representative of the signal variations and noise conditions encountered in the target communication scenario was selected. No explicit study was made on the impact of the dataset size.

3.2.2 Training Labels

The training labels consist of the ideal channel matrix. The ideal channel matrix represents the proper relationship between the transmitted and received signals without the inclusion of noise. This matrix is impossible to establish in a real-world scenario but is attainable in a simulator. How well the machine learning model transfers to a real world scenario is dependent on the quality of the mathematical model that defines the training data. Noise in 5G-LLS is added to the signal after the channel matrix has altered it. The characteristics of the channel are affected by fading, path loss, and multipath propagation. Noise, on the other hand, arises from sources unrelated to the channel itself and is not included in the labels. The ideal channel matrix is constructed dependent on the channel model and is calculated differently for each channel model.

Chapter 4

Evaluation Methodology

Chapter 4 focuses on the evaluation methodology for the channel estimation experiments within 5G-LLS. The metrics used for evaluating both the system-level performance and the performance of the machine learning models are presented. Additionally, the 5G-LLS, including the channel models and system simulation parameters are described.

4.1 Metrics

The metrics for determining performance should be split into two sections: the performance of the machine learning models and the performance of the communication system. This section introduces Bit-Error Rate (BER) to measure the system performance and MSE to measure the channel estimation algorithm performance. Lastly, Symbol-Error Rate (SER) is introduced to cover the middle ground between the two.

4.1.1 System-level Performance

BER is a widely adopted performance metric of communication systems to determine the accuracy of bits in data transmissions. It can be described as the likelihood of a bit being incorrectly interpreted at the receiver [34].

$$\text{BER} = \frac{\text{Number of Error Bits}}{\text{Number of Total Bits}} \tag{4.1}$$

The BER is strongly influenced by the strength of the signal and the noise, which is, in turn, dependent on the wireless channel's condition. A higher SNR enables a lower BER as environmental factors make the signal less distorted. This ratio is referred to as SNR, or S/N, and is defined as follows, where P is the power expressed in dBm:

$$\text{SNR}_{\text{dB}} = 10 \cdot \log_{10} \left(\frac{P_{\text{signal}}}{P_{\text{noise}}} \right)$$

The performance of the communication system is typically displayed as a 2D graph, with BER on the y-axis on the SNR on the x-axis. The 2D graph shows how well the system performs over multiple SNR ratios, which is essential as the noise inherent in the wireless channel is constantly changing [20]. It should be noted that BER over SNR measures the total system performance, where channel estimation plays a crucial role, but this means this measurement also accounts for modulation and decoding. To better understand how the channel estimation performs, the MSE is positioned on the y-axis. The MSE is also the metric to be optimized.

4.1.2 Machine Learning Model Performance

The machine learning model utilizes an MSE loss function and is thereby best evaluated in the testing phase by comparing the ideal channel matrix with the estimated channel matrix over a MIMO grid. The MSE loss function is suitable as it provides an average error over the whole grid, which can also be easily applied to the LS, MMSE, and LMSSE methods for comparison. The MSE definition is shown in equation 4.2, where N is the total number of data points in the dataset. y_i represents the true values, whereas $\hat{y}_i$ represents the predicted or estimated values.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{4.2}$$

As mentioned in the previous section, the channel estimation algorithm is best measured using MSE. As for the system-level performance, the best representation is given throughout SNR values due to the unpredictability of noise.

A symbol represents a group of bits translated from the phase within a resource element, which depends on the modulation scheme used. It can be calculated similarly to the Bit Error Rate (BER) and is visualized with a constellation graph to see the distribution of estimated phases, as seen in Section 2.1.2. The distribution of estimated phases provides valuable information about the accuracy of the channel estimation. If the estimated symbols are tightly clustered around the ideal symbol locations, it indicates high channel estimation and demodulation accuracy. Conversely, if the estimated symbols are scattered or deviate significantly from the ideal locations, it suggests errors in the channel estimation process. Lastly, this constellation graph is titled with its SER, which displays the ratio of errors in the estimation.

$$\text{SER} = \frac{\text{Number of Error Symbols}}{\text{Number of Total Symbols}} \tag{4.3}$$

4.2 5G Link Layer Simulator

5G-LLS is a Link Layer Simulator, initially developed by Tietoevry to simulate 4G traffic, but has been modified and re-implemented based on 5G requirements and standards. The simulator is developed by both employees of the company as well as through thesis work [6][39]. Figure 2.1.2 displaying the 5G NR Processing Chain (as seen in Chapter 2), displays all implemented modules in the simulator.

