Resumen

Este trabajo final de máster tiene como objetivo investigar la implementación de funcionalidades de inteligencia artificial/aprendizaje automático (AI/ML) para mejorar la precisión de posicionamiento de la localización de dispositivos. En este proyecto, se utiliza el nuevo sistema radio de quinta generación (5G) y se centra en el interior de las fábricas. Esta configuración objeto de estudio es un escenario relevante para el *Industrial Internet of Things* (IIoT) dentro del sector industrial, ya que las características específicas de este escenario crean un entorno desfavorable para un posicionamiento eficaz. Se trata de un escenario de implementación típico estandarizado en las versiones del *Third Generation Partnership Project* (3GPP).

El estudio se basa en el uso de un simulador MATLAB proporcionado para generar la información necesaria sobre las ubicaciones de los dispositivos inteligentes o *User Equipment* (UE) y sus mediciones de canal con las estaciones base 5G, los Puntos Transmisor Receptor. Este simulador crea el modelo de canal, la geometría del entorno y las señales de referencia de posición alineadas con el 3GPP. Con el objetivo de mitigar los efectos de la propagación multitrayecto, se han desarrollado diferentes métodos de AI/ML en Python. Se han explorado varios modelos AI/ML investigando diferentes entradas, como la respuesta al impulso del canal u otras características significativas del canal, así como varias salidas del modelo, como la posición directa del UE o los ángulos o tiempos intermedios de las señales de radio.

En consecuencia, este proyecto evalúa el rendimiento del posicionamiento asistido y directo con AI/ML frente a los métodos tradicionales en términos de precisión y complejidad, considerando al mismo tiempo diferentes estrategias de implementación. Se han simulado diferentes configuraciones de escenarios para evaluar la capacidad de generalización de los métodos de AI/ML. Además, otro objetivo ha sido estudiar la viabilidad real de la AI/ML para el posicionamiento de dispositivos 5G y, en ese caso, en qué dirección es más interesante investigar en el futuro.

Palabras clave: Inteligencia Artificial, localización, dispositivos IIoT, Python, señales de radio, 5G, 3GPP.

Resum

Aquest treball final de màster té com a objectiu investigar la implementació de funcionalitats d'intel·ligència artificial/aprenentatge automàtic (AI/ML) per a millorar la precisió de posicionament de la localització de dispositius. En aquest projecte, s'utilitza el nou sistema ràdio de cinquena generació (5G) i es centra en l'interior de les fàbriques. Aquesta configuració objecte d'estudi és un escenari rellevant per al *Industrial Internet of Things* (IIoT) dins del sector industrial, ja que les característiques específiques d'aquest escenari creen un entorn desfavorable per a un posicionament eficaç. Es tracta d'un escenari d'implementació típic estandarditzat en les versions del *Third Generation Partnership Project* (3GPP).

L'estudi es basa en l'ús d'un simulador MATLAB proporcionat per a generar la informació necessària sobre les ubicacions dels dispositius intel·ligents o *User Equipment* (UE) i els seus mesuraments de canal amb les estacions base 5G, els Punts Transmissor Receptor. Aquest simulador crea el model de canal, la geometria de l'entorn i els senyals de referència de posició alineades amb el 3GPP. Amb l'objectiu de mitigar els efectes de la propagació multitrajecte, s'han desenvolupat diferents mètodes d'AI/ML en Python. S'han explorat diversos models AI/ML investigant diferents entrades, com la resposta a l'impuls del canal o altres característiques significatives del canal, així com diverses eixides del model, com la posició directa de l'UE o els angles o temps intermedis dels senyals de ràdio.

En conseqüència, aquest projecte avalua el rendiment del posicionament assistit i directe amb AI/ML enfront dels mètodes tradicionals en termes de precisió i complexitat, considerant al mateix temps diferents estratègies d'implementació. S'han simulat diferents configuracions d'escenaris per avaluar la capacitat de generalització dels mètodes d'AI/ML. A més, un altre objectiu ha sigut estudiar la viabilitat real de l'AI/ML per al posicionament de dispositius 5G i, en aquest cas, en quina direcció és més interessant investigar en el futur.

Paraules clau: Intel·ligència Artificial, localització, dispositius IIoT, Python, senyals de ràdio, 5G, 3GPP

Abstract

This master thesis aims to investigate the implementation of artificial intelligence/machine learning (AI/ML) functionalities for improving the positioning accuracy of device localization. In this project, the fifth-generation (5G) New Radio system is used and focuses on indoor factories. This setup held under study is a relevant scenario for the Industrial Internet of Things (IIoT) within the industrial sector, as the specific characteristics of this scenario create a disadvantageous environment for effective positioning. It is a typical deployment scenario standardized in Third Generation Partnership Project (3GPP) releases.

The study is based on using a provided MATLAB simulator for generating the required information about the smart devices or User Equipment (UE) locations and their channel measurements with the 5G base stations, the Transmitter Receiver Points. This simulator creates the channel model, environment geometry, and position reference signals aligned with the 3GPP. Pursuing the goal of mitigating the multipath propagation effects, different AI/ML methods have been developed in Python. Several AI/ML models have been explored investigating different inputs, such as the Channel Impulse Response or other significant channel features, as well as various model outputs, such as the direct UE position or intermediate angles or times of the radio signals.

Consequently, this project evaluates the positioning performance of the assisted and direct AI/ML positioning versus the legacy methods in terms of accuracy and complexity while considering different deployment strategies. Different scenario configurations have been simulated regarding the generalization ability of the AI/ML methods evaluation. Furthermore, another objective has been studying the actual viability of AI/ML for 5G device positioning and in that case, which direction is more worthwhile for future investigation.

Keywords: Artificial Intelligence, localization, IIoT devices, Python, radio signals, 5G, 3GPP.

List of Acronyms

3GPP	3rd Generation Partnership Project
5G	Fifth-Generation
AI	Artificial Intelligence
AoA	Angle of Arrival
AoD	Angle of Departure
BS	Base Station
CDF	Cumulative Distribution Function
CIR	Channel Impulse Response
CNN	Convolutional Neural Network
CRLB	Cramer-Rao Lower Bound
DL	Downlink
FNN	FeedForward Neural Network
FR1	Frequency Range 1
FR2	Frequency Range 2
gNB	gNodeB
GPS	Global Positioning Systems
IIoT	Industrial Internet of Things
IoT	Internet of Things
KPI	Key Performance Indicator
LMF	Location Management Function
LOS	Line-of-Sight
ML	Machine Learning
NLOS	Non Line-of-Sight
NN	Neural Network
NR	New Radio
PDF	Probability Density Function
PDP	Power Delay Profile
RAT	Radio Access Technology
RSRP	Reference Signal Received Power
SNR	Signal to Noise Ratio
ToA	Time of Arrival
ToD	Time of Departure
TRP	Transmission and Reception Point
UE	User Equipment
UL	Uplink

Contents

1 Introduction

1.1 Background and Motivation

Location Services

Device or User Equipment (UE) localization is the ability of a location server to determine the position of a connected device inside a coordinate system. This ability is important for a wide range of applications, such the industrial, navigational, commercial, or emergency services [1]. For obtaining the UE position, there are several location services.

One of the most popular technologies for UE positioning of the past decades has been the Global Positioning System (GPS). It was started in the 1970s by the US Department of Defense as a navigation aid for the military. After that, it was improved and opened for civilian purposes around the 1980s. During this decade, mobile phones were also introduced to society, and from that moment, location services gained a lot of popularity. Since then, location services have become an essential aspect of telecommunications, continuously improving the accuracy and reliability of positioning methodologies. GPS also presents several drawbacks, such as its high location errors and low reliability for some specific areas or scenarios, such as tunnels or indoor buildings.

Since the advent of GPS technology, location services have undergone significant advancements and transformations. One significant development has been the integration of GPS with other positioning technologies, such as Wi-Fi and cellular network-based positioning. The rise of 5G networks, advancements in satellite navigation systems, autonomous vehicles, and the Internet of Things (IoT) will further shape the landscape of location-based services.

In recent years, the industry has been experimenting with a trend towards more interconnected and smarter automated factories, Industry 4.0. This advent created a surge in the use of intelligent devices that gather data and require monitoring to ensure accuracy and reliability in their given industrial environment. These devices are the well-known Internet of Things. The Industrial Internet of Things (IIoT) helps communications, automation, and self-monitoring inside the factories. The IIoT allows new data exchange in the production processes, logistic systems, and manufacturing environments, offering the possibility to organize them with less human intervention.

Positioning accuracy is one of the most important Key Performance Indicators (KPIs) in IIoT use cases. One use-case example can be the positioning inside a smart logistic center, where it is essential to locate assets and moving objects on the warehouse floor for the regular operations of the automated transportation systems. From the navigation robotic perspective, knowing the location information is also required for robust and safe navigation of intelligent systems. In this context, 5G benefits IIoT smart devices, providing the opportunity to achieve higher levels of mobility, flexibility, reliability, and security inside the factories [2].

Enhancing the current location accuracy of robots or other smart devices inside the factory could make a significant impact. It could improve the performance of different industrial tasks, helping to move toward a more automatic, efficient, and smarter industry. For these reasons, this project focuses on the indoor factory scenario, where it will be really beneficial to have accurate UE device positioning. The efficient location of smart devices in an indoor factory can help with space usage optimization, real-time alerts and monitoring, energy efficiency, or workflow plan improvement of the manufactured or stored products.

5G and Beyond Systems

UE positioning in 5G systems can be performed using radio-based signals e.g., instead of using GPS. UE location of the devices can be determined based on the positioning measurements based on reference signals from one or more gNBs or devices (uplink or downlink signals) [3]. Positioning can also be performed by measuring the time it takes for signals to travel the distance to the receiver and backward. The measurements for positioning can be by means of Received Signal Strength Indicator (RSSI), ToA, or AoA. Accurate positioning is typically obtained in case there are multiple Light-Of-Sight (LOS) links between the UE and the neighbor gNBs to compute the position estimation using multi-lateration techniques, as explained later.

One of the most challenging scenarios is the indoor scenario, such as smart industry factories. It is often difficult to distinguish the characteristics of the links where there are multiple paths between the radio transmitter and the receiver. False detection could happen e.g. when identifying a noise spike as a LOS component or in dense multipath environments when observing a Non-Light-Of-Sight (NLOS) link and using it as a LOS component. The false detection leads to poor positioning accuracy when deriving the position of the UE based on these measurements.

Solving these problems is currently being discussed in the 3rd Generation Partnership Project (3GPP) implementation of Artificial Intelligence/Machine Learning (AI/ML) algorithms due to the fact that they could be adequate for positioning in 5G and 6G technologies. AI/ML methods are known as important tools that have been widely used in various applications due to their ability to identify trends and patterns, their versatility, accuracy, and capability to handle multidimensional and different data types. Hence, considering the deployment of AI/ML techniques could improve UE positioning accuracy.

These AI/ML methods and algorithms could be considered beyond 5G systems, particularly to improve positioning accuracy to achieve the required accuracy for specific use cases [3]. Moreover, UEs in this context may have better performance than UE in previous communication generations before 5G. Together with the gNB's high processing capability, the possibility to obtain rich radio channel characteristics beyond 5G systems can facilitate/enable the deployment of these AI/ML techniques.

Finding an improved solution for positioning systems could be beneficial. Current systems do not meet the requirements for some variety of scenarios in which they must operate. These limitations are more apparent in indoor factory environments, where there are large metallic objects and dense cluttering that interfere and cause a low accuracy positioning. Hence, it is relevant to develop a more reliable and precise positioning system that can operate effectively in challenging scenarios only making use of the 5G radio signals and its features.

1.2 Methods and Objectives

Goals and objectives

The aim of this project is to enhance the accuracy of legacy UE localization techniques using AI/ML methods. Device location in a 5G network faces challenges such as environment variations, shadowing, scattering, and blind spots, generating NLOS situations. This project aims to cope with these issues by implementing AI/ML methods. This objective also implies investigating the viability of the solution itself and the direction for future work in the same field. Hence, this report aims to find an approach for a viable solution that settles up the first steps of the promising directions toward the future of AI/ML within 5G and beyond systems.

Mainly, the project is focused on the objective of investigating the feasibility of deploying the AI/ML model on the UEs side. The objective is for the AI/ML models to achieve better accuracy than the existing location methodology for some scenarios. As the first milestone, the benefit of AI/ML for positioning versus legacy methods has to be proven. The next challenge has been the study of where the AI/ML model should be deployed. Together with that, various deployment aspects have to be taken into account such as the signaling procedure, computation load, memory, power consumption, and dataset collection.

Real-life scenarios can present different adversities, such as a rich amount of NLOS, diverse clutter parameters, or synchronization errors. This project's main scenario is an Indoor Factory Deployment (e.g., with Dense Clutter with High BS Height (InF-DH)) because of its challenging nature in terms of NLOS links probability. Hence, the channel model will mostly consider the indoor channel model. Furthermore, the model will be exposed to different conditions, such as different drops or different clutter parameters.

Not only the AI/ML model structure has been investigated, but also the input data selection, the preprocessing, and the potential postprocessing. All of the different possibilities create a work environment with many degrees of freedom, but due to time and material constraints, only part of them could be investigated. For evaluating the differences in performance and accuracy, a comparison between the results with the proposed AI/ML positioning solution and the legacy methods has been conducted.

Finally, another goal of this project is to emphasize the relevance of an accurate and robust positioning methodology using 5G networks for the factories of the future. In the project context, in an indoor factory, enhanced positioning techniques could improve the tracking of equipment, vehicles, workers, or assets. This potentially benefits some typical industrial aspects such as process optimization, autonomous systems, assets tracking, or worker safety, among others [4]. Hence, this project also aims to discuss how enhanced 5G localization methods could improve the current efficiency, productivity, and safety of typical factory operations.

Approach and Methodology

The first step of the project has been investigating and studying the available literature of previous work, the current 3GPP discussion, and related topics, including the specifications of the current communication system, the different ranging techniques, available localization methods, and suitable ML/AI algorithms.

The research has been based on the RAN1 3GPP [5], which has already been investigating the field and has started working on further standardization. Following the 3GPP perspective, there are two different approaches for implementing the AI/ML positioning methods based on the output given by the AI/ML model. There is direct and assisted positioning:

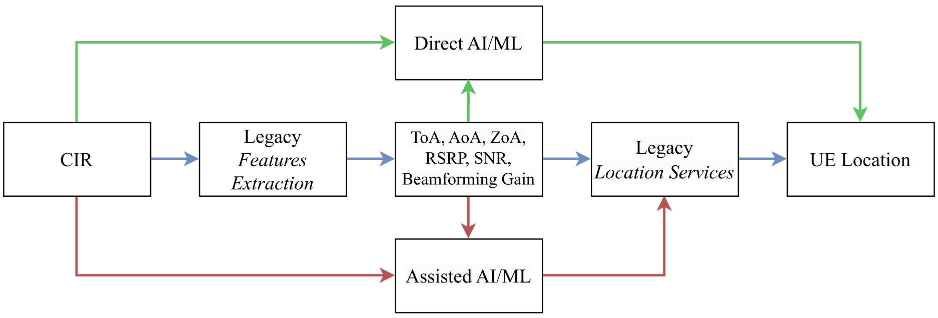


Figure 1.1: *Schematics of the AI/ML assistance in the UE positioning process*

Direct AI/ML positioning method e.g., fingerprinting, produces the UE location directly as the output based on different parameters and characteristics of the channel. It can be seen as a mapping relationship between the channel features and the location coordinates.

On the other hand, the assisted AI/ML positioning does not produce the UE location. It estimates some intermediate relevant parameters, such as NLOS/LOS probability, ToA, or AoA. This information will be subsequently used by the legacy or methods to compute the UE position. The assisted AI/ML positioning enhances the LOS/NLOS identification to assure that the inputs given to the geometry-based method improve the final location accuracy. Part of the research has also been focused on evaluating the different input options for obtaining the outputs that give the best performance.

The present thesis project started more focused on assisted AI/ML positioning but then ended up studying also the possibilities and performance of the direct AI/ML positioning methodology.

The AI/ML model research and creation stage also included the generation of the training and inference datasets. These datasets consist of simulated UE positions and their corresponding channel measurements with the gNBs for different relevant scenarios. The generated data follows the temporal and spatial consistency procedures for getting the correct correlation between parameters. In this phase, data investigation and preprocessing have also been conducted, as well as the accuracy limitation. The created AI/ML model has been trained and subsequently hypertuned to optimize their performance.

The last part of the project has been mostly analyzing the proposed solution's performance and limitations. The evaluation has been based on accuracy and computation efficiency, generalization capability and feasibility for future work. Moreover, a study and a general discussion about its industrial applicability will be performed. The purpose is to research and reflect on the areas and operations within a factory that could potentially benefit from an enhanced location system.

1.3 Platform and tools

For the development of this project, several tools have been used. It can easily break down the project into two main parts: the data generation of the simulated scenarios part and the creation of the AI/ML algorithms and their performance evaluation.

The first task, the generation of the dataset for the interesting scenarios, was achieved with MATLAB software and the simulator developed at Sony. This MATLAB simulation tool applies the channel model e.g., including parameters agreed on the 3GPP to generate the needed signal measurements between the different gNBs and UEs of the selected scenario.

On the other hand, for processing the data obtained, for coding the AI/ML algorithms and its posterior training and inference performance Python was used. This programming language was selected due to its versatility with big datasets management and all the possibilities that it offers for machine learning algorithms creation. Some of the most relevant libraries used are Pandas, Scikit-learn, TensorFlow, Numpy, Scipy, and Matplotlib.

In the last stages of the project, access to a big Sony server was provided. This fact makes it easier to run the hyperparametrization of the models as well as the creation of different datasets to evaluate the generalization of the solutions to different scenarios.

1.4 Outline

In this section, an overview of how the report of this project is organized is presented:

- **Background: Chapters 2 - 3**. Chapter 2 is focused on the different measurements and algorithms for 5G localization and the 3GPP approaches for enhancing them. During Chapter 3, an overview of ML and the architectures of NN and CNN, as well as an introduction to optimization and dataset management, is presented.

– **Methodology: Chapters 4 - 5**. Chapter 4 introduces the used simulator configuration and the characteristics of the scenario, and its assumptions. Chapter 5 explains the different AI/ML model architectures developed for 5G positioning and their properties.

– **Results and conclusions: Chapters 6 - 7**. In Chapter 6, the results of the performance of the AI/ML methods are presented together with a discussion. The last Chapter 7 summarizes the conclusions of the projects and proposes some future work actions.

Finally, the Appendix contains for each model architecture the schematics, learning curves, and graphics of the performance, such as CDF for the estimation or Confusion Matrix for the label prediction. In the Appendix can also be found some figures of the CIR present the different approaches for processing it.

2 UE positioning in 5G NR

The positioning system is normally divided into two phases, the measurements and positioning estimation phases. During the measurement, as its name indicates, the different parameters of the channel are measured, obtaining the CIR and other relevant parameters of the channel. Once these parameters are obtained by measurement, the localization stage begins. There exist different techniques for doing this measurement presented subsequently [6] [7].

2.1 Architecture Overview

The architecture of this project follows the concepts in Figure 2.1 [8]. In 5G NR positioning architecture, there are several key concepts involved: UE, gNBs, and LMF. Each one of these elements has its responsibilities depending also on the model, as will be explained later in this Chapter. Also, the operations performed by each element depend on the system characteristic (Uplink or Downlink).

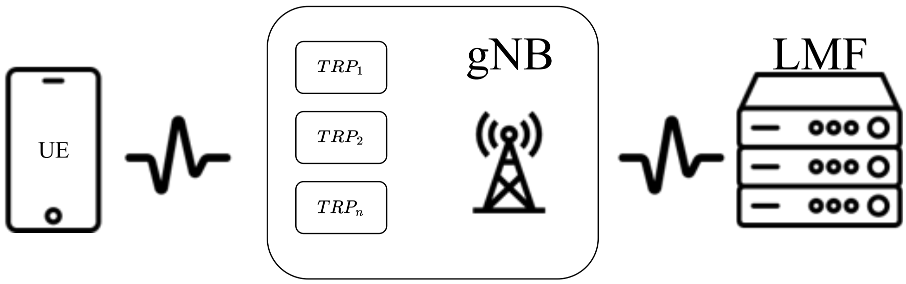


Figure 2.1: 5G Transmission and Reception Point Architecture

The overall operation for positioning can be summarized as follows:

- First, there is a **positioning request** initiated by UE/gNB. Following the previous Figure 2.1, the UE performs the request.
- After this request, a **Positioning Measurement Request** is performed. The gNBs and LMF coordinate to perform the positioning measurement. The gNBs play a crucial role in gathering the needed measurements, such as time, angle, or signal strength, from multiple resources.
- Once these measurements are obtained, the gNBs **transmit** the measurements to the LMF. This element receives and estimates the positioning from the measurements using different algorithms to determine the UEs location (in this example). The estimation may involve techniques like TDOA or AOA, even hybrid approaches. There exist different supported techniques [9].

2.2 Basic measurements for positioning systems

Before the localization phase, the measurements of angles, times, or power of the radio signal are needed. In the 3GPP TR 38.805 [9], different types of radio-based positioning solutions are introduced and standardized. This section introduces various measurements that are necessary to provide in the context of the 5G scenario.

AoA

The Angle of Arrival (AoA) is a technique that determines the angle at which received signals arrive at the receiving antenna. It provides information about the direction and the azimuth angle of the original transmitted signal. It gives the benefit of having directional information, which is useful for positioning problems. Also, another advantage of AoA is the capability and information it brings regarding a multipath scenario, as multipath components arrive from many different directions.

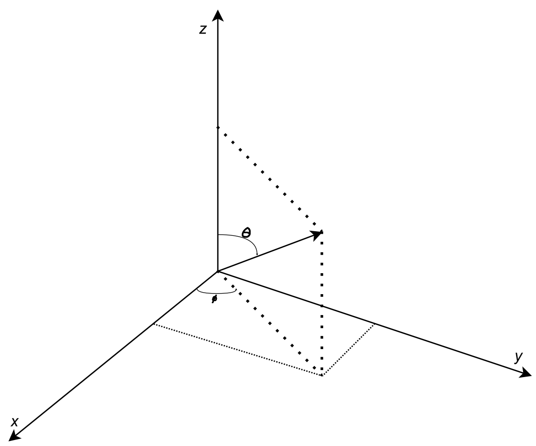


Figure 2.2: Definition of Azimuth and Zenith ((ϕ and θ)

This parameter is fully dependent on the characteristics of the antenna due to a multiple antenna array is needed to capture the direction of the incident signal. By computing the steering vector, it can be obtained the direction in which the signal is arriving.

ToA

Time of Arrival (ToA) refers to the time it takes an electromagnetic signal to complete the distance from the transmitter to the receiver. It does not have to be the shortest distance but the distance allowed by the scenario. Technically, it is determined by measuring the time difference between transmission and reception, giving us the TDoA, calculated by comparing the phase and time of a PRS sent to determine time.

TOA is a crucial parameter for distance calculation, applying the equation of velocity. It is also important to calculate the Ranging for a unique antenna, allowing us to know by the time in which range where the UE device can be located.

TDoA

Time Difference of Arrival (TDoA) is based on the usage of different ToAs and their subtraction, giving the timing difference. This parameter is used for precise positioning applications, as it works well with a large number of receivers, leading to an environment with a higher amount of obstacles or NLOS. Synchronization has to be ensured among the different transmitters.

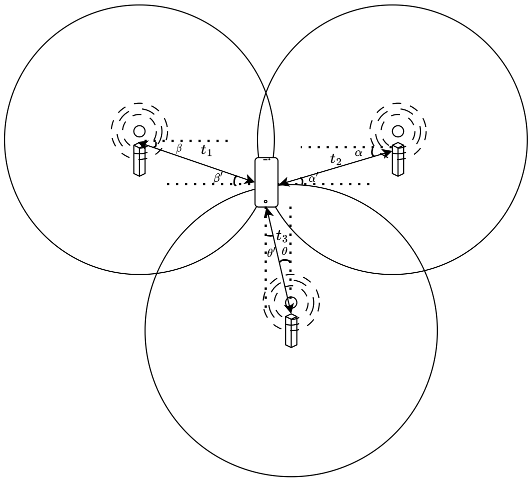


Figure 2.3: *Definition of ToA/D, AoA/D*

RSSI

RSSI stands for Received Strength of Signal Indicator and represents the received power of the received signal, typically measured in dB. It is really useful for proximity detection and for establishing a range of possible positions for the UE device.

The parameter might not bring full information about the communication link as it is affected by several factors such as distance, weather conditions, obstacles, and interference. For the studied indoor scenario of the project, it can be assumed that no weather conditions will affect the propagation properties, nor will interference exist from external sources.

