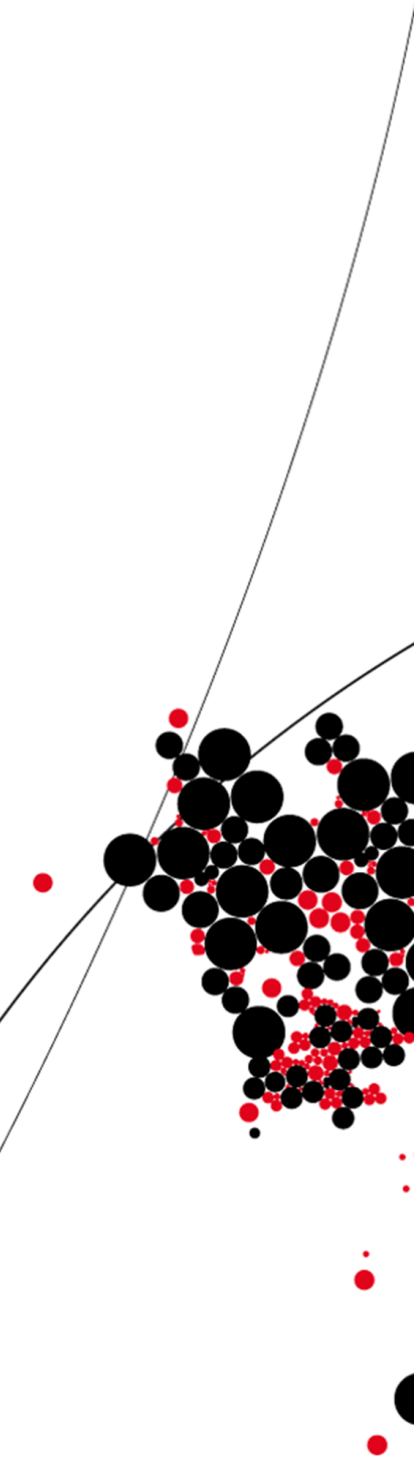
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Direction-of-Arrival Estimation Using A Machine Learning Framework

Frank Hijlkema

M.Sc. Thesis
Maart 2022



Supervisors:
prof. dr. ir. M. J. G. Bekooij
dr. A. Alayón Glazunov
ir. M. L. Lima de Oliveira

Computer Architectures for
Embedded Systems Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

In recent years, radar technology has emerged in various civil applications, and one of them are the automotive vehicles. Modern cars have several radars that are used to improve driver experience for instance in providing parking assistance. These radar sensors are placed at different positions on the vehicle to gather information about object around the car. The echoed signal from the objects are impinging on a Minimum Redundancy Antenna (MRA) which is used to estimate the Direction of Arrivals (DoAs). The DoA is calculated using Maximum Likelihood Estimation (MLE) which is considered as the state of the art algorithm for MRAs. However, the high computational complexity and the imperfections that are present in antenna arrays makes this approach computationally expensive.

This thesis investigates the opportunities of using machine learning for DoA estimation in these automotive radar systems. We have focussed on different aspects of the DoA estimation problem for instance imperfections, combined azimuth and elevation estimation, cost minimization and number of targets estimation. These aspects have been treated as classification problems and were considered only for fully connected neural networks. The DoA estimation results obtained with our neural networks are compared with the results obtained with MLE.

Our results have indicated that machine learning can be an attractive approach for the existing challenges of traditional algorithms. Using this approach, the impact of imperfections is mitigated compared to traditional algorithms. Furthermore, making use of machine learning can decrease the required computational load by a factor of 200 to a modest memory usage of 256 kB. This reduction is obtained through combining a shallow neural network with MLE which makes it attractive to apply neural networks in automotive radar systems.

Contents

List of Figures

List of acronyms

AoA	Angle of Arrival
DNN	Deep Neural Network
DoA	Direction of Arrival
ESPRIT	Estimation of Signal Parameters via Rotational Invariance Technique
FFNN	Feed Forward Neural Network
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MR	Minimum Redundancy
NN	Neural Network
MIMO	Multiple Input Multiple Output
MUSIC	Multiple Signal Classification Algorithm
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
SGD	Stochastic Gradient Descent
SNR	Signal Noise Ratio
SVR	Support Vector Machine

Chapter 1

Introduction

1.1 Context

For a long time, radars were solely used for air defence purposes, such as long range air surveillance and short-range detection of low altitude targets. In the last decade, improvements have been made in radar technology which makes that it is now also used for many civil applications.

In commercial applications, the radar was used for weather observers and air traffic controllers, which contribute to the safety of air traffic in airports. A more recent development in the field of radar technology is the application in the automotive industry. Modern cars already have radars that provide lane steering assistance[1] and support parking assistant. In the coming years, the automotive industry is mainly focussing on the development of semi-self driven cars.

Estimating the Direction of Arrival (DoA) of multiple waves impinging on an array is a well known problem in the field of signal processing. Numerous methods have been proposed to address this problem, such as multiple signal classification (MUSIC)[2], estimation signal parameter via rotation (ESPRIT)[3] and maximum likelihood estimation (MLE). These conventional methods have a high computational cost and have degradation in performance in presence of noise and reverberation. All sources impinging on the antenna array are mapped in space. In essence, the DoAs are estimated by finding the inverse of this mapping. In practice, antenna arrays usually suffer from imperfections in gain, phase, mutual coupling, and antenna positions errors. This will adversely affect the inverse mapping and degrades estimation performance of the algorithms.

To overcome the difficulties mentioned above, a data-driven-based method is introduced to find the direction of arrivals. This method has shown superior results in estimations of the direction of arrival[4] [5]. By considering DoA estimation as a function approximation problem, Artificial Neural Networks (ANN) show the abilities to establish the mapping between a correlation matrix and directions of signal sources. As a result, factors such as imperfections are implicitly taken into account.

1.2 Problem Statement

The company *NXP Semiconductors* has created a 77 GHz single-chip radar 1.1 transceiver which is based on multiple-input multiple-output (MIMO) principle. This chip is planned to be used in self-driving cars. Currently, NXP is working on a DoA estimation algorithm for a sparse antenna array of 8 elements. A Maximum likelihood Estimation algorithm is considered as the state of the art technique that produces adequate estimation results for a single snapshot, because it seems to perform well in unfavourable conditions involving low SNR, high correlated or coherent sources and small array apertures.

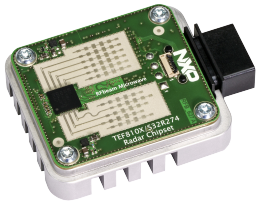


Figure 1.1: Radar chip

A major drawback of this algorithm is the exponential grow in computations that is required to estimate one extra source. By expanding a 1D search grid to an 2D search grid, the number of computations is increased by C^{2k}, where C is the computational cost for one cycle and k is the number of signal sources. Due to the high computational costs, the real-time requirements could not always be met any more.

Another practical problem is the existence of antenna imperfections. Those imperfections occur due to the non-idealities in antenna arrays such as mutual coupling, position, gain- and phase errors. These imperfections could be partly linearly compensated, but this can result in the fact that there does not always exist an inverse mapping.

Taking these factors into account, we are interested in the exploration of the opportunities for applying machine learning in radar systems. Prior research has been shown that the DoA estimation problem can be solved by neural networks. This thesis investigates the feasibility of implementing neural networks in different parts of the DoA estimation problem.

The main research objectives for this thesis are:

1. How to achieve a good estimation performance for small/low-cost NN for azimuth and elevation estimation?

2. What are the effects of antenna imperfections on the estimation performance?

3. Could we develop an efficient NN approach to estimate the number of sources?

4. How can we reduce the computational load and memory requirements of NN used for DoA estimation?

1.3 Related work

Direction of arrival estimation is an important topic in array signal processing. In the early '90s, El Zooghby [10] started to implement a Radial Basis Function Network (RBFN) to estimate DoAs. They found a good neural network performance and have shown the ability to generalize data sets that are derived from different signal conditions. J. Fuchs[11] described a high-resolution direction-of-arrival estimation approach for a single snapshot case and came to the same conclusion that neural networks can achieve a good estimation performance.

A proposition was made [12] for a DoA estimation method based on deep neural networks that could achieve better estimation results than the methods based on support vector machine (SVM). Unfortunately, this method is only applicable for angles in the range of $\theta \in [-60, 60]$. This interval could be enlarged to $\theta \in [-90, 90]$ by using a framework[13] that consists of two neural networks, a Linear classifier network (LCN) and a Convolutional neural network (CNN). The LCN is used to split the angle range of $\theta \in [-90, 90]$ into smaller regions, where each subregion contain one CNN for the estimations of the DoAs. This is an interesting structure due

to the wide range of angles that could be detected. O. Bialer[14] present a model that combine classification and regression to estimate number of sources and DoAs. The classification part estimates the number of sources, which is assumed to be between 1 and 4. Based on the output, there is a set of regression neurons to estimate DoAs. This model is able to achieve 90% accuracy for high SNR values and get a RMSE error of 0.5. Generally, deep neural networks offer a high estimation performance, at the cost of being hard to train. Therefore, L. Wanli [15] has implemented a residual technique that makes it possible to train a very deep neural network in a controllable manner.

Another area of research is the minimization of neural network parameters [16]. One of the minimization methods will be to choose a neural network structure that requires fewer computations, for instance a SqueezeNet[17]. An alternative way to minimize the computations is to use Ternary weights CNN[18]. Ternary weights CNN has three weight values of +1, 0, -1 so they can be represented in 2-bit per weight, which can give a huge reduction in memory usage.

Until now, we have shown that there are different neural network structures used to estimate the DoAs and there are various optimization techniques to minimize the network parameters. The main contribution of our work will be to explore if there is a more efficient neural network structure that optimize performance for minimum computational costs. Furthermore, we will consider the effect of imperfections in antenna arrays and investigate if neural networks are able to deal with such imperfections.

1.4 Thesis outline

The remainder of this thesis is organized as follows. Chapter 2, provides background information about the radar model that we consider and some basic knowledge about neural networks. In chapter 3, we introduce the neural network structure and describe the training strategy. In chapter 4, we will present the different experiments that were conducted and presents for each experiment the estimation performances of the neural network and compare this to the obtained estimation performance for the MLE algorithm. In the last chapter 5, we will draw conclusions based on the results and provide recommendations for future work.

Chapter 2

Background information

This section consist of three parts; in the first part the radar model will be elaborated, in the second part we cover a state of the art algorithm currently in use, and in the third part we will provide background information about neural networks.

2.1 Radar model

In this section, we introduce the in this thesis considered Rx antenna array geometry and present the corresponding signal model.

2.1.1 Array geometry

Consider a M element MRA array with sensor elements located at $\mathbf{p} = [p_1, p_2, \ldots, p_M]^T$ where $p_1, p_2, \ldots p_M$ are integers. The first sensor element in this array is assumed as reference sensor, i.e., $p_1 = 0$. Fig. 2.1 depicts the geometry of the considered MRA consisting of 2 identical 4-element minimum redundancy sub-arrays.

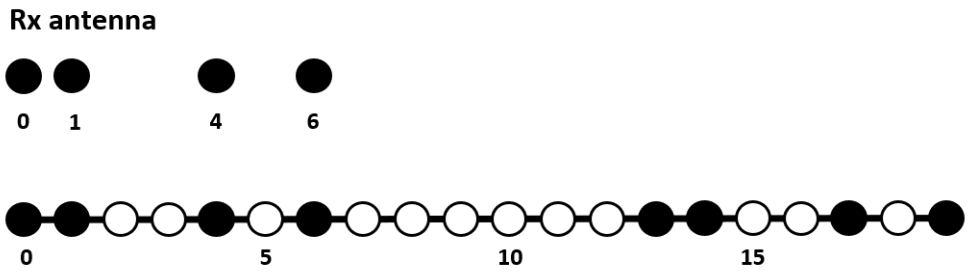


Figure 2.1: Virtual antenna array due to spacing of the Tx antennas

A Uniform Linear Array (ULA) has all antennas spaced at a fixed distance, which is typically $\frac{1}{2}\lambda$. In MRAs, a minimum number of antenna elements is selected from a ULA array. The selection is such that each possible angle can be unambiguously estimated.

2.1.2 Signal model

Consider a MRA array with M identical elements located at positions $\mathbf{p}$. The received signal $\mathbf{x}(t) = [x_1(t), x_2(t), ..., x_M]^T$ is modeled as

$$\mathbf{x}(t) = \mathbf{A}(\theta)\mathbf{s}(t) + \mathbf{n}(t) \tag{2.1}$$

where $\mathbf{x}(t) \in \mathbb{C}^M, \mathbf{A} \in \mathbb{C}^{M \times K}$ is the steering matrix, $\mathbf{s}(t) \in \mathbb{C}^{\mathbb{K}}$ is a vector containing the complex amplitudes of the transmitted signal and $\mathbf{n}(t) \in \mathbb{C}^{\mathbb{M}}$ denotes the Gaussian noise vector. The $M \times K$ matrix is modeled as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}(\theta_1) & \mathbf{a}(\theta_2) & \dots & \mathbf{a}(\theta_k) \end{bmatrix} \tag{2.2}$$

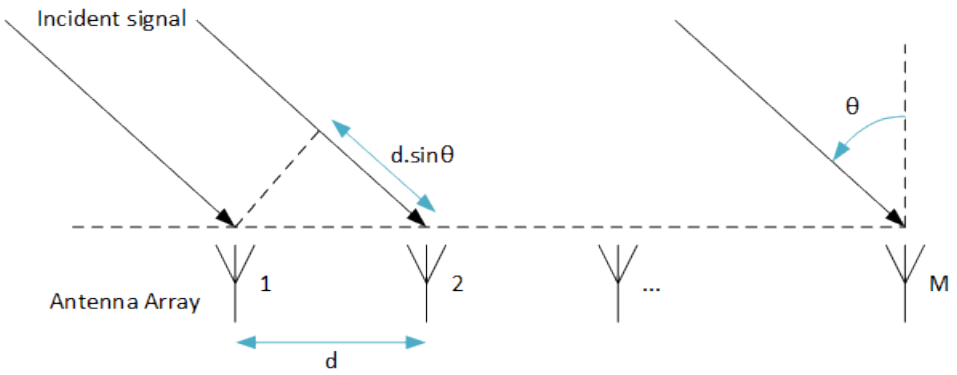


Figure 2.2: Geometric phase difference

The k-th collumn $\mathbf{a}(\theta_k) \in \mathbb{C}^M$ is the steering vector that corresponds to the signal $s_k(t)$ arrived from θ_k direction:

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 & e^{-j\frac{2\pi}{\lambda}p_2 d\sin(\theta)} & \dots & e^{-j\frac{2\pi}{\lambda}p_M d\sin(\theta)} \end{bmatrix}^T \tag{2.3}$$

where p_m denotes the location, λ the wavelength and d the minimum inter-element spacing of $\frac{\lambda}{2}$ for the highest frequency of the impinging signals.