4.2.1 Channel Models

Two different channel models are used to examine the different channel estimation methods: a flat MIMO channel model, where there is a one-to-one correlation between each sender and receiver pair, and a correlated MIMO channel, where each transmitting antenna correlates to all receiving antennas to some degree[40].

The first channel, labeled as an *EYE*-channel, where the system acts as a SISO configuration in the sense that each transmitting antenna only sends to one corresponding receiving antenna. While simple, it gives a good view of how the different methods perform and compete against each other on their own.

The other channel, labeled as an *OFF*-channel, is a correlated interference channel MIMO channel model where each transmitting antenna is sending data to each receiver, and with each receiving antenna experiencing a interference from the other transmitting antennas depending on the spacing between the transmitting antennas. Here, each receiver must deal with a superposed signal and account for data coming from all transmitters, which increases the complexity of the channel estimation and gives a more realistic view than the EYE-channel model.

Furthermore, the same module that models the channel adds AWGN noise to the signal, scaled according to the SNR decibel value which is a commonly used noise model designed to mimic the random effects that occur in nature [30].

Table 4.2.1: Parameter Specification

Parameter	Specification
Subcarrier Spacing	15 kHz
Number of PD-SCH Subcarriers	300
Number of OFDM Symbols	14
MIMO Configuration	4x4
Number of Physical Resource Blocks	12
Interpolation Type	Linear
Modulation Scheme	QPSK

4.2.2 System Simulation Parameters

As mentioned in Section 2.1.3, the 5G system parameters can be modified to suit environmental requirements. Table 4.2.1 displays the parameters that were used during the simulations.

4.2.3 Software Requirements

The implementation requires Matlab's Deep Learning Toolbox, particularly the Deep Learning Toolbox Converter for ONNX Model Format, an add-on necessary for importing ONNX Networks.

Chapter 5

Results

This chapter presents the results and evaluation of the CNN compated with the conventional algorithms for channel estimation on two different channel models. The simulations have been run within 5G-LLS and evaluation is done with curves representing BER, MSE and SER versus SNR for a QPSK modulation scheme, with measurement points on each SNR integer in the span. Section 5.1 presents the results for the Fast Fading EYE Channel and Section 5.2 presents the evaluation for the Spatially Correlated OFF channel.

5.1 Flat Fading Channel Performance

Figure 5.1.1 displays the BER over a full range of SNR values over the Flat Fading EYE Channel. It is important to note that at the lowest SNR level where the signal is still detectable, the receiver's performance is akin to random guesswork, with a probability of approximately 50% of correctly decoding the transmitted signal. This poor result is due to the signal being severely corrupted by noise, making it challenging to distinguish the transmitted symbols accurately.

The BER vs SNR results in Figure 5.1.1 displays a similar result for the CNN and the LMSSE in terms of BER over SNR, both outperforming LS and MMSE. The difference between CNN and LMSSE are almost indistinguishable for the whole span. The first assumption would thereby be that the CNN does not manage to outperform the LMSSE. However, this is not entirely true, as proven by Figure 5.1.2.

Figure 5.1.2 displays the MSE for all algorithms using a logarithmic scale, where CNN is significantly lower than both LMMSE and the others, which means that the CNN performs better channel estimation, but that the LMSSE's MSE still manages to lie above the threshold of where the symbols can be correctly demodulated. At SNR an SNR of 20 dB, the CNN MSE is 80% lower than that of the others. It indicates that using a more complex modulation scheme would favor the CNN, as a more advanced modulation scheme requires more precise channel estimation and puts a more significant emphasis on MSE for demodulation.

