DESIGN OF DEEP LEARNING BASED METHOD FOR OPTIMIZING MIMO COMMUNICATION

THESIS

ELECTRICAL ENGINEERING

COMMUNICATION AND INFORMATION SYSTEMS

Declared qualified to obtain

a *Master Teknik* degree

Mahdin Rohmatillah
Student ID: 166060300111002

UNIVERSITY of BRAWIJAYA
FACULTY of ENGINEERING
MALANG
2018

Acknowledgement

This master's thesis was done in National Sun Yat-Sen University, Taiwan as a result of double degree cooperation with University of Brawijaya, Indonesia.

First and foremost, I would like to thank God, The Almighty, for giving me His blessing and opportunity to make everything possible. Secondly, I would like to express my deepest appreciation and gratitude to my research supervisor in National Sun Yat-sen University, Professor Chao-Kai Wen for his advice and invaluable guidance throughout my research. I would also like to thank my research supervisors in University of Brawijaya, Hadi Suyono, ST., MT., Ph.D and Rahmadwati, ST., MT., Ph.D. This work will not be done without their precious support and encouragement

I also want to give much respect and appreciation for all of my lab mates in Communication Technology (CT) Lab, the best lab in the world (:D).

I dedicated this thesis to my family for the love, constant encouragement, and endless support. My deepest gratefulness goes to my wonderful mother and father, Binti Maqsudah and Sholeh Hadi Pramono, for her praying that manage me to reach this far. Special thanks and appreciation for all of my brothers and sister, Arif Ulumuddin, M. Hamdani Azmi and Somadevi for their patience, support, and heartwarming love. The last is for all of my best friends who always understand and give much support to me.

Kaohsiung, July 2018

Mahdin Rohmatillah

ABSTRACT

Multiple Input Multiple Output (MIMO) communication system, a system implementing multiple antennas at the transmitter and receiver, has been developed rapidly in order to improve the effectiveness of communication among users. However, trade-off phenomenon between performance and computational complexity always become the hugest dilemma suffered by researchers. As an alternative solution to the aforementioned problem, this research proposes an optimization in both of spatial diversity and spatial multiplexing MIMO communication system using end-to-end learning based model, specifically, it adapts autoencoder model. Four models are introduced in this thesis which each two of them address a problem about data detection task and channel estimation task that has not been addressed in the previous research. The proposed models were evaluated in one of the most common channel impairment which is Rayleigh fading with additional Additive White Gaussian Noise (AWGN). The results show that these deep learning based models for MIMO communication system result in very promising results by outperforming the baseline methods (methods widely used in conventional MIMO communication). In perfect CSIR (Channel State Information in Receiver side) case, the proposed models achieve BER nearly 10^{-5} at SNR 22.5 dB. While in channel estimation case, the proposed models can exceed the baseline performance even by only transmitting 2 pilots.

Keywords: Deep Learning; MIMO Communication; Spatial Diversity; Spatial Multiplexing

Table of Contents

List of Tables

Table **Page**

List of Figures

Figure **Page**

List of Abbreviations

AWGN............................Additive White Gaussian Noise

BER...............................Bit Error Rate

CNN...............................Convolutional Neural Network

CSI.................................Channel State Information

MAP...............................Maximum a Posteriori

MIMO............................Multiple Input Multiple Output

ML.................................Maximum Likelihood

MLP...............................Multilayer Perceptron

NN.................................Neural Network

OFDM............................Orthogonal Frequency Division Multiplexing

PReLU...........................Parametric Rectified Linear Unit

ReLU.............................Rectified Linear Unit

RNN...............................Recurrent Neural Network

SNR...............................Signal to Noise Ratio

Chapter 1

Introduction

1.1 Background

The increase of demand on better quality of service in telecommunication including high data rate, reliable and secured wireless communication has put significant pressure on wireless communication researchers to develop a new method for satisfying users' expectations. However, this problem has become more complicated due to the limitation of radio frequency spectrum and several wireless channel impairments, for instance Rayleigh fading. As a solution, in recent years, Multiple-Input Multiple-Output (MIMO) systems have arose as one of the most promising methods in the wireless communication system (Oestges *et al*, 2010).