2.1.3 2D signal model

Consider a 2D non-uniform linear array with M identical elements in the x-direction located at position $\mathbf{p}$ as mentioned in section 2.1.1. Now, 2 of these elements are shifted in the z-direction by a distance λ as is illustrated in Fig. 4.6. The received signal $\mathbf{x}(t) = [x_1(t), x_2(t), ..., x_M]^T$ is modelled as

$$\mathbf{x}(t) = \mathbf{A}(\theta, \phi)\mathbf{s}(t) + \mathbf{n}(t) \tag{2.4}$$

where $\mathbf{x}(t) \in \mathbb{C}^M, \mathbf{A} \in \mathbb{C}^{M \times K}$ is the steering matrix, $\mathbf{s}(t) \in \mathbb{C}^{\mathbb{K}}$ is a vector containing the incident signals. $\mathbf{n}(t) \in \mathbb{C}^{\mathbb{M}}$ denotes the Gaussian noise vector. The $M \times K$ matrix is modeled as:

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}(\theta_1, \phi_1) & \mathbf{a}(\theta_2, \phi_2) & \dots & \mathbf{a}(\theta_k, \phi_k) \end{bmatrix} \tag{2.5}$$

where $\mathbf{a}(\theta_k, \phi_k) \in \mathbb{C}^M$ is the steering vector that corresponds to the signal $s_k(t)$ arrived from (θ_k, ϕ_k) direction:

$$\mathbf{a}(\theta) = \begin{bmatrix} e^{-j\pi(z_1\cos(\phi)+p_1(\sin(\theta)\sin(\phi)))} & \dots & e^{-j\pi(z_M\cos(\phi)+p_M(\sin(\theta)\sin(\phi)))} \end{bmatrix}^T \tag{2.6}$$

where $p_M \in \mathbb{N}$ denotes the location of the M^{th} sensor in the x-direction and $z_M \in \mathbb{W}$ denotes the displacement of the sensor elements in the z-direction

2.2 State of the Art DOA estimation

The received signals from the sensor array are used to identify the incoming signal directions. There are various algorithms that are able to find the signal directions which can be classified into three categories: classical-algorithms, subspace-based algorithms and maximum likelihood based algorithms. In this thesis we only consider the most popular subspace based, and maximum likelihood techniques and disregard classical algorithms, since they require a high computational cost and have poor performance, making them impractical for applications

2.2.1 Multiple Signal Classification (MUSIC)

MUSIC[2][19] is a high resolution subspace based algorithm based on eigenvector decomposition. The decomposition is performed for the covariance matrix and results in a signal subspace and noise subspace. Since the signal subspace is spanned by the array steering vector of the received signals, this makes the steering vector orthogonal to the noise subspace. The inner product of the steering vector and the noise subspace is null for the particular Direction of Arrival. According to the received signal vector mentioned in section 2.1.2, we obtain the following covariance matrix:

$$\mathbf{R_x} = E\left\{\mathbf{x}(t)\mathbf{x}^H(t)\right\} = \mathbf{A}\mathbf{R_s}\mathbf{A}^H + \sigma^2\mathbf{I} \tag{2.7}$$

The covariance matrix ($\mathbf{R}_x$) of the received signals K has M eigenvalues $[\lambda_1\lambda_2\ldots\lambda_M]$ and M associated eigenvectors making the subspace $\bar{E} = [\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_M]$. Sorting the eigenvalues from smallest to largest, the subspace $\bar{E}$ can decompose in two subspaces:

$$\bar{E} = \underbrace{[\bar{e}_1, \ldots, \bar{e}_K}_{\bar{E}_M}, \underbrace{\bar{e}_{K+1}, \ldots, \bar{e}_M]}_{\bar{E}_S} \tag{2.8}$$

$$\begin{bmatrix}\bar{E}_M & \bar{E}_S\end{bmatrix} \tag{2.9}$$

where $\bar{E}_M$ is an $M \times (M-K)$ noise subspace spanned by the noise eigenvectors. These eigenvectors in combination with the steering vectors are used to express the MUSIC spectrum function:

$$P_{MUSIC}(\phi) = \frac{1}{\mathbf{a}(\phi)\mathbf{E}_M\mathbf{E}_M^H\mathbf{a}(\phi)} \tag{2.10}$$

The steering vector a(θ) is orthogonal to each column of matrix $\bar{E}_M$ and therefore the dominator of equation 2.10 should be zero for the right angles. Due to the presence of noise, the dominator is a small value. This result in a spectrum that contains sharp peaks. Varying the value of θ, the DoAs can be estimated by finding the location of these peaks.

2.2.2 Estimation of Signal Parameter Rotation Invariance Technique (ESPRIT)

ESPRIT[3] is another subspace based algorithm, based on exploiting the rotation invariance in the signal subspace. This subspace is created by dividing a M element array in two identical sub arrays of size S in a translational invariance structure [20]. Figure 2.3 shows a translational structure of an four element array.

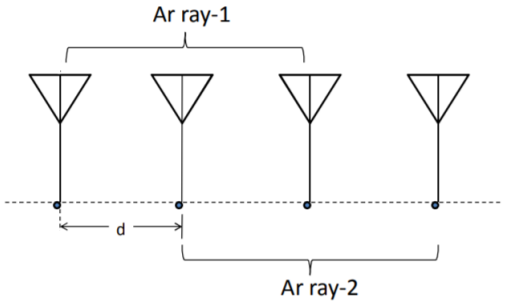


Figure 2.3: Subarray structure

Array-2 is shifted by a distance d from array-1 to create the translational structure. The output of the two sub-arrays are given by

$$\begin{aligned} x_1(t) &= \mathbf{A}s(t) + n_1(t) \\ x_2(t) &= \mathbf{A}\Phi s(t) + n_2(t) \end{aligned} \tag{2.11}$$

where $\Phi \in \mathbb{C}^{K\times K}$ is a diagonal matrix called the rotation operator and K is the number of impinging signals

$$\Phi = diag\{e^{-j\frac{2\pi}{\lambda}d\sin(\theta_1)}, \ldots, e^{-j\frac{2\pi}{\lambda}d\sin(\theta_k)}\} \tag{2.12}$$

Now we can estimate the correlation matrices in the two subarrays as

$$\begin{aligned} \mathbf{R_{x11}} &= E\left\{\mathbf{x_1}(t){\mathbf{x_1}}^H(t)\right\} \\ \mathbf{R_{x22}} &= E\left\{\mathbf{x_2}(t){\mathbf{x_2}}^H(t)\right\} \end{aligned} \tag{2.13}$$

After eigen-decomposing $\mathbf{R}_{11}$ and $\mathbf{R}_{22}$ as is done in equation 2.8, we get the two signal subspaces E_{S1} and E_{S2}. Defining a $2K \times 2K$ matrix C from the two subspaces such that

$$C = \begin{bmatrix} E_{S1}^H \\ E_{S2}^H \end{bmatrix} \times \begin{bmatrix} E_{S1} & E_{S2} \end{bmatrix} = E_C \Lambda E_C^H \tag{2.14}$$

E_c can be obtained by eigen value decomposition of C such that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_K$ and $\Lambda = diag\{\lambda_1\lambda_2 \ldots \lambda_K\}$.

$$E_C = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix} \tag{2.15}$$

The rotation operator is estimates as

$$\Phi = -E_{12}E_{22}^{-1} \tag{2.16}$$

From K eigenvalues of Φ, the angles of arrival can be estimated

$$\theta_i = \sin^{-1}\frac{arg(\lambda_i)}{2\pi d} \tag{2.17}$$

2.2.3 Maximum likelihood (ML)

Maximum Likelihood (ML) is another direction of arrival estimation algorithm that determines DoAs by maximizing the log-likelihood function. The log-likelihood function can be expressed as

$$L = const - KN\ln\hat{\sigma} - \frac{1}{2\hat{\sigma}^2}\sum_{t=1}^{N}\left\|x(t) - As(t)\right\|^2 \tag{2.18}$$

Maximizing the log-likelihood function is equivalent to minimizing

$$\sum_{t=1}^{N} \left\| x(t) - \hat{A}\hat{s}(t) \right\|^2 \tag{2.19}$$

The solution of 2.19 that results in the lowest cost is the orthogonal projection of x(t) on the subspace spanned by the columns of A. Hence:

$$s(t) = (A^H A)^{-1} A^H x(t) \tag{2.20}$$

The two subspace based algorithms can achieve a near optimal performance in case sources are uncorrelated but degrades rapidly when sources are correlated. In practice, sources are correlated most of the time, due to having to deal with multipath effects. Some methods, such as spatial smoothing [21], can help to improve robustness against correlation but have a negative effect on the resolution. Additionally, these algorithms do not perform well in combination with arrays that have a low number of sensor elements. ML is a more general algorithm which does not suffer from the same limitations of the other algorithms, and therefore a good choice to compare results with a NN. The main drawback of MLE is its high computational cost.

2.3 Artificial Neural Networks

Deep learning is a subset of machine learning and based on artificial neural networks. Examples of artificial neural networks are the Boltzmann machine or the multilayer perceptron, which is also the most commonly used neural network. A neural network is a system composed of many simple processing elements (neurons). Each output of a neuron is related to the input and can be thought as a system that receives, process and transmits information. An interconnection of neurons makes that a neural network can behave as a brain.

The historical background of neural networks is that they were developed keeping in mind the way human brains works. The human brain has a biological neural network which has billions of interconnections. As the brain learns, these connections are formed, changed or could be removed. Fig.2.4 shows a structure of a typical brain neuron. The signals are aggregated in the nucleus and when the nucleus exceeds a threshold, the axon passes a signal to the other neurons. This is similar to the perceptron, which outputs a signal depending on the activation function applied on the input. The neuron can have two states, and is either 'activated' or 'inhibited' based on the level of aggregation.

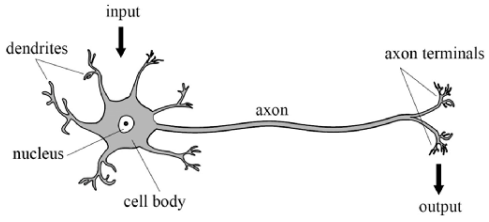


Figure 2.4: Human neuron

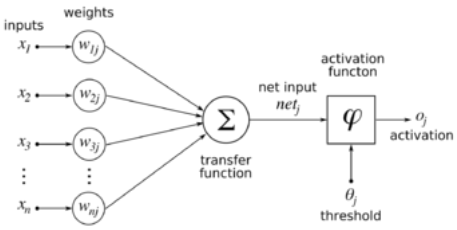


Figure 2.5: An Artificial neuron

A neuron of an artifical neural network is a mathematical processing element with n inputs $(x_1, x_2, ..., x_n)$, one bias and a single output y. The neuron has n weights and a weight w_n will be multiplied with the corresponding input x_n. The most common transfer function is the summation of input values followed by an activation function to improve performance. A

network of interconnected neurons is called a neural network.

$$y = f(\sum_{i=1}^{N} x_i * w_i + b) \tag{2.21}$$

2.3.1 Topology

The basic structure of a neural network consist of an input layer, several hidden layers and one output layer, such a structure is depicted in figure 2.6. Each layer is composed of artificial neurons that are connected to all neurons in the adjacent and forward layer. There are no connections permitted in between the neurons belonging to the same layer. When a neural network has more than 3 hidden layers, it is called a deep neural network.

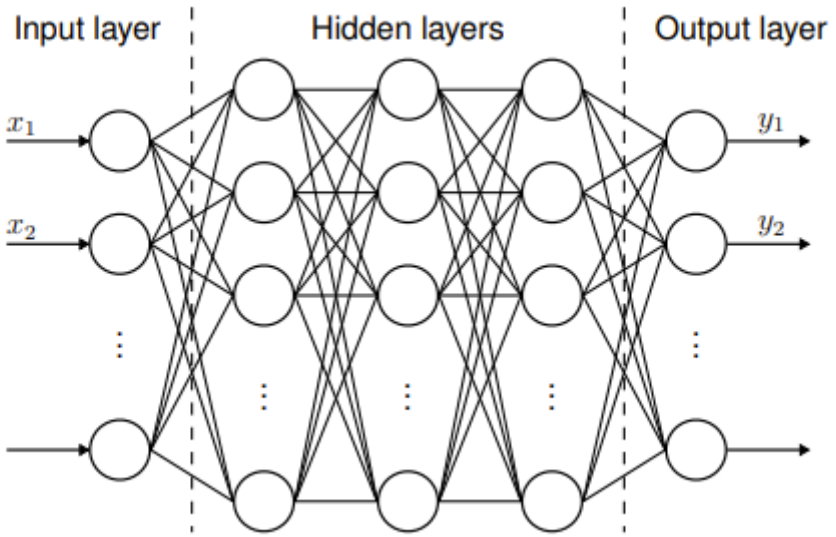


Figure 2.6: Multi-layer structure

In a typical neural network, the information flows from the input to the output without any feedback loop. In more advanced neural networks, artificial neurons can have feedback loops that has the ability to memorize information from previous cycles or they can have sparse connections. The 'state' in such network keep changing until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. A useful application of such networks are prediction problems.