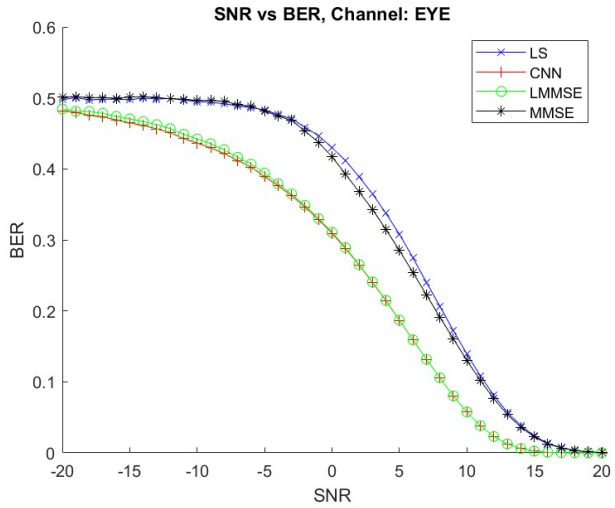

Figure 5.1.1: EYE Channel BER Performance

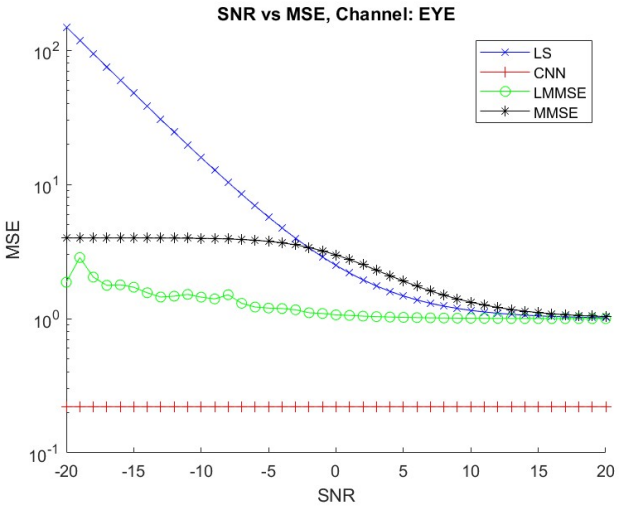

Figure 5.1.2: EYE Channel MSE Performance

For reference, Figure 5.1.3 displays how the different algorithms manage to demodulate the symbols for SNR levels 10, 15 and 20. The blue dots represent the correctly labeled demodulation and the red dots the mislabeled. It is important to acknowledge that the red data points, despite appearing to dominate in quantity, coincide with the blue data points. Therefore, although a visual observation may in some instances suggest a majority of red points, the SER for both the LS and MMSE methods is only twice that of the CNN and LMSSE methods (see Table 5.1.1 for SER values). The centered center of mass displays the algorithms' struggle to correctly identify phases closer to the thresholds.

Additionally, in Figure 5.1.2, the conventional algorithms' MSE all decrease as the SNR increases, as it also does in the BER graph. In contrast, the CNN's MSE curve exhibits a linear and seemingly unchangable trend, which can be attributed to the CNN's ability to learn from similar scenarios in its training data and construct the channel matrix