The utilization of several antennas either at transmitter or receiver or at both of them has become more popular nowadays due to its ability to maintain a reliable communication in a wireless channel with some impairment predominantly by fading. This reliable communication can be maintained because multiple antennas technology provides benefits in a communication system which are spatial multiplexing or spatial diversity gain, array gain and interference reduction (Biglieri *et al*, 2007)

The main idea behind MIMO communication is that signals which are sampled in the spatial domain at both transmitter and receiver are combined in a certain method that they either add diversity to improve the quality in term of Bit Error Rate (BER) of the communication and/or create effective multiple parallel spatial data pipes that will result in increasing the data rate (Oestges *et al*, 2010).

Basically the MIMO communication process is that first, input bits are first encoded through several process that eventually result in spatial data streams. These data streams are then transmitted by several antennas to the receiver and propagate through certain channel impairments, for example Rayleigh fading. The received signal will be decoded in the receiver side until the estimated bits are obtained.

However, the process of encoding and decoding mentioned in previous paragraph are very challenging. For years, researchers have been developing algorithms in multiple antennas technology in order to improve its performance either in detection task or channel estimation task or other tasks but the issue of a trade-off between performance improvement and computational complexity always become a main restriction and consideration. As a solution, deep learning, an approach shining nowadays, is introduced in multiple antennas communication system.

Deep Learning actually is the development of the basic Neural Network (NN) which firstly introduced in 1940 by McCulloch et al in a form of electronic brain. Years after years this method has been developed very well that eventually result in some breakthrough models and methods. Those advanced models result in very high performance and can handle so many works in several domains, especially in computer vision started by digit recognition (Hinton *et al*, 2006).

1.2 Motivation

Recently, there are some publications implementing method from deep learning field in MIMO communication system in order to improve its performance. As a result, they perform very well and even result in better performance compare to the baseline methods.

Some of the most interesting results of machine learning implementation in a communication system are paper titled "An Introduction to Deep Learning for the Physical Layer" (O'shea *et al*, 2017) and "Deep Learning-Based Communication Over the Air" (Doner *et al*, 2018) which introduce deep learning as an end-to-end system in SISO communication. This end-to-end model means that transmitter, channel impairments, and receiver are represented by one or several neural network layer (dense) then interpret the whole system as an autoencoder, a powerful method for performing unsupervised learning (Baldi *et al*, 2012). Since they show good results, researches related to autoencoder implementation in MIMO communication has been developing rapidly, for instance its application in channel decoding (Gruber *et al*, 2017) and Orthogonal Frequency Division Multiplexing (OFDM) (Ye *et al*, 2018). However, the need of improvement in this topic is still required especially in end-to-end learning based model in order to make it feasible to be implemented in the real world condition.

1.3 Objective

This thesis aims to provide a new method in optimizing the performance of both spatial diversity and spatial multiplexing MIMO communication system. The proposed method is based on deep learning field implementation, which is end-to-end learning autoencoder. The deep learning method is selected as a solution because it has been proven as to work well with low computational complexity. The computational complexity in deep learning method emerge only in training stage. Once we obtain the well trained weights, we just need to load them and pass the data for testing stage.

1.4 Thesis Organization

This master thesis focuses on the design of end-to-end learning based models for MIMO communication in both spatial diversity (2x1) and spatial multiplexing (2x2) scheme. The proposed models are basically inspired of the autoencoder implementation where model try to replicate its input to its input. Chapter 1 discusses the background and the motivation of the research, Chapter 2 deeply explains several related theories referred as a basic reference in building this research including the baseline method. Chapter 3 explains a detailed design of deep learning based models and their properties (layout of all used layers and their hyperparameters tuning). Moreover, there is a clear explanation about the comparison between the proposed research and the previous research. Chapter 4 discusses about the simulation results obtained from Spyder software using keras with tensorflow (Abadi *et* al, 2016) backend for the deep learning based method and Matlab for the baseline method. Chapter 5 presents conclusion of this thesis which includes the summary and opportunities for the future research.