2.3 Challenges from indoor scenarios

The main problem of the indoor scenarios is obtained due to the nature of the reflection of the transmitted and received waves across space, as mentioned in the previous chapter. The existence of more "blocking objects" in the channel makes a huge impact on the received signal.

2.3.1 LOS/NLOS

The appearance of the blocking objects generates that sometimes is not possible to have a direct link between the transmitter and the receiver. The wireless communication channel has the disadvantage of the appearance of metallic objects that generate multipath components or shadowing behavior. Having a direct Line of Sight (LOS) means that the signal has a stronger component due to the direct vision. In contrast, Non-line-of-sight (NLOS) generates not having a strong component together with different multipath components having similar magnitude, with the behavior on the distribution explained in the previous chapter.

In the CIR, the existence of a LOS/NLOS scenario can be observed by the reception of the different multipath components in the delay domain for a certain timestamp. A CIR with a high magnitude component and low delay values means a high probability of the existence of a LOS component. Although, NLoS component can be reflected in the CIR as a high noise graph with multiple low-medium density components at the whole range of delay values.

2.4 3GPP AI/ML positioning

3GPP has discussed since Release 16 the importance of future steps regarding positioning for 5G networks, concretely this project focuses on indoor scenarios. As said in the introduction, this is an interesting research field due to the reality of GPS coverage inside buildings. It is a research topic for which different companies are tackling the problem.

Nevertheless, it is not until the 3GPP Release 18 that the introduction of AI/ML methods for NR interface is studied. The proposed AI/ML methods are investigated for CSI feedback enhancement, beam management and positioning accuracy enhancement. This last one is the AI/ML implementation on which this project is focused.

The positioning system is formed mainly by three agents, the User equipment (UE), gNB (5G base station), and the Location Management Function (LMF). Each of the components has been held under research, and also in relation to the positioning model where it should be deployed. Under this premise, and looking at Figure 2.4, the system relation between different blocks proposed in the 3GPP papers can be seen.

The positioning model process is shown in Figure 2.4. First, there is the dataset construction based on UE or gNB input measurements from simulations or direct field data. This data is sent to two different blocks, the model management block and the model inference block. The first one is divided into four different parts:

- **Model acquisition**. It refers to the process of obtaining ML models, either by selection or design of the model.
- **Model (de)activation**. It is the process of activating the deployed model in the system, after acquisition, training, and validation.

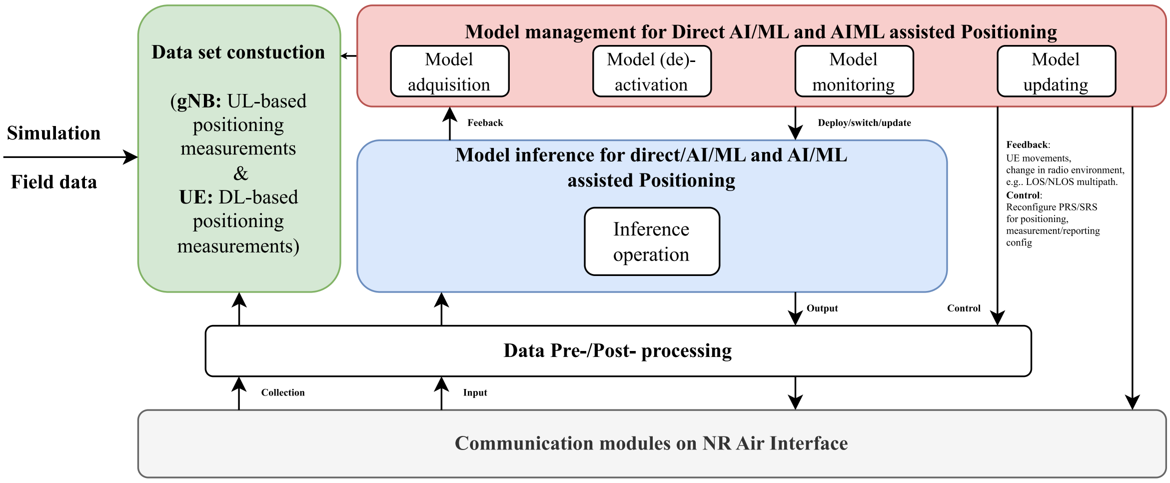


Figure 2.4: *Schematics of the AI/ML methods for 5G UE positioning functionalities*

- **Model monitoring**. It refers to the process of evaluating the performance of the previously activated model to asses how the model is functioning in the real world.

- **Model updating**. If the deployed model does not fulfill some performance levels and needs an update, such as fine-tuning for improving it.

Another vital process that can be observed is the processing part. Preprocessing and postprocessing are vital in the data analysis pipeline to ensure accuracy and meaningful results. It helps to improve data quality, making it more suitable for the model using preprocessing. Additionally, for refining the predictions, postprocessing can give valuable insights into the model output.

Furthermore, it can be found the Model Inference block is in charge of the application of new unseen data to the previously trained model in order to produce the desired output prediction. In this block, the input data can vary depending on the application and type of model that is being used, but it is referred to the normal activity the model will face when deployed.

Finally, all blocks have a division between direct and assisted positioning. This difference in the way of positioning comes from where the model is deployed in the system.

2.4.1 Direct positioning

Direct positioning is a technique used to estimate the exact position of the UE using the received signals without prior knowledge of the environment. This input consists of measurements obtained from the wireless channel, with features previously mentioned, such as RSSI, TOA, or AOA, together with the characteristics of the channel given by the CIR.

After gathering the data, the next step is to process it and introduce it to an AI/ML

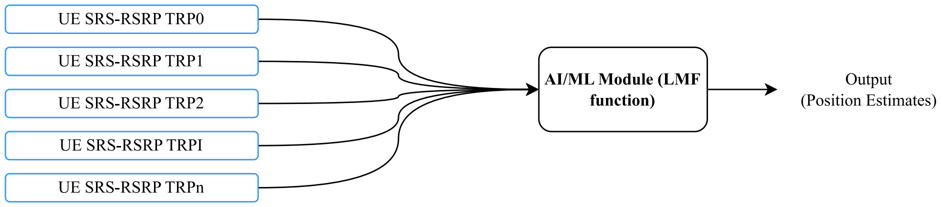


Figure 2.5: Direct Positioning Scheme

model that computes the estimated position of the UE. The model can be allocated on the UE-side, gNB-side, or at the LMF, such as in Figure 2.5

2.4.2 Assisted positioning

Assisted positioning is another technique that enhances the accuracy of positioning by estimating some intermediate parameters of the channel, such as time, angle, or power signal. In this report, these parameters will be called channel features and will be further analyzed in Chapter 4.

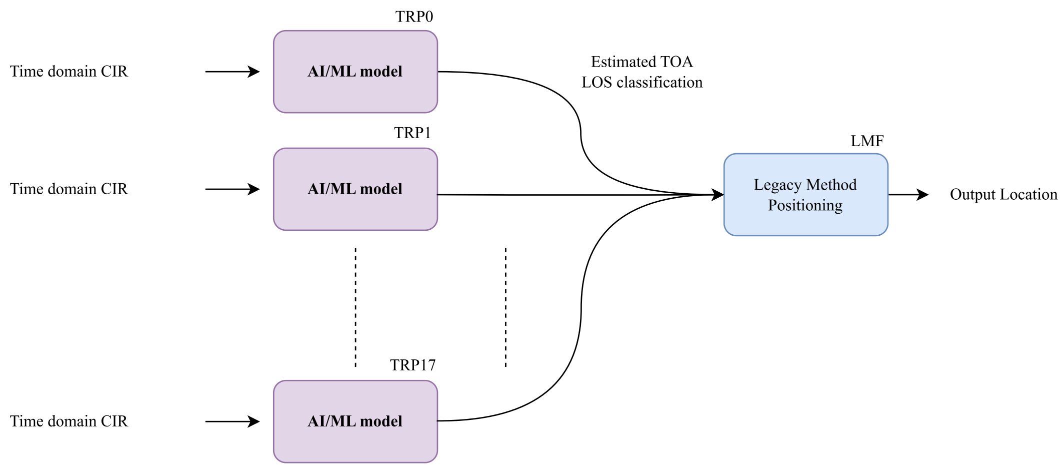


Figure 2.6: Assisted Positioning Scheme

The model again can be allocated inside the gNB or UE, collecting the received data and predicting some parameters for helping with the position calculations. Figure 2.6 displays an assisted positioning scheme, specifically in this case, with the model deployed at the Transmitter Receiver Points (TRPs) or gNBs.

Nevertheless, the assisted methods can be deployed at the gNB or UE side, as the system can work both in UL and DL scenarios. The input data can include different types of information, such as CIR or reference signal measurements, along with channel features that demonstrate behavior of the same nature.

3 Machine Learning Overview

3.1 Introduction to Machine Learning

Machine Learning (ML) is encompassed within the field of Artificial Intelligence (AI) and describes the algorithms that have the ability to learn patterns from data and the correlation of the different input features with the outcome. A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E [10].

In the beginning, AI basically consisted of hard-coding the formal and mathematical rules and relations that describe and solve some tasks by automating them. This made the computer able to solve a series of problems, not only easy and repetitive ones but also intellectually highly difficult ones for human beings. All were inside a sterile and controlled environment, facing a lot of difficulties when requiring the computers to have real-world reasoning or more subjective decision-making.

ML appeared from the need for the systems to acquire their own knowledge, extracting the describing patterns directly from raw data. Gathering it from experience prevented the human operators to hard-code and formally specify all the knowledge the algorithm needs.

Overfitting and Underfitting

During the training of the ML algorithms, there are several problems that can impact the performance and generalization of the model. Two of the most common problems that should be taken into account are overfitting and underfitting.

On the one hand, overfitting occurs when the ML system has found and learned patterns in the training data that are not representative of the wider dataset. It can be seen that the model has memorized some inherent patterns of the data rather than the underlying and relevant relations for the specific task. This phenomenon leads to an ML algorithm that may perform well on the training data but poorly on new and unseen data.

There are several possible causes for overfitting. The most common is having a model with too many parameters relative to the amount of training data, leading to a too complex model or also a model that has been trained for too long. There exist some techniques to avoid or mitigate its effects, such as the early stopping or implementing dropout on the layers.

Underfitting, on the other hand, happens when the model is too simple to be able to

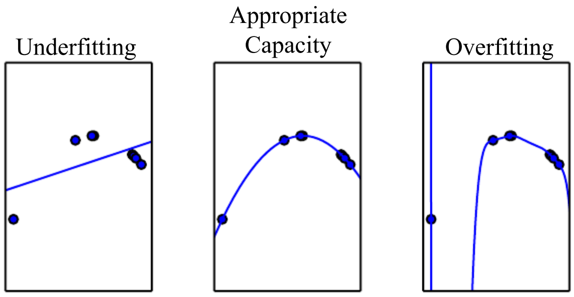


Figure 3.1: *Exemplification of three models underfitting (left), well-adapted (center), and overfitting (right) the data*

capture the underlying patterns of the data that describe the relation to predict. It can be viewed as a model that is too general to learn and capture all the important relations and features. Underfitting leads to a model that performs poorly both on the training and unseen data.

3.2 Learning Strategies

When it comes to training the algorithm, there are different learning strategies varying depending on the type of methodology applied or the availability of data. This project is focused on the supervised one, but there also exists unsupervised learning, when the expected output is never provided to the algorithm, and also the reinforcement one, similar to how humans learn with a signal system of reward and penalization [11].

Supervised

In this type, the algorithm is provided with a set of inputs, and also with the expected outputs for every input. The objective of this methodology is that during the training, the algorithm modifies its own parameters to adapt, in the best possible way, its own outputs to the expected ones. After the training, the idea is that it has already learned the association or mapping between the input and output, so when giving new inputs, it should predict the correct labels or classes.

3.3 Neural Networks

A Neural Network (NN) is a ML algorithm that process the input data and generates an output response. The name is due to the fact they try to artificially imitate how the biological nervous systems process the information. For instance, the human brain is a very complex, parallel, and non-linear system, which the NNs try to imitate.

The architecture of this algorithm is a group of interconnected neurons that are organized in different layers. The resulting network is able to learn from its experience, plastically adapt to changes, and with a non-linear behavior that allows it to process information resulting from non-linear phenomena.

Parts and Architecture

- **Neuron**: The simplest element forming the algorithm is a mathematical linear function. This equation gives an output proportional to the input, depending on the weight (that will change with the training) and an independent variable.

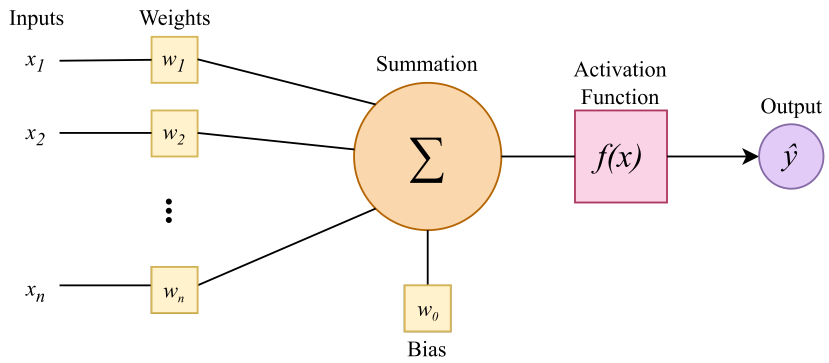


Figure 3.2: *Schematics of the neuron parts*

- **Activation Functions**. They are responsible for changing the linear behavior into non-linear. This allows the networks to adapt to the non-linearity behavior of the working data or phenomena. There exist different types of activation functions. Some of the most used ones are: *sigmoid*, *hyperbolic tangent*, or *ReLU*.

- **Fully Connected Layer**. It is the set or group of neurons that are not connected to each other but to the previous and next layers. During the training, each layer specializes itself to detect certain concrete characteristics or patterns. As the complexity of the problem increases, more neurons might be needed, organized in wider layers or into more layers. There are only two layers with a restricted number of neurons, which are the input for which the number of neurons matches the number of inputs, and equivalently for the output layer.

- **Dropout**. It consists of a regularization method that can be used to prevent overfitting and improve generalization performance. The dropout technique is based on randomly dropping some fraction of the neurons in a layer during each training iteration. The fraction of neurons that will be dropped during the training is defined beforehand. This way method forces the remaining neurons of the layer to learn more robust features and reduces the dependence on certain neurons or sets of neurons.

Grouping and connecting different fully connected layers together, with an input and output layer, gives as a result the known Neural Network (NN). The NN is sometimes also referred as Multilayer Perceptron.

3.4 Introduction to Deep Learning

The evolution of neural networks through a more complex architecture has led to algorithms of Deep Learning (DL). Networks with greater depth are able to execute a higher number of instructions in sequence, giving them greater capacity due to their later instructions being referred to earlier instructions. More depths and more multisteps allow DL to reach more abstract and complex mapping to extract from the input [12].

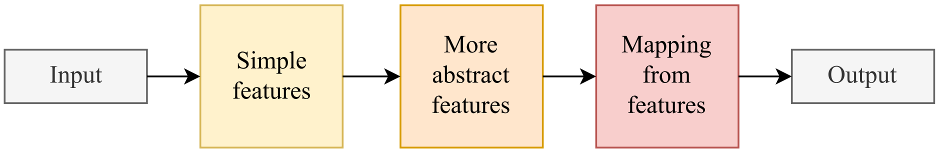


Figure 3.3: *Flowchart showing the parts of a Deep Learning system*

DL is a subfield of Machine Learning focused on training deep neural networks. One NN is considered a deep neural network depending on the depth of the network architecture. Typically, a NN with three or more layers is inside the DL domain as it has one or more hidden layers.

DL has a greater amount of composition of functions than traditional ML, making it a powerful and flexible tool for representing nested and hierarchy functions. It can build out complicated functions from simpler ones and their relation.

3.4.1 CNN

As presented before, the input to a NN has to be an array or matrix of values. However, a NN can also be able to process information from inputs with more dimensions, but first, it needs to experience some spatial transformations. The described behavior can be performed by a Convolutional Neural Network (CNN).

CNN is a specific type of NN that implements convolutional layers that are responsible for performing the transformations needed to process the information from a multidimensional input. Following, the architecture of the convolution block of a typical CNN is presented.

Parts and Architecture

- **Convolutional Layers**. They consist of several filter or "kernel" layers of one or more dimensions, which will apply several transformations to the input data. They are responsible for extracting the different characteristics or features from the input information. The convolution layer is followed by an activation function as the regular NN to transform the output into a non-linear function.

- **Pooling Layers**. In the features extraction phase, alternating with convolutional layers, there are the reduction layers. They are responsible for reducing the dimension of the input data, making the later layers less sensitive to disturbances on the input but active to more complex and abstract features. There exist several types of dimension reduction layers or pooling. The most popular are maximum or average, preserving in that way the most relevant characteristics the filter has detecting depending on the application.

– **Flattening**. The feature map will be multidimensional, so a flattering will be needed to be able to feed the regular NN already presented (which cannot have multidimensional inputs).

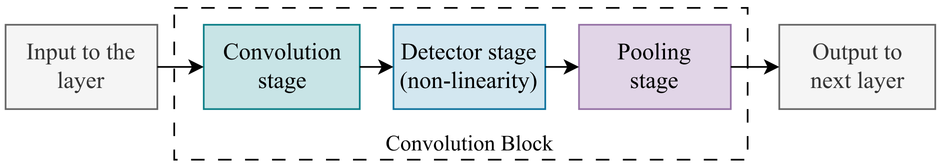


Figure 3.4: *Flowchart showing the typical stages of a convolution block*

Finally, the flattering layer is followed by a regular NN that will continue learning patterns from the extracted complex features. For a better understanding of the functionality of CNN, one example could be processing RGB images. The first stage of the network is responsible for detecting simple features such as edges or corners and their orientation. As the network goes deeper, the convolution layers are able to detect more complex features such as textures, patterns, or shapes. After, the extracted feature map is fed into the NN that learns from it to perform the specific task, such as classify or estimate.

3.5 Optimization for Machine Learning

For learning and improving the performance of the overall NNs, the key can be found inside the linear functions of each neuron. These linear functions contain a series of weights (w) that can be modified to adapt the response of NN to a given input. Hence, the training of the models becomes a gradient-based optimization problem of the weights of each linear function inside each neuron. In this way, the weights can be seen as the optimal parameters θ to be found for reducing a cost or loss function $J(\theta)$, which typically includes a performance measure evaluated on the training set [12].

The selection of the loss function is really dependent on the problem to be solved. There exist many ways to evaluate the performance of the model depending on the nature of the task. There is also the possibility to create a custom loss function that can be adequate to the needs of the specific problem. This one can be a combination of other loss functions or a completely new creation. Below some of the most used loss function is presented. The election depends on the problem nature and the NN architecture used.

– For regression problems: Mean Squared Error (MSE) or Mean Absolute Error (MAE).

– For binary classification problems: Binary Cross-Entropy.

– For categorical classification: Categorical or Sparse Categorical Cross-Entropy.

Regarding the modification of these weights, it is a task done using different optimization techniques that identify which and how each neuron is affected. There exist several algorithms to find the needed weight values for better performance in the final prediction. Some of the most popular ones available in Python are [13]:

– **Stochastic Gradient Descent (SGD)**. A basic type of optimizer that performs the gradient descent with a fixed learning rate.

– **Root Mean Square Propagation (RMSprop)**. An optimizer based on the SGD, but instead of having a fixed learning rate, the RMSprop divides it by a moving average of the root mean square of the gradients.

– **Adaptive Gradient Algorithm (Adagrad)**. It adapts, in this case, the learning rate for each parameter based on its historical gradients.

– **Adaptive Moment Estimation (Adam)**. This optimizer uses an adaptive learning rate for each parameter individually. It can be seen as a combination of RMSprop and Adagrad.

Early Stopping

As mentioned before, one method to prevent overfitting is implementing early stopping in the training phase. It is a regularization technique that consists of stopping the weights optimization over the training data before it starts overfitting the data. For this, during the training, the validation performance is monitored to stop the training when it degrades or reaches a plateau. In practice, a common usage of early stopping is to save the parameters associated with the lowest validation error and use those instead of the last iteration ones as final model parameters.

3.5.1 Hyperparmeters Optimization

The majority of deep learning algorithms have several hyperparameters that control different aspects of their behavior. These hyperparameters can significantly affect time, memory cost, or performance quality, so choosing the right values is an essential task. The hyperparameters are the values set before training, such as the number of neurons per layer, number of layers, learning rate, dropout, batch size, or type of activation function. This is itself an optimization problem where there exist some good values of these hyperparameters that minimize an objective function, such as validation errors, sometimes under some constraints.

There exists two basic different approaches for finding the right values of each hyperparameter: choosing them manually or automatically. Making the selection by hand requires a deep understanding of what each hyperparameter does and how the model can achieve a good generalization, while automatic selection reduces this need but is much more computationally costly.

The objective of the hyperparameter optimization is to find the values for each hyperparameter that maximizes the performance of the model using the given dataset. For using automatic hyperparameter selection, the first step is defining the range of values that each hyperparameter can take, known as hyperparameter or search space, and then applying one of the automatic search algorithms. There exist several techniques based on the way the algorithms explore the hyperparameter space. Some examples are:

- **Grid Search**. It is a simple approach based on defining a grid of hyperparameter values (a finite set of the search space) to search through exhaustively. It becomes really computationally expensive if the number the hyperparameters is high.
- **Random Search**. This technique involves randomly sampling the hyperparameter space, which is more efficient than a grid search for high-dimensional spaces.
- **Bayesian Search**. This optimizer models one objective function, such as the validation accuracy, as probability distribution and uses this distribution to guide the search for good hyperparameters.

3.6 Dataset Management

The process of data management involves the collection, processing, and analyzing the data before feeding it to the ML to ensure a better and more meaningful pattern extraction. The quality and format of the data are crucial for creating and training an accurate and robust model.

After gathering the needed data, the next step is to ensure its quality. This includes removing outliers and detecting missing values or other anomalies. Independently of the nature of the data, categorical, text, or numerical, it is important that the dataset has more or less balance among them. When there exists a big difference in the number of instances of each class, the ML can become biased towards the predominant class, not learning correctly.

3.6.1 Splitting the data

Once the quality of the data has been assured, dividing it into the dataset is necessary for ensuring the suitable generalization of the model to unseen data. The dataset has to be split into separate subsets for training, validation, and testing.

The process of dividing the data is done by randomly selecting instances from the dataset and assigning them to each subset. It is typically done randomly, but it is

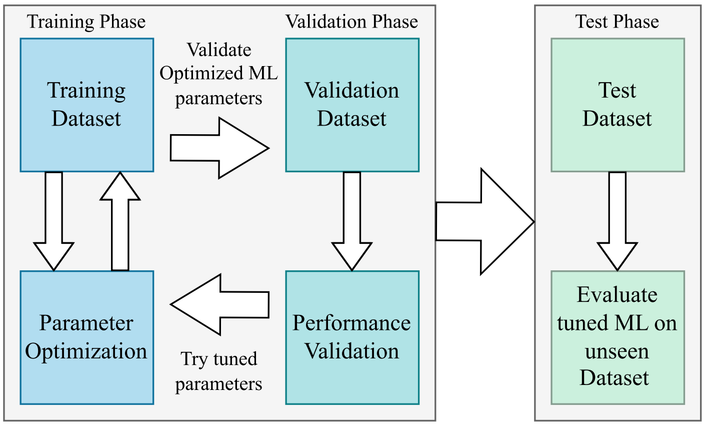


Figure 3.5: *Flowchart of the training, validation, and test phases in ML*

important to check that the testing and training subsets are representative of the overall population. The percentages of each subset vary a lot depending on the size and specific requirements, but some guidance values that have been followed in this project are:

- **Training Set** (60-80%). Used for training the model.
- **Validation Set** (10-20%). Evaluates the performance during training and hyperparameter tuning.
- **Test Set** (10-20%). Evaluates the final performance after training with this new and unseen dataset.

There also exists another technique called cross-validation. It is a common practice when there is not so much availability of data or when the model has a large number of hyperparameters to tune. This method consists of making several equally sized partitions or "folds" of the dataset, selecting one as a test and the rest as a train set. This process is repeated several times, every time with the test set being a different one. The final performance of the model is an average of the results.