2.3.2 Activation functions

The activation function is an important component of neural networks. A neuron without activation function is linear and cannot form networks that are able to be trained for complex problems. By adding an activation function, the network will be able to solve non-linear (complex) problems. Commonly used activation functions are listed below:

- ReLU: ReLU is an abbreviation for rectifier linear unit and is a widely used activation function in neural networks. It only propagates positive values through the network and sets all negative values to zero, see equation 2.22. An advantage of ReLU over other activation functions is the reduction in amount of active neurons. This reduces the computational time and results in that the training of a neural network is much faster [22].

$$f(x) = max(0, x) \tag{2.22}$$

- Sigmoid: The Sigmoid function is also known as the logistic curve and has the shape of an 'S'. One main property of the sigmoid function is that the output bounds always between [0,1]. The sigmoid function is defined as follows:

$$f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} \tag{2.23}$$

- Softmax: The Softmax is an activation function that is mostly used as the last layer of a multi-class classification problem. It has the property to turns a vector of K real values into a vector of K real values that sum to 1.

$$f(\vec{x}_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{2.24}$$

2.3.3 Learning algorithms

Each model is intended to be used for a various purpose and has to be trained differently. This section covers several learning methods, which can be classified in three different categories:

Supervised learning
Supervised learning uses a training set that consists of pairs of input and desired outputs [23]. The goal of this algorithm is to learn the mapping between the input and the output. Subsequently, the learned mapping has to be used to estimate the outputs given new inputs. If the output takes a finite set of discrete values that indicate the class labels of the input, the learned mapping corresponds to the classification of the input data. If the output takes continuous values, then the neural network performs regression [24].

Unsupervised learning
Unsupervised learning[25] uses a training set that consists of inputs without any assigned desired output. The goal of this algorithm is to cluster input points together that are close to each other, hence assigning a label to each input. Some common tasks that fall under the category of unsupervised learning are clustering or dimensionality reduction.

Reinforcement learning
Reinforcement learning[25] does not have a training dataset but receives feedback from the environment only after selecting an output for a given input or observation. The feedback indicates the degree to which the output, known as action in reinforcement learning, executes the goals of the learner.

2.3.4 Back propagation

The training algorithm that is used to optimize fitting between input and target data is called back propagation. This algorithm optimize a cost function that minimize the sum of squared errors (SSE), where the errors are the differences between the actual output value and the target value. This needs to be done for all trainings samples. To optimize the cost function there are several optimizers and the most popular ones are mentioned below:

Stochastic Gradient Descent (SGD)
SGD is one of the most popular algorithms used in machine learning. This algorithm updates model parameters(θ) in the negative direction of the gradient by taking a subset or a mini-batch of data size(m)

$$g = \frac{1}{m}\nabla_\theta \sum_i L(f(x^{(i)};\theta), y^{(i)}) \tag{2.25}$$

$$\theta = \theta - \eta \cdot g \tag{2.26}$$

where θ is the model parameter, η the learning rate, g the gradient descent and L the loss function. An important property of SGD is that the computation-time per update does not increase with the size of the training dataset.

Adaptive Moment Estimation (Adam)
Adam optimizer combines the advantages of Adaptive Gradient Algorithm (AdaGrad) with momentum and Root Mean Square Propagation (RMSProp). This algorithm computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. Some of Adam's advantages are that the magnitudes of parameters updates are invariant to rescaling of the gradient, its step sizes are approximately bounded by the step size hyperparameter and it works with sparse gradients. The first and second order moment can be calculated through equation 2.27 and 2.28 respectively.

$$m_t = \beta_1 \cdot m_{i-1} + (1 - \beta_1) \cdot \frac{\partial L}{\partial \theta} \tag{2.27}$$

$$v_t = \beta_2 \cdot v_{i-1} + (1 - \beta_2) \cdot (\frac{\partial L}{\partial \theta})^2 \tag{2.28}$$

where β_1 and β_2 are hyper-parameters that are in the range [0,1]. The model parameters can be updated by the following equation:

$$\theta = \theta_{i-1} = -a \cdot \frac{\hat{m}_i}{\sqrt{\hat{v}_i} + \epsilon} \tag{2.29}$$

where $\hat{m}_t = \frac{m_t}{(1-\beta_1^t)}$ and $\hat{v}_t = \frac{v_t}{(1-\beta_2^t)}$.

Root Mean Square Propagation (RMSProp)
This optimizer makes use of a exponentially decaying averaging technique to optimize a model. The use of a decaying moving average allows the algorithm to discard early gradients and focus on observed partial gradients. The learning rate of this optimizer gets adjusted automatically and it chooses a different learning rate for each parameter. The learning rate is divided by the average of the exponential decay of the squared gradients. The model parameters can be updated by equation 2.30

$$\theta_{i+1} = \theta_i - \frac{\eta}{\sqrt{(1-\beta)(\frac{\partial L}{\partial \theta}_{i-1})^2 + \beta \frac{\partial L}{\partial \theta} + \epsilon}} \cdot \frac{\partial L}{\partial \theta} \tag{2.30}$$

where θ is the model parameter, η the learning rate, g the gradient descent and L the loss function.

Training a deep neural network that can generalize well to new data can be a challenging problem. A NN models could have too few layers and therefore not able to learn the problem, whereas a model with too many layers learns too well with the consequences of overfitting. To prevent overfitting and optimize performance, there are so-called regularization techniques that can be applied.

2.3.5 Regularization techniques

1. **Dropout**: Dropout[27] is a regularization technique to prevent overfitting and provides a way of approximately combining exponentially many different neural network architectures efficiently. The term "dropout" refers to dropping out a random number of neurons along with all its incoming and outgoing connections. During training, there will be an exponential number of "thinned" neural networks that share all weights. At test time, the shared weights of all "thinned" neural networks are averaged and scaled down to use in an "unthinned" neural network without dropout.

2. **Batch Normalization**: Batch normalization[28] is a technique that reduce internal covariate shift. Covariate shift happens when the distribution of each layer's input changes constantly resulted in the change of parameters in the previous layer. It accomplishes this via a normalization step to normalize each element of a layer to zero mean and unit variance followed by a re-scaling(γ) and offsetting(β). These steps greatly enhance training speed.

$$\hat{x_i} = \gamma \cdot \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.31}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

3. **Early stopping**: Early stopping is a regularization technique that stops training when the model performance stops improving on the validation data-set. This regularization technique prevents overfitting and saves the best model parameters.

4. **Weight decay**: Weights decay[29] is a regularization technique to penalize large weight by adding a regularization term to the loss function. The $L2$-norm is used specifically as a regularization term to penalize large weights. This prevents overfitting and keeps the weights small.

$$L(\mathbf{w}) = L_0(\mathbf{w}) + \frac{1}{2}\lambda \sum_{i=1} \|w_i\|^2 \tag{2.32}$$

Chapter 3

Our DoA estimation approach

This chapter describes our neural network-based approach for DoA estimation. This problem can be divided into two sub-problems; estimate the direction of arrivals and estimate the number of sources. The first section explains the DoA estimation whereas the second section covers the estimation of the number of sources. Finally, the last section provides the training strategy to achieve optimal performance.

3.1 DoA estimation

Previous studies have shown that neural networks can be used to implement regression or classification formulation to estimate the DoA of a single source[30]. For multiple sources, implementation issues arise for regression because only a predefined number of DoAs can be estimated. Therefore, it was decided to construct a classification framework where each class refers to a discrete angle. For a practical antenna array pattern, the antenna gain for azimuth angles $\phi > 70^{\circ}$ are significantly reduced which makes it hard to estimate these angles. Taking this into consideration, it was decided to consider a field of view of $[-70^{\circ} \leq \phi \leq 70^{\circ}]$. The DoA estimation framework is described in the sections 3.1.1 and 3.1.2

3.1.1 Data preperation

Data preparation is an important preprocessing step before training a neural network. In [31] the effectiveness is shown of a good data preparation scheme to enhance the neural network learning process and reduce the complexity of neural network models. The data preparations steps applied to this framework will be described in this section.

Covariance Matrix

The steering vector represents the set of phase delays experienced by a plane wave as it reaches each element in an array of sensors. The covariance matrix is used to express the phase delays between all elements. Under the assumptions of uncorrelated sources and spatially white Gaussian noise, the covariance matrix $\mathbf{R_x}$ can be expressed as

$$\mathbf{R_x} = E\left\{\mathbf{x}(t)\mathbf{x}^H(t)\right\} = \mathbf{A}\mathbf{R_s}\mathbf{A}^H + \sigma_n^2\mathbf{I_M} \tag{3.1}$$

Where $\mathbf{R_s} \in \mathbb{C}^{K\times K}$ is the signal correlation matrix [32], σ_n^2 is the noise variance and $\mathbf{I_M}$ is an identity matrix of $(M \times M)$. Due to multipath propagation or unfriendly jamming signals, it is hard to obtain the true covariance matrix. Hence, the estimated covariance matrix can be

expressed as follows:

$$\hat{\mathbf{R}}_{\mathbf{x}} = \frac{1}{N} \sum_{k=1}^{N} \mathbf{x}(t)\mathbf{x}^{H}(t) \tag{3.2}$$

The diagonal elements of the covariance matrix represent the source power and the off-diagonal elements represent source correlations. Since the correlation matrix is hermitian, either the upper or lower triangular matrix can be discarded without losing information.

Data Normalization

J. Sola [33] shows that network performance is enhanced as input variable ranges are "equalized" by normalization. If data is not normalized, feature scales can be very dissimilar, the bigger one will have a higher contribution to the output error and as a result, the error reduction algorithm will be focused on the features of higher values, neglecting the information from the small valued features. In our approach, the input data is normalized so that the squared sum of elements needs to be 1.

3.1.2 Model structure

This section describes the model structure and explained how MLE is employed within this neural network.

Input layer

As a neural network does not deal with complex numbers, each complex-valued entity of $\mathbf{R}_{\mathbf{x}}$ is considered as two real values: one for the real part and one for the imaginary part. The input to the NN contains all elements of $\mathbf{R}_{\mathbf{x}}$ belonging to the upper triangular part and can be organized as follows:

$$\mathbf{x} = [\mathrm{Re}(r_{1,1}), \mathrm{Re}(r_{1,2}), \mathrm{Im}(r_{1,2}), \mathrm{Re}(r_{1,3}) \ldots, \mathrm{Re}(r_{2,2}), \mathrm{Re}(r_{2,3}), \mathrm{Im}(r_{2,3}), \ldots, \mathrm{Re}(r_{M,M})]^{T} \tag{3.3}$$

Hidden layer

All hidden layers are fully connected and have a ReLu activation function. This is an often used activation function because it has a simple mathematical formula that results in a lower training time and will be also desirable for future implementations on an embedded system.

Output layer

The neural network has an output resolution of 1° that results in a total amount of 141 classes. The output neurons have a softmax activation function, such that each output class can be interpreted as a probability. The estimated classes are the ones with the highest probability. All the probabilities of the classes are put in a vector $\mathbf{O} = [o_1, o_2, \ldots, o_{141}]^{T}$ which is used in a post-processing step.

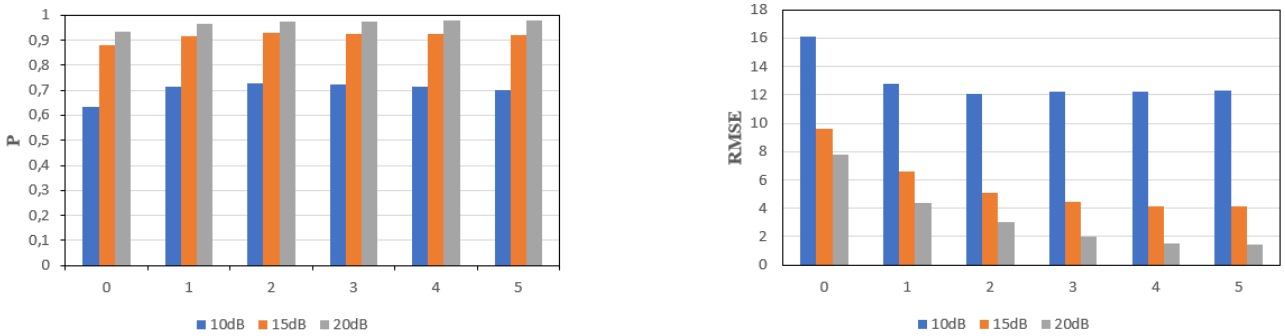
MLE

MLE is an additional part of the NN model and is used to filter outliers in the results of the neural network. This construction was based on the action that NN can have outliers, whereas the MLE algorithm estimates the correct angles for the same snapshot. The correct angle had in this case often not one of the highest probabilities but the fourth or fifth highest probability.

After more detailed analyses it was noticed that the probability differences between a correctly estimated class and a miss-predicted one would often be minimal. With this in mind, MLE is used as a post-processing step to filter these outliers. It uses the M results of the NN that have the highest probability estimations and calculates the least square error 3.4. The combinations of angles that give the lowest error are defined as the most likely combinations of angles.

$$[\Theta] = \arg\min_{\theta} \|\mathbf{x} - \mathbf{As}\|^2 \tag{3.4}$$

The number of sources that are needed to determine the correct DoAs depends on the number of impinging signals. We assume that at most 3 sources are present which is also the maximum number of sources that can be estimated in our experiments. To explore how many sources are needed to estimate the correct signals, we have set up a small experiment where we consider a different number of sources for the same test dataset. Results are presented in Fig. 3.1 where the number of sources is represented on the x-axis and the estimation performance, expressed in accuracy and RMSE, represented on the y-axis. The value 0 on the x-axis represents that we skip MLE where 6 means that we use the 3 output classes that have the highest probabilities and 6 extra candidate angles produced by the neural network which are the $[4^{th}, 5^{th}, \ldots, 9^{th}]$ highest probabilities.



(a) Performance (b) RMSE

Figure 3.1: MLE over different number of sources

It can be observed that the estimation performance does not improve significantly any more if we consider more than 3 additional candidate angles. This results into an input vector of $N+3$ candidate angles where the 3 extra candidate angles are $(N+1)^{th}$, $(N+2)^{th}$ and $(N+3)^{th}$ alternative best estimations. The total number of combinations that have to be checked can be calculated by equation 3.5, where L is the length of the input vector and N is the estimated number of sources. This shows that adding an extra source comes at a cost.

$$Total\ \ number\ \ of\ \ combinations = \frac{L!}{(L-N)!} \tag{3.5}$$

3.2 Number of Source estimation

Another problem related to direction of arrival estimation is the estimation of the number of sources. Rogers [34] presents a deep-learning-based approach of 15 layers that can achieve an accuracy of 82% for estimation the number of sources. This model uses the covariance matrix as input and can estimate up to 5 sources. Burel [35] introduce a completely different approach and use the statistical information of the output. This method has been shown that (when it is possible to apply MLE) the neural network approach provides almost the same precision as MLE with less computation time. We will consider both approaches to explore which one performs better on a minimum redundancy array.