Chapter 2

Related Theory

This chapter discusses about theories referred as a basic reference in building this research. The related theories are divided into two parts, deep learning related theories and theories of baseline method that is later used as a comparison with the proposed neural network based method.

2.1 Deep Learning

2.1.1 Deep Feed Forward Network

Deep feed forward network or sometimes called Multilayer Perceptron (MLP) is one of the quintessential or fundamental deep learning model. The term deep comes from the depth of the hidden layer or layer between the input and the output layer. While, These models are called feedforward because information flows through the function being evaluated from *x* which then go through the intermediate computations used to define f, and eventually to the output y. This network aims to find the best function of f^* which suitable to the desired task. For example, this network define $y = f(x; \theta)$ using the best obtained parameter θ resulted from learning process (Goodfellow *et al*, 2016). This network has been developed to more advanced network, for instance Recurrent Neural Network (RNN) (Mikolov *et al*, 2010) feed forward network with feedback connection and Convolutional Neural Network (CNN) (Krizhevsky *et al*, 2012) which implements parameter sharing method.

Process of feed forward neural network is depicted by Figure 2-1. There are three type of nodes or layers which are, input layer, hidden Layer and output layer. Input layer gives information from the dataset to the network by directly passing them to the next layer, in other word there is no computational process in this layer. Hidden layer perform a massive computational process in order to find the best representation or features of given dataset. This layer called as a hidden layer as it does not have direct connection with the “world”. The last, output layer is a layer that provide a transfer information from the network to be implemented in the real world situation such as regression.