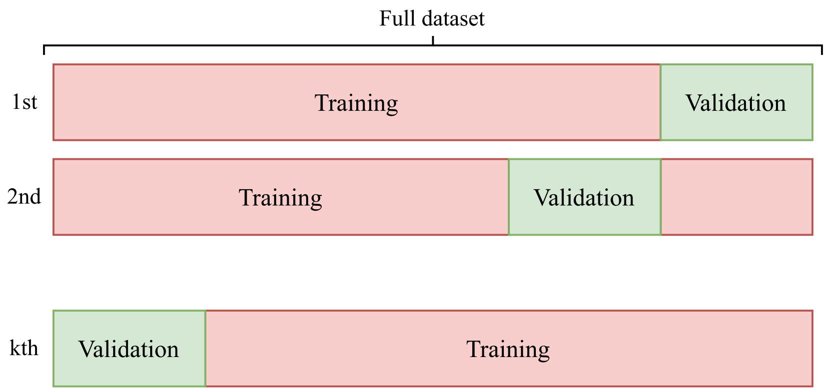


Figure 3.6: *Schema of the cross-validation technique for k-th folds*

3.6.2 Preprocessing

The preprocessing process involves transforming the raw data into a different format from where the ML algorithm can learn the relevant patterns for the problem to solve. It is a critical step in AI/ML that totally affect the performance result. The choice of a transformation or a different parameter can lead the model to have completely distinct behaviors. Overall, having good transformations and a correct feature selection can improve the generalization ability, improve the performance and reduce the complexity of the model.

Feature Selection

In addition to the transformations, another relevant technique is the feature selection. It consists in identifying and selecting the most relevant parameters from the data to learn the prediction. It helps to remove the redundant features that can introduce noise or bias and also extra complexity to the model.

There exist different tools that can be really helpful for identifying the most significant features and disregarding the irrelevant ones. Some common tools for visualizing the data relations and distributions among the parameters are histograms, scatter plots, or correlation matrices.

Transformations

One key technique in preprocessing is the transformations that can be applied to the data, such as standardization, normalization, or encoding. Applying these, the impact of outliers or the difference in scale can be reduced, and categorical variables can be handled better by the model. There exist different types of scalers depending on the specific transformation they apply to the data. Some of the more common ones, available in Python, are presented thereafter.

- **MinMax Scaler**. It converts the data to a fixed range between 0 and 1. This is done by subtracting the minimum value and dividing it by the difference between the maximum and minimum.
- **Normalizer**. This scaler transforms each sample independently using a specified norm, such as L1-norm or L2-norm.
- **Standard Scaler**. This makes the data have a zero mean with a standard deviation of one. The computation is removing the mean value and dividing each by the standard deviation.
- **Robust Scaler**. It works by subtracting the median value and dividing it by the Interquartile Range (IQR), making it less sensitive to outliers than the standard scaler.

4 Scenarios Simulation Setup

4.1 3GPP Channel Model Scenario

For the study of NR positioning and the evaluation of its performance, 3GPP has defined use scenarios and the corresponding channel models as baseline assumptions. The Indoor Open Office (IOO) and the Indoor Factory (InF) scenarios are two of these use case scenarios. The thesis work focuses on the InF scenario.

4.1.1 Explored Indoor Factory Scenario Types

3GPP defines different scenario types with various clutter characteristics. These clutter characteristics are defined in the simulation mainly by three parameters corresponding to {clutter density, clutter height, clutter size}. Each of them has the possibility of having different values constrained to the ranges of:

- Density $\in [20\%, 60\%]$
- Height $\in [2m, 6m]$
- Size $\in [2m, 10m]$

This degree of freedom for choosing the parameter values generates different channel characteristics, each with more or less multipath propagation effects. The multipath propagation effect is common in different scenarios with different channel characteristics. Each configuration has different NLOS effects probabilities for them to happen on the links between the UE and the gNBs.

In Indoor Factory's case, variants exist depending on the values of the presented parameters [9]. The IOO and InF scenarios are presented in [1]. For the InF case, two of the most common configurations are the indoor factory-sparse high (InF-SH) and the indoor factory-dense high (InF-DH). This project focuses on the second one, the InF-DH, because of the density of the clutters and clutter size that allow having a slightly higher percentage of LOS probability. The description of its characteristics is described subsequently in Table 4.1.

The probability for each link to be LOS depends on scenario clutter settings. Table 4.2 collects these probabilities for the InF-DH scenario. These values are obtained with the FR1 assumption. However, the project will focus on FR2, so the numbers in the table are only used as guidance for selecting the most interesting scenarios.

When selecting the scenario for training the AI/ML models, the most important factor, in this case, has been data balance. Having a high number of LOS and NLOS on the

Scenario Parameter	Properties
External wall and ceiling type	Concrete or metal walls and ceiling with metal-coated windows
Clutter type	Small to medium metallic machinery and objects with irregular structures. For example, assembly and production lines surrounded by mixed small-sized machines
Typical clutter size	2m
Clutter density	$\geq$ 40%

Table 4.1: Description of InF-DH parameters [14]

Environment Clutter Setting	LOS Probability
{40%, 2m, 2m}	0.449
{50%, 2m, 2m}	0.352
{60%, 2m, 2m}	0.268
{40%, 6m, 2m}	0.014
{50%, 6m, 2m}	0.025
{60%, 6m, 2m}	0.008

Table 4.2: LOS Probabilities for different InF-DH clutter settings

dataset allows the algorithm to learn more efficiently and distinguish the relevant characteristics between different classes.

For that purpose, the best possible scenario would be the one with a ratio of 0.5 between LOS and NLOS links. Hence, the closest LOS probability value for the presented InF-DH scenario is the cluster setting environment {40%, 2m, 2m} with approximately a ratio of 0.449 between both classes. This scenario was selected for training the different AI/ML models.

If a low ratio of LOS links exists in comparison with NLOS, the AI/ML model might not be able to efficiently discriminate between them. The reason is that the algorithm needs a sufficient number of samples to identify each class's characteristics. For instance, if the model is trained with a cluttered setting {60%, 6m, 2m}, it will not learn the relevant patterns of the LOS class due to 99.2% of the dataset would be NLOS and only an 0.8% would be LOS. For this example, a huge dataset will be needed to have a sufficient quantity of LOS samples.

Furthermore, once the relevant patterns of each class have been learned, the model is also supposed to identify them in other scenarios with different balances. For generalization purposes, unseen and more severe NLOS clutter-setting have been simulated. These different environment cases vary not only the clutter settings but also the UE locations to unseen ones. Moreover, it has also been investigated for the same clutter setting, a different factory layout, meaning a distinct distribution of the clutters inside it. The results of the comparison are presented in 6.

4.1.2 InF Scenario Assumptions

Antenna System Characteristics

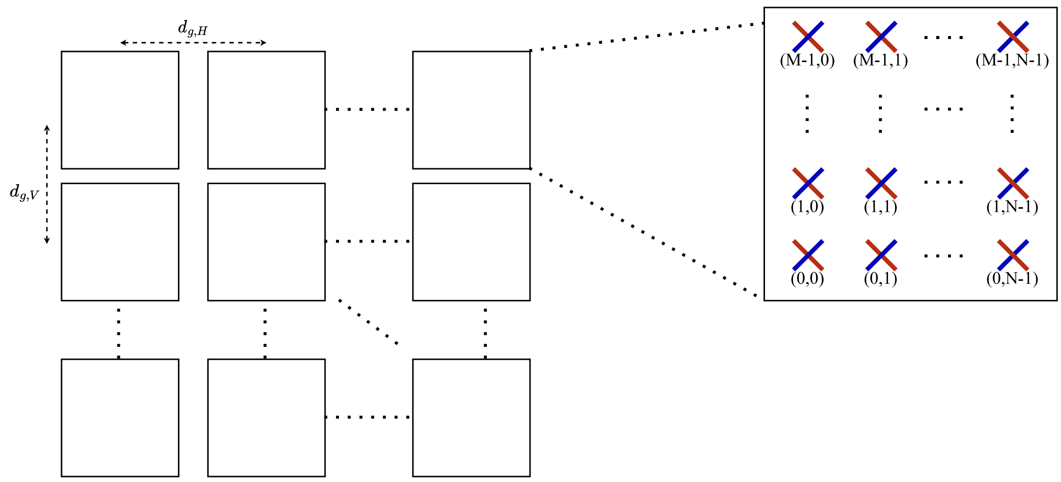


Figure 4.1: *Figure of the cross-polarized panel array antenna model*

The BS antenna has a uniform rectangular panel array with $M_g N_g$ panels, being M_g number of columns and N_g number of rows. The antenna system can be observed in Figure 4.1.

Antenna panels are uniformly spaced with $d_{g,H}$ horizontal distance and $d_{g,V}$ vertical distance. Each antenna panel is constructed by M×N number of antenna elements, where M is the number of rows and N is the number of columns. Antenna elements are uniformly distributed with d_h horizontal distance and d_v vertical distance.

Furthermore, antennas' polarization can be either single-polarized (P=1) or dual-polarized (P=2). Hence, the antenna array can be described as (M, N, P, M_g, N_g). This antenna system is defined on the 3GPP for study purposes [9] [14].

For this project, the antennas that have been selected depend on the frequency range FR1 and FR2.

- **FR1 Antenna Configuration**

  As it appears in the 3GPP [9], the configuration introduced in the simulator has been (4,4,2,1,1), this means 4 by 4 dual-polarized antennas. The Noise Figure (NF) is 5 dB, and maximum transmitted power (PTx) is 24 dBm.

  The UE has a (1,1,2,1,1) antenna configuration with an isotropic antenna model. The UE NF is 9 dB, and the PTx is 23 dBm.

- **FR2 Antenna Configuration**

  In the higher frequency scenario, the configuration is (4,8,2,1,1), meaning 4 by 8 dual-polarized antennas. This has a Noise Figure (NF) of 7 dB, and maximum transmitted power (PTx) of 24 dBm.

The UE for the FR2 has a (1,4,2,1,1) antenna configuration with an isotropic antenna model. Its NF is 13 dB, and its PTx is 23 dBm.

Layout Dimensions and BS Distribution

The indoor factory can either be a small or big hall. The project case, InF-DH, corresponds to the small hall setup, with dimensions of $L \times W = 120 \times 60m$ and a total height of 10m. The studied scenario also contains 18 gNBs with an 8m height placed on a square lattice with D = 20m gNB-to-gNB spacing, located D/2 = 10m from the walls. The schematics of the layout configuration described hall can be shown in the following 4.2.

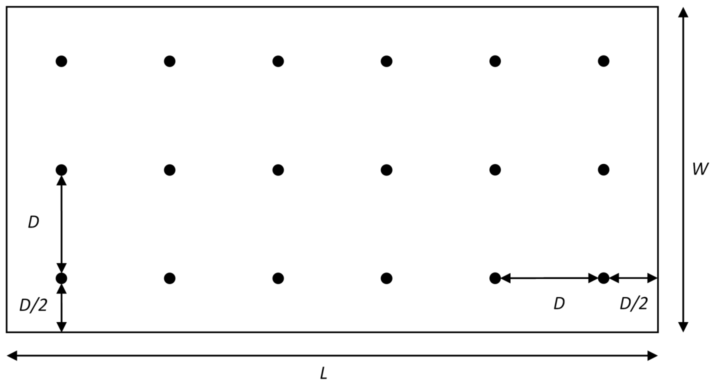


Figure 4.2: *Layout of the Indoor Factory Scenario*

Other Assumptions

- The z-coordinate, as the height of the UE is assumed constant for the InF scenario, it has not been considered in the UE location estimate. Then, the location prediction is only made for the horizontal x and y planes.
- The channel generated in the simulator follows 3GPP standards [14] with noise following a Zero Mean Circularly Symmetric Complex Gaussian (ZMCSCG) behavior.

4.2 Data generation

4.2.1 UE Distribution

For the data generation of the UE locations and its signal channel measurements, each dataset has been simulated with 4000 UEs deployed in the presented scenario. It has 18 gNBs, and as each UE is linked to every single gNB, there are 72000 links in total. The UE distribution on the factory layout can be seen in Figure 4.3.

The main motivation for this number of UEs is due to computational limitations, the lack of a more powerful computer limited the scale of the simulations. The following Figure shows the distribution of the UEs in the described layout and its orientation, with the gNBs located in a fixed position in every simulation [1].

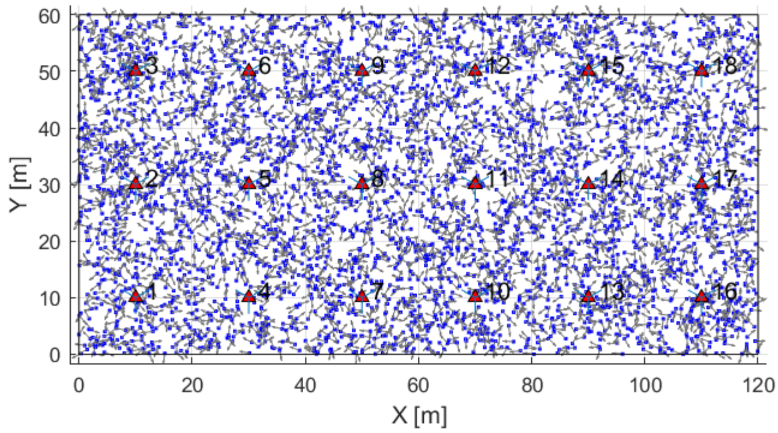


Figure 4.3: *Location distribution of 4000 UEs throughout the entire hall*

There exist different simulation drops of the UE location inside the factory room. Instead of having all the devices spread through the room (60 x 120m), a convex hull in the area delimited by the exterior gNBs has been considered. This convex hull condition will affect the device density as there is a lower space (40 x 100m) to allocate the same amount of UEs. This simulation assumption aims to study the positioning performance only in a restricted area. The convex-hull simulation approach is represented in Figure 4.4

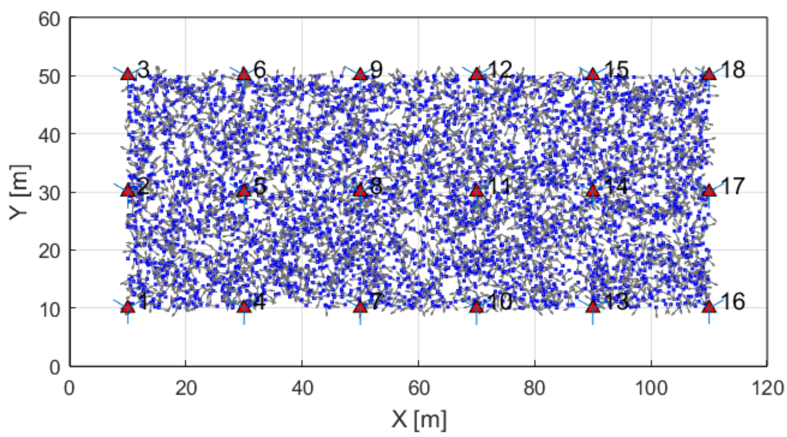


Figure 4.4: *Location distribution of 4000 UEs through the hall using convex hull*

Spatial consistency

The generated data follows the spatial consistency procedure defined by the 3GPP TR 38.901 [14]. This consistency guarantees that channel parameters in different locations correlate depending on their distances in the two-dimensional horizontal plane. Spatial consistency is important in this setup because there exists a high number of closely located user devices generated by drop-based simulations.

The different UE locations along the entire layout could also be seen as having a single UE moving across the factory and taking samples in different time instants. One could say that all the UE dropped on the layout is equivalent to having one UE describing one trajectory, as the interference among the UEs is not considered in this project.

4.2.2 Dataset structure

The MATLAB simulator of the presented InF-DH scenario generates the different ground truth position references and their channel signal measurements for the 18 links of each UE. As the simulations have been done uplink (UL), the measurements represent the arrival times and angles, as well as the transmitter corresponding to the UE and the receiver being the gNBs. All the parameters obtained for each link are compiled in table 4.3.

Parameter	**Abbreviation**
Reference Signal Received Power (dB)	RSRP_dB
Receiver Beam Forming Gain	RxBeamFormingGain
Mean-Path-to-Average Ratio (MPAR)	quality_1
First-Path-to-Average Ratio (FPAR)	quality_2
Signal-to-Noise Ration (SNR)	quality_3
Time of Arrival	TOA2Reference
Azimuth angle of Arrival (Local Coordinates System)	AOALCS
Azimuth angle of Arrival (Global Coordinates System)	AOAGCS
Zenith angle of Arrival (Local Coordinates System)	ZOALCS
Zenith angle of Arrival (Global Coordinates System)	ZOAGCS
Time of Arrival - Ground Truth	TOA_GT
Azimuth angle of Arrival (LCS)	AOALCS_GT
Azimuth angle of Arrival (GCS) - Ground Truth	AOAGCS_GT
Zenith angle of Arrival (LCS) - Ground Truth	ZOALCS_GT
Zenith angle of Arrival (GCS) - Ground Truth	ZOAGCS_GT
LOS or NLOS Condition - Ground Truth	LOSCondition_GT
Signal-to-Noise Ratio	SNR_GT
Distance UE and gNB - Ground Truth	distance_GT
Receiver ID (gNB ID)	RxDevice
X, Y of the Receiver (gNB) - Ground Truth	Rx_GT_x, Rx_GT_y
Transmitter ID (UE ID)	TxDevice
X, Y of the Transmitter (UE) - Ground Truth	Tx_GT_x, Tx_GT_y

Table 4.3: Parameters simulated per link with their used abbreviation

Apart from the presented data, the complex CIR is also provided for each link with a sampling window of 220. The motivation for this concrete number is reducing the signaling and processing complexity, besides the InF hall that makes the 220 a big enough sampling window. To motivate the election the following estimations have been done.

The first step roughly estimates the maximum distance any radio signal can travel inside the factory hall. Taking into consideration that for the study case InF-DH the size is 120m x 60m:

$$d_{max} = \sqrt{(x_{max})^2 + (y_{max})^2} = \sqrt{(120)^2 + (60)^2} = 134.164m \tag{4.1}$$

Successively, the sampling frequency has to be computed for the selected 5G numerology. In the project case, the Fast Fourier Transform (FFT) size is 4096, and as the numerology μ is 3, it implies the Subcarrier Spacing (SCS) equals 120 kHz:

$$f_{sampling} = SCS \times FFT = 120,000 \times 4096 = 491.52MHz \tag{4.2}$$

The last step is to derive the minimum needed samples for covering the maximum possible distance. This is done by taking into account the calculations 4.1 and 4.2:

$$\text{window}_{sampling} = \frac{d_{max} \times f_{sampling}}{c} = \frac{134.164 \times 491.52 \times 10^6}{3 \times 10^8} = 219.814 \text{ samples} \tag{4.3}$$

As can be seen, the minimum number of samples to cover the estimated distance needs to be equal to or superior to 219.814. For that reason, the selected final value has been 220.

4.3 Dataset Analysis and Preprocessing

The presented dataset is then processed in Python for organizing and investigating the data, as well as applying some preprocessing to the raw data before feeding it to the AI/ML model.

Recalling the generation of the datasets, the maximum number of UEs to be simulated at once is 4000. Nevertheless, when processing and feeding them to the algorithm, the final datasets include 8000 UEs by merging two generated datasets together resulting in more data for the AI/ML algorithm to learn. This merge makes each final dataset of 8000 UEs, with 18 connections with gNBs each, hence, 144000 different links. Once again, the motivation of the 8000 UEs at a time instead of higher datasets is due to computational limitations.

4.3.1 Features Selection

The resulting dataset presents several parameters per link, representing different measurements of the channel characteristics, angles, or times, collected on Table 4.3. However, not all of them need to be relevant for the AI/ML model to find the needed pattern in the data to make predictions.

In order to sort these features and select the most significant ones for the required tasks, different analyses have been performed. The analysis is presented using two different tools, a histogram and a correlation matrix.

Histogram

A histogram is a statistical graphical representation of the data distribution using a series of vertical bars. It is a useful tool to identify patterns or trends, give insight into each feature distribution and examine its range of values.

Concretely, the presented histogram represents the distribution of the features classifying them between LOS and NLOS links. The purpose is to analyze which features have a more distinct distribution for each class in order to potentially select them for the AI/ML methods. The histogram of the features is introduced in Figure 4.5.

For the plotted histogram4.5, it can be observed that there is not a clear feature or group that presents a clear different distribution for NLOS and LOS. Another observation is that the range of values that each parameter take is truly varied, so standardization of parameter might be required for a good performance of the AI/ML model.

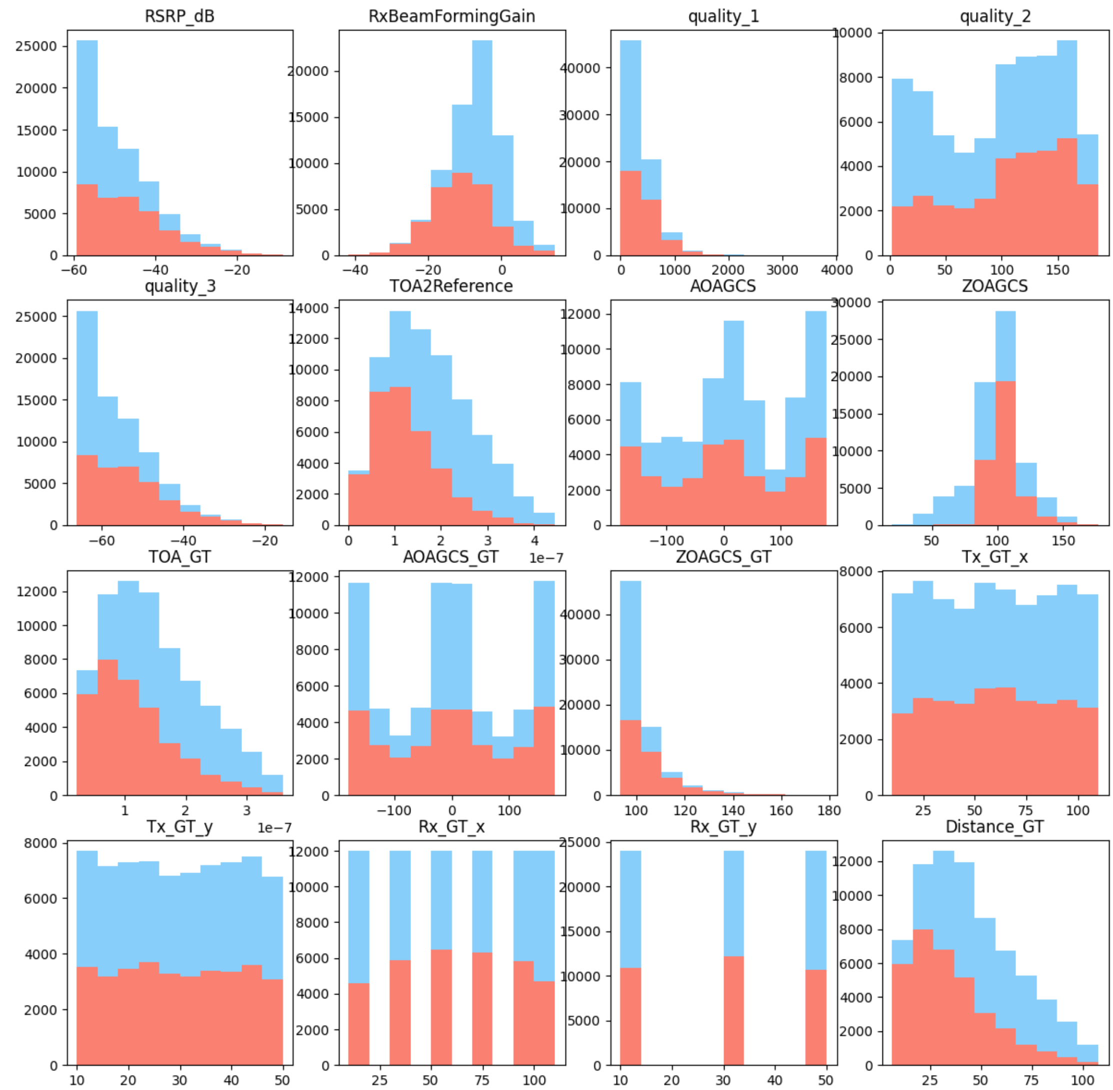


Figure 4.5: *Histogram of the features value distribution for LOS (red) and NLOS (blue) for the dataset configuration InF-DH 40%, 2m, 2m*

Correlation Matrix

The correlation matrix is another statistical tool that studies the dependence/relation between different variables displayed in a coefficient matrix. Each coefficient represents the degree of association between two different features. The sign of each correlation coefficient measures if it is a direct or inverse relationship, and the coefficient magnitude indicates how strong this relation is.

This tool maps the relations of the different parameters, helping to select the potentially more relevant features to have a good performance on the AI/ML models. For this motive, the focus is to find which features have the strongest correlation coefficient with the feature that needs to be estimated. The correlation matrix corresponding to the studied parameters is represented in Figure 4.6.