3.2.1 Estimation based on Input data

The neural network topology consists of one input layer, four hidden layers and one output layer. The input and hidden layers are both used in the same way as mentioned in section 3.1.2. The only difference is in the output layer, where we consider 4 output neurons $\mathbf{O} = [0, 1, 2, 3]$. Each neuron represents a number of objects that are detected. This neural network is trained by the same training data-set and settings as mentioned in section 3.3.

3.2.2 Estimation based on Output data

This neural network topology has 2 hidden layers and one output layer. There is no input layer since the network is added to the network for DoA estimation. The hidden layers consist of 141 neurons each where the output has 4 neurons $\mathbf{O} = [0, 1, 2, 3]$. This neural network is trained on the same dataset as the neural network for DoA estimation, but solely for different target values.

3.3 Training of our neural network

As stated in the preliminaries, the objective is that the neural network can estimate DoAs as well as the number of sources. The neural network is trained in a supervised manner, which requires a training and validation dataset. The training dataset has an order of size 6×10^6 samples so that we cover as many combination of angles as possible. The total number of combinations can be calculated by equation 3.5 and will be $\frac{141!}{141-3} = 2,74 \times 10^6$ for one SNR, assuming that all sources lay on the search grid. We consider at least 30 different SNRs and therefore it is not possible to train the NN for all these SNRs. The validation set is 6 times smaller than the training dataset and has a size of approximately 1.000.000 samples.

Hyper parameters

Hyperparameters are parameters that are tuned for machine learning classifiers to improve their performance and prediction accuracy. This optimization process is basically to run different training models where one parameter is changed each time. An alternative approach could be an automated parameter optimization algorithm such as grid search. The drawback of this non-deterministic method is that results may differ every time you run the training algorithm on the same dataset. Furthermore, these are time-intensive tasks and therefore, we have applied default parameter settings that are used in all training runs.

Hyper-parameters	**Value**
Optimizer	Adam
Momentum	$\beta_1 = 0.9$, $\beta_2 = 0.999$
Learning Rate	0.001
Batch size	128
Max Epochs	30

Post-training

Post-training is a training phase applied after a neural network is fully trained. This phase is intended to optimize the model structure and reduce memory usage of the neural network as well. There are several optimization techniques[36] which can be applied such as *weight pruning* or *quantization*. These methods are typically applied at different design stages in sequential order, but Tung[36] introduced a compression method that combines weight pruning and quantization in a single learning framework that performs pruning and quantization jointly, and in parallel with fine-tuning.

In our models we will consider a weight quantization method. This has the potential to move from floating-point representation to low-precision fixed point integer values represented in 8 or 16-bit. Furthermore, it has multiple other benefits over the full-precision counterpart in terms of latency, power consumption and area efficiency, according to Gholami[37].

Chapter 4

Experimental results

In this chapter, we will elaborate on several experiments and evaluate the performance of deep neural networks for different aspects of the DoA estimation problem. The results of these models are compared to the results obtained with the MLE algorithm, which is considered as the state-of-the-art DoA estimation algorithm for MRA arrays. The results are composed of two types of test data sets. In one dataset, we only consider angles of the sources that correspond with the search grid of the neural network. In the other data set, we consider DoAs that do not necessarily lie on a grid and are taken from a uniform distribution.

The model structure, which is used in the experiments, is shown in Fig. 4.1 and requires 55M multiply-accumulate(MAC) operations. This NN model with many parameters is chosen such that it has enough degrees of freedom to approximate the trainings set for the DoA estimation problem. In later experiments, optimizations are evaluated that make NN models more suitable for usage in embedded applications. The covariance matrix ($\hat{\mathbf{R}}_{\mathbf{x}}$) is used as input, as explained in section 3.1.2, and each output represents an DoA in the range $[-70^{\circ}, 70^{\circ}]$.

As mentioned in section 3.2 we use a dataset of 6×10^6 data samples containing one, two or three sources. The results presented in this section were obtained from 1 snapshot, estimated on a 1° resolution and compared to the-state-of-the art algorithm MLE. This algorithm obtains also results for a search grid of 141 points spaced at 1° so that we represent a discrete range of [−70°, 70°]. Furthermore, all sources are of equal power and have a minimum distance of $\Delta\theta = 3°$. This minimum distance is used otherwise the MLE cannot resolve them as 2 sources at a low SNR.

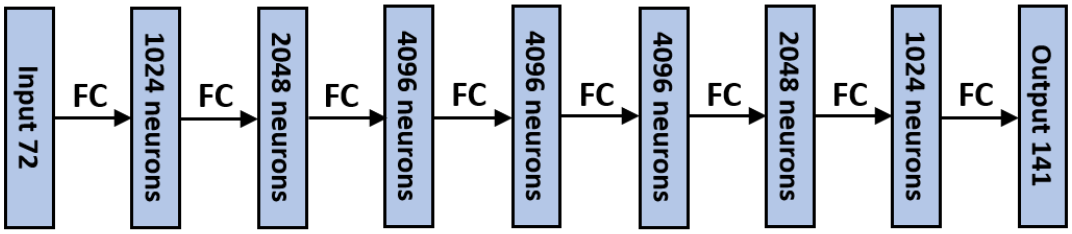
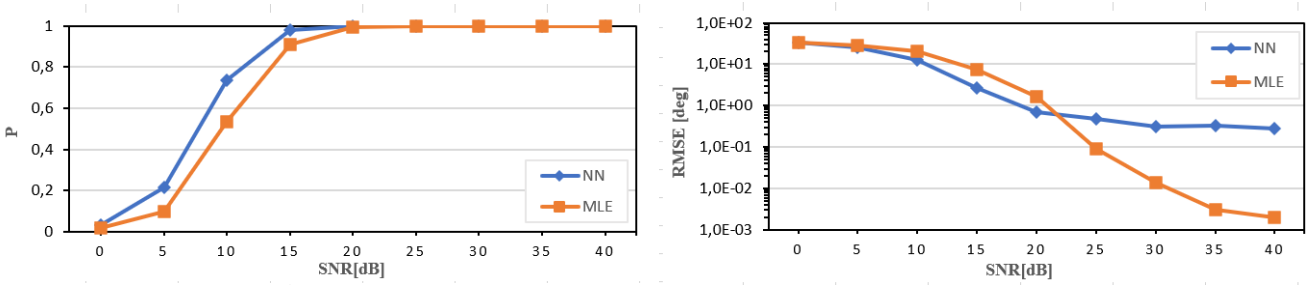


Figure 4.1: Neural network structure

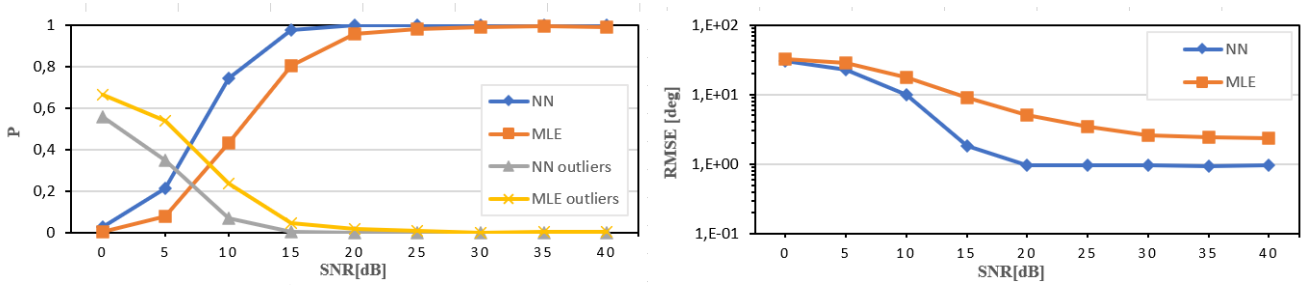
4.1 1D (Azimuth) DoA estimation

The main purpose of this experiment is to explore if neural networks are able to estimate up to 3 sources correctly. This is evaluated by a test dataset that contains 10^5 observations for each SNR. Most neural network evaluations are based on the RMSE value because this gives a good indication of whether a network performs well. The definition of RMSE is such that it can only be computed when the estimated sources $\hat{k}$ is equal to the labeled number of sources. Therefore, we have decided to create a test dataset that contains only 3 sources. Also, the RMSE value can result in misleading interpretations of the prediction with a large difference between estimated DoA and the true DoA. Therefore, there is an extra metric introduced called outliers. This metric indicates the percentage of how many outliers there are. An estimation is considered as an outlier if the difference between the true target value and estimated value exceeds 10°.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.2: Estimation performance of 3 sources, grid resolution of $\Delta\theta = 1^{\circ}$



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.3: Estimation performance of 3 sources, uniformly distributed DoAs

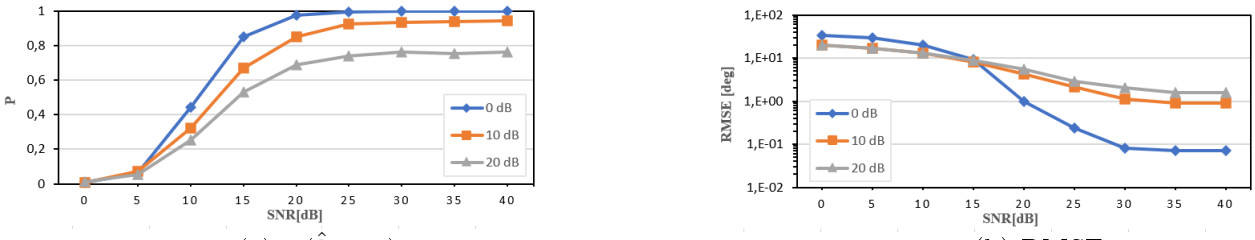
As can be observed from Fig. 4.2a, the performance of both algorithms increases with the SNR, until it stabilizes at 100% for SNRs of 20dB and higher. Both figures show that the NN outperforms MLE in the SNRs lower than 20dB, whereas MLE estimate more accurately for the SNRs higher than 20dB. The most obvious reason for this behavior will be the spacing in the used search gird which is 1 degree. Now, MLE can regenerate the exact same snapshot as is given. This also explains the RMSE value for MLE which converges towards zero for higher SNRs. Contrary to MLE, the NN does not converge to zero and has a minimum RMSE of 0.3° for higher SNRs. It seems to be likely that this occurs due to the statistical properties of neural networks that can result in small miss-predictions of 1°. This assumptions are confirmed in Fig. 4.2a where we have achieved an estimation performance of 100% what indicates that all estimations are predicted correctly for a resolution of 1°.

A detailed analysis of the estimations has given a good insight into how miss-predictions can occur. The probabilities of the output classes have shown tiny Gaussian distributions over the correct estimated class. In the case when one class is more pronounced than another one, i.e. a

powerful and weak source, the more pronounced DoA results in a wider Gaussian distribution which negatively affects the output estimation algorithm. An example for this could be the case if output i is recognized as a DoA, the algorithm neglected the output classes $i-1$ and $i+1$. However, the outputs $i-2$ and $i+2$ could have been higher classes compared to other outputs in the distribution, resulting in a higher likelihood of miss-predictions.

The RMSE value of Fig. 4.3b gives a better indication of how both algorithms perform in a radar system. For SNRs 15dB and lower, MLE has degraded performance by 10% and contains 20% more outliers than the NN have. This is mainly caused by the fact that the search grid of MLE is $\Delta\theta = 1^{\circ}$ while all DoAs are spread uniformly. We have deliberately chosen this search grid to create a fair comparison in estimation performance between the NN and MLE. A better search grid should be a uniform grid so that MLE is able to regenerate a more similar snapshot compared to the given one. However, the NN has achieved equivalent performance, compared to the NN model which was trained for a grid resolution of $\Delta\theta = 1^{\circ}$, and has an accuracy of 99% for SNRs 15dB or higher. These are promising results and indicate that if a neural network has enough degrees of freedom, it will be able to estimate DoAs well for a minimum output resolution of 1°.

Returning to the estimation problems of weakened sources, Fig. 4.4 shows the DoA estimation performances considered non-equal powered sources. These results were obtained from a grid resolution of $\Delta\theta = 1^{\circ}$ and have shown that NNs are able to estimate 92% correctly for SNRs 25dB and higher. Analyzing the estimation behavior in more detail revealed that 2 of the 3 sources are always correctly estimated for SNRs 25dB and higher. This is related to the estimation behavior of NNs that have snapshots with pronounced sources, which dominates the probability distribution with respect to weakened sources. This behavior is intensified when the power differences are increased to 20dB. The maximum estimation performance for these NNs is reduced to 76% for SNRs of 25dB and higher. Now, the NN is not able anymore to estimate always 2 sources correctly and have a maximum accuracy of 96% for estimating 2 DoAs. Overall, the estimation performance for 2 DoAs is 20% higher than the one for 3 DoAs in all SNRs. From these results it can be concluded that a NN can estimate DoAs well for sources that have a small power difference, but the estimation performance drastically degrades when the power difference becomes higher than 10dB. This is not desirable in radar systems and should be investigated more in-depth if there are optimizations that can improve estimation performance for power differences of 10dB and higher.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.4: Estimation performance of 3 sources, max power variations of 10dB, 15dB and 20dB

4.2 2D DoA estimation

In the previous experiment, Azimuth estimation was considered which has shown excellent performance for estimating 3 DoAs. These were promising results, but in practise signal sources are located in 2D space so we have to investigate if machine learning is able to estimate azimuth and elevation as well. In this experiment, the existing 1D MRA as mentioned in 2.1.1 is considered as a 2D antenna array where the antenna elements at position 2 and 6 are now shifted in the z-direction, see Fig. 4.5.