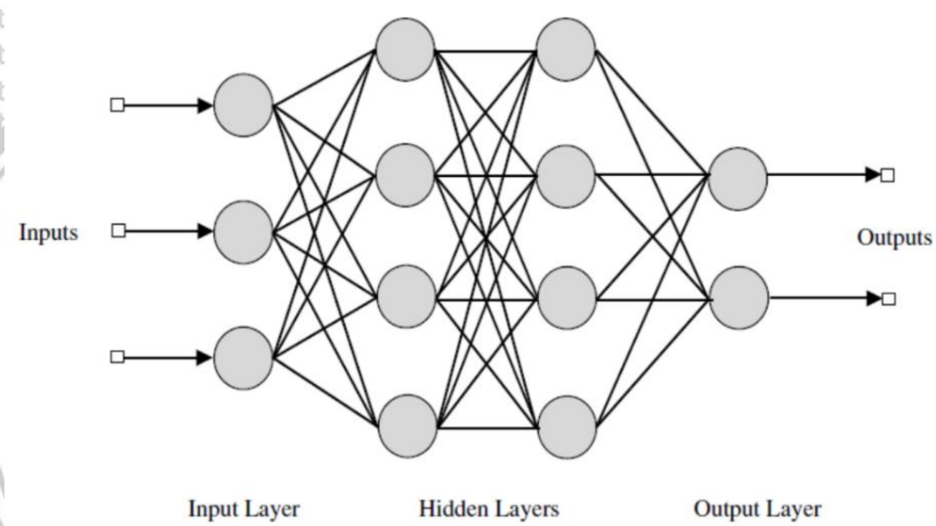


Figure 2-1: Feed Forward Neural Network

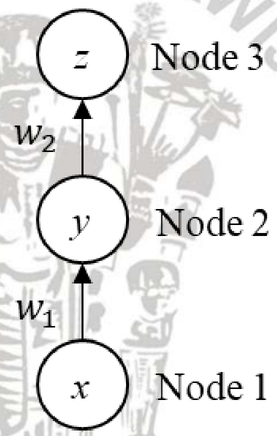
2.1.2 Backpropagation

Backpropagation (LeCun *et al*, 1989) is a method to compute the gradient for updating weights or parameter θ based on the cost function $J(\theta)$ acquired from feed forward process. The term back-propagation is often misunderstood as meaning the whole learning algorithm for multi-layer neural networks. Actually it just means the method for computing gradients in such networks. Moreover, backpropagation is generally considered as something very specific to multi-layer neural networks, but once its derivation is understood, it can easily be generalized to arbitrary functions. The basic idea of the back-propagation algorithm is that the partial derivative of the cost J with respect to parameters θ can be decomposed recursively by taking into consideration the composition of functions that relate θ to J, via intermediate quantities that mediate that influence, for example the activations of hidden units in a deep neural network.

The process of weight updating is done by applying chain rule. The illustration of chain rule algorithm is shown by Figure 2-2 (Goodfellow *et al*, 2016). Let assume *z* is the output node, *y* is hidden node, and *x* is the input node, *w* is the model weight or parameter θ, and *J* is the cost function. Let take one case for example, In order to know how much change affected by w_2 in total error, we must calculate partial derivative of $\frac{\partial J}{\partial w_2}$ which equal to following equation

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial a_3} \frac{\partial a_3}{\partial net_z} \frac{\partial net_z}{\partial w_2} \tag{2-1}$$

a_3 is the output of activation function in the node 3, net_z is the input of the node *z* or node



3. The obtained derivation is then used for parameter update.

Figure 2-2: Illustration of Chain Rule Algorithm

Algorithm 1 explain about backpropagation algorithm based on Andrew Ng explanation (Ng. Andrew, 2012) let assume that we have large training set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$. Then, backpropagation process can be explained through the following algorithm. Notation Δ is the variable that will be used to compute $\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta)$. δ denotes the error of the specific node. The last $D_{ij}^{(l)} = \frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta)$. Subscript i and j represent interconnection between node *i* and *j*, while superscript *l* denotes the l^{th} layer of the network.

Initialization: $\Delta_{ij}^{(l)} = 0$ for all l, i, j

for i = 1:m **do**

Set $a^{(1)} = x^{(i)}$

Perform forward propagation to compute $a^{(l)}$ for l = 2, 3, …, L

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$ (2-2)

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \dots, \delta^{(2)}$

$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$ (2-3)

end

if $j \neq 0$ **then**

$D_{ij}^{(l)} := \frac{1}{m} \Delta_{ij}^{(l)} + \lambda \theta_{ij}^{(l+1)}$ (2-4)

else

$D_{ij}^{(l)} := \frac{1}{m} \Delta_{ij}^{(l)}$ (2-5)

end

Algorithm 1: Backpropagation Algorithm

2.1.3 Autoencoder

Autoencoder is a Neural Network (NN) which is categorized as an unsupervised learning and has a function to replicate its input to its output. Basically, autoencoder consists of encoder and decoder which contain a code to describe the input ($h = f(x)$) and creates a reconstruction of the input from the hidden layer respectively ($r = g(h)$). The output of autoencoder, actually, is a compressed representation of the input that sometimes autoencoder can be used for feature reduction. The illustration of autoencoder model is shown by Figure 2-3.