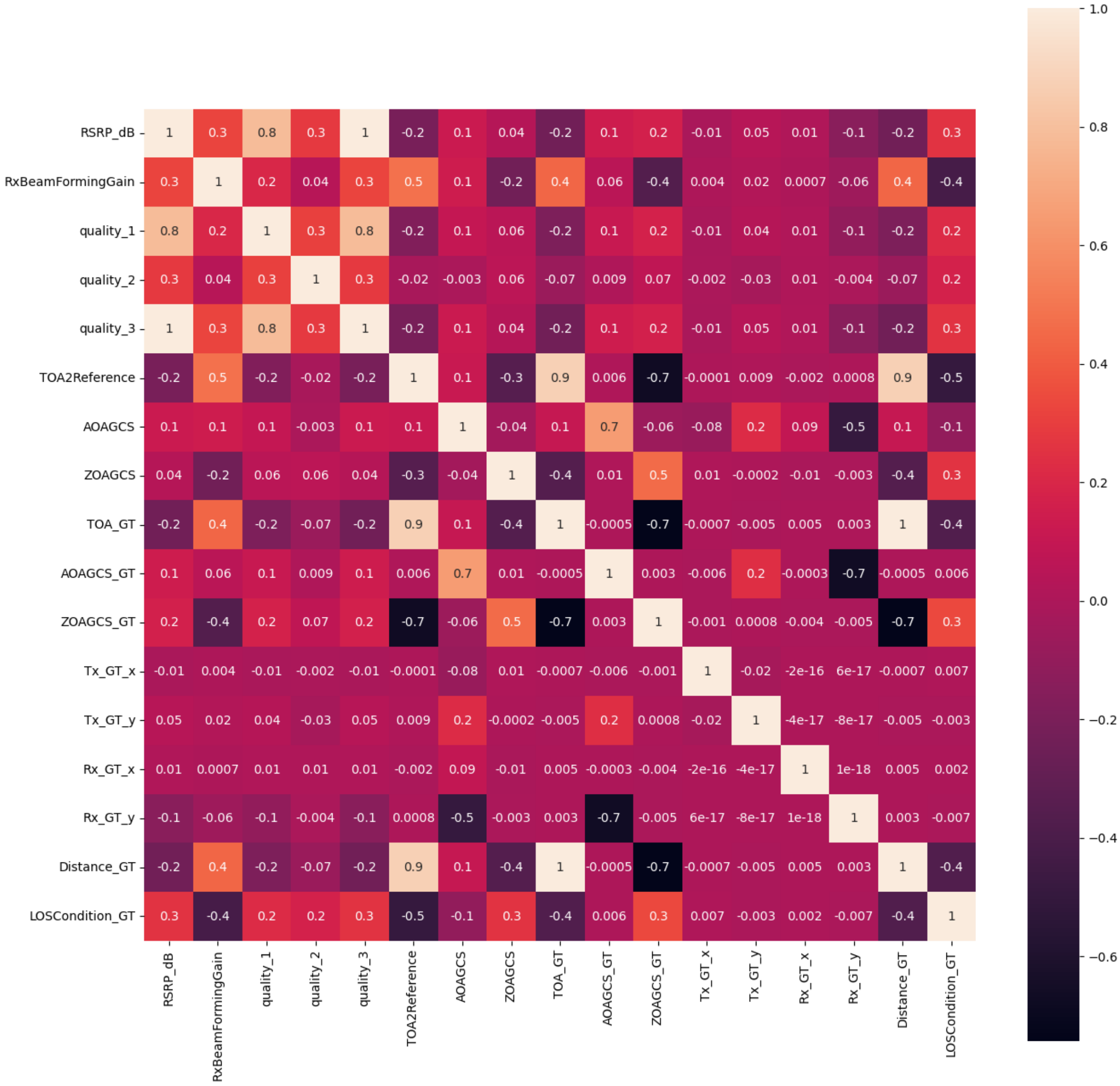


Figure 4.6: *Correlation matrix for the different features*

As the project has different estimation tasks, if the AI/ML method is for assisted or direct positioning, the inspection of the correlation matrix is performed separately for each case. As can be seen, when it comes to the Azimuth and Zenith angles, only the Global Coordinate System has been considered due to the interesting coordinates being the global ones, not the relatives.

For the assisted AI/ML location, the objective is to enhance the current channel estimation measurements. This includes the angles, times, and LOS probability, so the target features to be estimated are AOAGLS_GT, ZOAGLS_GT, TOA_GT, and LOSCondition_GT. It should be noted that other Ground Truth (GT) features cannot be selected due to they are not available in a not-simulated scenario, except the gNB's exact location can be known beforehand.

In the case of direct AI/ML positioning, the estimation aims are the UE position inside the hall, so the focus is to select the more related features with the features that represent that. Instead of the direct Tx_GT_x and Tx_GT_y, the target is the actual

Distance_GT, which represents the distance between the UE and the gNB. The motivation is that the exact coordinates of the transmitter itself are not so correlated with the other features, but the distance presents higher dependence. A similar approach is to observe instead the correlation with the distance. This distance combined with the exact location of the gNB, described the UE location.

From the matrix, the higher coefficients in module with the parameters to estimate, the better it is for the AI/ML algorithm to find the correct relation. Accordingly, the features that have strong relationships with the assisted approach are RSRP_dB, RxBeamFormingGain, quality_1, quality_2, quality_3, TOA2Reference, AOAGLS, and ZOAGLS. In the case of direct positioning, the final selected features have been RSRP_dB, RxBeamFormingGain, TOA2Reference, AOAGLS, ZOAGLS, Rx_GT_x, Rx_GT_y, and RxDevice.

4.3.2 Transformations

Following the previous feature analysis, the histogram reflected the data has different ranges of values for each feature indeed, meaning some standardization or normalization may be needed. The tried transformations are presented in the following.

- **Standardization**. The scalers investigated are Standard Scaler and Robust Scaler.
- **Normalization**. This other type of transformation modifies the data to have its parameters inside a specific range of values. The scalers explored in this project have been MinMax Scaler and Normalizer.

Implemented Approach

The same technique has been applied to all the channel features selected. For the case of the CIR, based also on the different model architectures that are being presented in the next chapter, there exist two different approaches for applying the correspondent transformation.

There exist models that process each link separately, estimating the required output for each one. Another typology is the models that directly process a chunk of 18 links corresponding to the same UE. Depending on which one is being used, the data from each link is standardized or normalized differently.

- **Single-link model**. In this architecture, the AI/ML algorithm process each link independently of the rest. For this reason, the transformation of the values has been done through the whole dataset. This means that for this operation, the values from the 220 samples of all the links of the training dataset have been used at once.

– **Multi-link model**. In this other typology, the AI/ML model is fed with a chunk of the 18 links of the same UE every time. For this case, the selected transformation is applied individually to each 18-link group, so the computations are only done with the values from the 18 links of each UE.

5 Developed AI/ML model architecture for 5G positioning

This section presents the Deep Learning models investigated for the project. First of all, the different methods created are divided into the single-link and multi-link models, which differ mainly in preprocessing and input date.

Within these two model typologies (single-link and multi-link), two different architectures have been studied (Architecture I and II). The difference between Architecture I and II is based on whether the CIR is included as an input or not, together with other channel signal parameters. To explore the feasibility of each architecture, a simple and a complex model have been created, with different numbers of neurons and parameters.

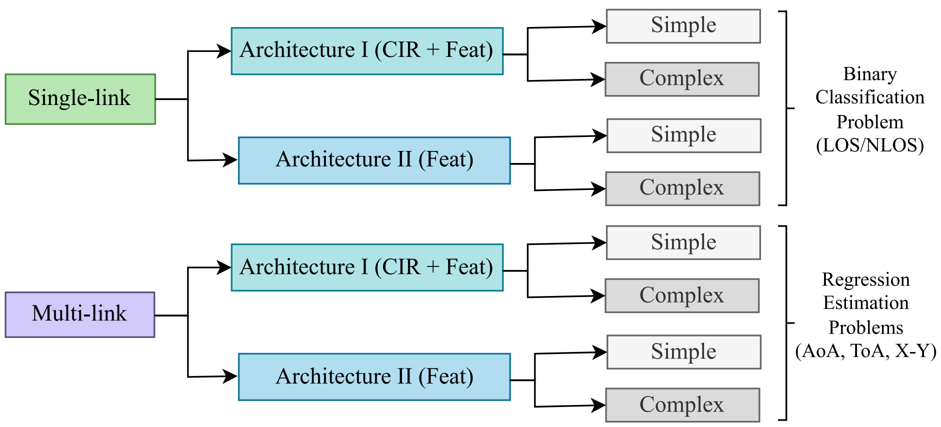


Figure 5.1: *Schematic of the developed models for 5G UE positioning*

The schema of the different developed models is shown in the previous Figure 5.1. It has been observed during the exploration that the single-link performs better when solving labeling problems, and the multi-link models work better with regression estimation.

Even if it could be thought that it is the other way around, the single-link model performs better than the multi-link one for LOS classification. This has been observed through trial and error, but there might be different explanations for this to happen.

The single-link model is able to focus on just one set of link characteristics, allowing it to be more precise. This model only outputs one label at a time, while the multi-link model outcomes 18 different values. The 18 needed labels per input obligate the model to have a broader and more complex perspective when investigating the data, which may be not so effective.

Furthermore, it also affects the nature of the label, might be that other links could not bring any extra valuable information for determining one link label. For example, that one link has a LOS condition does not have to be related to the LOS/NLOS condition of any other link.

Finally, to achieve the best performance in reference to the best possible model structure, a hyperparameter optimization of each architecture was conducted. These hyperparametrization results are presented in Chapter 6, giving further insight into each AI/ML method and its final structure. Concretely, a description of their I/O, layer structures, loss functions, and their optimizers is provided.

5.1 Single-Link Model

This AI/ML model type is based on feeding the algorithm with a single link at a time. In an uplink or downlink scenario, each link can be seen as a different connection between a UE and a gNB or vice versa. For each unique connection, some useful patterns are supposed to be obtained by the AI/ML models.

Inside the Single-link Model typology, there exists architecture I, which includes channel signal measurements and the complex CIR, and architecture II, which only uses some channel signal measurements.

5.1.1 Architecture I

Input and Output

For the case of the input, as presented, this architecture has two different input datasets that are fed into separate points of the NN and documented in Table 5.1.

Input	Size Dimension	Preprocessing	Description
Complex CIR	220 x 2 x 1	MinMax Normalization	A window of 220 samples of the Channel Impulse Response (Real and imaginarypart of each sample) of the link is fed before the Convolution Blocks of the NN
Channel Signal Parameters	6 x 1	MinMax Normalization	AOA, TOA, ZOA, RSRP, Beam Forming Gain and FPAR

Table 5.1: Inputs of the single-link model with Architecture I

Regarding the outputs of the system, this typology is only focused on providing the estimation of if the link has LOS or NLOS conditions. Specifically, the model outputs, which is the LOS probability of the link, is presented in Table 5.2.

Output	Size Dimension	Description
LOS Probability	1 x 1	As the input of one link, the model outputs one number between 0 and 1, representing 0 as a NLOS condition and 1 as a LOS link

Table 5.2: Output of the single-link model with Architecture I

Layers Structure

The presented NN architecture is the one resulting after the hyperparametrization. This architecture can be divided into the convolution block and the fully connected block. Figure 5.2 illustrates the final structure, showing the number of neurons per layer depending on the value of units, as well as the I/O dimensions. It is important to mention that FeedForward Neural Network (FNN) part, in reality, is only 1D. The previous figure represents the network in 2D but only for visualization purposes.

The CNN architecture has been selected because one input is the CIR of one link, which can be seen as a 2-dimensional image. CNNs are one type of architecture that works better with multidimensional inputs. The reason is the nature and composition of its layers, helping the NN to get a better understanding to reach the desired solutions.

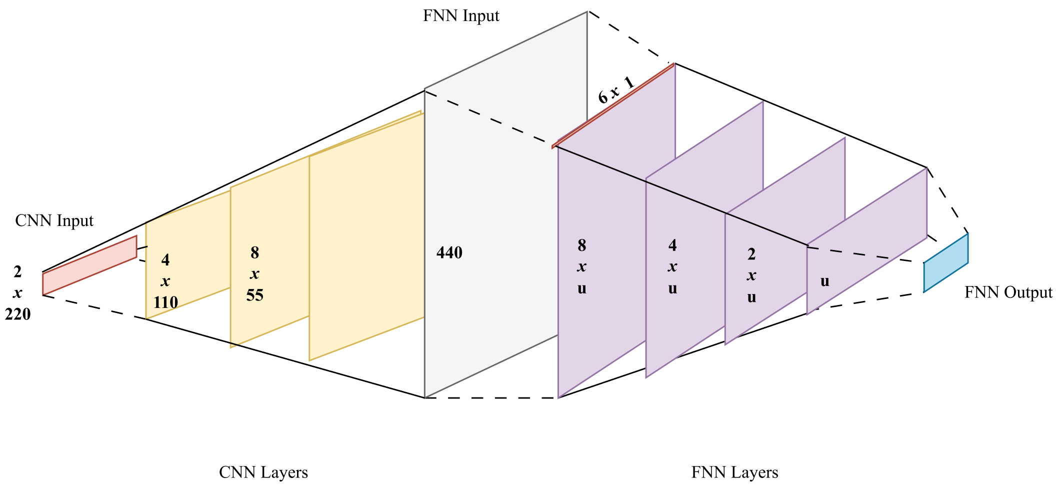


Figure 5.2: *Schema of the layers structure for the single-link model with Architecture I*

The model starts with the CNN input layer composed by the CIR, fed to the convolution block. The convolution block has 3 stages, each containing a regular Convolutional Layer, with ReLU as their activation function, followed by a Max Pooling Layer. The fully connected NN is connected directly to the flattened layer and has 4 different layers with different dimensions. It is on the stage that follows the flattened layer where the second input, the features, is processed. After this block, the dimension of the NN reduces until it outputs only the LOS probability of the received link.

The u factor shown in the previous Figure 5.2 is a variable that represents the number of units or neurons that each layer has. The specific value that this parameter takes is collected in Tables 5.3 and 5.4, as the number of neurons determines the complexity of the model.

Hyperparametrization

In this part, the hyperparametrization strategy is presented. Concretely, a Bayesian search has been used for exploring the search space. The hyperparameter space that has been investigated is presented in the next Table 5.3. The hyperparametrization is done independently for both simple and complex models.

From Table 5.3, the space is explored by steps in the range of the given numbers per investigated parameter. For different possible parameters to explore some set of them have been chosen, and for each of the a range of reasonable values.

The hyperparametrization can also explore not only numerical values, but also different types of activation functions, optimizers, or loss functions. There are some many possibilities that in this search only covers a defined set space inside the whole possibilities of the hyperparameter space.

It is interesting to understand that hyperparametrization does not explore all the options, but it follows a Bayesian-regression behavior and converges to the best hyperparameters by a series of iterations, so it does not simulate every single possible architecture.

Parameter	**Ranges / Types**
Units of neurons (u)	[6, 106]
Filter size	[4, 64]
Kernel size	[2, 3]
Strides	[1, 2]
Pool size	[2, 3]
Dropout	[0, 0.25]
Learning rate	[1e-5, 1e-2]
Batch size	[16, 128]
Epochs	[30, 220]
Activation function	{ReLU, tanh}
Optimizer	{Adam, SGD}
Loss function	{Binary Cross Entropy, Hinge}

Table 5.3: Search space explored for the single-link model with Architecture I

Final Parameters

Of the different loss functions investigated, the one that gave the best performance was the Binary-Cross Entropy for LOS/NLOS prediction. The exact formulation is represented in the next equation 5.1.

$$\text{Binary Cross-Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p_i) + (1 - y_i)\log(1 - p_i) \tag{5.1}$$

Where:

- N corresponds to the total number of samples
- y_i represents the true label (0 or 1) of the ith sample
- p_i is the predicted probability for the ith sample.

The parameters that characterize the simple and complex model found for this concrete architecture are collected in Table 5.4. After the hyperparameter space have been searched, the optimal values obtained from the Bayesian hyperparametrization are described in Table 5.3.

	Simple	**Complex**
Units of neurons (u)	48	106
Filter size	4	
Kernel size	2	
Strides	2	
Pool size	3	2
Dropout	0.05	0
Learning rate	1.77e-4	1.933e-5
Batch size	64	32
Epochs	120	
Activation function	Tanh	ReLU
Optimizer	Adam	
Loss function	Binary Cross Entropy	
Number of parameters	112,509	506,213

Table 5.4: Parameters of the simple and complex single-link model with Architecture I

5.1.2 Architecture II

Input and Output

This second architecture differs from the previous one in the fact that the model only takes the channel signal parameters as inputs, but not the CIR. This variant aims to explore the achievable accuracy when predicting the same characteristics by only depending on the channel measurements.

Input	**Size Dimension**	**Preprocessing**	**Description**
Channel Signal Parameters	6 x 1	MinMax Normalization	AOA, TOA, ZOA, RSRP, Beam Forming Gain and FPAR

Table 5.5: Input of the single-link model with Architecture II

In relation to the outputs of this architecture, the output is the same as that of the previous architecture, which is explained in Table 5.6.

Output	Size Dimension	Description
LOS Probability	1 x 1	A number between 0 and 1, being 0 a totally NLOS condition and 1 a perfect LOS link

Table 5.6: Output of the single-link model with Architecture II

Layers Structure

As the FNN does not take the CIR as input, this architecture does not have the Convolution Stage. Instead, it consists only of an FNN, where the inputs are the presented parameters. During the hyperparametrization, it has also been explored the option to add an initial fully connected stage to help to process the input data, which corresponds to the turquoise layers in Figure 5.3. The motivation is that the input can have a reduced dimension in comparison with the NN layers. This is heavily dependent on the final number of units of the layers.

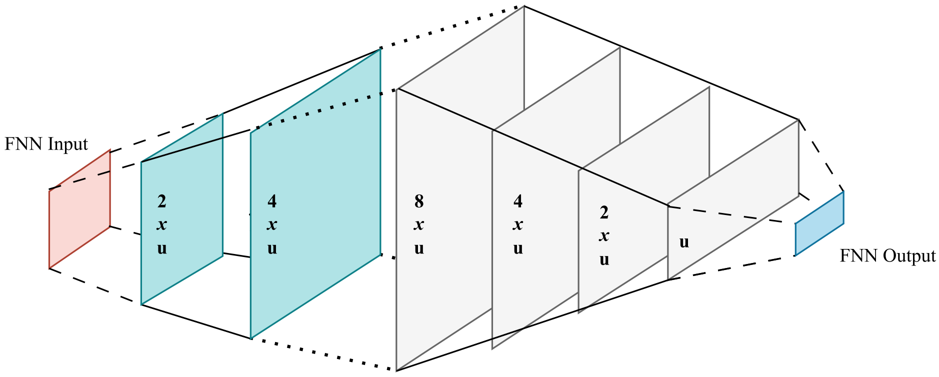


Figure 5.3: *Schema of the layers structure for the single-link model with Architecture II*

Hyperparametrization

As for the previously presented architecture, the Bayesian search has been used for exploring a defined search space. The hyperparameter space that has been investigated is presented in the following Table 5.3. As architecture II has different characteristics from architecture I, they have consequently different search space.

Parameter	Ranges / Types
Units of neurons	[6, 106]
Dropout	[0, 0.25]
Deep Dropout	[0, 0.1]
Learning rate	[1e-5, 1e-2]
Batch size	[16, 128]
Epochs	[30, 220]
Activation function	{ReLU, tanh}
Optimizer	{Adam, SGD}
Initial Stage	{True, False}

Table 5.7: Search space explored for the single-link model with Architecture II

Final Parameters

Finally, the characteristic parameters after the investigation for this specific architecture are collected in Table 5.8.

	Simple	Complex
Units of neurons	6	24
Dropout	0	
Deep Dropout	0	0.05
Learning rate	8.749e-4	3.376e-4
Batch size	64	80
Epochs	220	
Activation function	ReLU	
Loss function	Binary Cross Entropy	
Optimizer	Adam	
Initial Stage	False	
Number of parameters	1,897	25,729

Table 5.8: Parameters of the simple and complex single-link model with Architecture II

5.2 Multi-Link Model

For this model typology, a different understanding or perspective of the scenario has been given. As described in the previous section, each UE has a connection with each of the 18 gNBs. This project adopts the assumption introduced by 3GPP. In this approach, the 18 links of each UE are used at once for getting all the different connections at once, trying to feed the model with all the available information.

Hence, the dataset was divided into chunks representing the links observable by each UE, having 18 links in each chunk. This was a reorganization of all the available links, organized in sets of 18 links from the same UE. This division was intended to help the model to obtain certain patterns and to maybe make it easier to relate to the positions, angles, and times.

This architecture is dependent on always having 18 links available. The model is expected to constantly have the same input size. In the case of having fewer received links e.g., 17 or 16, due to signaling errors, some preprocessing is needed for the input to complete the missing data. One solution for this situation could be simply to repeat the received measurement of another link twice, take the missing link's previous values, or some kind of averaging of the rest available links' measurements. Another possible solution for this situation could be to do some interpolation between a neighbor gNB. As the gNB ID of the link missing information is known this is a feasible solution. Moreover, in principle, nearby gNB links are expected to have a higher correlation.

5.2.1 Architecture I

Input and Output

Regarding the inputs, this architecture is composed of two inputs, fed into separate points of the NN as in the previous typology. The main difference is that now, the input is not only one but 18 links at once. The specifications of the inputs are collected in Table 5.9.

Input	Size Dimension	Preprocessing	Description
Complex CIR	220 x 2 x 18	MinMax Normalization	A CIR window sample of 220 samples from the 18 links of the same UE fed before the Convolution Block
Channel Signal Parameters	6 x 18	MinMax Normalization	AOA, TOA, ZOA, RSRP, Beam Forming Gain, and FPAR, fed to the model after the Convolution Block, Rx_GT_x and RX_GT_y

Table 5.9: Inputs of the multi-link model with Architecture I

For the outputs of the system, this other model typology has proven to give better performance when performing regression parameter estimation. This has made the multi-link model more useful for predicting angles, times, or positions instead of only LOS labeling of each link. Therefore, this typology has been investigated for estimating several different outputs, all of them introduced in Table 5.10.

Layers Structure

The main difference between the multi-link and single-link models when it comes to the architecture is the convolutional stage. The CNN that composes the first part of the structure is more complex due to it has to handle an extra data dimension due to the composition of this other input.

Output	Size Dimension	Description
TOA	1 x 18	The estimated value of the correct Time of Arrival of each link in a LOS conditions channel
AOA	1 x 18	Estimation of the Azimuth angle of Arrival value for each link with a LOS conditions channel
X, Y	2 x 1	Estimation of the horizontal plane position of the UE from the 18 links

Table 5.10: Outputs of the multi-link model with Architecture I

This fact causes the CNN of the structure to have, in this multi-link case, 2D convolutions and 2D max poolings, transforming the dimension of the 18 link's CIR as the next Figure 5.4 shows.

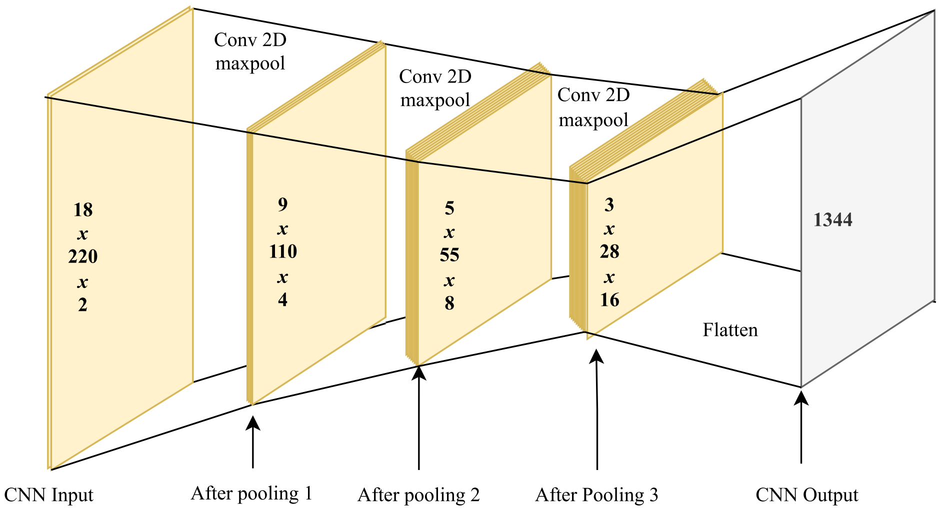


Figure 5.4: *Schema of the convolutional stage for the multi-link model with Architecture I*

This design has been chosen following typical CNN designs, where convolutional stages are placed one after the other. After an exhaustive process of trial and error, this structure has been chosen. Allowing the extraction of most details from the images given as inputs.