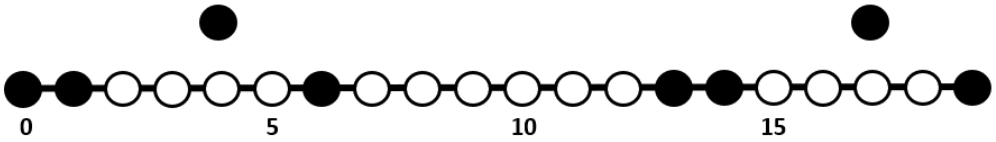


Figure 4.5: 2D Antenna array

The 2D DoA estimation problem has massively increased the complexity through the exponential rise of classifications. The number of output classifications can be defined as $(azimuth \times elevation)^k$, where azimuth and elevation are the values of the field of view that is considered for each estimation direction and k is equivalent to the number of sources. The field of view were predefined for azimuth $-70^\circ \leq \phi \leq 70^\circ$ and elevation $78^\circ \leq \theta \leq 90^\circ$. The geometry of all parameters is visualized in Fig. 4.6 and illustrates that an elevation angle of 90° corresponds to a horizontal view perpendicular to the x-y plane.

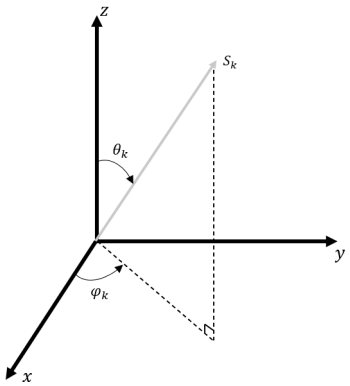


Figure 4.6: Geometry definitions

The total number of classes is equal to $(141 \times 13)^3 = 6 \times 10^9$, assuming that we consider for each combination a class. This can not be resolved anymore as a classification problem and therefore I did an experiment where I have split the 2D DoA estimation problem into two 1D estimation problems so that we estimate azimuth and elevation separately. We consider two similar neural networks that have a network structure as mentioned in Fig. 4.1. Both NNs use the $\hat{\mathbf{R}}_x$ as input, as was explained in section 3.1.2, and have 141/13 output classes. A model overview is visualized in Fig. 4.7 and shows two NNs with a post-processing step MLE. The main objective of the post-processing is to couple azimuth and elevation angles with the help of MLE to give information in the 2D space. Both neural networks perform estimations where the 5 highest outputs are passed to MLE. Subsequently, MLE regenerates snapshots for all possible combinations of azimuth and elevation angles and selects the one with the smallest difference from the received snapshot. This snapshot contains the best combinations of azimuth and elevation DoAs, from which we obtain information about the 2D location for two separate estimation networks.

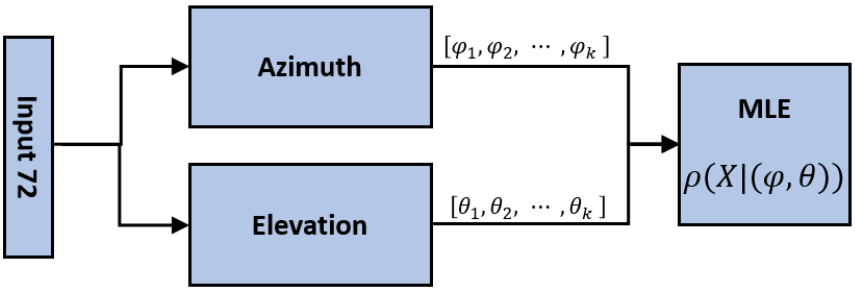
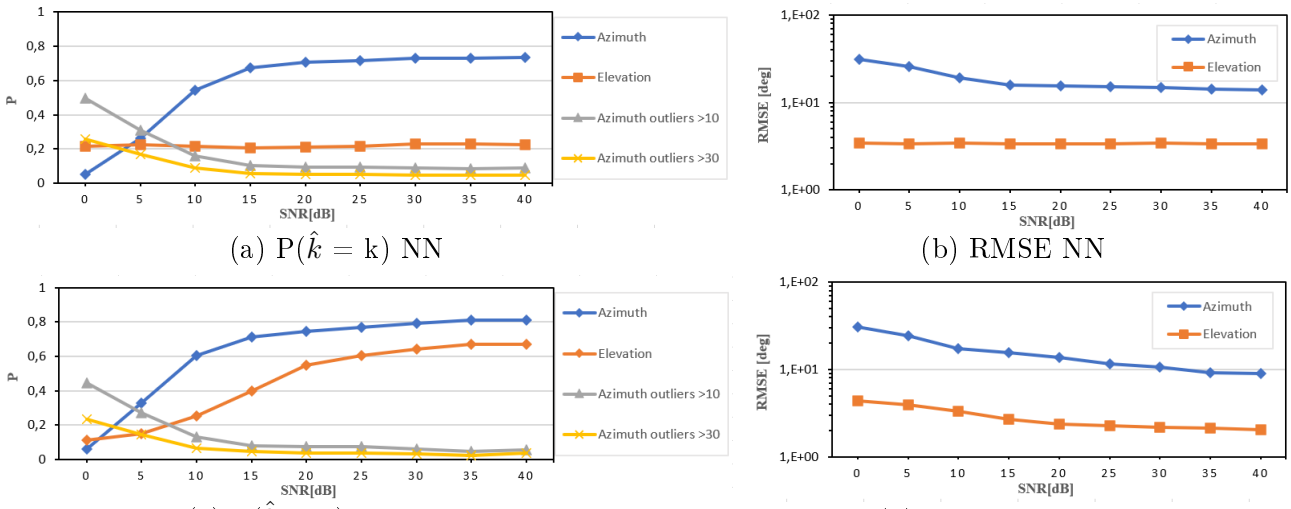


Figure 4.7: 2D neural network structure

The results are depicted in Fig. 4.8. First, we have considered the estimation performance of 3 DoAs for each NN. Fig. 4.8a and Fig. 4.8b has been shown that the azimuth estimation performance is stabilized at 73% for SNRs of 20dB and higher. This is lower than expected because in experiment 4.1 was concluded that we could achieve an accuracy of 99% for SNRs of 20dB and higher. If we have a look at the RMSE value for this NN, we observed that this value will not become below 14° for all SNRs. This is extremely high and implies that a NN is not able to estimate 3 azimuth DoAs correctly anymore. This high RMSE can be related to the high percentage of extreme outliers. In total 5% of all DoAs have an estimation error of at least 30° which means that 15% of the snapshots cannot be resolved. When the estimations were analyzed in more detail it was noticed that for SNRs higher than 20dB, two of the three DoAs are always correctly estimated. This confirms that 5% of the DoAs, which are extreme outliers, are responsible for this high RMSE.

Conversely, Elevation has a maximum estimation performance of 21% for an average RMSE of 3°. The actual elevation accuracy will be a bit higher because we have assumed that there are always three DoAs present. As a result, the NN will always estimate three DoAs. The NN is not able to check for multiple angles on the same elevation, while the snapshots do contain such angles. This means that there were more miss-predictions observed than there are. Despite this assumption, it can be concluded that a NN is not able to estimate three elevation DoAs.



(a) $P(\hat{k} = k)$ NN

(b) RMSE NN

(c) $P(\hat{k} = k)$ NN+MLE

(d) RMSE NN+MLE

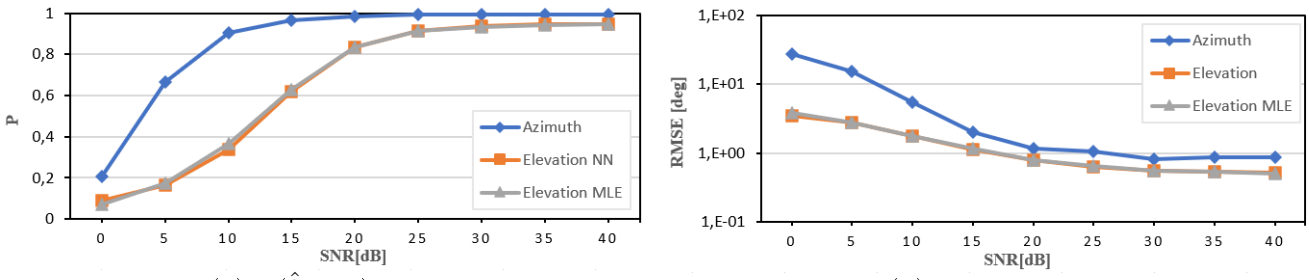
Figure 4.8: Estimation performance 2D antenna array structure, 3 sources and a grid resolution of $\Delta\theta = 1°$

The same experiment is also applied for the NN in combination with the post-processing

step. Now, the post-processing step is used to determine the couples and elevation as well. We have replaced the NN, used for elevation, with a pre-defined vector that contains all elevations and let MLE determine the snapshot which has the smallest differences with the received snapshot. By doing it this way, we could determine DoAs which have the same elevation.

The estimation performances are shown in Fig. 4.8c and Fig. 4.8d respectively. It can be observed that the elevation performance is improved to a maximum of 67%. This is a significant improvement of 46% and has been only achieved through replacing the NN with a predefined vector and letting MLE determine the elevations. Also, the azimuth estimation performance has been improved by 7% and stabilizes at 80% for SNRs 20dB and higher. This improvement could be related to MLE by utilizing all elevations. In other words, azimuth- and elevation estimations are no longer independent problems. This estimation approach could only be implemented when there are a low number of angle options because the computational time dramatically increases when considering one extra elevation angle. By analyzing the RMSE values in Fig. 4.8d, it can be observed that they are improved to a minimum of 9° for azimuth and 2° for elevation. This is still high and not acceptable for implementation in radar systems.

As already was noticed, a NN is able to estimate 2 of 3 azimuth DoAs correctly. Therefore, it would be interesting to explore if a NN is able to achieve good results for estimating only 2 DoAs. We use the same structure as illustrated in Fig. 4.7 and thus estimate azimuth and elevation separately. The results of these NNs are depicted in Fig. 4.9. Now, the NN azimuth estimation performance has achieved an accuracy of 99% and an RMSE of 1° for SNRs 20dB and higher. This was expected based on the observations in the same experiment for 3 sources. Furthermore, the NN elevation estimation performance is improved by 25%, compared to the best estimation approach in the previous experiment, since we only estimate 2 DoAs as opposed to 3. The maximum estimation performance of this NN is 95% correctly for SNRs of 30dB and higher. This improvement in estimation performance has also led to a smaller RMSE which is dropped below 1° for SNRs of 20dB and higher. We also have considered a similar experiment with MLE as was done for 3 sources where we have replaced the elevation NN with a predefined vector that contains all elevations. These results are presented by the metric *Elevation MLE* and show similar performance. In other words, it does not matter which algorithm to use so you have to pick the most efficient one. Currently, both NNs are computationally intensive, but for future experiments, it would preferable to explore the use of smaller neural networks.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.9: Estimation performance 2D antenna array structure, 2 sources and a grid resolution $\Delta\theta = 1^{\circ}$

So far, we only have considered the estimation performance of both directions separately, but we have to estimate the correct coordinate. From previously observed results has been concluded that a NN is not able to estimate 3 DoAs correct and therefore we only obtain results for 2 coordinates. Now, we apply an extra condition to check if the combination of azimuth and elevation is correct. The results are presented in Fig. 4.10. It can be observed that we can achieve at least an estimation performance of 80% for SNRs of 25dB and higher. This is roughly 10% lower than the estimation performance of elevation and thus means that

10% of the estimations are wrongly coupled. This percentage is increased while considering lower SNR values because MLE is sensitive to noisy data.

The overall conclusion could be that NNs in combination with MLE have shown that they can estimate the azimuth and elevation of 2 sources, of which at least 80% are correctly for SNRs 20dB and higher. This is a potentially good result but might not be good enough for implementation in radar systems.

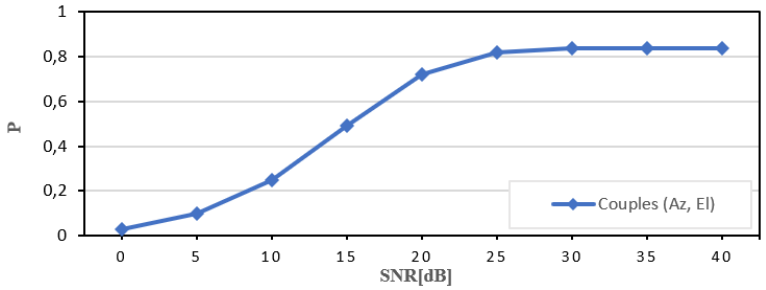


Figure 4.10: Estimation performance of 2D DoAs, 2 sources and a grid resolution $\Delta\theta = 1^{\circ}$

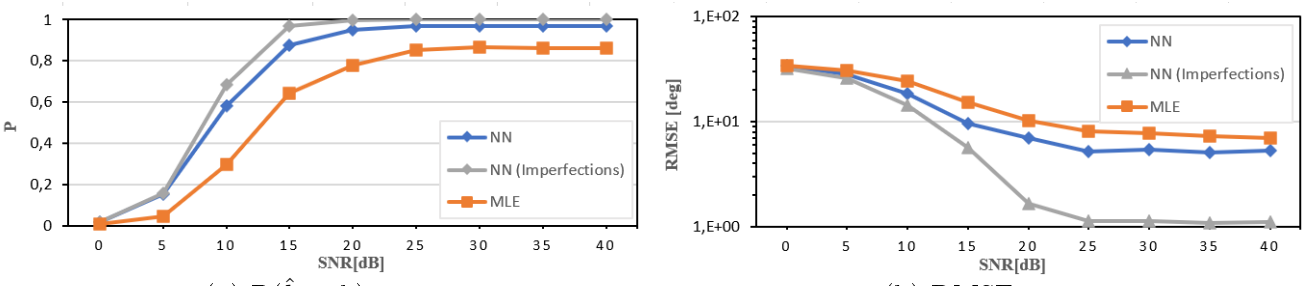
4.3 Antenna mitigation

In this experiment, we will quantify the performance degradation due to the effect of imperfections in the antenna array. Some of these imperfections are mutual coupling, antenna positions errors, phase- and gain inconsistencies. These imperfections cause deviations in the antenna array response, resulting in significant performance degradation of DoA estimation algorithms such as MLE. These imperfections can be added to a predefined model in section 2.1.2 and can be formulated as:

$$\mathbf{x}(t) = \sum_{k=1}^{K} \mathbf{a}(\theta_k, \mathbf{e})\mathbf{s}(t) + \mathbf{n}(t) = \mathbf{A}(\theta, \mathbf{e})\mathbf{s}(t) + \mathbf{n}(t) \tag{4.1}$$

where $\mathbf{a}(\theta_1, \mathbf{e})$ is the array response included imperfections($\mathbf{e}$). In this thesis we only consider gain and phase errors which can be linearly added as shown in equation 4.2:

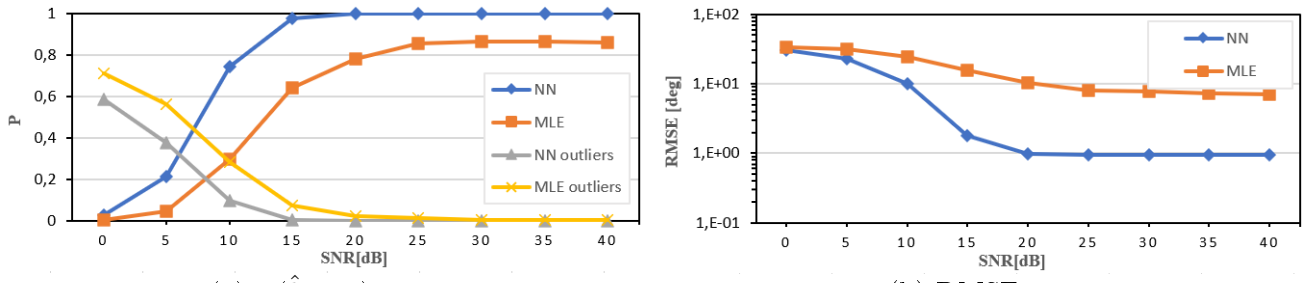
$$\mathbf{a}(\theta_k, \mathbf{e}) = diag(exp(ja_p\mathbf{e}_p)) \times (\mathbf{I}_M + diag(a_g\mathbf{e}_g)) \tag{4.2}$$



(a) P($\hat{k}$ = k) (b) RMSE

Figure 4.11: Estimation performance given antenna imperfections, 3 sources and a grid resolution of $\Delta\theta = 1^{\circ}$

For this experiment, we assume that the number of impinging signals is equal to $k = 3$. The results are obtained for an antenna array with imperfections and are presented in Fig. 4.11. We observe 3 different metrics; a trained neural network from a previous experiment (NN), a new NN trained for an antenna array with imperfections and MLE. Initially, observations



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.12: Estimation performance given antenna imperfections, 3 sources and uniformly distributed DoAs

confirm that imperfections cause drastic performance degradation for MLE. This was expected, since, as a result of an antenna array response with imperfections, an inverse mapping does not even always exist. In the case that multiple combinations of DoAs result in an identical inverse mapping, MLE is not able to differentiate between two snapshots. The maximum accuracy that has been achieved is 86% for SNRs 25dB and higher. Contrary to MLE, the NNs have shown that they do not degrade in estimation performance compared to the NNs of experiment 1 and achieve an accuracy of 99% for SNRs 25dB and higher. One explanation for this performance gap that has arisen through imperfection will be that the used MLE is not aware of these imperfections and the maximum likelihood one does not correspond anymore to the true DoAs.

The RMSE values of the metrics are depicted in Fig. 4.11b and give a considerably worse perception of the performances compared to Fig. 4.11a. Now, we observe that *NN* and *MLE* have high RMSE values for all SNRs. In particular, the NN of the previous experiment 4.1 which has achieved an accuracy of 97% has an average RMSE of 5° for SNRs 25dB and higher contrary to 1° for *NN(Imperfection)* which is trained on data with imperfections. This should indicate that *NN* has a few extreme outliers because the total amount of outliers is lower than 1%. The RMSE value could be reduced by using MLE, as mentioned in section 3.1.2, to 2° which implies that MLE is an appropriate method to filter outliers.

However, The NN which is trained for data with imperfections has an RMSE of 1°. When the output was analyzed in more detail it was observed that there were no outliers, but many estimations errors of 1°. These errors are not marked as miss-estimation and therefore the metric *NN(Imperfection)* has achieved an high estimation performance. This estimation performance is similar to that achieved in experiment 4.1, where we consider the same antenna array without imperfections. It has been only degraded in RMSE to 1° contrary to 0.3° for an antenna array without imperfections.

To explore the more practical case, we consider a grid that contains uniformly distributed DoAs. In Fig. 4.12, we compare a NN trained on data with imperfections to MLE. The performance and RMSE curves look very similar to the one achieved in the previous experiment where we consider DoAs which have a minimum grid resolution of $\Delta\theta = 1^{\circ}$. A possible explanation for this is that the dataset consisted of more angles close to the boundaries of the grid, which is most unfortunate. Another, less likely explanation would be that during training the NN is converged to a more optimal model. This trained NN on data with imperfections has achieved similar estimation performance as well as RMSE value to the one in experiment 4.1. Both NN achieves an estimation accuracy of 99% and an RMSE value of 1° for SNRs of 20dB and higher. Therefore, it can be concluded that NNs, which are trained for antenna arrays with imperfections, can estimate 3 sources correctly without degradation in estimation performance. These are promising results and overcome the difficulty of imperfections in MLE. However, This NN implementation requires a huge amount of MAC operations. In the next section, we investigate in more depth which optimizations can be applied on a NN.

4.4 Shallow NN

Experiment 4.3 have shown that NNs are able to learn the mapping between input and output to achieve an accuracy of 99% for SNRs of 20dB and higher. However, the bottleneck of these NNs is the high amount of MAC operations that are required to estimate 3 signal sources. This experiment investigates where optimizations can be applied to reduce the number of computations. This will be accomplished through gathering previously found observations which will be merged to compose an optimal model. The optimizations have to be made on the input- and hidden layers since the output should have the same amount of classifications. Currently, the upper triangular part of $\hat{\mathbf{R}}_x$, included the diagonals, is used as input for the NNs. To reduce the number of computations on the input layer, it will be explored if the DoA estimation problem can be solved to only consider the off-diagonals. This should result in a reduction of 16 neurons to a total of 56. The major improvement has to be made in hidden layers i.e. minimize the number of hidden layers and reduce the number of neurons per layer. This will be explored in the next part of this section.

Experiment 4.1 has been shown that a NN outperforms MLE for all SNRs lower than 20dB where the opposite is true for SNR higher than 20dB. A possible explanation for this could be that a neural network provides a statistical probability for each output, where MLE generates a calculated optimal value for a combination of angles. In lower SNR regions, the influence of noise is greater, relative to high SNR regions. For this reasons, MLE offers a lower accuracy in lower SNR values. Here, a NN approach is beneficial, as the statistical probability of achieving the correct angle gives a higher likelihood of achieving the correct result. In higher SNR regions, MLE has a great accuracy of calculation, which would outperform an approach based on statistical probability, such as a NN.

A possible explanation for this could be that a NN provides a statistical probability for each output that is calculated for an average SNR value, whereas MLE calculates an optimal value based on a combination of angles for a specific SNR value. In lower SNR regions, the influence of noise is greater, relative to high SNR regions. For this reason, MLE is more influenced by the low SNRs and offers a lower accuracy as result. In the higher regions, MLE achieves high accuracy and has no longer influence of noise, whereas an NN provides estimations for the same weights as were used for the lower SNR values. A recommendation for the future would be to investigate if a NN, which is only trained for high SNRs, can have a lower RMSE value than when an NN is trained for all SNRs. If that would be the case, it confirms the assumption above and gives a better understanding of the difference in estimation behavior.

To use the advances of both algorithms, we have combined the NN with MLE as mentioned in section 3.1.2. Now, the goal will be to minimise network size without loosing performance. This has resulted in a final model of 3 hidden layers of 160, 360, 160 neurons respectively. Fig. 4.13 shows the NN model structure which is used in this experiment.

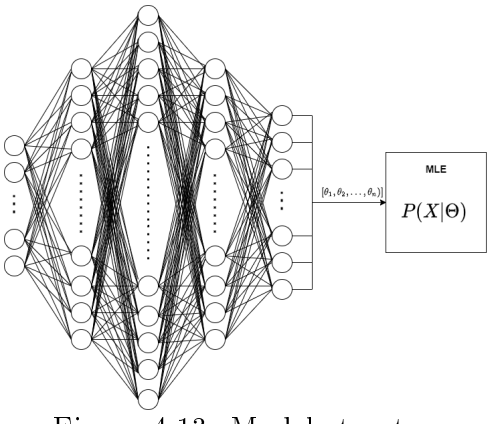
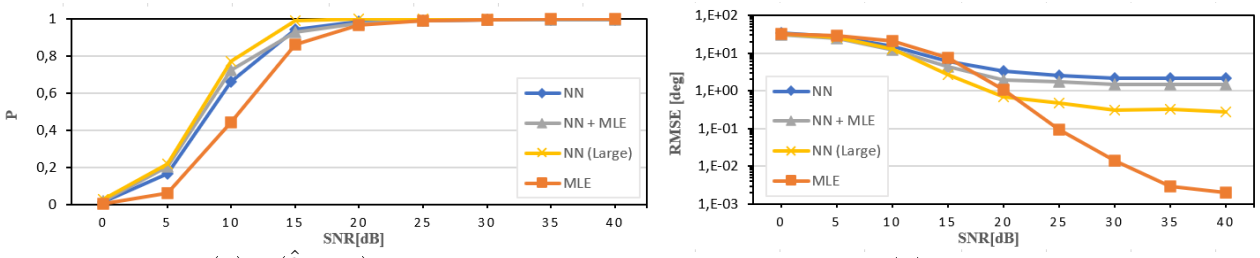


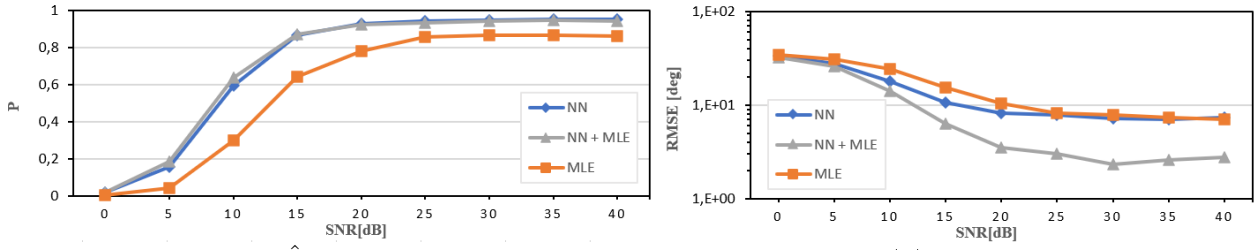
Figure 4.13: Model structure

This shallow NN is trained on a training dataset of 7×10^6 observations so that we consider as much as possible combinations of angles that could be processed on our hardware. The results are obtained for a test-dataset of 10^5 observations for each SNR. Moreover, it will be important to train small NNs for many more epochs than large sized NNs to reduce the training error because small changes have more influences on the output. Therefore, we have decided to train this NN for at least 400 epochs. Due to time constraints, we didn't have explored the minimum number of epochs that is required.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.14: Estimation performance shallow NN, 3 sources and a grid resolution of $\Delta\theta = 1^{\circ}$



(a) $P(\hat{k} = k)$ (b) RMSE

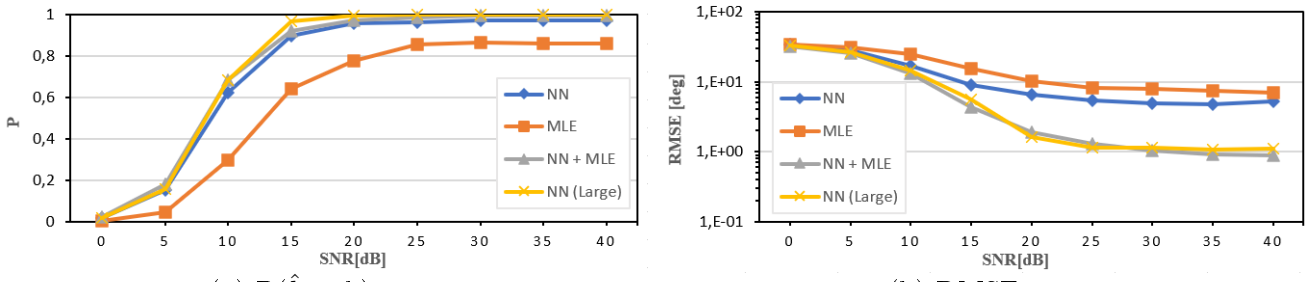
Figure 4.15: Estimation performance shallow NN given antenna imperfections, 3 sources and a grid resolution of $\Delta\theta = 1^{\circ}$

In Fig. 4.14, four frameworks were compared: the shallow NN(*NN*), the shallow NN + MLE(*NN+MLE*), NN of experiment 4.1(*NN(Large)*) and MLE. It can be observed that the *NN* has a average estimation performance degradation of 2% compared to *NN(Large)* for SNRs lower than 20dB. This is also confirmed by Fig. 4.14b where the RMSE values of *NN* are all higher than the one for [NN(Large)]. If we look specifically to the RMSE values for SNRs higher than 20dB, the shallow NN converged to 2.3° contrary to *NN(Large)* which is converged to 0.3°. This degradation is compensated by the addition of MLE and is improved to an RMSE 1.3°. This is still higher than achieved by *NN(Large)* but we have reduced the number of MAC operations by a factor 200 to 1.68×10^5 which is much lower than *NN(large)* and *MLE* require. These results are obtained for an theoretical optimal antenna array. Now, we will consider the behavior of a similar NN structure which is trained for an antenna array with imperfections.