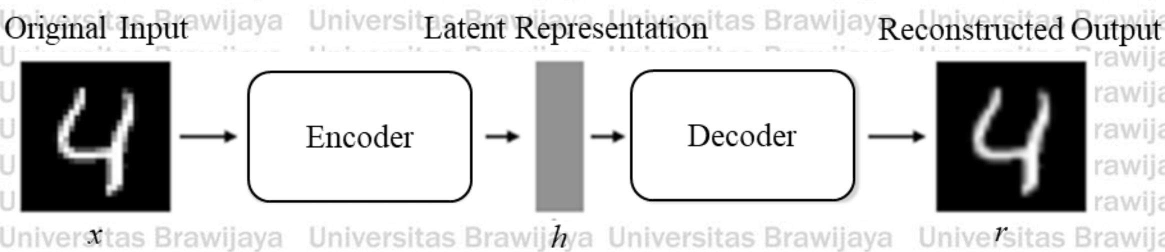


Figure 2-3: Architecture of an Autoencoder

2.1.4 Activation Function

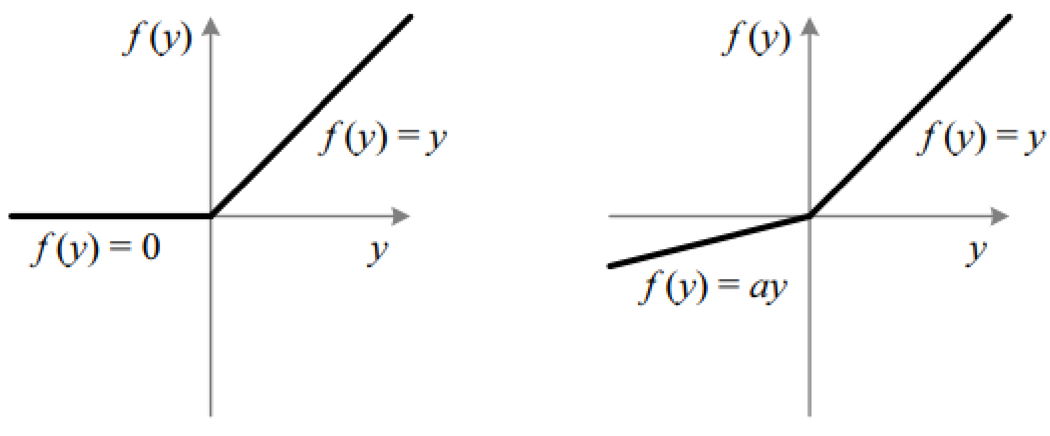
Activation functions are really important for in deep learning field in order to make the model understands about the relation between the inputs and response variable. They introduce non-linear properties to our Network. Their main purpose is to convert an input signal of a node in a network to an output signal that subsequently used as an input in the next layer. Sometimes activation function is called as a neuron internal state (Fausset *et al*, 1994).

2.1.4.1 PReLU (Parametric Rectified Linear Unit)

PReLU (He *et al*, 2015) is an activation function which generalizes the traditional Rectified Linear Unit (ReLU) which has zero gradient when the input is negative value. The output of PReLU is given by

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases} \tag{2-6}$$

Here, y_i is the output of the nonlinear activation function of f on the i^{th} channel and a_i denotes a coefficient controlling the slope of the negative part. The value of coefficient a varies on different channel. Figure 2-4 shows the difference between the output of ReLU and PReLU.



(a) (b)

Figure 2-4: Activation Output (a) ReLU (b) PReLU

The update value of coefficient a_i is done by using backpropagation algortithm, similar to weight update process. The gradient of a_i for one layer is given by

$$\frac{\partial \varepsilon}{\partial a_i} = \sum_{y_i} \frac{\partial \varepsilon}{\partial f(y_i)} \frac{\partial f(y_i)}{a_i} \tag{2-7}$$

ε represents the objective function, while $\frac{\partial \varepsilon}{\partial f(y_i)}$ indicates the gradient which is propagated from deeper layer. Next, the gradient of activation is given by

$$\frac{\partial f(y_i)}{a_i} = \begin{cases} 0, & \text{if } y_i > 0 \\ y_i, & \text{if } y_i \leq 0 \end{cases} \tag{2-8}$$

Eventually, the coefficient a_i is updated by using the momentum method as shown by the following equation

$$\Delta a_i \coloneqq \mu \Delta a_i + \epsilon \frac{\partial \varepsilon}{\partial a_i} \tag{2-9}$$

μ and ϵ denotes the momentum and learning rate respectively.