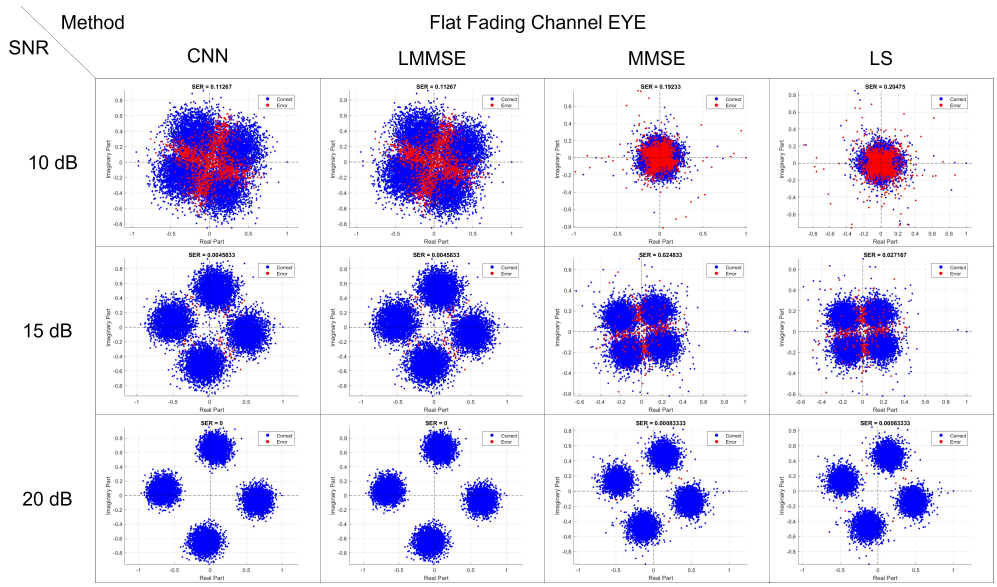

Figure 5.1.3: Constellation Graph for EYE Channel using QPSK

Table 5.1.1: SER Values for Different Algorithms at Various SNR Levels for Flat Fading Channel

SNR (dB)	Algorithm			
	CNN	LMMSE	MMSE	LS
10	0.11	0.11	0.19	0.205
15	0.005	0.005	0.025	0.027
20	0.0	0.0	0.001	0.001

accordingly. Its gentle slope can also be attributed to the underlying simplistic channel model that allows the machine learning model to distinguish the different patterns offered.

5.2 Spatially Correlated Channel Performance

The Spatially Correlated OFF channel provides a comparative result with the EYE channel, though the curves are not as steep and never reach the same low BER for the same SNR span. The BER values are consistently higher for the OFF channel across all SNR levels.

The performance of the CNN and LMMSE algorithms is compared in terms of MSE, as shown in Figure 5.2.2. The CNN consistently achieves lower MSE values than the LMMSE algorithm. However, both algorithms exhibit similar results when examining the system-level performance across different SNR levels, as seen in Figrue 5.2.1, and suggests that, even here, under the current modulation scheme, the MSE achieved by LMMSE is still sufficiently low to discern the symbols in a constellation graph.

In Figure 5.2.3, an interesting pattern emerges for LMMSE and CNN, where many dots are still centered around the origin. There becomes too much interference in

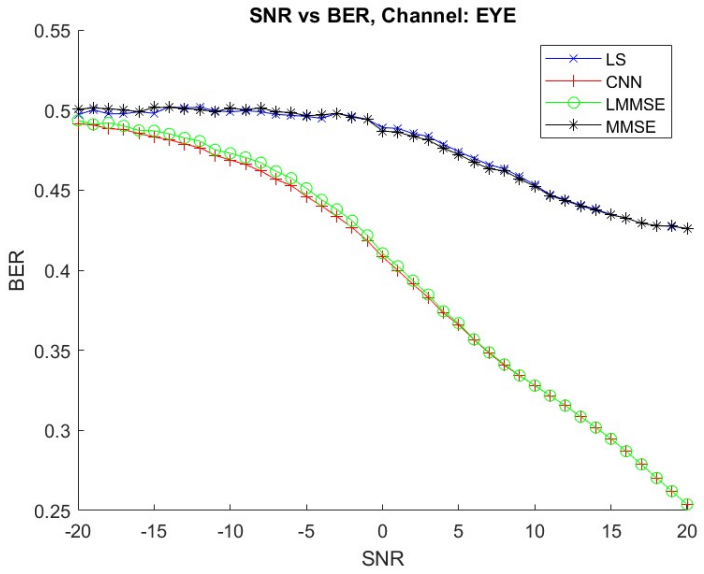

Figure 5.2.1

the superposed signal. Which means the interference is too high for the signal to be accurately demodulated. The specific SER values are visible in Table 5.2.1

The dissimilarity between the two algorithms in terms of MSE can be attributed to the nature of the training data. CNN can make accurate predictions by leveraging the mappings learned during training. Furthermore, the results shed light on the impact of noise. Despite knowing the channel matrix through channel estimation, noise remains a factor that cannot be fully compensated for, making it inherently challenging to eliminate.