Hyperparametrization

As for the previous models, a Bayesian search has been used for exploring a concrete search space. The hyperparameter space investigated is collected in next Table 5.11.

Parameter	Ranges / Types
Units of neurons (u)	[6, 106]
Filter size	[4, 64]
Kernel size	[2, 3]
Strides	[1, 2]
Pool size	[2, 3]
Dropout	[0, 0.25]
Learning rate	[1e-5, 1e-2]
Batch size	[16, 128]
Epochs	[30, 220]
Activation function	{ReLU, tanh}
Optimizer	{Adam, SGD}

Table 5.11: Search space explored for the multi-link model with Architecture I

Final Parameters

The parameters of the best model found in the searched space are shown in Table 5.12.

	Simple	Complex
Units of neurons (u)	16	32
Filter size	64	
Kernel size	3	
Strides	2	
Dropout	0.1	0.05
Learning rate	1.396e-3	7.763e-4
Batch size	48	64
Epochs	110	70
Activation function	ReLU	
Optimizer	Adam	
Number of parameters	582,882	880,386

Table 5.12: Parameters of the simple and complex multi-link model with Architecture I

For this model case, the loss function finally used has been the Mean Absolute Error. The Mean Squared Error has also been manually investigated, but the results were not so satisfactory.

The same loss function was used for both the estimation of the location in the case of direct positioning and for the case of estimating time and angle parameters for the assisted positioning method. The explicit mathematical formula is presented in equation 5.2.

$$\text{Custom MAE} = \alpha \cdot \frac{1}{N} \sum_{i=1}^{N} |\mathrm{a}_i - \hat{\mathrm{a}}_i| + \beta \cdot \frac{1}{N} \sum_{i=1}^{N} \left|\mathrm{b}_i - \hat{\mathrm{b}}_i\right| \tag{5.2}$$

Where:

- N corresponds to the total number of samples
- α and β are weights assigned to penalize one error more than the other
- $\hat{a}_i$ represents the estimated value of the variable $texta$, which can be the X coordinate of the UE location, or the value of ToA
- a_i is the ground truth value of a variable $texta$, corresponding to the X coordinate position, or also could be the true ToA
- $\hat{b}_i$ is another different the predicted variable $textb$, that could be the estimated Y coordinate location, or analogously the predicted value of AoA.
- b_i corresponds to the true value of the variable $textb$, such as the Y true position or the ground truth AoA

The final values for both α and β were 0.5. There was not any specific reason to penalize more errors in one specific X or Y coordinate nor for the AoA and ToA estimation. Both predicted values were equally important, and none was drastically worst than the other, so the coefficients penalize the error equally.

Other loss functions, such as RMSE, were manually tried. The most promising results were obtained with the MAE. For this reason, it was continued investigating with this last one.

5.2.2 Architecture II: Neural Network

Input and Output

This case is identical to the previous architecture, except for not including the CIR as input to the NN. The inputs and output are still processed for 18 links at once, representing all the UE links with the 18 gNBs. Both the input and the different possible outputs are presented in Tables 5.13 and 5.14.

Input	**Size Dimension**	**Preprocessing**	**Description**
Channel Signal Parameters	6 x 18	MinMax Normalization	AOA, TOA, ZOA, RSRP, Beam Forming Gain, and FPAR, fed to the model after the Convolution Block

Table 5.13: Inputs of the multi-link model with Architecture I

Output	Size Dimension	Description
TOA	1 x 18	The estimated value of the correct Time of Arrival of each link in a LOS conditions channel
AOA	1 x 18	Estimation of the Azimuth angle of Arrival value for each link with a LOS conditions channel
X, Y	2 x 1	Horizontal position estimation of the UE from the 18 links

Table 5.14: Outputs of the multi-link model with Architecture I

Layers Structure

The structure for this method is really similar to the single-link architecture II, but a little bit more complex due to the input, in this case, being of 18 different links. For the hyperparametrization, the initial stage implementation has been investigated again. The structure of the FNN is shown in Figure 5.5.

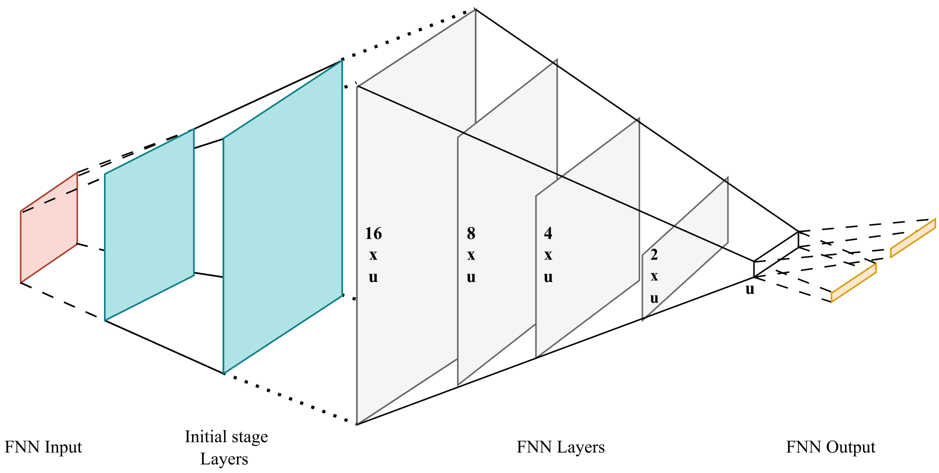


Figure 5.5: *Schema of the layers structure for the multi-link model with Architecture II*

Hyperparametrization

The search space explored for this architecture, doing a Bayesian search, is collected in next Table 5.15.

Final Parameters

For this last architecture, the final parameters of the simple and complex model found by the hyperparametrization are introduced in Table 5.16.

Parameter	**Ranges / Types**
Units of neurons	[2, 128]
Dropout	[0, 0.25]
Deep Dropout	[0, 0.1]
Learning rate	[1e-5, 1e-2]
Batch size	[16, 128]
Epochs	[30, 220]
Activation function	{ReLU, tanh}
Optimizer	{Adam, SGD}
Initial Stage	{True, False}

Table 5.15: Search space explored for the multi-link model with Architecture II

	Simple	**Complex**
Units of neurons	32	64
Dropout	0.1	0.15
Deep Dropout	0.05	0
Learning rate	3.724e-4	1.060e-3
Batch size	32	128
Epochs	220	
Activation function	ReLU	
Optimizer	Adam	
Initial Stage	False	
Number of parameters	248,866	845,890

Table 5.16: Parameters of the simple and complex multi-link model with Architecture II

5.3 Models Summary

In this final section, a summary of the 4 model architectures is presented in Table 5.17 for comparing and pointing out the differences between them.

Model	**Inputs**	**Outputs**	**Type**	**Complexity**	**No. Parameters**
Single-link I	CIR + Features	LOS Probability	CNN	Simple	112,509
				Complex	506,213
Single-link II	Features	LOS Probability	FNN	Simple	1,897
				Complex	25,729
Multi-link I	CIR + Features	AOA, TOA or (X,Y)	CNN	Simple	582,882
				Complex	880,386
Multi-link II	Features	AOA, TOA or (X,Y)	FNN	Simple	248,866
				Complex	845,890

Table 5.17: Summary of the different studied DL architectures

6 Performance Evaluation

This chapter presents the performance results of the models for different simulated scenarios and discusses the outcomes. The evaluation compares all the models presented in Chapter 5, investigating the impact of different CIR preprocessing, the importance of the hyperparametrization, as well as the generalization ability of the best models.

6.1 Performance Evaluation Results

For evaluation purposes, different parameters need to be calculated and compared to study how the models perform. The results obtained are also necessary for future research.

In order to make the different models comparable to each other, AI/ML has been trained with exactly the same dataset. Specifically, 8,000 UEs, and 144,000 links the InF-DH{40%, 2m, 2m} with the clutter distribution inside the factory will remain constant. Moreover, to increase the robustness of the model, batches are shuffled during the training of the weights.

The evaluation test consists of testing the model with unknown data, which the model has never seen before. This evaluation test has also been identified. It consists of a dataset of 4000 UEs (72000 links) in the same presented case with the same factory layout but with different UE positions. Both the training and test UE distribution corresponds to the presented convex hull approach, but in the generalization section, the non-convex hull distribution is also simulated to evaluate its impact.

Analysis Criteria

For the analysis of the results, different criteria have been used depending on the type of variable predicted, mainly depending on if it is a labeling or regression problem.

When predicting the label for the LOS or NLOS condition of a link, the evaluation tools used to compare the performance of the model's results are introduced in Table 6.1.

For the regression estimation parameters, different metrics have been used to evaluate the model's effectiveness. These criteria for studying the results are presented in Table 6.2.

Criteria	Description
Accuracy	Metric that measures the ratio of correct predictions out to the total prediction number
Precision	Provides the rate of true positives over the sum of true positives and false positives, showing the correctness of positive predictions and avoiding false positives
Recall	Ratio calculates the true positives over the sum of true positives and false negatives, capturing the ability to avoid false negatives
F-Score	The harmonic mean of precision and recall, between 0 and 1, with higher values indicating higher prediction performance
Precision-Recall Area Under the Curve (PR AUC)	It represents the area under the curve generated by the plot of precision and recall, and it assesses how balanced the predictions are in binary classification
Confusion Matrix	A graphical tool that displays the count of true positives, true negatives, false negatives, and false positives for a more insightful evaluation of the classification errors

Table 6.1: Criteria used for the labeling estimations analysis

Criteria	Description
Mean Absolute Error (MAE)	Metric that calculates the average absolute difference between predicted and actual values
Mean Squared Error (MSE)	Measure that computes the average squared difference between predictions and actual values
Root Mean Square Error (RMSE)	The average size of the errors by taking the square root of the average squared differences between predicted and actual values
R-squared	Measures the proportion of the variance in the dependent variable explained by the regression model, indicating how well the model fits the observed data
Cumulative Distribution Function (CDF)	Statistic tool that provides the distribution of one variable, used to compare the prediction error for different percentiles

Table 6.2: Criteria used for the regression estimations analysis

6.1.1 Assisted AI/ML Positioning

This part of the results evaluates the performance of the presented models as AI/ML-assisted methods. For the assisted approach, the models aim to predict the LOS probability of each link, its time of arrival, and its azimuth angle of arrival. Finally, in order to compare if there has been an actual improvement in comparison with the legacy methods, a traditional position computation has been derived using the enhanced parameters in contrast to the measurement.

LOS Classification Performance

The LOS Classification performance is evaluated with the metrics presented in Table 6.1. The models evaluated are the single-link model architecture I and II for investigating the relevance of the CIR. Moreover, from each architecture, one simple and complex model has been tried for further study of the implementation process. The results of all the models are introduced below in Table 6.3.

	Architecture I		***Architecture II***	
	Simple	**Complex**	**Simple**	**Complex**
Accuracy	0.914	0.914	0.921	0.916
Precision	0.917	0.927	0.938	0.927
Recall	0.898	0.887	0.890	0.891
F-Score	0.908	0.906	0.909	0.909
PR AUC	0.8717	0.875	0.887	0.877

Table 6.3: Evaluation of the LOS classification results for the single-link models

As it can be seen in Table 6.3, looking at the results, there is no big difference between simple and complex models. This behavior could be attributed to the simplicity of the LOS prediction task.

In simpler tasks, using a complex model may not necessarily result in better predictions. In fact, complex models might learn intricate relationships within the dataset that do not accurately represent the true underlying patterns. As a result, the model can generate incorrect predictions or misguided interpretations of the data.

Time Estimation

For the TOA estimation performance, the evaluating criteria are the ones corresponding to a regression problem, introduced in Table 6.2. In time estimation, it has been observed that the multi-link model outperforms the single-link ones. For that reason, these will be the ones that are going to be presented. The following Table 6.4 shows the obtained results of the simple and complex models corresponding to architecture I and II of the multi-link typology.

	Architecture I		***Architecture II***	
	Simple	**Complex**	**Simple**	**Complex**
MAE (ns)	8.579	8.280	7.771	7.488
MSE (ns^2)	118.552	111.821	104.077	93.174
RMSE (ns)	10.888	10.575	10.202	9.653
R-squared	0.980	0.981	0.982	0.984

Table 6.4: Evaluation of the TOA estimation for the Multi-link models

Angle Estimation

In the case of the azimuth angle of arrival, as with the time, the criteria used for the result analysis are the regression ones, as Table 6.2 describes. The approach is the same as the one used for TOA. The same Multi-link model typology is the one investigated. In fact, the models created predict both at the same time the TOA and AOA of the 18 links of each UE. As before, the different complexity models from each architecture are evaluated, reflecting the results in the next table 6.5

	Architecture I		***Architecture II***	
	Simple	**Complex**	**Simple**	**Complex**
MAE (°)	8.119	6.715	6.322	6.8
MSE ($°^2$)	532.281	337.307	418.546	383.449
RMSE (°)	23.071	18.366	20.458	19.582
R-squared	0.956	0.972	0.965	0.968

Table 6.5: Evaluation of the AOA estimation for the Multi-link models

Position Estimation Computation

As presented, for comparing the performance of the assisted methods with the legacy methods, the location computation of the UE has been done. For the legacy methods, the derivation of these locations has been done directly with the angles and time measurements, and then another calculation is presented when the estimated AoA, ToA, and LOS Probability have been used for obtaining the position.

The method used for computing the actual UE location consists of the Maximum Likelihood Estimator with different cost functions that estimate the coordinates using the AoA, ToA, or a combination of both. The direct measurements and the AI/ML enhanced ones have been used for both approaches, to ensure that the results are comparable. For studying the differences in the performance, the CDF of the Euclidean distance error has been compared for various percentiles.

The calculations for obtaining the UE positioning have been done for the best assisted AI/ML model in AoA and ToA prediction, selected in section 6.2. This model is the Multi-link complex with Architecture II, and its corresponding performance results are introduced in next Table 6.6.

6.1.2 Direct AI/ML Positioning

For this approach, first, the quality of the prediction is evaluated using the metric tools presented in Table 6.2. The results for the different models on both the X and Y coordinates are shown in Tables 6.7 and 6.8.

		CDF of the Absolute Position Error (m)				
		50%	**67%**	**80%**	**90%**	**95%**
Legacy Methods	**TOA**	8.175	13.223	17.902	24.251	30.821
	AOA	0.078	2.309	7.664	16.019	23.605
	TOA + AOA	0.051	2.185	5.528	9.993	14.246
Assisted AI/ML	**TOA**	3.341	4.261	5.386	6.768	8.111
	AOA	3.509	4 477	5.759	7.341	9.143
	TOA + AOA	3.279	4.219	5.265	6.614	7.875

Table 6.6: Evaluation of the location estimation for legacy and AI/ML assisted methods in the scenario case InF-DH{40%, 2m, 2m}

	Architecture I		*Architecture II*	
	Simple	**Complex**	**Simple**	**Complex**
MAE (m)	3.194	2.471	2.774	2.990
MSE (m^2)	16.415	10.362	13.263	14.493
RMSE (m)	4.051	3.219	3.642	3.807
R-squared	0.980	0.988	0.984	0.983

Table 6.7: Evaluation of the X coordinate estimation for the Multi-link models

	Architecture I		*Architecture II*	
	Simple	**Complex**	**Simple**	**Complex**
MAE (m)	2.574	2.314	2.229	2.388
MSE (m^2)	10.783	8.581	8.264	9.082
RMSE (m)	3.284	2.929	2.875	3.014
R-squared	0.918	0.935	0.937	0.931

Table 6.8: Evaluation of the Y coordinate estimation for the Multi-link models

Position Estimation

In this part, as done with the AI/ML assisted methods, the Euclidean absolute position error is compared with the Legacy methods. This is done to study the different error distributions for each of the presented models and which is the performance of the traditional methods. The CDF of the error for each of the representative percentages is presented in Figure 6.1 and in Table 6.9.

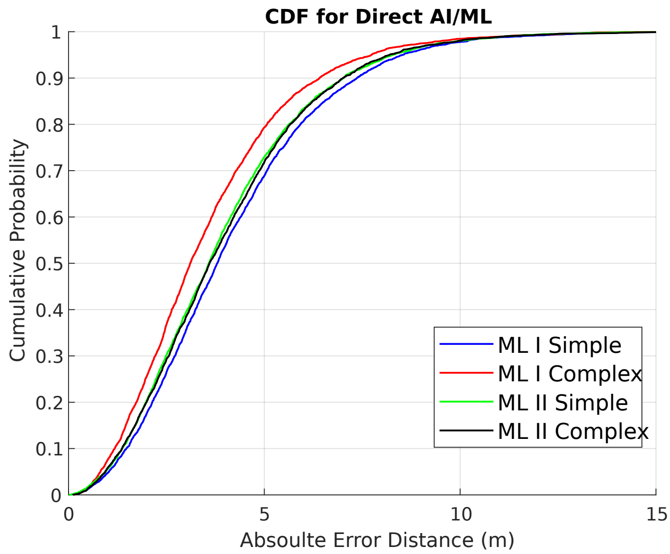


Figure 6.1: *CDF of Absolute Distance Error for the developed direct AI/ML positioning*

		CDF of the Absolute Position Error (m)				
		50%	**67%**	**80%**	**90%**	**95%**
Legacy Methods	**TOA**	8.175	13.223	17.902	24.251	30.821
	AOA	0.078	2.309	7.664	16.019	23.605
	TOA + AOA	0.051	2.185	5.528	9.993	14.246
Direct AI/ML	**Multi-link I Simple**	3.820	4.874	5.948	7.373	8.579
	Multi-link I Complex	3.197	4.142	5.186	6.385	7.677
	Multi-link II Simple	3.574	4.608	5.708	7.083	8.437
	Multi-link II Complex	3.408	4.448	5.539	6.925	8.368

Table 6.9: Evaluation of the location estimation for legacy and AI/ML Direct methods in the scenario case InF-DH{40%, 2m, 2m}

6.2 Model Selection

The selection of the best models has been done based mainly on the prediction performance but also taking into account the complexity of the model. The final selection is presented in the following schema 6.2, and the motivation is explained next.

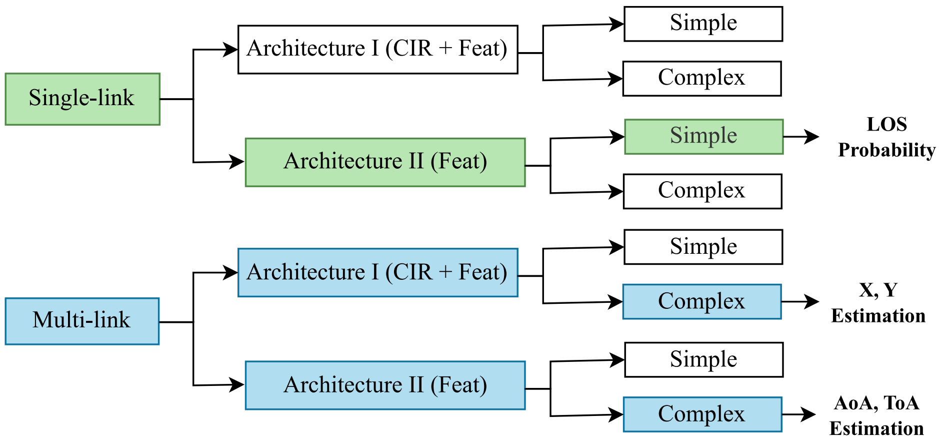


Figure 6.2: *Schema of the final model selection for 5G UE positioning*

Assisted AI/ML Positioning

For the LOS prediction, as Tables 6.3 shows, the performances of all the described models are satisfactory and really similar. For this case, the next aspect that has been taken into account for the selection has been the complexity. As the performance of all is alike, the choice was the one with fewer parameters, corresponding to the **Single-Link II simple model**.

Inside assisted positioning, but for the signal channel parameter estimation, the selection is motivated by Tables 6.5 and 6.4. In this case, the architecture with only the signal channel parameters outperforms the CIR and also has less computational complexity without having to process the CIR. Between the complex and simple models of this architecture, there is no excessive difference in terms of performance, and the complexity variation is not extreme in neither of them. This has led to ultimately selecting the **Multi-Link II complex model** for ToA and AoA estimation, due to a slightly better estimation over the simple, in expenses of a not excessive complexity gain.

<table>
<tr><th rowspan="1">Model</th><th>Inputs</th><th>Outputs</th><th>Type</th><th>Param.</th><th colspan="3">Performance</th></tr>
<tr><td rowspan="3">Single-Link II simple</td><td rowspan="3">Features</td><td rowspan="3">1x LOS Probability</td><td rowspan="3">FNN</td><td rowspan="3">1,897</td><td colspan="2">LOS</td><td>Metrics</td></tr>
<tr><td colspan="2">0.921</td><td>Accuracy</td></tr>
<tr><td colspan="2">0.909</td><td>F-score</td></tr>
<tr><td rowspan="3">Multi-Link II complex</td><td rowspan="3">Features</td><td rowspan="3">18x ToA
18x AoA</td><td rowspan="3">CNN</td><td rowspan="3">845,890</td><td>ToA</td><td>AoA</td><td>Metrics</td></tr>
<tr><td>7.488</td><td>6.8</td><td>MAE</td></tr>
<tr><td>0.984</td><td>0.968</td><td>R^2</td></tr>
</table>

Table 6.10: Summary of the AI/ML methods selected for assisted 5G positioning

Direct AI/ML Positioning

In the case of the AI/ML models for direct positioning, the final choice is based on Tables 6.7 and 6.8. For the X coordinate location estimation, the best performance is obtained with the Architecture I complex model. Nevertheless, the best estimation for the Y is done with the Architecture II simple model, but this performance is barely the same that the Architecture I complex model. The finally selected model for direct positioning is the **Multi-Link I complex model** because, for almost the same accuracy on the Y coordinate, it gives better X prediction than the Architecture II simple model.

Model	**Inputs**	**Outputs**	**Type**	**Param.**	**Performance**		
					X	*Y*	*Metrics*
Multi-Link I complex	CIR + Features	1x X 1x Y	CNN	880,386	2.417	2.314	*MAE*
					0.988	0.935	R^2

Table 6.11: Summary of the selected AI/ML methods selected for direct 5G positioning

6.3 Further Exploration

Once the best models have been selected, further investigation has been applied to the found solutions. The motivation for further exploring the CIR preprocessing types is to improve the current model performance due to it being an input with complex shape and characteristics. The other exploration consists of studying the robustness of the selected models to changes, observing their behavior against new and different scenarios, and at the same time, testing their performance in heavy NLOS scenarios also against the legacy methods.

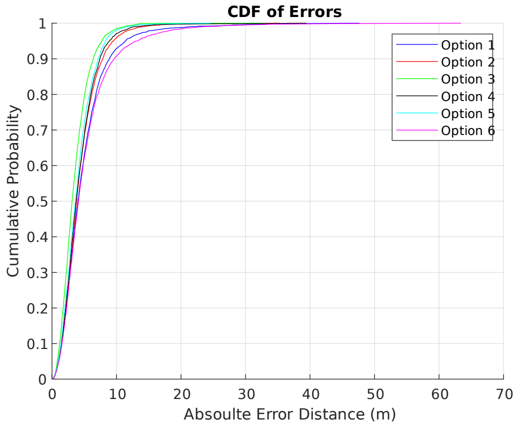
6.3.1 CIR Investigation

This part presents the different investigations of the CIR processing for improving the performance of the CNN. Due to the nature of the CIR, extra actions could be needed to effectively process the CIR on the AI/ML models and take all the potential and advantage from it. Different approaches were explored, such as different preprocessing techniques or the addition of extra dimensions to the convolutional stage. The tried methods are:

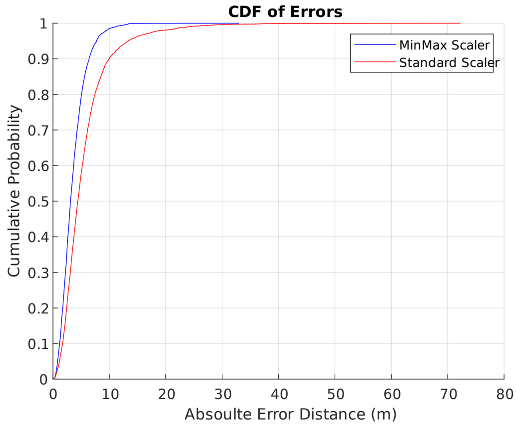
- **Option 1**. Enter as input the real and imaginary part of the CIR
- **Option 2**. Transform the input, instead of the raw CIR, and use the amplitude and phase.
- **Option 3**. Combine both of the previous options and input the real part, imaginary part, amplitude, and phase of the CIR.
- **Option 4**. Apply periodic function to the raw phase e.g., sinus.
- **Option 5**. Transform the CIR to dB and feed the combined option.