The results are presented in Fig. 4.15 and has shown that the performance of the *NN+MLE* is degraded to an average RMSE of 2.7°. This is 1.4° higher than was achieved for an antenna array without imperfections and implies that the *NN* has not enough degrees of freedom to achieve similar performance as in Fig. 4.14a. Therefore, we have decided to increase the size of the hidden layers by 20 neurons so that the hidden layers have sizes of 180, 380, 180 respectively. The performance of this new model structure is depicted in Fig. 4.16a, and has competitive performance compared to *NN(Large)*. The new trained *NN* has an estimation performance of 97% contrary to 99% for the *NN+MLE* for SNRs higher than 20dB. This performance reduction can be directly related to the omission of MLE. The RMSE values in Fig. 4.16b show the main differences between the *NN* and *NN+MLE* where the *NN* has an average RMSE of 5° for higher SNRs contrary to 1° for *NN+MLE*. This implies that a NN without MLE has many

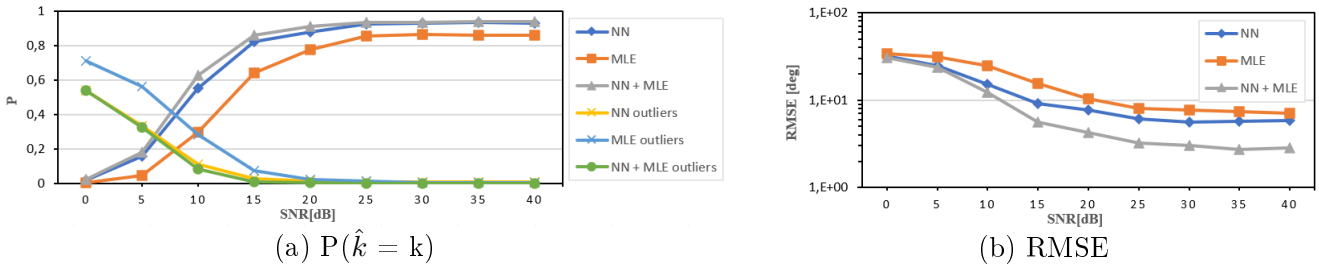
more outliers than the one including MLE and confirms the assumptions that MLE is capable of filtering outliers.

To determine if the shallow NN can achieve competitive results for a grid where the DoAs are uniformly distributed, a new NN has to be trained on data samples that contain uniformly distributed DoAs. The results obtained for this NN are presented in Fig. 4.17. It can be observed that the overall performance has been degraded to a maximum accuracy of 94% for SNRs of 25dB and higher. This was not noticed in previous experiments, so it will be very likely that this is caused by the increase in the complexity of the problem. Especially, similar behavior was observed when we had introduced imperfections. The hidden layers of this model were enlarged by 20 neurons and have achieved competitive performance with an RMSE of 1°, see Fig.4.16b. Another explanation for this degradation could be the low output resolution. Since the framework does not increase the output resolution, the NN will not offer the exact DoAs anymore. However, MLE is only a beneficial addition to an NN when we can offer the right set of DoAs because then we can regenerate a similar snapshot compared to the given one. Currently, we consider an output resolution of 1° and do not offer the exact set of DoAs with the consequence that multiple combinations of DoAs can result in the same inverse mapping. After analyzing the output data, it was observed that there are snapshots where the highest probabilities of the outputs are the correct estimated DoAs but are then not correctly estimated by MLE. This effect was not observed for a grid resolution of $\Delta\theta = 1^{\circ}$ and therefore confirms our hypothesis. The opposite is also true because the addition of MLE has resulted in an RMSE value that is converged to 3° instead of 5° for the NN without MLE. The performance of a shallow NN could be enhanced by increasing the output resolution to 0.5° or considering a regression model which has multiple outputs and letting MLE estimate the correct DoAs because we have observed that MLE can filter outliers well.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.16: Estimation performance of enlarged shallow NN given antenna imperfections, 3 sources and a grid resolution of $\Delta\theta = 1^{\circ}$

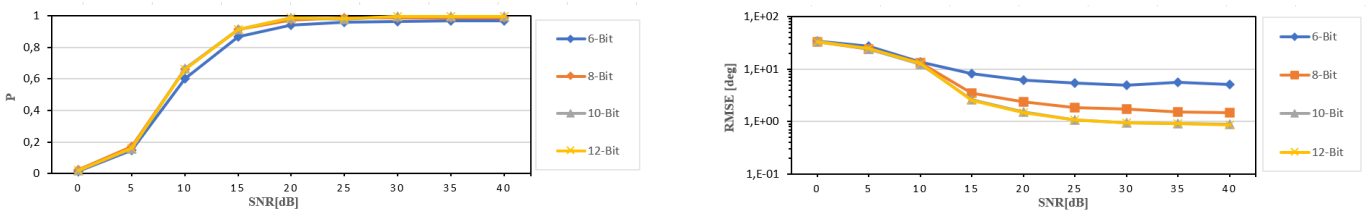


(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.17: Estimation performance of enlarged shallow NN given antenna imperfections, 3 sources and uniformly distributed DoAs

4.4.1 Quantization

Through the exponential growth of computational abilities of the last decade, floating-point arithmetic has become by far the most used arithmetic. These numbers have a much larger dynamic range which is especially important when processing extremely large data sets or data sets where the range may be unpredictable. However, floating-point operations are more complex and require more area and power which highly affect the hardware costs. To optimize these costs, fixed-point representation tends to be preferred. Therefore, the weights of this NN are represented in several bit widths to explore the minimum bit width that is required. When the weights are analyzed, it was noticed that all weights are in between [-2, 2]. This means that all weights were represented as signed values and need 1 bit for the integer part. All the remaining bits of the mantissa can be used for the fractional part. The proposed bit widths for this experiment are 6, 8, 10 and 12 bits. The results are obtained by a test-set containing 10^3 observations and are presented in Fig. 4.18.



(a) $P(\hat{k} = k)$ (b) RMSE

Figure 4.18: Estimation performance of quantized neural network, 3 sources and a grid resolution of $\Delta\theta = 1^{\circ}$

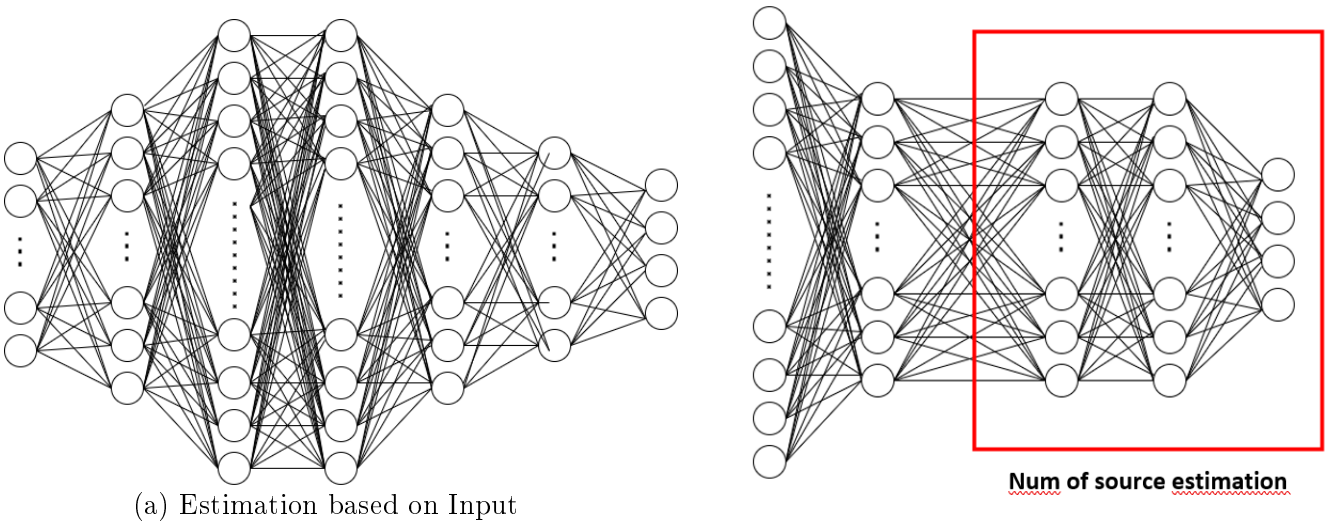
It can be observed that almost all bit-width can achieve an estimation performance of at least 96% correctly. The difference in estimation performance is visible in Fig. 4.18b where it can be observed that an increased bit-width gives a significant reduction in RMSE value relative to a lower bit-width. The RMSE values are stabilized for a minimum bit-width of 10 which means that we can reduce our shallow neural network to a memory size of 265 kB. Compared to our non-optimized model we have achieved with all our optimizations a memory reduction of more than 200 times.

4.5 Number of Source estimation

With the exception of a few methods, most traditional DoA estimation methods require prior knowledge of the number of signal sources. If the estimated number of sources does not match the actual one, MLE is not able to estimate internally a snapshot that is similar to the one provided as input. This limitation does not appear in data-driven algorithms because there is no prior knowledge required to estimate DoAs. This makes that we can consider it as two separate problems that can be addressed individually. This section proposes two source estimation approaches that make use of different input data.

Fig. 4.19 illustrates a schematic overview of the two approaches. The first approach, which is shown Fig. 4.19a, estimates the number of sources based on the correlation matrix of the antenna data. This approach can run parallel to the NN for DoA estimation to estimate the number of DoAs immediately. This small NN has 5 hidden layers of sizes: [100 250 250 100 25] and 4 output neurons which require a total number of 122.600 MAC operations.

Fig. 4.19b presents the second approach, which is based on the statistical information of the output layer. This means that we will use instead of the covariance matrix the output probabilities of the NN that is used to estimate the DoAs to estimate the number of sources. We have extended the previously defined NN in section 4.4 with 2 extra hidden layers and

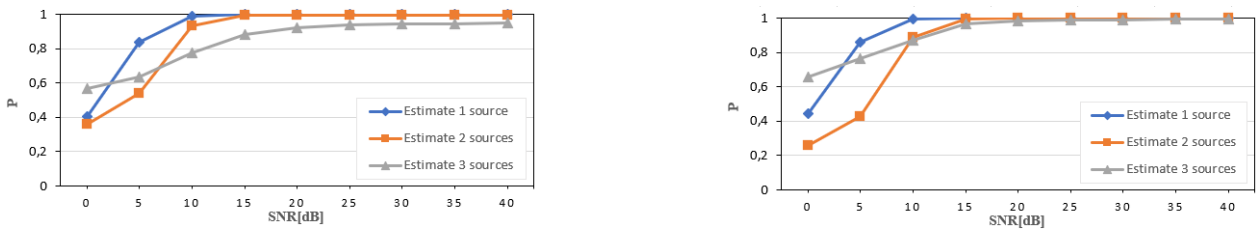


(a) Estimation based on Input

(b) Estimation based on output

Figure 4.19: Neural network structures number of source estimation

one output layer of 4 neurons. This additional piece of NN requires 40.326 MAC operations, which means that we can reduce the number of MAC operations by a factor of 3 if we use the statistical data of the output instead of the input data. The estimation performance of both approaches is presented in Fig. 4.20



(a) Estimation based on Input

(b) Estimation based on output

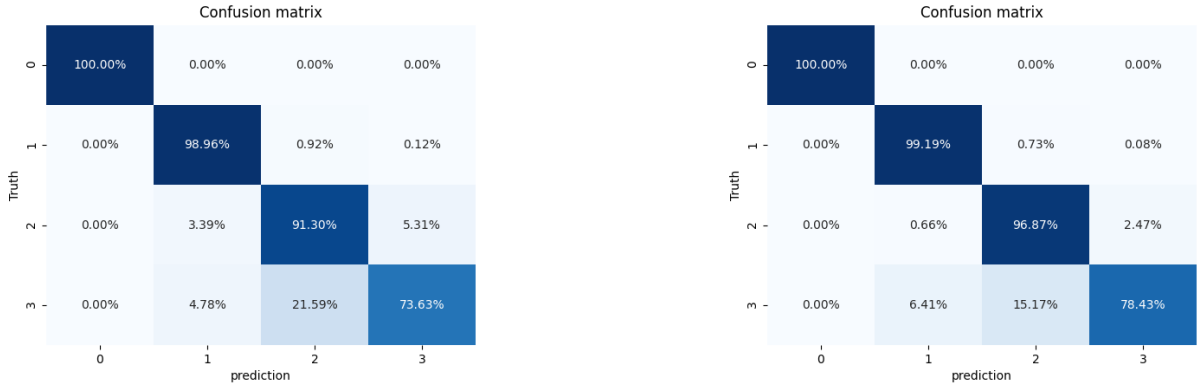
Figure 4.20: Estimation performance of the number of sources

It can be observed that both approaches can achieve a good estimation performance for 1 and 2 DoAs. The main difference is noticed for estimating 3 DoAs, where estimations based on output data achieve a 5% higher accuracy than when using the ($\hat{\mathbf{R}}_{\mathbf{x}}$) as input. The maximum accuracy that has been achieved is 99% for SNRs 20dB and higher. This, in combination with the low number of MAC operations makes this an attractive approach fur future implementation in radar systems. The only disadvantage will be the time delay between the estimation of the DoAs and the number of sources because it is processed in sequential order.

This experiment showed that it would be interesting to use the output data of the NN for DoA estimation to estimate the number of sources. In an earlier experiment, we have compared the difference in estimation performance of the number of sources for snapshots used for azimuth (1D) estimation and snapshots used for azimuth and elevation (2D) estimation. Both NNs were structured as shown in Fig. 4.19a, because the 2D NN approach does not provide probabilities for each output class. The results were obtained for a SNR of 10dB and are presented in Fig. 4.21a and Fig. 4.21b. An interesting observation in the confusion matrix for snapshots of 2D estimation is the high estimation performance of 96,9% for 2 sources. This is significantly higher than the estimation performance that is achieved for snapshots which are used for 1D estimation. With analyses for more SNRs was concluded that the estimation performance for snapshots, used for 2D estimation, was only slightly improved to a maximum

of 81% for 3 sources. Another remarkable observation is the estimation behavior for 3 sources in the confusion matrix of Fig. 4.21b. It can be observed that 6.4% of the estimations for 3 sources is predicted as 1 instead of 2 sources. This percentages is reduced to approximately 5% for higher SNRs but is still high in case that 3 sources has to be estimated.

In contrast, Fig. 4.21a shows that the estimation performance of 3 sources is 73.6% for SNRs of 10dB and increase to a maximum of 93.5% as can be observed in Fig. 4.20a. For these snapshots, it was observed that 5% of the miss-predictions for the 3 source case was predicted as 1. This percentage has been reduced to below 1 for SNRs of 20dB and higher. This will be less problematic since the percentage of the miss-predictions is far less.



(a) Snapshots used for 1D estimation (b) Snapshots used for 2D estimation

Figure 4.21: Confusion matrices 1D vs 2D number of source estimation, SNR 10dB

Chapter 5

Conclussion and Future work

Our main goal was to explore the use of neural networks for DoA to overcome some difficulties of traditional algorithms in radar systems. We will investigate this by doing several experiments where each experiment considered a different aspect of the DoA estimation problem. In this chapter, we state our conclusions and indicate a direction for future work.