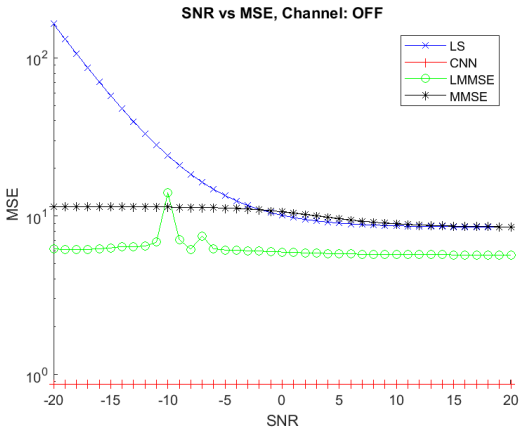

Figure 5.2.2

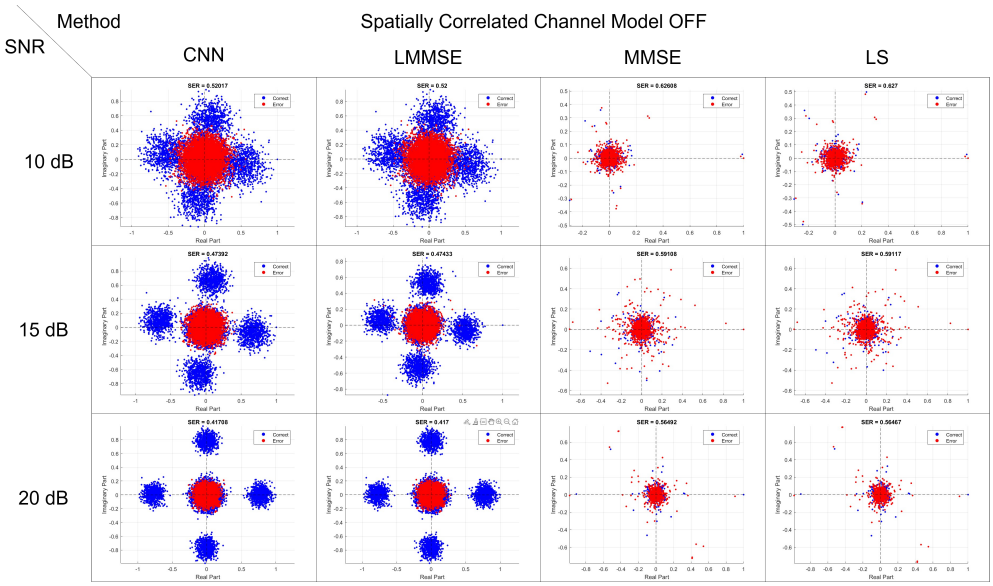

Figure 5.2.3: Constellation Graph for OFF Channel using QPSK

Table 5.2.1: SER Values for Different Algorithms at Various SNR Levels for the Spatially Correlated Channel

SNR (dB)	Algorithm			
	CNN	LMMSE	MMSE	LS
10	0.52	0.52	0.627	0.627
15	0.474	0.474	0.591	0.591
20	0.417	0.417	0.565	0.565