2.1.4.2 Softmax

Softmax is a generalization of logistic regression in order to handle multiple classes classification task (Duan *et al*, 2003). Softmax function are mostly used in the output layer to represent the probability over *J* different classes. Occasionally, softmax function is put in the hidden layer to behave as a decision maker between one of *J* different option for some internal variables. The output of softmax activation is given by

$$f_i(x) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \text{ for i} = 1, \dots, \text{J} \tag{2-10}$$

2.1.4.3 Linear

Linear activation function is an activation which its output is identical to its input. Figure 2-5 shows the output of linear activation function which follows the following equation

$$f(x) = x \tag{2-11}$$

Mostly, this activation is put in the output layer, not in the hidden layer. Because It doesn't help the network to understand with the complexity or various parameters of usual data that is fed to the neural networks.

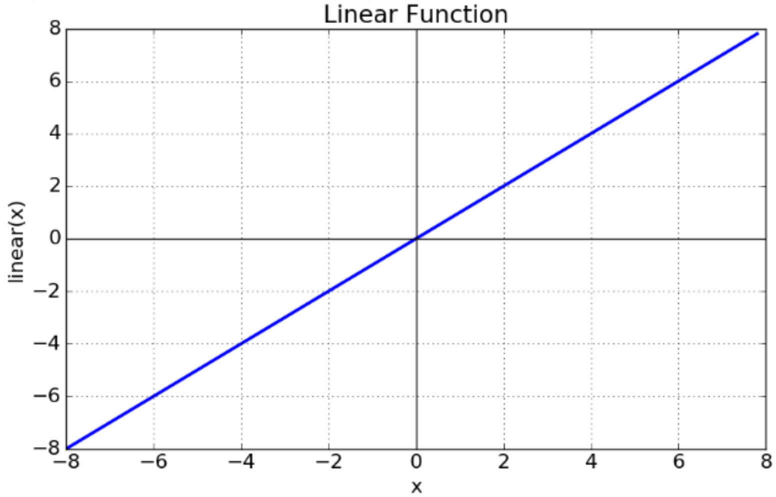


Figure 2-5: Linear Activation Function

2.1.5 Batch Normalization

Proposed in 2015, batch normalization is a method of adaptive reparameterization motivated by the difficulty of training deep models (Ioffe *et al*, 2015). Batch normalization applies a normalization to mini batch H (H') of activations of the layer by using following equation

$$H' = \frac{H-\mu}{\sigma} \tag{2-12}$$

where μ is the mean of each activation function over *m* data in batchsize which is defined by

$$\mu = \frac{1}{m}\sum_i H_i \tag{2-13}$$

and σ is a vector containing the standard deviation given by

$$\sigma = \sqrt{\delta + \frac{1}{m}\sum_i (H-\mu)_i{}^2} \tag{2-14}$$

However, the normalization of H (H') will reduce the performance of the model as the output of activation function is shifted or scaled by randomly initialized parameter. Then, instead of directly using H', batch normalization applies gamma parameter (standard deviation) and beta parameter (mean) to the equation. Therefore the batch normalization utilizes $\gamma H' + \beta$ that only allows SGD do the denormalization by changing only these two weights for each activation, instead of losing the stability of the network by changing all the weights.

2.1.6 Optimizer

2.1.6.1 Adam

Adam (Kingma *et* al, 2014) derived from adaptive momentum estimation, is a method for minimizing $\mathbb{E}[f(\theta)]$ with respect to its parameter θ that only requires first order gradient with require a little number of memory. Adam takes advantages of two previous methods, Adagrad (Duchi *et* al, 2011) and RMSProp (Tieleman *et al*, 2012) which works well with sparse gradients and in on-line and non-stationary setting. Several advantages of Adam are first; magnitudes of parameter updates are invariant to rescaling of gradient. The second, its stepsize are approximately bounded by the stepsize hyperparameter. The third, Adam does not require a stationary objective. The last, it works with sparse gradients. Moreover, it also naturally performs a form of step size annealing.