– **Option 6**. Apply a smoother function to the CIR e.g., averaging every three samples.

This CIR exploration was done for only the AI/ML direct positioning, applying the different options to the best-selected model. This model was the Multi-Link I complex, and the results obtained from the presented approaches are collected in the next table 6.12. Moreover, in Figure 6.3, the CDF of the different approaches are presented, together with a CDF comparison between the Min Max and Standard Scaler, both for Option 3.



(a) CDF of the CIR preprocessing options



(b) CDF of the Min Max and Standard Scaler

Figure 6.3: *CDF of the error for the different CIR preprocessing options (left) and for the Min Max Scaler for option 3 (right)*

	CDF of the Absolute Position Error				
	50%	**67%**	**80%**	**90%**	**95%**
Option 1	4.019	5.292	6.671	8.693	11.731
Option 2	3.815	4.899	6.1631	7.858	9.506
Option 3	3.197	4.142	5.186	6.385	7.677
Option 4	3.735	4.906	5.998	7.421	8.8983
Option 5	3.631	4.651	5.799	7.187	8.264
Option 6	4.114	5.496	7.082	9.625	13.10

Table 6.12: Results of the different CIR techniques investigation for the simple Multi-link model II

The CIR investigation option 3 from the previous figure has the same results as the best results from Table 6.11. The best results obtained are when CIR is used combining amplitude, phase, real and imaginary parts.

6.3.2 Model Sensitivity

Once the performance of the different models has been evaluated within the same scenario, their generalization has to be analyzed. This section explores the model performance to different environment changes and how well or badly they adapt the predictions for the new scenarios. For testing the sensibility to changes, several configurations have been simulated, varying different characteristics.

The different explored scenarios are collected in Table 6.13, where Scenario 0 represents the training scenario, and the rest are test scenarios for generalization evaluation.

	Convex Hull	**Clutter Case**	**Factory Layout**	**LOS Prob.**
Scenario 0	True	{40%, 2m, 2m}	A	0.4340
Scenario 1	False	{40%, 2m, 2m}	A	0.4402
Scenario 2	False	{60%, 2m, 2m}	A	0.2659
Scenario 3	False	{40%, 6m, 2m}	A	0.0245
Scenario 4	False	{40%, 2m, 2m}	B	0.4332
Scenario 5	False	{60%, 2m, 2m}	B	0.2581
Scenario 6	False	{40%, 6m, 2m}	B	0.0245

Table 6.13: Different scenarios explored for the complex Multi-link model I

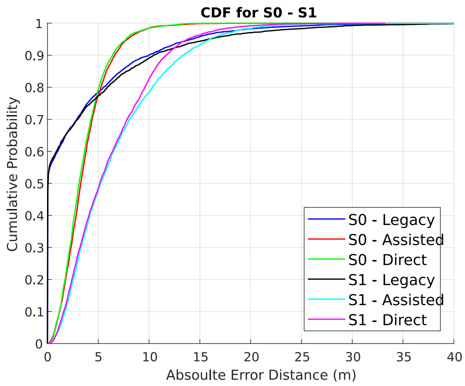
The models used have been the ones selected, presented in section 6.2, due to them demonstrating the best performance. These trained models are then tested with datasets of the presented scenarios, comparing their performance with the Legacy Methods in the following Table 6.9. Figure 6.4 compares the generalization performance when having different clutter settings, factory layout, and convex hull conditions. Additionally, figures present the performance when modifying the factory layout but with the same clutter and convex hull conditions, and vice versa.

Factory Layout represents the disposition of the cluttering objects inside the room, leading in Table 6.13 to different LOS probabilities.

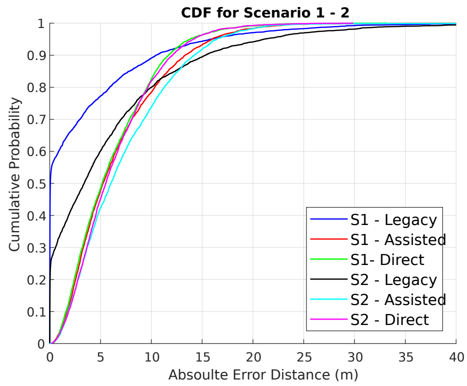
It is important to comment that from the assisted and legacy methods, only the position estimation that combines the ToA + AoA is shown. The reason to omit the others is that only using the AoA or the ToA always leads to the worst results, so only the best position estimation is presented corresponding to combining both the ToA + AoA.

The sensitivity to each condition is evaluated by comparing their performance in the presented scenarios. Comparing Scenario 0 and 1, the effect of the convex hull condition can be noticeable. When comparing Scenario 1 with 2 and 3, the impact of LOS probability is studied, maintaining the layout and convex hull constant.

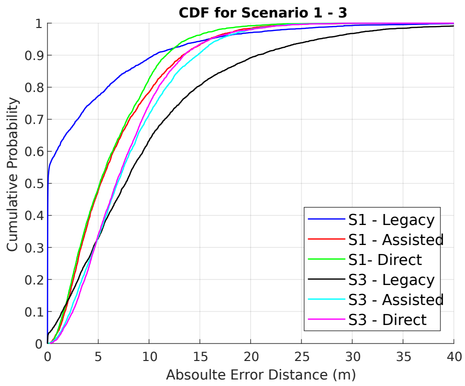
Finally, their performance on Scenarios 1 - 4, 2 - 5, and 3 - 6, is analyzed to observe the effect of a totally new factory layout. The factory layout can be understood as how is the clutter distributed inside. This generalization is relevant because it studies the effects of training the AI/ML model with data from one factory and then deploying it into a different one with similar clutter characteristics but a totally different distribution inside the hull.



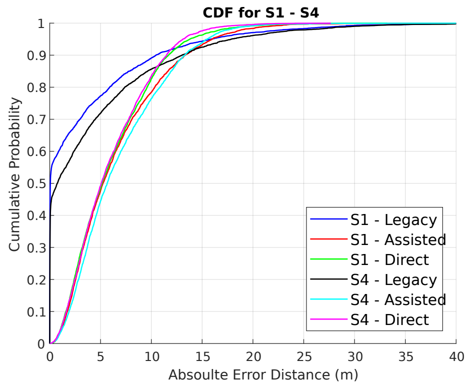
(a) CDFs for Scenario 0 and 1



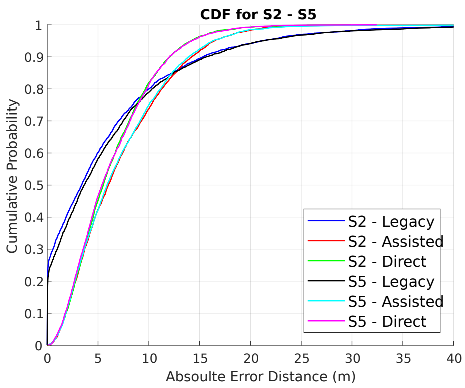
(b) CDFs for Scenario 1 and 2



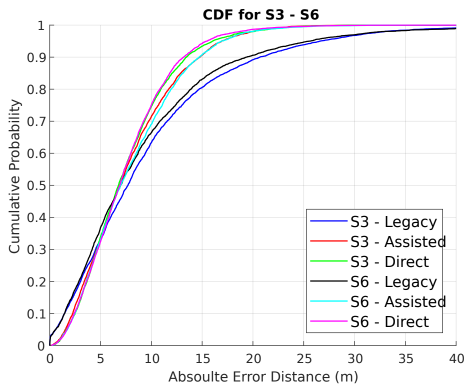
(c) CDFs for Scenario 1 and 3



(d) CDFs for Scenario 1 and 4



(e) CDFs for Scenario 2 and 5



(f) CDFs for Scenario 3 and 6

Figure 6.4: *Comparison of CDF for AoA +ToA assisted, direct, and AoA + ToA legacy for the different Scenarios*

		CDF of the Absolute Position Error (m)				
		50%	**67%**	**80%**	**90%**	**95%**
Scenario 0	**Legacy Methods**	0.051	2.185	5.528	9.993	14.246
	Assisted AI/ML	3.279	4.219	5.265	6.614	7.875
	Direct AI/ML	3.197	4.142	5.186	6.385	7.677
Scenario 1	**Legacy Methods**	0.774	3.870	7.372	12.325	17.463
	Assisted AI/ML	5.780	8.738	11.176	13.863	16.057
	Direct AI/ML	5.161	7.393	9.592	11.691	13.768
Scenario 2	**Legacy Methods**	3.057	6.370	9.827	14.790	19.832
	Assisted AI/ML	6.016	8.665	11.274	14.225	16.440
	Direct AI/ML	5.531	7.650	9.661	12.074	14.174
Scenario 3	**Legacy Methods**	7.263	10.645	14.599	20.002	26.131
	Assisted AI/ML	6.883	9.1984	11.708	14.766	16.938
	Direct AI/ML	6.769	8.945	10.934	13.462	16.011
Scenario 4	**Legacy Methods**	0.753	3.775	7.393	13.019	17.873
	Assisted AI/ML	5.649	8.192	10.851	13.292	15.495
	Direct AI/ML	5.080	7.250	9.3355	11.491	13.108
Scenario 5	**Legacy Methods**	3.644	6.905	10.315	15.642	21.163
	Assisted AI/ML	5.890	8.592	11.176	13.899	16.261
	Direct AI/ML	5.382	7.727	9.801	12.019	14.198
Scenario 6	**Legacy Methods**	6.987	10.119	13.895	19.350	25.614
	Assisted AI/ML	6.998	9.597	11.950	14.704	16.931
	Direct AI/ML	6.906	8.816	10.844	13.182	15.481

Table 6.14: Evaluation of the sensibility of the models to different scenarios

6.4 Discussion

After the presentation of all the results, this section discusses the evaluation carried out and the comparison of the different approaches.

This project, as presented in Chapter 4, is based on UpLink simulations due to requiring less computational time and resources than DownLink, which causes the interesting parameters to be the TOA and AOA. In the case of working with DownLink measurements, the process for both assisted and direct AI/ML approaches will be exactly identical but using TOD and AOD instead.

Analogously, the assisted models are now trained to predict a specific time and angle, but it would not require any significant effort to train them to predict any other possible interesting channel parameter. Instead e.g., zenith angle. It is a matter of only changing and selecting the desired inputs and outputs from the dataset.

During the training and testing evaluations, it has been observed that there is a need for at least around 8,000 UEs to allow the AI/ML models to learn properly. Training with fewer UEs distributed on the factory layout e.g., 4,000 UEs, leads to poorer estimation performance. Analogously, When using 12,000 UEs for training, apparently there has not been any big improvement. For this motive, it was decided to stick the training to 8,000 UEs, as data availability and gathering are always a constraint in real-life scenarios.

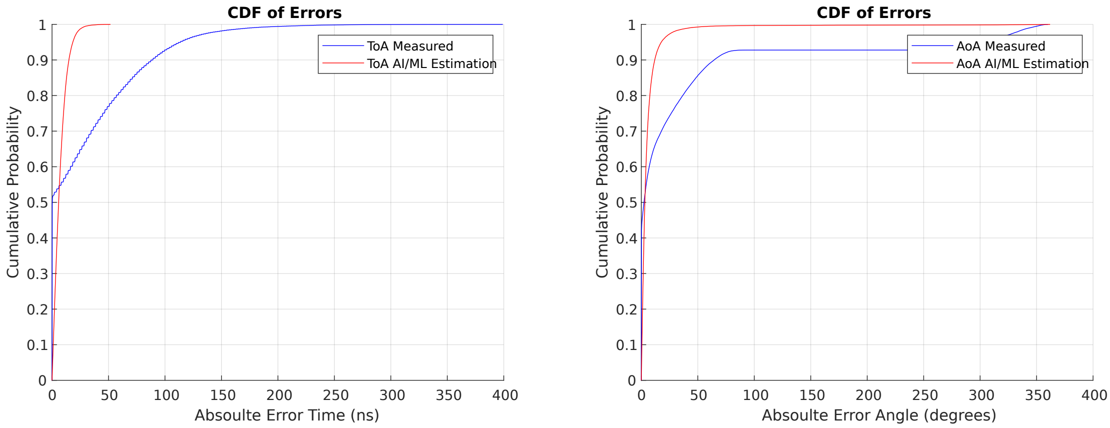
LOS Probability, ToA, and AoA estimation

In the AI/ML assisted methods for enhancing 5G positioning, one possibility is to predict if the link between the gNB and the UE suffers from NLOS or not. The final selected model achieved a 92.1% of accuracy when discriminating if a link is LOS or NLOS. This result, together with a 90.9% of F-score, indicates a good performance of the model for this task. This can be observed in Table 6.3.

Another positive aspect is the model does not directly output the label of the link but the probability of it being NLOS or LOS. This probability can be really useful in different situations, for example, to know the quality of the directly measured channel signal parameters. Moreover, this is achieved with a really simple model with only 1,897 parameters that take only a few features as input.

For the AoA and ToA prediction, as can be observed in the introduced Tables 6.4 and 6.5, the best models are able to predict them with acceptable performance. The R-squared for both the angle and time is over 95%, and the RMSE has no disproportionate values. The next figures show the error CDF for the ToA and AoA estimated with AI/ML versus the directly measured in the UE.

The comparison of the different architectures shows that the use of the CIR has not had a big impact when estimating these parameters. For this motive, the complexity and computational cost that it has to process the CIR is not justified for predicting the LOS probability and ToA and AoA parameters. Instead of the CIR and as presented in 5, they are introduced signal parameters such as RSRP, Beam Forming Gain, FPAR, and ZOA.



(a) CDF of the ToA error

(b) CDF of the AoA error

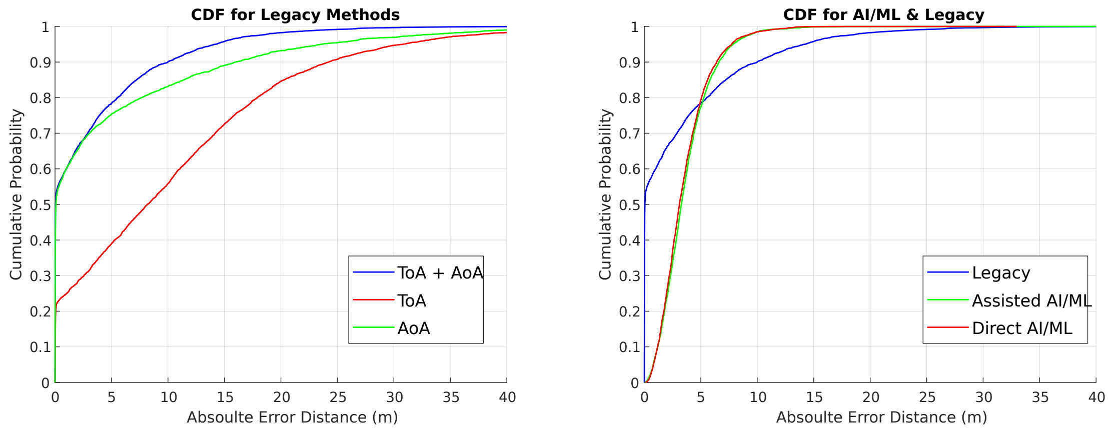
Figure 6.5: *CDF of the error for ToA (left) and AoA (right) estimated and measured*

Nevertheless, for the links with LOS conditions, when predicting AoA and ToA it is most likely that the quality and precision of the predictions are not satisfactory as the directly measured values. The assisted AI/ML has more potential to overpass legacy measurements in heavy NLOS conditions, where the directly measured times and angles can not be trusted.

Position Estimation

In terms of position results, the comparison among assisted, direct, and legacy methods is presented in Tables 6.6 and 6.9. Both tables have been fused in Table 6.15. Figure **??** represent the CDFs for the different legacy methods, and Figure **??** shows a comparison between the assisted and direct AI/ML with the legacy methods. Some observations are introduced next.

- The legacy method performs really well for half of the UEs, corresponding to the 50% CDF, that for the best legacy of ToA + AoA has a value of 0.051 m. This value indicates the absolute distance error, indicating 50% of the UEs have an error smaller than this one. Nevertheless, when increasing the percentage of the CDF, it can be noticed that all the legacy methods get worse errors, reaching 14.25 m in the 95% CDF with the best one, corresponding to the ToA + AoA.
- The developed AI/ML method for assisted positioning offers worse results for lower CDFs. In the case of 50% of the CDF for the ToA + AoA, the assisted has an absolute error of 3,28 m. However, this methodology equals the legacy performance of around 80% of the CDF. From that moment, the assisted outperforms the legacy, reaching 95% CDF with an error of 7.875 m.



(a) CDF for the ToA + AoA, ToA, and AoA legacy methods

(b) CDF for the legacy, assisted, and direct AI/ML methods

Figure 6.6: *CDF of the Absolute Distance Error for different legacy methods (left) and for comparing legacy, assisted, and direct AI/ML (right)*

		CDF of the Absolute Position Error (m)				
		50%	**67%**	**80%**	**90%**	**95%**
Legacy Methods	**TOA**	8.175	13.223	17.902	24.251	30.821
	AOA	0.078	2.309	7.664	16.019	23.605
	TOA + AOA	0.051	2.185	5.528	9.993	14.246
Assisted AI/ML	**TOA**	3.341	4.261	5.386	6.768	8.111
	AOA	3.509	4 477	5.759	7.341	9.143
	TOA + AOA	3.279	4.219	5.265	6.614	7.875
Direct AI/ML	**Multi-link I Complex**	3.197	4.142	5.186	6.385	7.677

Table 6.15: Evaluation of the location estimation for legacy and AI/ML methods in the scenario case InF-DH40%, 2m, 2m

– The AI/ML direct model developed has the same behavior as the assisted, being worst than the legacy for the 50% CDF with an error of 3.197 m, and offering better results for higher CDF, achieving 95% CDF only 7.677 m of error. Nonetheless, the direct positioning approach does not only surpass the legacy method but also the assisted performance, improving it by around 0.20 m of error for all the CDFs.

– The legacy and assisted methods offer better results combining AoA+ToA. Using only AoA in legacy methods improves the ToA results, leading to the conclusion that AoA brings more information than time. Regardless, it is the combination of both time and angles that allows the legacy and assisted model to achieve the best positioning results.

Tables 6.7 and 6.8 reflect a better performance in the results for the Y coordinate over the X. This can be explained due to the nature of the given values of X and Y. The Y coordinate has less value range, while X has a wider range of values due to the factory layout having more length than width. The higher variance on the X coordinate can make it harder to estimate, giving slightly worse results than for the Y.

The more satisfactory results of the AI/ML methods in front of the legacy for high CDF percentages could be explained due to they learn from the data. They have been trained with the available data of every link, while the legacy methods do not have the opportunity to identify and use underlying patterns in the signal channel for the scenarios. They can only compute the MLE with the available measurements.

For the direct methodology, the best model has CIR and features as inputs, while the assisted for the AoA and ToA prediction takes only features. Different prediction goals require different models. Including the CIR has shown a better approach for X and y positioning cases. A possible explanation is this prediction may need a more complex architecture and more data information than angle and time, which can be seen as intermediate parameters in the position calculation.

Hyperparametrization

From the different available search methods to explore the hyperparameter space, Bayesian optimization has been selected. The motivation is that it is more efficient when exploring the search space than others, such as grid or random search. Exploring the defined hyperparameter space, the Bayesian optimizator looks for the hyperparameter values that lead to the best performance, resulting in the best model.

Bayesian hyperparameter optimization has been demonstrated to improve and refine the results. Table 6.16 illustrates the performance changes of the selected models before and after the hyperparametrization for the AI/ML assisted positioning.

Applying hyperparametrization has also given some guidance toward the best solution and a deeper understanding of the project. It has helped to comprehend the relevance of different hyperparameters and neural network structures, as well as to identify and investigate all the possible elements that can compose a NN.

	ToA Estimation		***AoA Estimation***		***LOS Prob***	
	MAE	R^2	**MAE**	R^2	**Accuracy**	**PR-AUC**
Before	12.645	0.957	10.989	0.934	0.894	0.856
After	7.341	0.984	6.226	0.973	0.921	0.887

Table 6.16: Comparison of the AI/ML Assisted Positioning Hyperparameter Optimization

One relevant result of hyperparametrization is that the initial stage of the NN has never appeared in the best performing models. It was an initial group of layers before the regular NN that was supposed to help to reduce the large dimension gap between the input and the NN layer. As this structural option has never been selected for having a good model performance, it has led to the conclusion that this research branch may not be productive.

The main constraint of the hyperparametrization has been the running time and complexity, as some of these hyperparameter optimizations have taken days to be performed. The project time and computational limitations are reflected in constraints when defining the search space. More resources mean more hyperparameter space to explore and find a better combination.

In summary, Bayesian hyperparametrization has improved every model generated. For this reason, it is highly encouraged to perform a hyperparameter optimization in future model research for achieving the best possible solutions.

CIR Investigation

It is believed that the CIR has huge potential due to it containing a lot of information about the channel signal. For the case of AI/ML direct positioning, including it as an input has slightly improved the estimation performance than only including channel features.

The main problem is the CIR is a really complex signal, multidimensional, and with a lot of peaks and large value variations, and AI/ML models are delicate and sensible to this type of data. To solve this, during the project, different techniques to exploit this potential have been investigated, including different CNN architectures and various CIR preprocessing.

As has been presented in Table 6.12, there has been no great improvement when applying the explored options. The best option is to feed the AI/ML model with all the available information the CIR can give, including amplitude, phase, and real and complex parts. This was the final preprocessing step applied to the CIR. Testing to make the signal smoother, applying the sinus function to the phase, or using a smoother averaging function, has shown no substantial improvement in the model performance.

In conclusion, the CIR slightly improves the direct positioning results from only features, although it is expected to have a bigger impact and enhance the results. There might exist a different preprocessing step that helps the model to get more of the available information contained on the CIR. Future investigation may be needed to succeed in better location results by the CIR usage.

Model Sensitivity

The model generalization ability is reflected in Table 6.14. Performance is evaluated for the different scenarios defined in Table 6.13, where the convex-hull condition, the layout, and the LOS probability are modified for studying the sensibility of the different methods.

First of all, the AI/ML models have been trained with convex hull assumption but tested with the non-convex hull simulated scenarios. This directly leads to a deterioration in all the results from the AI/ML methods, but not in the legacy methods ones.

The assisted AI/ML models do not generalize quite well to this change in the convex hull condition, leading that, in most cases, the legacy method performs better, even for high CDFs. Nevertheless, for the AI/ML direct positioning, despite the overall location estimation getting worst, it is still able to satisfactorily surpass the legacy methods for high CDFs.

Another trend that can be observed is that the LOS probability plays a vital role in the performance of legacy and AI/ML models. Legacy methods are highly dependent on this parameter due to they use the measured times and angles directly. This derivates in large location error in scenarios with low LOS probability, where most of the links are in NLOS conditions.

The AI/ML models show a different behavior. They appear to be more robust to this change. For these heavily NLOS scenarios, they offer a significant improvement in comparison to the legacy method for every CDF percentage.

Moreover, when studying how the models are affected by a change in the factory layout, it can be observed there is no difference in the performance when modifying the factory layout from A to B. This demonstrates that the AI/ML models could be trained with data from some concrete layout and then deployed in another factory with a different layout without having any notable deterioration in their estimation performance.

The proposed methods offer good generalization and robustness to the proposed scenarios. Their learning ability seems to have allowed the models to analyze and identify underlying relations and patterns, being able to generalize satisfactorily even to different and unseen scenarios, such as having almost no LOS links.

In summary, the developed solutions and their future investigation are highly encouraged for highly NLOS scenarios, where legacy methods present really poor performance. Some typical scenarios with this very low LOS probability are Indoor Factories with a high amount of scattering objects, as this project investigates.

7 Conclusion

7.1 Summary

The contents of this thesis work document the theory and methodology for creating and implementing AI/ML methods for enhancing the UE location in 5G networks, including all the significant stages in the development process. They include model generation, training, validation and testing, and their inference performance on new different scenarios.

The primary goal of this project was to achieve an accuracy enhancement of the current 5G positioning techniques. The gathered results demonstrate that the objective has been achieved, as the application of the AI/ML methods surpasses the performance of the legacy methods from above 80% of the error CDF and significantly improves the accuracy of the position estimation in heavily NLOS scenarios.