Chapter 6

Conclusions

This thesis work has evaluated the use of convolutional neural networks to estimate radio channels in 5G NR by comparing them with traditional methods, including LMMSE, MMSE and LS. The results of the work indicated that the machine learning approach using a CNN significantly improved the MSE of the channel estimation procedure compared to all algorithms. Most notably it significantly outperformed the closest competitor, LMMSE. For the flat fading channel, the CNN's MSE was only 22% of that of the LMMSE, and for the spatially correlated channel at 22% of LMMSE's MSE. Consequently, the results suggest that there is indeed a potential for using machine learning techniques to enhance the accuracy of channel estimation in wireless communication systems. While the CNN outperformed the MMSE and LS algorithms using both metrics, the performance of the overall communication system in the 5G-LLS was not improved by the channel estimation improvements alone compared to LMMSE. The reason was that the rivaling LMMSE algorithm offered a sufficiently low MSE to ensure that the received symbols were demodulated correctly for the QPSK modulation scheme.

Additional research and fine-tuning are required to fully exploit the advantages of machine learning for channel estimation in Tietoevry's 5G-LLS. Future investigations should focus on testing the CNN on more sophisticated channel models and exploring advanced machine learning techniques such as an RNN or an ELM.

In sum, the thesis work has successfully developed a Python framework for channel estimation experimentation and demonstrated the potential of machine learning techniques in improving channel estimation accuracy within 5G-LLS. This work lays a solid foundation for future research endeavors aimed at advancing the field of channel estimation in 5G telecommunications for Tietoevry and its application in wireless communication networks.

Bibliography

[1] In: *5G Wireless: A Comprehensive Introduction*. Addison-Wesley Professional, 2021. ISBN: 9780136767206.

[2] *5G NR Physical channels and modulation*. 2020.

[3] *5G NR Services provided by the physical layer*. 2018.

[4] Aggarwal, Karan, Mijwil, Maad M, Al-Mistarehi, Abdel-Hameed, Alomari, Safwan, Gök, Murat, Alaabdin, Anas M Zein, Abdulrhman, Safaa H, et al. "Has the future started? The current growth of artificial intelligence, machine learning, and deep learning". In: *Iraqi Journal for Computer Science and Mathematics* 3.1 (2022), pp. 115–123.

[5] Ahmadi, Sassan. "Chapter 4 - New Radio Access Physical Layer Aspects (Part 2)". In: *5G NR*. Ed. by Sassan Ahmadi. Academic Press, 2019, pp. 411–654. ISBN: 978-0-08-102267-2. DOI: https://doi.org/10.1016/B978-0-08-102267-2.00021-X. URL: https://www.sciencedirect.com/science/article/pii/B978008102267200021X.

[6] Albawi, Saad, Mohammed, Tareq Abed, and AlZawi, Saad. "Understanding of a convolutional neural network". In: *2017 International Conference on Engineering and Technology (ICET)*. IEEE. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186.

[7] Alzubaidi, Lina, Zhang, Jun, Humaidi, Ahmed Jasim, and al., et. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (2021), p. 53. DOI: 10.1186/s40537-021-00444-8. URL: https://doi.org/10.1186/s40537-021-00444-8.

[8] Anand, P., Singh, Y., Singh, A., Singh, PK., Felseghi, RA., and Raboaca, MS. "IoVT: Internet of Vulnerable Things? Threat Architecture, Attack Surfaces, and Vulnerabilities in Internet of Things and Its Applications towards Smart Grids". In: *Energies* (2020).

[9] Anonymous. *Channel Estimation*. Accessed on March 4, 2023. URL: https://www.hs-osnabrueck.de/fileadmin/HSOS/Forschung/Recherche/Laboreinrichtungen_und_Versuchsbetriebe/Labor_fuer_Hochfrequenztechnik_und_Mobilkommunikation/Forschung/digitale_Funksysteme/Channel-Estimation.pdf.

[10] Anonymous. *Simulate channel models for wireless systems*. Accessed on May 16, 2023. URL: https://se.mathworks.com/discovery/channel-model.html.

[11] Anonymous. *What is a Neural Network?* Accessed on May 6, 2023. 2020. URL: https://www.ibm.com/topics/neural-networks.