The process of Adam in optimizing the model weight is started by obtaining gradient with respect to stochastic objective at time step *t* where, *t* ← *t* + 1 that is

$$g_t \leftarrow \nabla_\theta f_t(\theta_t - 1) \tag{2-15}$$

Then, the biased first and second raw momentum estimate, m_t and v_t respectively, are updated by following equation

$$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1)g_t \tag{2-16}$$

$$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2){g_t}^2 \tag{2-17}$$

β_1 and β_2 are the exponential decay rate for the moment estimates. After that, parameter θ is updated preceded by changing the stepsize number, that is

$$a_t = a\sqrt{1-{\beta_2}^t}/(1-{\beta_1}^t) \tag{2-18}$$

$$\theta_t \leftarrow \theta_{t-1} - a_t m_t/(\sqrt{v_t}\ \hat{\in}) \tag{2-19}$$

${\beta_1}^t$ and ${\beta_2}^t$ are exponential decay rate for the moment estimates in a certain timestep and $\hat{\in}$ denotes the epsilon used for avoid zero division.

2.1.7 Loss Function

Loss function is a function that calculates loss between given an input to a target. In a simple way, loss function is a method of evaluating how well the model fit with the given dataset. Loss function is very useful for updating model weights as the output of it will become a guidance in training process (Goodfellow *et al*, 2016).

2.1.7.1 Log-cosh

Log-cosh is a function used in regression task that performs better or than L2. This function is defined as

$$L(y, \hat{y}) = \sum_{i=1}^{n} \log(\cosh(\hat{y}_i - y_i) \quad (2\text{-}20)$$

$\hat{y}_i$ and y_i is the predicted output and target of i^{th} dataset. and Log(cosh(x)) is approximately equal to $\frac{x^2}{2}$ for small x and $abs(x) - \log(2)$ for large x. It implicitly shows that log-cosh works nearly identical to mean squared error, but this function is not strongly affected by occasional wildly incorrect prediction like mean squared error. However, log-cosh loss has a drawback that it still suffers from the problem of gradient and hessian for very large off-target being constant (Neuneier *et al*, 1998). Figure 2-6 shows the plot of log-cosh vs prediction value where true value is equal to 0

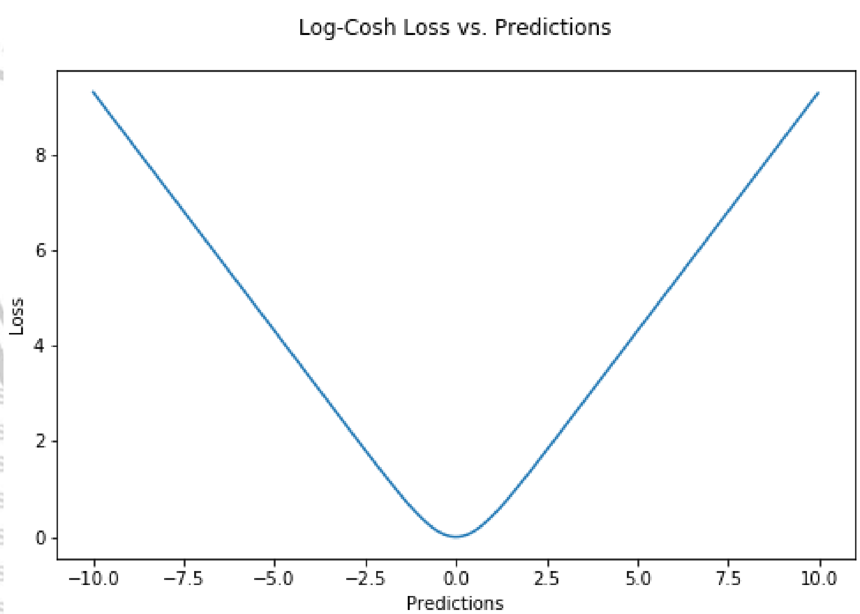


Figure 2-6: Log-Cosh Loss vs Predictions

2.1.7.2 Categorical Cross Entropy

Categorical cross entropy is a loss function which suitable or appropriate to handle multi class-classification task. Sometimes it is also called negative log likelihood. The equation of categorical cross-entropy loss function (ℓ_{CE}) is given by

$$\ell_{CE}(y,\hat{y}) = \frac{-1}{|y|}\sum_{i=0}^{|y|-1}[(y_i\, log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)] \quad (2\text{-}21)$$

$\hat{y}$ is the output of the last neural network layers while $\hat{y}_i$ and y_i denote estimated output and target of data respectively.