For fulfilling the objectives of the project, several types of AI/ML models have been developed. The characteristics for each of them depend on the nature of the prediction needed, divided into two groups: the ones estimating regression problems as the AoA, ToA, or the X-Y UE position, and the ones that output the LOS probability. From each of the explored models, different types of complexity have been studied to analyze their impact on performance.

Regarding data and feature selection for feeding the AI/ML models, this report also offers a broad explanation of the final selection and its preprocessing. A selection of a few signal channel parameters, combined with the complex CIR and their amplitude and phase for some models, have been proven to be the most suitable inputs to the models. The preprocessing that has been proven to give the best performance for both has been the regular Min Max normalization.

During the generation and training process, several architectures and layer structures have been tested, disregarding different approaches. For selecting the most promising channel features, a lot of trials have been done until a satisfactory combination has been found, also discarding different normalizers and scalers for preprocessing the data due to poor estimation performance. This project exploration has also observed that 8,000 is an acceptable number of UE on the factory layout for having reasonable estimation results.

Finally, this investigation has also demonstrated that by applying Bayesian hyper-parametrization to the final selected models, an improvement in their performance can be achieved.

The project indicates that AI/ML models are suitable for 5G positioning even in indoor factory scenarios with high NLOS issues. Moreover, the deployment on the UE side will take advantage of analyzing the 18 links with the gNBs at once. For the AI/ML methods, when estimating regression problems, this approach was more beneficial. In the case of deploying the models on the gNBs, this approach would not be possible, or it may require extra signaling to transmit this information.

The generalization ability of the final models has also been investigated, showing reasonable robustness against new unseen scenarios with different settings. Furthermore, the AI/ML methods, especially the direct positioning, have shown really good performance when predicting the UE location in extreme NLOS scenarios, outperforming the legacy methods' estimations.

Concerning the complexity of the developed models, all the developed and finally selected models can be considered as small AI/ML models as they have less than a million parameters. Small models are typically used when computational resources are limited, such as UE devices. However, even having limited computational power, small models are still effective in certain scenarios, as these project results have confirmed.

Hence, this thesis report gives a better understanding of the indoor factory 5G positioning problem, presents some viable solutions to mitigate its problems using AI/ML methods, and establishes a starting direction for further research on this topic.

7.2 Contribution

This thesis project has contributed to the field and Sony in various manners. The main contribution corresponds to the development of three final AI/ML models in Python for enhancing the 5G location services. All the AI/ML methods created are considered small models because they have relatively few parameters and still perform with satisfactory accuracy.

The most simple model developed has only 1,897 parameters, uses only signal channel parameters, and predicts the LOS probability of each link between a UE and a gNB. The other model generated for AI/ML assisted localization predicts the AoA and ToA of each link with 845,890 parameters, using the same parameters as the previous one. Lastly, the third model corresponds to the AI/ML direct positioning and predicts the exact UE position, making use of some signal channel parameters together with the CIR, composed of 880,386 parameters.

Following the previous contributions, it has been proven that the deployment of AI/ML methods in 5G location services is a viable solution with great potential. This work supports the future investigation path toward a more accurate UE positioning through the development and implementation of AI/ML methods.

Overall, this thesis has demonstrated the practical viability of AI/ML 5G positioning methods, obtaining less than 8 meters of absolute error for the 95% of the UEs in an indoor factory setting.

Relevance within the industrial sector

An accurate and robust positioning methodology using 5G networks is really relevant for the industrial factories of the future. In the project context, in an indoor factory, enhanced positioning techniques can improve the tracking of equipment, vehicles, workers, or assets. This potentially benefits some typical industrial aspects such as process optimization, autonomous systems, assets tracking, or worker safety, among others.

Nowadays, in the industrial sector, where efficiency and productivity are paramount, a satisfactory positioning within indoor factories is crucial. Traditionally location services, such as GPS, often struggle when proving reliable and precise UE locations for the introduced scenarios. Nevertheless, 5G opens the possibility to overcome these limitations, prescinding from the other technologies and changing the way the current factories operate.

Crowded factories with IIoT devices are every day getting closer to become a reality. Deploying AI/ML models on these connected devices inside a factory is a potential solution for enhancing the positioning requirements of the smart robots, sensors, or machinery that operates inside a factory.

Accurate location using 5G in an indoor factory would allow for real-time tracking of the industrial elements. This aspect could enable more efficient and robust inventory management of the assets, reduce the search time for equipment or machinery, or monitor the working personnel for proactive safety measures. All of this information provides the possibility to improve the overall operation efficiency, reduce downtimes, or apply workflow process optimization to reduce unnecessary movements and identify inefficiencies.

Precise tracking all the relevant elements within a factory would generate a huge amount of data about the trends and patterns of its daily operations. The gathered information can also be used for predictive strategies and optimization purposes. This could benefit diverse aspects of the industrial plant, such as energy efficiency, supply chain processes, or safety protocols.

Having an accurate positioning is vital for the deployment of autonomous systems and robotics, essential elements in the interconnected and smarter Factory 4.0. For example, a robust positioning will benefit path planning and collision avoidance in typical robot operations, such as pick-and-place, assembly tasks, material handling, and transportation. Only using the 5G network to monitor the location of the device can also help in having a remote control and monitoring of an automated factory.

In conclusion, the AI/ML solutions presented help to enhance the 5G positioning in indoor factories, a really complex environment that requires having an accurate location service for many crucial applications.

7.3 Future Work

Although an enhancement of the 5G legacy positioning methods has been achieved using AI/ML models, there are still several avenues for further research on this topic. This project has been focused on studying the viability of this solution and give a starting point for this investigation. This section presents some proposed main points and approaches for further improvement and development of the project.

The main limitations of this work have been time and equipment. The thesis project has a duration constraint of only a few months. This factor affected the further development of some of the proposed alternatives due to the amount of time invested at the beginning of the project in extensive research of the background theory and proposed standards.

Regarding the equipment, the computational capacity is crucial not only for simulating the dataset for training and testing but also for applying hyperparametrization to the AI/ML models. As access to a big server was only possible in the last stages, the available datasets and viable hyperparmetrizations during most of the project were limiting. Nevertheless, some opportunities for further development of the project work are presented next.

– Increasing the sizes of the datasets. Investigating the actual impact of having a truly large dataset e.g., 50,000 UEs on the factory. This factor might help the models to continue learning extra information to improve their performance.

– Using data obtained from a real scenario instead of a simulated one. This would reflect how the models would function in real-life scenarios. Training with real data could also help to explain behaviors that are actually not introduced on the simulator.

– Studying and probing with different AI/ML typologies. This project has only been able to cover the investigation of CNN and FNN, but there exist other types of AI/ML algorithms that could be tested.

– Further exploration of the search space with the Bayesian Hyperparametrization. Having more time, together with adequate computational resources, would have allowed a larger hyperparameter space to explore. A bigger search space with a more exhaustive search potentially means finding better hyperparameters for the models.

– Extra CIR preprocessing techniques. Due to the complex nature of the channel signal, specific preprocessing methods may be needed to completely extract all the contained information.

– Further exploration of the models' generalization ability. Creating new scenarios with different clutter settings, characteristics, layouts, or sizes to continue evaluating the sensitivity and robustness of the models to various changes and their ability to cope with different severity when it comes to NLOS conditions.

– Investigating the effect of mixing datasets from different scenarios for training or for fine-tuning the models. The performance of the models will be affected depending on the characteristics of the training data, so exploring if mixing data from scenarios with distinct characteristics improves their performance and/or generalization ability.

– Combining the obtained results from the AI/ML assisted methods into the AI/ML direct methods. For example, feeding the direct model with also the estimated AoA and ToA or using the LOS probability prediction when deriving the UE position for weighting the quality of the different links.

– Introducing the Z-coordinate for prediction as well. In future researches, UE devices could be considered to have not constant height.

Bibliography

[1] M. Posluk, J. Ahlander, D. Shrestha, S. M. Razavi, G. Lindmark, and F. Gunnarsson. "5G deployment strategies for high positioning accuracy in indoor environments." In: *arXiv preprint arXiv:2105.09584v1* (2021).

[2] A. Nessa, B. Adhikari, F. Hussain, and X. N. Fernando. "A survey of machine learning for indoor positioning." In: *IEEE Access* 8 (2020), pp. 214945–214965.

[3] X. Lin et al. "Positioning for the internet of things: A 3GPP perspective." In: (2017).

[4] F. Halawa, H. Dauod, I. Gyu Lee, Y. Li, S. Won Yoon, and S. Hoon Chung. "Introduction of a real time location system to enhance the warehouse safety and operational efficiency." In: *International Journal of Production Economics* 224 (2020).

[5] 3GPP RAN. *Summary #6 of Evaluation on AI/ML for positioning accuracy enhancement.* 3GPP RAN #112bis-e. 2023.

[6] Y. Zhang and Y. Shu. "Investigation on Positioning Techniques for Indoor Office Scenario." MA thesis. Master's thesis, 2020.

[7] .

[8] Satyam Dwivedi, Ritesh Shreevastav, Florent Munier, Johannes Nygren, Iana Siomina, Yazid Lyazidi, Deep Shrestha, Gustav Lindmark, Per Ernstrom, Erik Stare, Sara M. Razavi, Siva Muruganathan, Gino Masini, Åke Busin, and Fredrik Gunnarsson. "Positioning in 5G networks." In: (2021).

[9] 3rd Generation Partnership Project. *3GPP TR 38.857 V17.0.0: Technical Report.* Tech. rep. Study on NR Positioning Enhancements (Release 17). Technical Specification Group Radio Access Network, 2021.

[10] Tom M Mitchell. *Machine Learning.* 1997.

[11] Jay H Lee, Joohyun Shin, and Matthew J Realff. "Machine learning: Overview of the recent progress and implications for the process systems engineering field." In: *Computers & Chemical Engineering* 114 (2018), pp. 111–121.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* `http://www.deeplearningbook.org`. MIT Press, 2016.

[13] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow.* " O'Reilly Media, Inc.", 2022.

[14] 3rd Generation Partnership Project. *3GPP TR 38.901 V16.1.0: Technical Report.* Tech. rep. Study on channel model for frequencies from 0.5 to 100 GHz (Release 16). Technical Specification Group Radio Access Network, 2019.

Appendix A

Economical Budget

A.1 Price Table 1: Labor

Code	**Units**	**Denomination**	**Salary** (€/h)
MO.MUII	h	Industrial Engineer	25

Table A.1: Labor price table

A.2 Price Table 2: Material

A.2.1 Software

In order to obtain the depreciation period of the licenses, it has been considered that each year has 220 labor days, with a full work day of 8 hours.

$$\text{hours}_{year} = \frac{8h}{1day} \times \frac{220days}{1year} = 1.760 \text{ h/year} \tag{A.1}$$

It is important to emphasize that Microsoft Office is free for all students and professors in academic institutions. Regarding the Python license, it is an open-source programming language that is free and available for everyone to use.

Code	**Units**	**Denomination**	**Price** (€/$year$)	**Price** (€/h)
MS.MAT	u	MATLAB license	860	0,49
MS.PY	u	Python license	0	0
MS.O365	u	Microsoft Office 365 license	0	0

Table A.2: Software price table

A.2.2 Hardware

For deriving the depreciation of the laptop, a service life of 5 years has been considered. For the useful hours per year, the previously calculated number has been used.

$$\text{hours}_{depreciation} = 5years \times \frac{1.760h}{1year} = 8.800 \text{ h} \tag{A.2}$$

Code	**Units**	**Denomination**	**Price** (€/*year*)	**Price** (€/*h*)
MH.OP	u	Laptop	900	0,1

Table A.3: Hardware price table

A.3 Price Table 3: Unit Prices

Code	**Units**	**Description of work items**	**Price** (€)
1	*Chap. 1*	*Necessities definition and objectives planification*	
1.1	u	Information research and alternatives study	155,12
1.2	u	Technical meeting	51,60

Table A.4: Unit prices table of Chapter 1

Code	**Units**	**Description of work items**	**Price** (€)
2	*Chap. 2*	*Data generation of the simulated scenarios*	
2.1	u	Simulator functionalities and parameter investigation	843,45
2.2	u	MATLAB Simulation of a specific scenario	16,52
2.3	u	Technical meeting	51,60

Table A.5: Unit prices table of Chapter 2

Code	**Units**	**Description of work items**	**Price** (€)
3	*Chap. 3*	*Creation, training and evaluation of the AI/ML methods*	
3.1	u	Generated data investigation and Python transformation	517,06
3.2	u	AI/ML creation and development	3.619,42
3.3	u	Test of AI/ML performance	25,90
3.4	u	AI/ML model hyperparameter optimization	29,05
3.5	u	Technical meeting	51,60

Table A.6: Unit prices table of Chapter 3

Code	**Units**	**Description of work items**	**Price** (€)
4	*Chap. 4*	*Information recompilation and processing*	
4.1	u	Project report writing	2.068,24

Table A.7: Unit prices table of Chapter 4

A.4 Price Table 4: Itemized Prices

Code	Units	Description	Efficiency	Price	Sum
1	Chap. 1	Necessities definition and objectives planning			
1.1	*u*	*Information research and alternatives study*			
MO.MUII	h	Industrial Engineer	6	25	150
MH.OP	h	Laptop	6	0,1	0,6
			CD		150,6
			CDC (3%)		4,52
			Total 1.1		155,12€
1.2	*u*	*Technical meeting*			
MO.MUII	h	Industrial Engineer	2	25	50
MH.OP	h	Laptop	1	0,1	0,1
			CD		50,1
			CDC (3%)		1,5
			Total 1.2		51,6€

Table A.8: Itemized price table of Chapter 1

Code	Units	Description	Efficiency	Price	Sum
2	Chap. 2	Data generation of the simulated scenarios			
2.1	*u*	*Simulator functionalities and parameter investigation*			
MO.MUII	h	Industrial Engineer	32	25	800
MH.OP	h	Laptop	32	0,1	3,2
MS.MAT	h	MATLAB	32	0,49	15,68
			CD		818,88
			CDC (3%)		24,57
			Total 2.1		843,45€
2.2	*u*	*MATLAB Simulation of an specific scenario*			
MO.MUII	h	Industrial Engineer	0,5	25	12,5
MH.OP	h	Laptop	6	0,1	0,6
MS.MAT	h	MATLAB	6	0,49	2,94
			CD		16,04
			CDC (3%)		0,48
			Total 2.2		16,52€
2.3	*u*	*Technical meeting*			
MO.MUII	h	Industrial Engineer	2	25	50
MH.OP	h	Laptop	1	0,1	0,1
			CD		50,1
			CDC (3%)		1,5
			Total 2.3		51,6€

Table A.9: Itemized price table of Chapter 2

Code	**Units**	**Description**	**Efficiency**	**Price**	**Sum**
3	Chap. 3	Creation, training and evaluation of the AI/ML methods			
3.1	*u*	*Generated data investigation and Python transformation*			
MO.MUII	h	Industrial Engineer	20	25	500
MH.OP	h	Laptop	20	0,1	2
MS.MAT	h	Python	20	0	0
			CD		502
			CDC (3%)		15,06
			Total 3.1		517,06€
3.2	*u*	*AI/ML creation and development*			
MO.MUII	h	Industrial Engineer	140	25	3500
MH.OP	h	Laptop	140	0,1	14
MS.MAT	h	MATLAB	140	0	0
			CD		3.514
			CDC (3%)		104,42
			Total 3.2		3.619,42€
3.3	*u*	*Test of AI/ML performace*			
MO.MUII	h	Industrial Engineer	1	25	25
MH.OP	h	Laptop	1,5	0,1	0,15
MS.MAT	h	MATLAB	1,5	0	0
			CD		25,15
			CDC (3%)		0,75
			Total 3.3		25,90€
3.4	*u*	*AI/ML model hyperparameter optimization*			
MO.MUII	h	Industrial Engineer	1	25	25
MH.OP	h	Laptop	32	0,1	3,2
MS.MAT	h	MATLAB	32	0	0
			CD		28,2
			CDC (3%)		0,85
			Total 3.4		29,05€
3.5	*u*	*Technical meeting*			
MO.MUII	h	Industrial Engineer	2	25	50
MH.OP	h	Laptop	1	0,1	0,1
			CD		50,1
			CDC (3%)		1,5
			Total 3.5		51,6€

Table A.10: Itemized price table of Chapter 3

Code	Units	Description	Efficiency	Price	Sum
4	Chap. 4	Information recompilation and processing			
4.1	*u*	*Project report writing*			
MO.MUII	h	Industrial Engineer	80	25	2000
MH.OP	h	Laptop	80	0,1	8
MS.O365	h	Microsoft Office 365	80	0,1	0
			CD		2.008
			CDC (3%)		60,24
			Total 4.1		2.068,24€

Table A.11: Itemized price table of Chapter 4

A.5 Price Table 5: Partial Budgets

Code	Units	Description	Efficiency	Price	Sum
1	*Chap. 1*	*Necessities definition and objectives planning*			
1.1	u	Information research and alternatives study	4	155,12	620,48
1.2	u	Technical meeting	2	51,60	103,20
			Total Chap. 1		723,68€

Table A.12: Partial budget of Chapter 1

Code	Units	Description	Efficiency	Price	Sum
2	*Chap. 2*	*Data generation of the simulated scenarios*			
2.1	u	Simulator functionalities and parameter investigation	1	843,45	843,45
2.2	u	MATLAB Simulation of a specific scenario	10	16,52	165,20
2.3	u	Technical meeting	2	51,60	103,20
			Total Chap. 2		1.111,85€

Table A.13: Partial budget of Chapter 2

Code	Units	Description	Efficiency	Price	Sum
3	*Chap. 3*	*Creation, training, and evaluation of the AI/ML methods*			
3.1	u	Generated data investigation and Python transformation	1	517,06	517,06
3.2	u	AI/ML creation and development	1	3.619,42	3.619,42
3.3	u	Test of AI/ML performance	20	25,90	518
3.4	u	AI/ML model hyperparameter optimization	4	29,05	116
3.5	u	Technical meeting	4	51,60	206,40
			Total Chap.	3	4.976,88€

Table A.14: Partial budget of Chapter 3

Code	Units	Description	Efficiency	Price	Sum
4	*Chap. 4*	*Information recompilation and processing*			
4.1	u	Project report writing	1	2.068,24	2.068,24
			Total Chap.	4	2.068,24€

Table A.15: Partial budget of Chapter 4

A.6 Price Table 6: Total Budget

Chapter 1: Necessities definition and objectives planning	723,68
Chapter 2: Data generation of the simulated scenario	1.111,85
Chapter 3: Creation, training, and evaluation of the AI/ML methods	4.976,88
Chapter 4: Information recompilation and processing	2.068,24
Total Material Execution	**8.880,65€**

Table A.16: Total budget of Material Execution

Total budget of Material Execution (PEM)	8.880,65
Industrial benefit (6%)	532,84
General expenses (3%)	1.154,48
Total Execution by Contract	**10.567,97€**

Table A.17: Total budget of Execution by Contract

Total budget of Execution by Contract	10.567,97
IVA (21%)	2.219,27
Total Bidding Base	**12.787,24€**

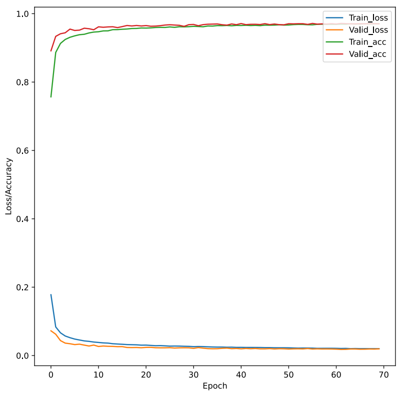
Table A.18: Total budged (Bidding Base)

The projected budget amounts to the stated quantity of TWELVE THOUSAND SEVEN HUNDRED EIGHTY-SEVEN EUROS AND TWENTY-FOUR CENTS.

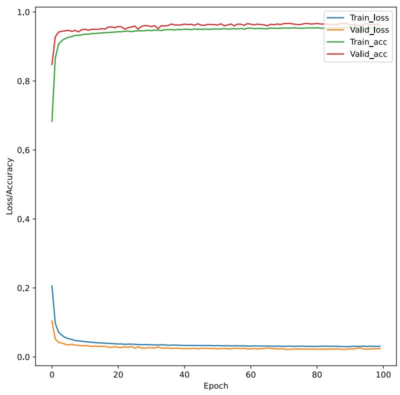
Appendix B

AI/ML Training Results

This first appendix collects all the figures corresponding to the developed AI/ML models, introduced in Chapter 5. From each model architecture and typology, the training curves are presented. Moreover, it is also shown the CDF of the absolute distance error, for the direct AI/ML location problems, or the Confusion Matrix, for the labeling prediction.

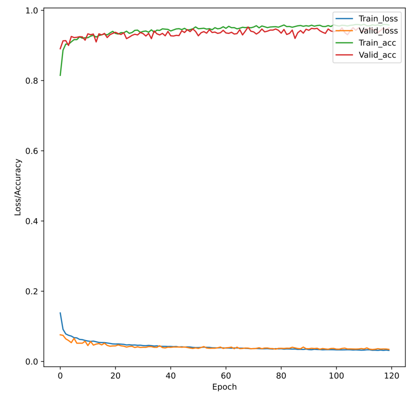


(a) ML I Complex - Training Curve for ToA and AoA estimation

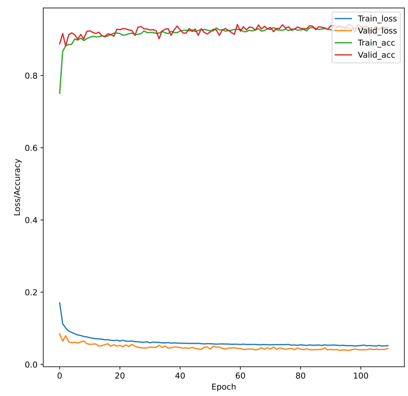


(b) ML I Simple - Training Curve for ToA and AoA estimation

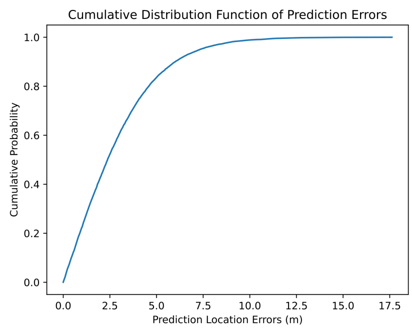
Figure B.1: *Training curves of the Multi-Link Architecture I models simple and complex for ToA and AoA estimation*



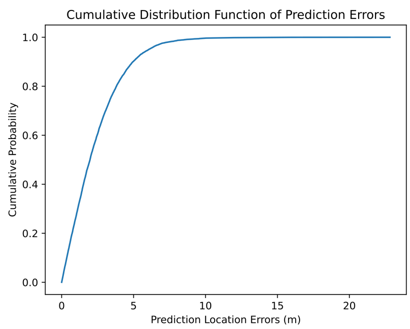
(a) ML I Complex - Training Curve for X-Y location estimation



(b) ML I Simple - Training Curve for X-Y location estimation

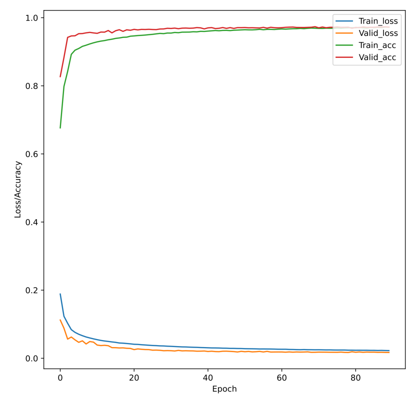


(c) ML I Complex - CDF of the absolute error location

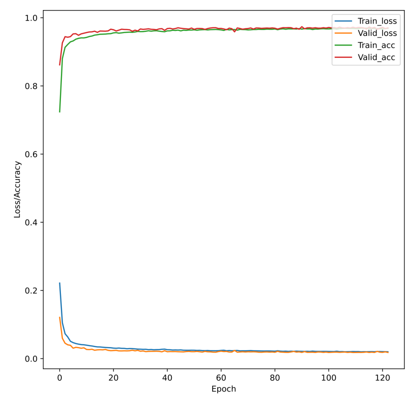


(d) ML I Simple - CDF of the absolute error location

Figure B.2: *Training curves and CDFs of the Multi-Link Architecture I models simple and complex for X and Y location estimation*