[12] Baheti, Pragati. "The Beginner's Guide to Deep Reinforcement Learning". In: *Blog* (2023). Accessed on May 16, 2023. URL: https://www.v7labs.com/blog/deep-reinforcement-learning-guide#h4.

[13] Buzzi, Stefano, I, Chih-Lin, Klein, Thierry E., Poor, H. Vincent, Yang, Chenyang, and Zappone, Alessio. "A Survey of Energy-Efficient Techniques for 5G Networks and Challenges Ahead". In: *IEEE Journal on Selected Areas in Communications* 34.4 (2016), pp. 697–709. DOI: 10.1109/JSAC.2016.2550338.

[14] Chaudhary, Amit. *How does back-propagation work in neural networks? With Worked Example*. Accessed on May 6, 2023. 2020. URL: https://towardsdatascience.com/how-does-back-propagation-work-in-neural-networks-with-worked-example-bc59dfb97f48.

[15] Cho, Yong Soo, Kim, Jaekwon, Yang, Won Young, and Kang, Chung G. "Channel Estimation". In: *MIMO-OFDM Wireless Communications with MATLAB®*. 2010, pp. 187–207. DOI: 10.1002/9780470825631.ch6.

[16] Cloud, Google. "What is Artificial Intelligence?" In: *Google Cloud* (n.d.). Accessed on May 17, 2023. URL: https://cloud.google.com/learn/what-is-artificial-intelligence.

[17] Cox, Christopher. "Chapter 4 - New Radio Access Physical Layer Aspects (Part 2)". In: *An Introduction to 5G The New Radio, 5G Network and Beyond*. Ed. by Christopher Cox. John Wiley and Sons Ltd, 2021, p. 96. ISBN: 978-1-11-960269-9. DOI: 10.1002/9781119602682.

[18] Ding, Chris and He, Xiaofeng. "K-Means Clustering via Principal Component Analysis". In: *Proceedings of the Twenty-First International Conference on Machine Learning*. ICML '04. Banff, Alberta, Canada: Association for Computing Machinery, 2004, p. 29. ISBN: 1581138385. DOI: 10.1145/1015330.1015408. URL: https://doi.org/10.1145/1015330.1015408.

[19] Dong, Chao, Loy, Chen Change, and Tang, Xiaoou. "Accelerating the SuperResolution Convolutional Neural Network". In: *arXiv preprint arXiv:1608.00367* (Aug. 2016), arXiv:1608.00367.

[20] Goldsmith, Andrea. *Wireless Communications*. Cambridge University Press, 2005.

[21] Hao, Karen. "What is AI?" In: *MIT Technology Review* (Nov. 2018). Accessed on April 16, 2023. URL: https://www.technologyreview.com/2018/11/17/103781/what-is-ai-we-drew-you-a-flowchart-to-work-it-out/.

[22] Huang, Gao, Song, Shiji, Gupta, Jatinder N. D., and Wu, Cheng. “Semi-Supervised and Unsupervised Extreme Learning Machines”. In: *IEEE Transactions on Cybernetics* 44.12 (2014), pp. 2405–2417. DOI: 10.1109/TCYB.2014.2307349.

[23] Khlifi, Abdelhakim and Bouallegue, Ridha. “Performance Analysis of LS and LMMSE Channel Estimation Techniques for LTE Downlink Systems”. In: *International Journal of Wireless & Mobile Networks (IJWMN)* 3.5 (Oct. 2011), p. 141. DOI: 10.5121.

[24] Kumar, Sandeep. *Supervised Machine Learning Algorithms You Should Know*. Accessed on May 17, 2023. Jan. 2020. URL: https://www.aitude.com/supervised-machine-learning-algorithms-you-should-know/.

[25] Li, Xuhong, Feng, Zhiyuan, and Wang, Anyi. “Research on MIMO-OFDM Deep Receiver in Mine Environment”. In: *2021 International Conference on Digital Society and Intelligent Systems (DSInS)*. 2021, pp. 292–295. DOI: 10.1109/DSInS54396.2021.9670585.