5.1 Conclusion

The initial problem was defined as estimating 3 azimuth signal sources by using a (simple) NN. It has been shown that a NN can estimate 99% of the observations correctly for SNRs of at least 15dB. This is even better than MLE does while the NN requires 18% less MAC operations. However, only for a discrete grid with $\Delta\theta = 1^{\circ}$ and SNRs higher than 20dB, the RMSE value of MLE will be at least 10% lower than the one achieved for a NN.

For the 2D estimation problem, we explored whether we could estimate the azimuth and elevation angles separately, i.e. with two NNs. We achieved a maximum of only 80% correctly estimated azimuth angles and 67% correctly estimated elevation angles within a tolerance of 1°. This is gradually improved by only considering 2 signal sources to a minimum accuracy of 90% for both algorithms for SNRs 25dB and higher. Both models are used in combination with MLE to estimate the coordinate of azimuth and elevation angle. The final 2D estimation performance that could be achieved was 82% for a maximum of 2 sources.

Another aspect that was considered were the antenna imperfections. These imperfections are difficult to take into account in traditional DoA estimation algorithms but do not affect the NN performance when the NN is trained on imperfect antenna array data. This is an interesting property of NNs and can be useful if there is no prior knowledge about imperfections of the antenna array.

The memory size of NN is the main area in which major improvements need to be made. It was found that we could achieve a major memory reduction to compose a hybrid approach where we combine a neural network with MLE. This has resulted in a major reduction in computation costs of a factor 200 with only a modest estimation performance degradation while considering DoAs on a grid $\Delta\theta = 1^{\circ}$. In future research, it should be investigated how well the small neural networks perform for arbitrary angles instead of angles that lay on the search grid.

Another part of the DoA estimation problem that we considered were estimating the number of sources. It has been shown that using the output probabilities of the DoA estimations as input for a second NN for the number of source estimations can achieve an estimation performance of at least 95% for SNRs of 15dB and higher for a maximum of 3 sources. This will be 10% higher compared to using the ($\hat{\mathbf{R}}_{\mathbf{x}}$) as input. Another advantage of using the output probabilities compared to the upper triangle of the covariance matrix ($\hat{\mathbf{R}}_{\mathbf{x}}$) is the reduction in the number of MAC operations by a factor 3 to a maximum of 40.326 MACs.

5.2 Future work

Although several aspects of the DoA estimation problem were considered, a potential shortcoming is that only artificially generated data was used instead of measured data. An interesting direction would be to explore whether snapshots can be used as input data of the NNs instead of the covariance matrix. This would reduce the size of the input layers of the NN.

Another issue that should be studied is how well the small neural networks perform for arbitrary angles instead of angles that lay on the search grid. I will mention a few suggestions which could improve the estimation performance of the shallow NN. The first experiment could be to enlarge the neural network by one or two layers to increase the degree of freedom. This should be improve estimation performance because experiment 4.1 has shown that a large NN can estimate well for uniformly distributed DoAs. A second experiment could be to double the number of outputs neurons to increase output resolutions so that MLE gets more accurate DoAs which could result in fewer miss-predictions. The only disadvantage of this approach is that the increased number of outputs will also lead to a more complex problem and thus a more complex NN. The last suggestion will be to explore whether a neural network model based on regression in combination with MLE could achieve a higher estimation performance. The regression model should have $K + 3$ outputs where K depends on the maximum number of sources that has to be estimated. Sequentially, MLE is used, in the same way as mentioned in the experiments, to estimate the correct DoAs. This should improve estimation performance through the more accurate DoAs that are provided to MLE.

Furthermore, the hybrid approach has shown a significant reduction in network size with only a very modest estimation performance degradation. According to the results, the trained weights can at least be represented in 10-bit fixed-point numbers. It would be preferable if it can be reduced to 8-bit fixed-point numbers. Therefore, it is necessary to investigate why the weights, in the last two layers of the number of source estimation parts, are a factor 4 higher than the weights of the remaining layers. These weights require more integer bits while we have to use the same amount of fractional bits to achieve an equivalent precision.

Bibliography

[1] Murat Dikmen and Catherine Burns. "Trust in autonomous vehicles: The case of Tesla Autopilot and Summon". In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2017, pp. 1093–1098. DOI: 10.1109/SMC.2017.8122757.

[2] Njeri P. Waweru, Dominic Bernard Onyango Konditi, and Philip Kibet Langat. "Performance Analysis of MUSIC, Root-MUSIC and ESPRIT DOA Estimation Algorithm". In: *World Academy of Science, Engineering and Technology, International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering* 8 (2014), pp. 209–216.

[3] R. Roy and T. Kailath. "ESPRIT-estimation of signal parameters via rotational invariance techniques". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.7 (1989), pp. 984–995. DOI: 10.1109/29.32276.

[4] Hongji Huang et al. "Deep Learning for Super-Resolution Channel Estimation and DOA Estimation Based Massive MIMO System". In: *IEEE Transactions on Vehicular Technology* 67.9 (2018), pp. 8549–8560. DOI: 10.1109/TVT.2018.2851783.

[5] Marija Agatonovic, Zoran Stanković, and Bratislav Milovanović. "High resolution two-dimensional DOA estimation using artificial neural networks". In: *2012 6th European Conference on Antennas and Propagation (EUCAP)*. 2012, pp. 1–5. DOI: 10.1109/EuCAP.2012.6206729.

[6] M. Li, Y. Lu, and B. He. "Array signal processing for maximum likelihood direction-of-arrival estimation". In: *Journal of Electrical and Electronic Systems* 3.1 (2013), p. 117. DOI: 10.4172/jees.1000117. URL: http://eprints.gla.ac.uk/88622/.

[7] R. Schmidt. "Multiple emitter location and signal parameter estimation". In: *IEEE Transactions on Antennas and Propagation* 34.3 (1986), pp. 276–280. DOI: 10.1109/TAP.1986.1143830.

[8] Fang-Ming Han and Xian-Da Zhang. "An ESPRIT-like algorithm for coherent DOA estimation". In: *IEEE Antennas and Wireless Propagation Letters* 4 (2005), pp. 443–446. DOI: 10.1109/LAWP.2005.860194.

[9] I. Ziskind and M. Wax. "Maximum likelihood localization of multiple sources by alternating projection". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 36.10 (1988), pp. 1553–1560. DOI: 10.1109/29.7543.

[10] A.H. El Zooghby, C.G. Christodoulou, and M. Georgiopoulos. "Antenna array signal processing with neural networks for direction of arrival estimation". In: *IEEE Antennas and Propagation Society International Symposium 1997. Digest.* Vol. 4. 1997, 2274–2277 vol.4. DOI: 10.1109/APS.1997.625423.

[11] Jonas Fuchs, Robert Weigel, and Markus Gardill. "Single-Snapshot Direction-of-Arrival Estimation of Multiple Targets using a Multi-Layer Perceptron". In: *2019 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*. 2019, pp. 1–4. DOI: 10.1109/ICMIM.2019.8726554.

[12] Zhang-Meng Liu, Chenwei Zhang, and Philip S. Yu. "Direction-of-Arrival Estimation Based on Deep Neural Networks With Robustness to Array Imperfections". In: *IEEE Transactions on Antennas and Propagation* 66.12 (2018), pp. 7315–7327. DOI: 10.1109/TAP.2018.2874430.

[13] Bo Hu et al. "DOA Robust Estimation of Echo Signals Based on Deep Learning Networks With Multiple Type Illuminators of Opportunity". In: *IEEE Access* 8 (2020), pp. 14809–14819. DOI: 10.1109/ACCESS.2020.2966653.

[14] Oded Bialer, Noa Garnett, and Tom Tirer. "Performance Advantages of Deep Neural Networks for Angle of Arrival Estimation". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 3907–3911. DOI: 10.1109/ICASSP.2019.8682604.

[15] Wanli Liu. "Super resolution DOA estimation based on deep neural network". In: *Scientific Reports* 10 (2020).

[16] Abhinav Goel et al. "A Survey of Methods for Low-Power Deep Learning and Computer Vision". In: *CoRR* abs/2003.11066 (2020). arXiv: 2003.11066. URL: https://arxiv.org/abs/2003.11066.

[17] Abhinav Goel et al. "A Survey of Methods for Low-Power Deep Learning and Computer Vision". In: *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*. 2020, pp. 1–6. DOI: 10.1109/WF-IoT48130.2020.9221198.

[18] Hyeonwook Wi et al. "Compressing Sparse Ternary Weight Convolutional Neural Networks for Efficient Hardware Acceleration". In: *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. 2019, pp. 1–6. DOI: 10.1109/ISLPED.2019.8824855.

[19] Pooja Gupta and Sambit Prasad Kar. "MUSIC and improved MUSIC algorithm to estimate direction of arrival". In: *2015 International Conference on Communications and Signal Processing (ICCSP)* (2015), pp. 0757–0761.

[20] M. Haardt and J.A. Nossek. "Unitary ESPRIT: how to obtain increased estimation accuracy with a reduced computational burden". In: *IEEE Transactions on Signal Processing* 43.5 (1995), pp. 1232–1242. DOI: 10.1109/78.382406.

[21] Dyonisius Dony Ariananda and Geert Leus. "Direction of arrival estimation for more correlated sources than active sensors". In: *Signal Processing* 93.12 (2013). Special Issue on Advances in Sensor Array Processing in Memory of Alex B. Gershman, pp. 3435–3448. ISSN: 0165-1684. DOI: https://doi.org/10.1016/j.sigpro.2013.04.011. URL: https://www.sciencedirect.com/science/article/pii/S0165168413001540.

[22] Rahul Parhi and Robert D. Nowak. "The Role of Neural Network Activation Functions". In: *IEEE Signal Processing Letters* 27 (2020), pp. 1779–1783. DOI: 10.1109/LSP.2020.3027517.

[23] M Roopa and S. Selva Kumar Raja. "Artificial neural network using back propagation algorithm in distributed MANETs". In: *2016 International Conference on Information Communication and Embedded Systems (ICICES)*. 2016, pp. 1–4. DOI: 10.1109/ICICES.2016.7518899.

[24] Qiong Liu and Ying Wu. "Supervised Learning". In: *Encyclopedia of the Sciences of Learning*. Ed. by Norbert M. Seel. Boston, MA: Springer US, 2012, pp. 3243–3245. ISBN: 978-1-4419-1428-6. DOI: 10.1007/978-1-4419-1428-6_451. URL: https://doi.org/10.1007/978-1-4419-1428-6_451.

[25] Osvaldo Simeone. "A Very Brief Introduction to Machine Learning With Applications to Communication Systems". In: *IEEE Transactions on Cognitive Communications and Networking* 4.4 (2018), pp. 648–664. DOI: 10.1109/TCCN.2018.2881442.

[26] Dokkyun Yi, Jaehyun Ahn, and Sangmin Ji. "An Effective Optimization Method for Machine Learning Based on ADAM". In: *Applied Sciences* 10.3 (2020). ISSN: 2076-3417. DOI: 10.3390/app10031073. URL: https://www.mdpi.com/2076-3417/10/3/1073.

[27] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), 1929–1958. ISSN: 1532-4435.

[28] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG].

[29] Stephen Hanson and Lorien Pratt. "Comparing Biases for Minimal Network Construction with Back-Propagation". In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 1. Morgan-Kaufmann, 1989, pp. 177–185. URL: https://proceedings.neurips.cc/paper/1988/file/1c9ac0159c94d8d0cbedc973445af2da-Paper.pdf.

[30] Laurréline Perotin et al. "Regression Versus Classification for Neural Network Based Audio Source Localization". In: *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. 2019, pp. 343–347. DOI: 10.1109/WASPAA.2019.8937277.

[31] Lean Yu, Shouyang Wang, and K.K. Lai. "An integrated data preparation scheme for neural network data analysis". In: *IEEE Transactions on Knowledge and Data Engineering* 18.2 (2006), pp. 217–230. DOI: 10.1109/TKDE.2006.22.

[32] Fabrizio Sellone and Alberto Serra. "A Novel Online Mutual Coupling Compensation Algorithm for Uniform and Linear Arrays". In: *IEEE Transactions on Signal Processing* 55.2 (2007), pp. 560–573. DOI: 10.1109/TSP.2006.885732.

[33] J. Sola and J. Sevilla. "Importance of input data normalization for the application of neural networks to complex industrial problems". In: *IEEE Transactions on Nuclear Science* 44.3 (1997), pp. 1464–1468. DOI: 10.1109/23.589532.

[34] John Rogers, John E. Ball, and Ali C. Gurbuz. "Estimating the Number of Sources via Deep Learning". In: *2019 IEEE Radar Conference (RadarConf)*. 2019, pp. 1–5. DOI: 10.1109/RADAR.2019.8835819.

[35] G. Burel and N. Rondel. "Neural networks for array processing: from DOA estimation to blind separation of sources". In: *Proceedings of IEEE Systems Man and Cybernetics Conference - SMC*. Vol. 2. 1993, 601–606 vol.2. DOI: 10.1109/ICSMC.1993.384940.

[36] Frederick Tung and Greg Mori. "Deep Neural Network Compression by In-Parallel Pruning-Quantization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.3 (2020), pp. 568–579. DOI: 10.1109/TPAMI.2018.2886192.

[37] Amir Gholami et al. "A Survey of Quantization Methods for Efficient Neural Network Inference". In: *ArXiv* abs/2103.13630 (2021).

[38] Jianyuan Yu et al. *Direction of Arrival Estimation for a Vector Sensor Using Deep Neural Networks*. 2020. arXiv: 2004.05671 [eess.SP].

[39] Jian Li and Petre Stoica. "MIMO Radar with Colocated Antennas". In: *IEEE Signal Processing Magazine* 24.5 (2007), pp. 106–114. DOI: 10.1109/MSP.2007.904812.

[40] Andreas Kirschner et al. “A design-algorithm for MIMO radar antenna setups with minimum redundancy”. In: *2013 IEEE International Conference on Microwaves, Communications, Antennas and Electronic Systems (COMCAS 2013)*. 2013, pp. 1–5. DOI: 10.1109/COMCAS.2013.6685236.