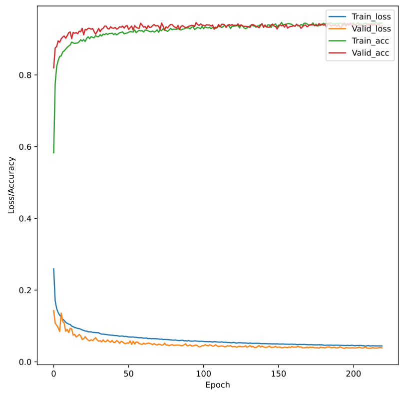


(a) ML II Complex - Training Curve for ToA and AoA estimation

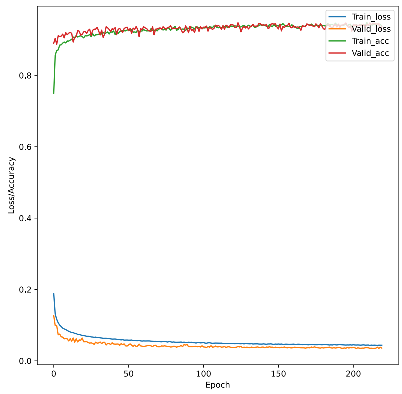


(b) ML II Simple - Training Curve for ToA and AoA estimation

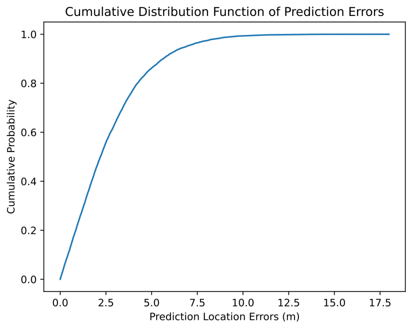
Figure B.3: *Training curves of the Multi-Link Architecture II models simple and complex for ToA and AoA estimation*



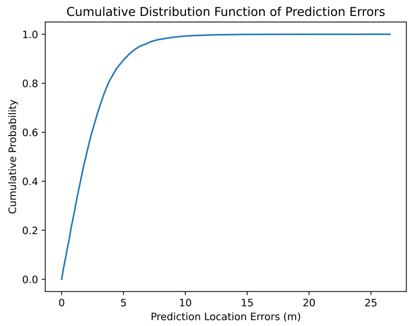
(a) ML II Complex - Training Curve for X-Y location estimation



(b) ML II Simple - Training Curve for X-Y location estimation

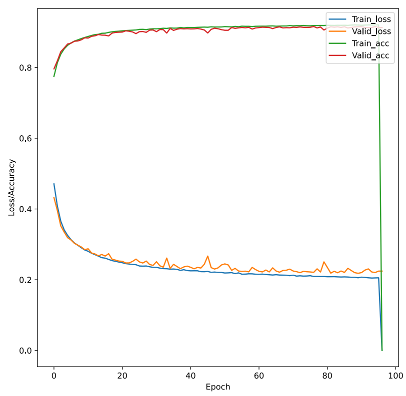


(c) ML II Complex - CDF of the absolute error location

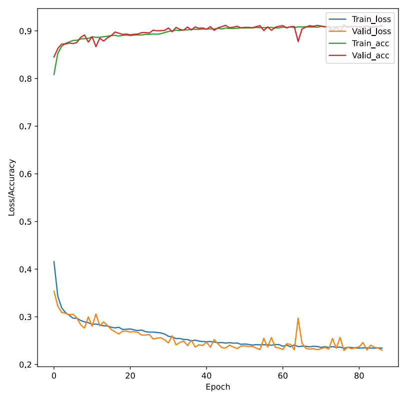


(d) ML II Simple - CDF of the absolute error location

Figure B.4: *Training curves and CDFs of the Multi-Link Architecture II models simple and complex for X and Y location estimation*

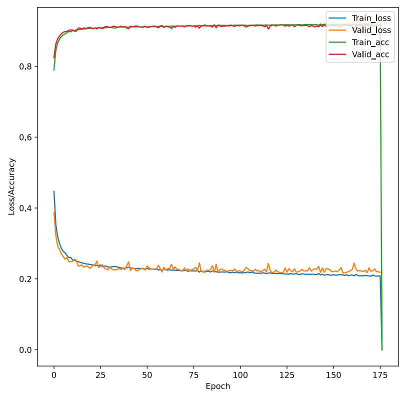


(a) SL I Complex - Training Curve for LOS prediction

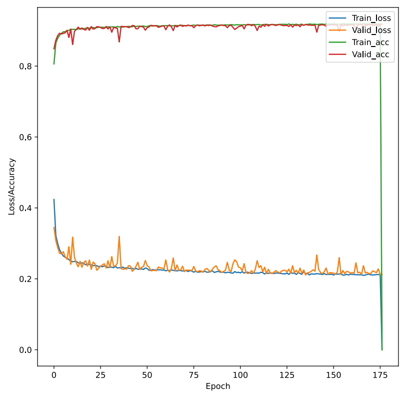


(b) SL I Simple - Training Curve for LOS prediction

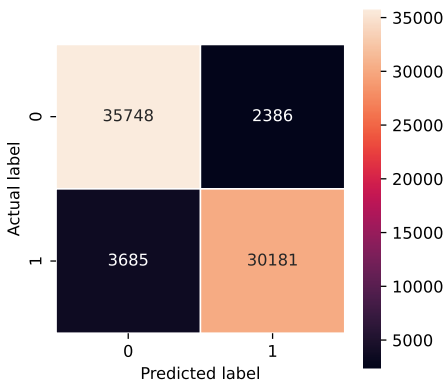
Figure B.5: *Training curves of the Single-Link Architecture I models simple and complex for LOS prediction*



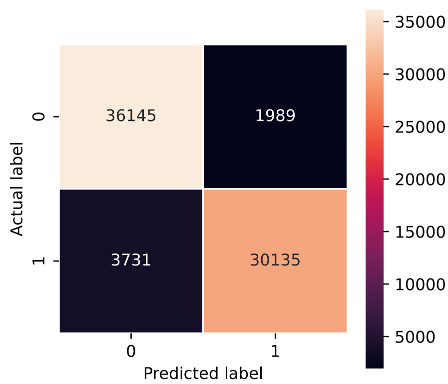
(a) SL II Complex - Training Curve for LOS prediction



(b) SL II Simple - Training Curve for LOS prediction



(c) SL II Complex - Confusion Matrix for LOS prediction



(d) SL II Simple - Confusion Matrix for LOS prediction

Figure B.6: *Training curves and Confusion Matrices of the Single-Link Architecture I models simple and complex for LOS prediction*

Appendix C

CIR Investigation

The current appendix introduces the figures corresponding to all the CIR investigated forms, described in Chapter 6. Among them, there are the real and complex parts, magnitude, phase, sinus of the phase, or the averaging smoothing function.

All the presented figures represent the normalized CIR, applying the MinMax Scaler. However, the results of using the Standard Scaler, also investigated, are the exactly the same except of the scale values. All of the values from the CIR in this case are bounded between 0 and 1, for Standard Scaler, there are any range restriction but he constraint to have zero mean and unit variance.

The figures from this appendix are the 18 link corresponding to one specific UE. As the plots are one after the normalization, the following figures represent directly the inputs of the CNN.

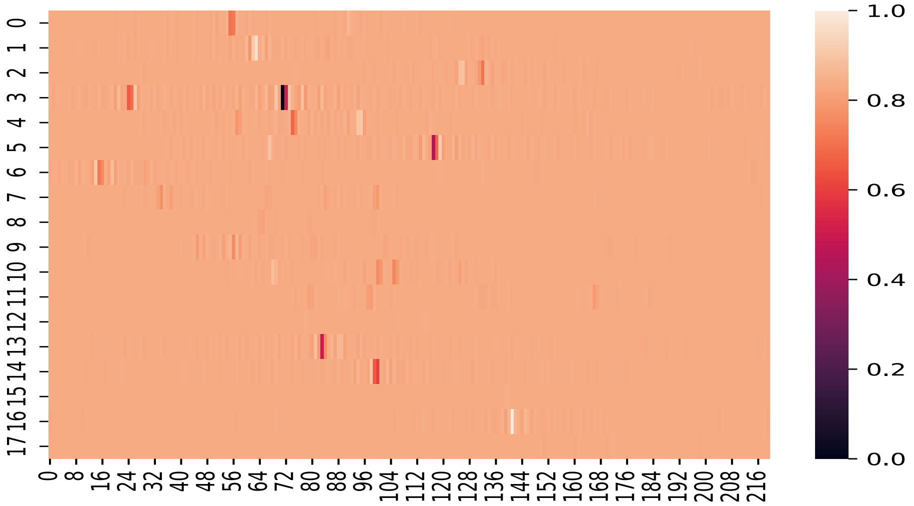


Figure C.1: *Option 1. CIR real part*

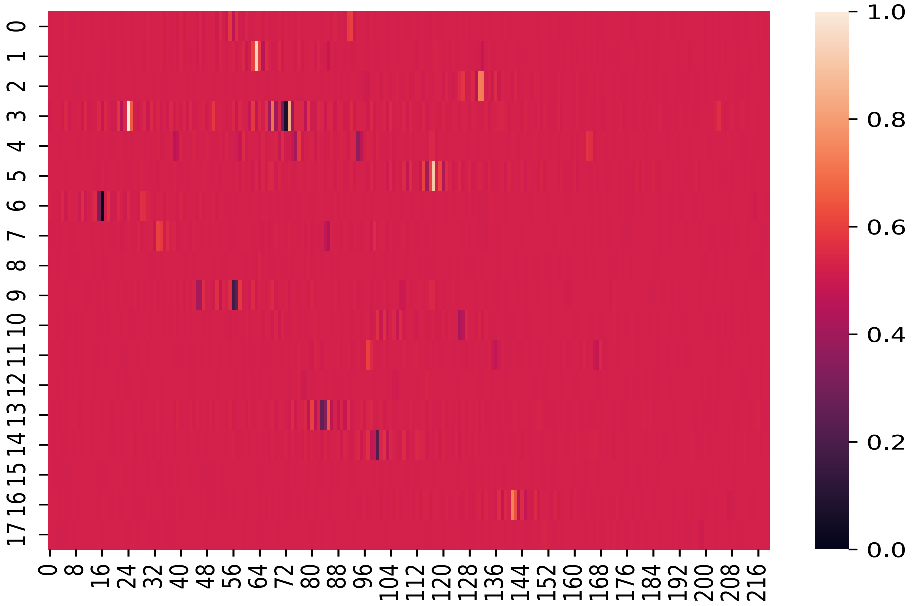


Figure C.2: *Option 1. CIR complex part*

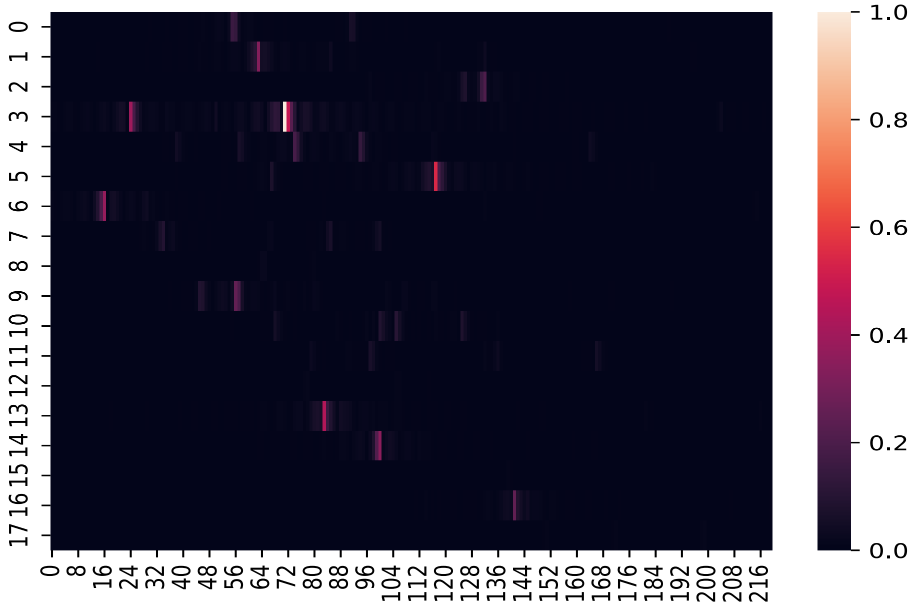


Figure C.3: *Option 2. CIR amplitude*

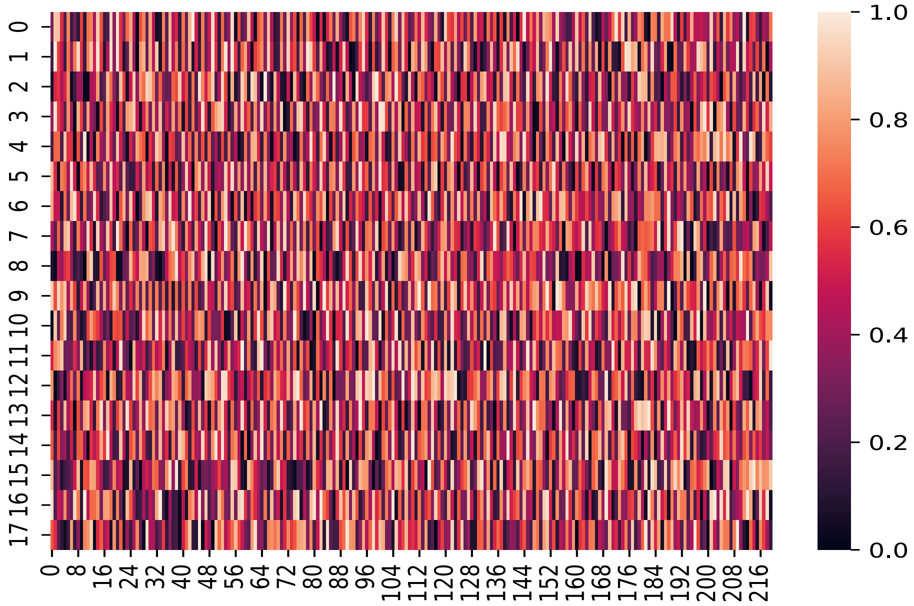


Figure C.4: *Option 2. CIR phase*

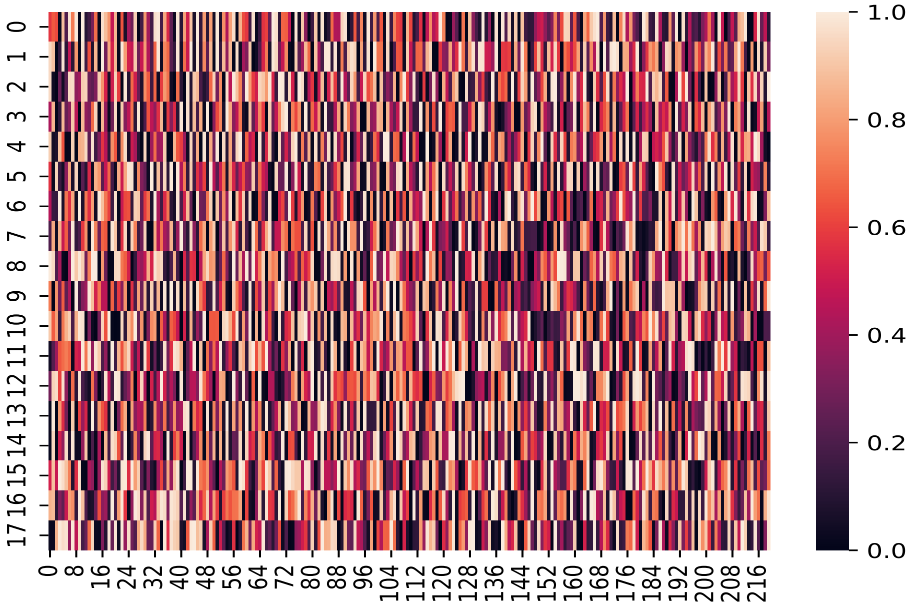


Figure C.5: *Option 4. CIR phase applying sinus function*

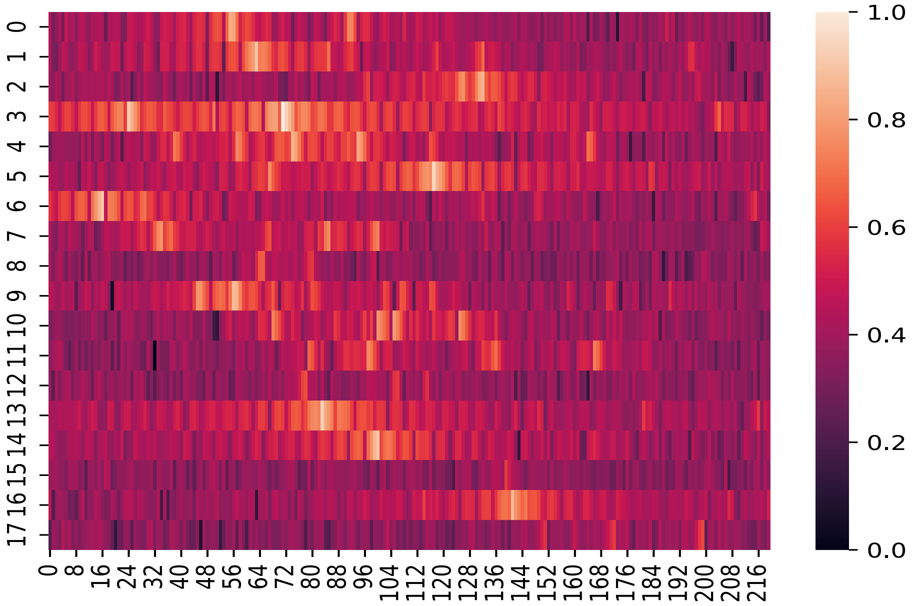


Figure C.6: *Option 5. CIR amplitude in dBs*

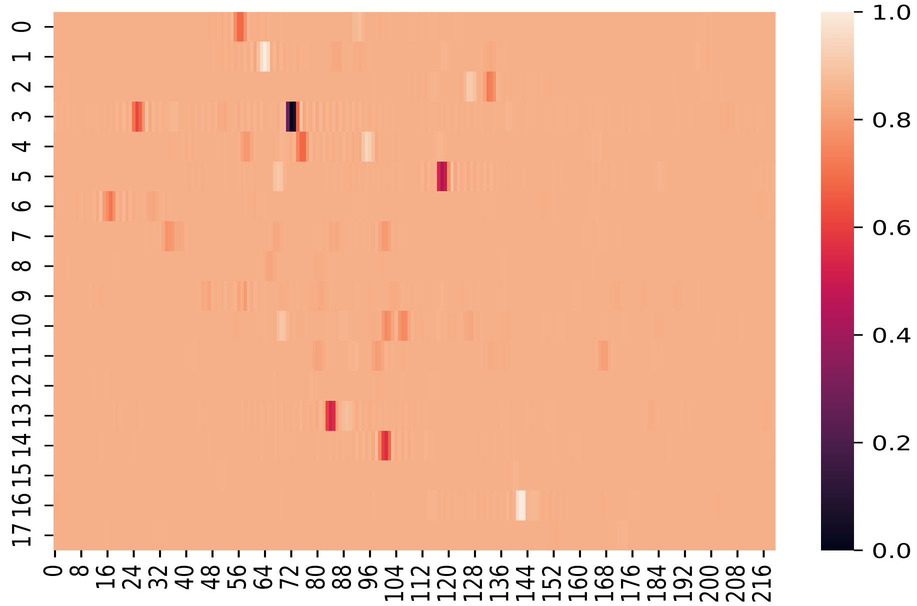


Figure C.7: *Option 6. CIR real part smoothed applying average every 3 samples*

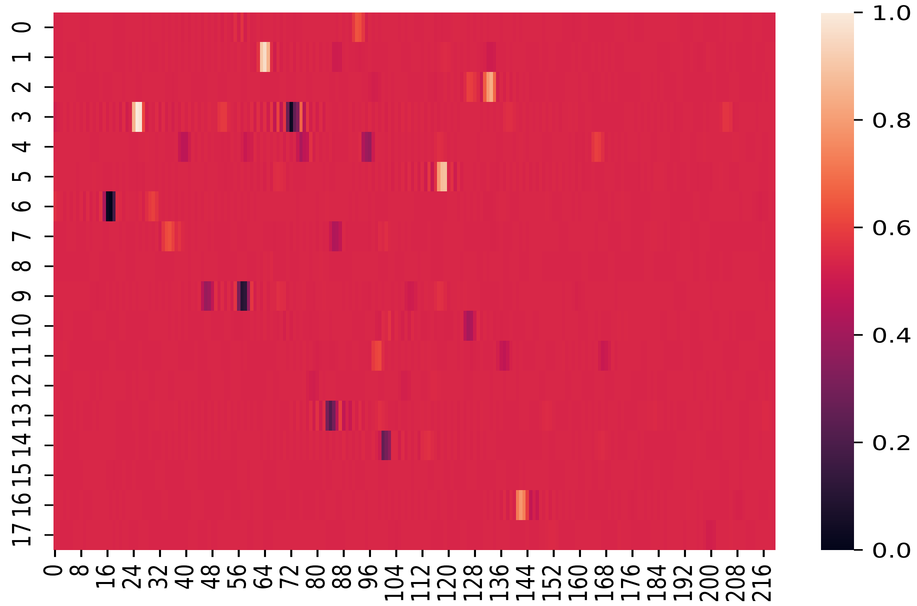
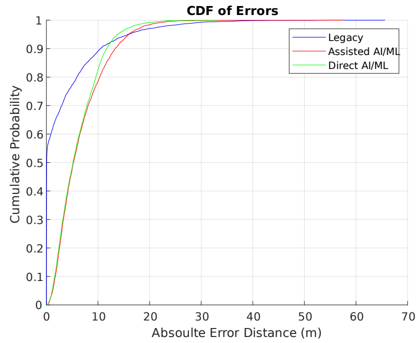


Figure C.8: *Option 6. CIR complex part smoothed applying average every 3 samples*

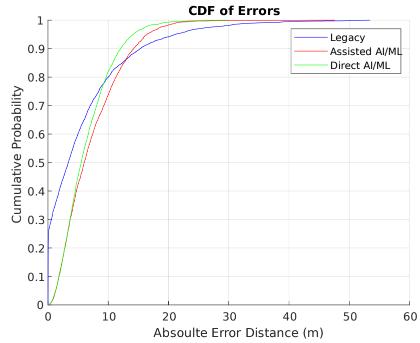
Appendix D

Generalization Scenarios CDFs

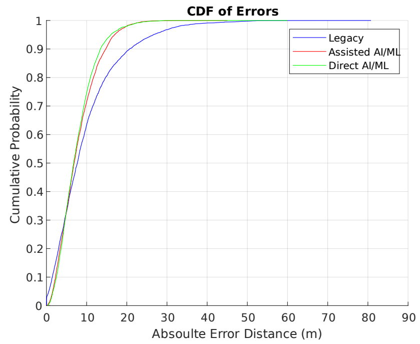
This last appendix presents the absolute positioning error CDF of the UEs for the assisted AI/ML, direct AI/ML, and legacy methods in the different scenarios presented in Chapter 6, Table 6.13.



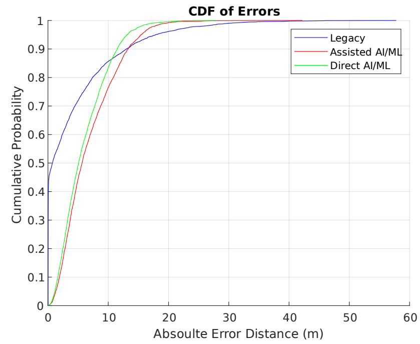
(a) CDFs for Scenario 1



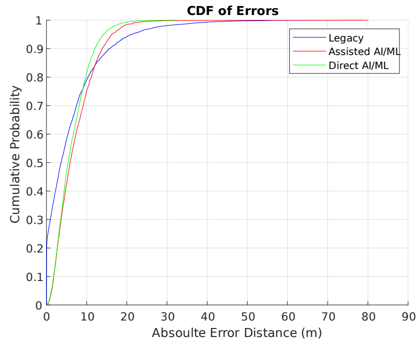
(b) CDFs for Scenario 2



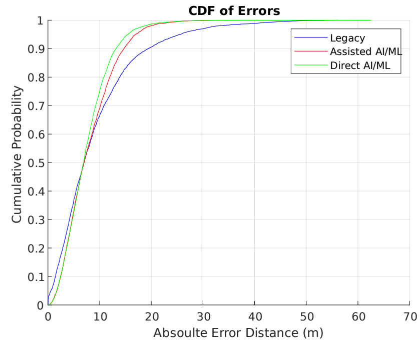
(c) CDFs for Scenario 3



(d) CDFs for Scenario 4



(e) CDFs for Scenario 5



(f) CDFs for Scenario 6

Figure D.1: *Comparison of CDF for AoA +ToA assisted, direct, and AoA + ToA legacy for the different Scenarios*