[26] Liu, Sicong and Huang, Xiao. “Sparsity-aware channel estimation for mmWave massive MIMO: A deep CNN-based approach”. In: *China Communications* 18.6 (2021), pp. 162–171. DOI: 10.23919/JCC.2021.06.013.

[27] Meenalakshmi, M., Chaturvedi, Saurabh, and Dwivedi, Vivek K. “Deep Learning-based Channel Estimation in 5G MIMO-OFDM Systems”. In: *2022 8th International Conference on Signal Processing and Communication (ICSC)*. 2022, pp. 79–84. DOI: 10.1109/ICSC56524.2022.10009461.

[28] Nair, Aswathy K and Menon, Vivek. “Joint Channel Estimation and Symbol Detection in MIMO-OFDM Systems: A Deep Learning Approach using Bi-LSTM”. In: *2022 14th International Conference on COMmunication Systems NETworkS (COMSNETS)*. 2022, pp. 406–411. DOI: 10.1109/COMSNETS53615.2022.9668456.

[29] Nelson, Joseph. *What is ONNX?* https://blog.roboflow.com/what-is-onnx/. Accessed on May 14, 2023. 2021.

[30] Noisecom. *What is AWGN?* https://noisecom.com/resource-library/articles/artmid/1867/articleid/1233/what-is-awgn. Accessed on May 25, 2023. 2023.

[31] O’Shea, Timothy and Hoydis, Jakob. “An introduction to deep learning for the physical layer”. In: *IEEE Transactions on Cognitive Communications and Networking* 3.4 (2017), pp. 563–575.

[32] Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, Desmaison, Alban, Kopf, Andreas, Yang, Edward, DeVito, Zachary, Raison, Martin, Tejani, Alykhan, Chilamkurthy, Sasank, Steiner, Benoit, Fang, Lu, Bai, Junjie, and Chintala, Soumith. *PyTorch*. https://pytorch.org/. Accessed on May 27, 2023. 2019.

[33] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. *Scikit-learn: Machine Learning in Python*. 2011. URL: https://scikit-learn.org/stable/index.html.

[34] Proakis, John G. and Salehi, Massoud. *Digital Communications*. McGraw-Hill Education, 2007.

[35] PyTorch Tutorials. *Optimizing Model Parameters*. https://pytorch.org/tutorials/beginner/basics/optimization_tutorial.html. Accessed on May 26, 2023.

[36] PyTorch Tutorials. *PyTorch Tensors*. https://pytorch.org/docs/stable/tensors.html. Accessed on May 26, 2023.

[37] Ru, Xin, Wei, Li, and Xu, Youyun. "Model-Driven Channel Estimation for OFDM Systems Based on Image Super-Resolution Network". In: *2020 IEEE 5th International Conference on Signal and Image Processing (ICSIP)*. 2020, pp. 804–808. DOI: 10.1109/ICSIP49896.2020.9339375.

[38] Ryu, Jaeku. *Communication Technology*. Accessed on April 23, 2023. URL: http://www.sharetechnote.com/html/Communication_Equalizer.html.

[39] Saad, Haea. *Enhanced Channel Estimation for MIMO-OFDM in 5G NR*. Tech. rep. Accessed: 4 May 2023. 2021. URL: http://www.diva-portal.org/smash/get/diva2:1574059/FULLTEXT01.pdf.

[40] Sklar, Bernard. *Digital Communications: Fundamentals and Applications*. 2nd. Upper Saddle River, NJ: Prentice Hall, 2001, p. 953.

[41] Soltani, Mehran, Pourahmadi, Vahid, Mirzaei, Ali, and Sheikhzadeh, Hamid. *Deep Learning-Based Channel Estimation*. 2019. arXiv: 1810.05893 [cs.IT].

[42] Zaier, Aida and Bouallegue, Ridha. "Channel Estimation Study for Block - Pilot Insertion in OFDM Systems under Slowly Time Varying Conditions". In: *2009 6th International Multi-Conference on Systems, Signals and Devices*. IEEE. 2009, pp. 1–5.