2.2 Baseline Method

2.2.1 Alamouti

Alamouti is a simple transmit diversity scheme improving the quality of signal at the receiver on one side of the link by simple processing across two transmit antenna on the opposite side (Cho *et al*, 2010). This scheme provides identical diversity order as MRRC consisting of one transmit antenna and two receive antennas. Alamouti scheme has been proven to provide an improvement in terms of error performance, data rate, or capacity of wireless communication. Figure 2-7 shows basic alamouti scheme proposed in (Alamouti, 1998)

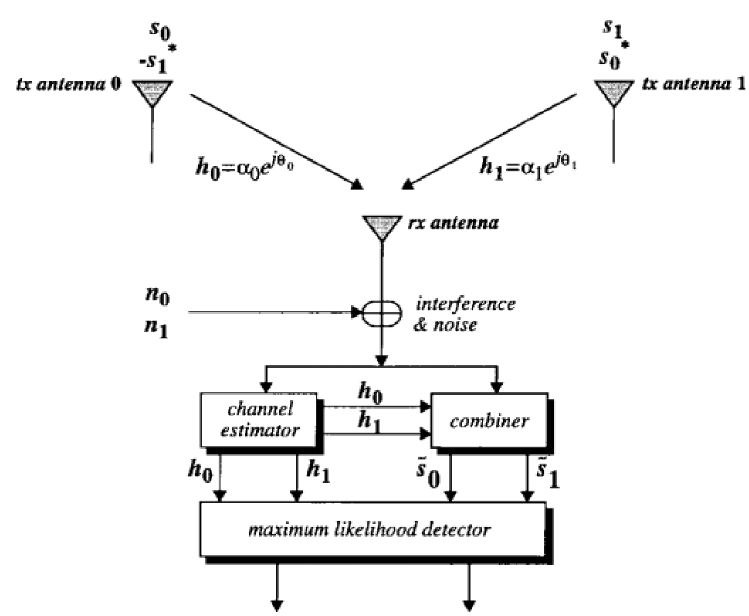


Figure 2-7: 2x1 Alamouti Scheme

Alamouti works as two signals are simultaneously transmitted over a given symbol perid. The transmitted symbols from antenna 1 and antenna 2 are shown in Table 2-1 or in other word the transmitted codeword is

$$\mathbf{X} = \begin{bmatrix} s_0 & -s_1{}^* \\ s_1 & s_0{}^* \end{bmatrix} \tag{2-22}$$

Table 2-1: Transmission Sequence for the Two-Branch Transmit Diversity Scheme

	Antenna 0	Antenna 1
Time t	s_0	s_1
Time t + T	$-s_1{}^*$	$s_0{}^*$

As depicted in Table 2-1, the transmitted codeword is a complex-orthogonal matrix that is,

$$\mathbf{SS}^H = \begin{bmatrix} |s_0|^2 + |s_1|^2 & 0 \\ 0 & |s_0|^2 + |s_1|^2 \end{bmatrix} = (|s_0|^2 + |s_1|^2)\mathbf{I}_2 \tag{2-23}$$

$\mathbf{I}_2$ denotes the 2x2 identity matrix.

Alamouti code has a diversity gain of 2, and this diversity analysis is based on ML signal detection at the receiver side. As explained in the paper (Alamouti, 1998), it is assumed that two channel gains, $h_0(t)$ and $h_1(t)$ are time-invariant over two consecutive symbol periods, that is,

$$h_0(t) = h_0(t + T_s) = h_0 = |h_0|e^{j\theta_0}$$

$$h_1(t) = h_1(t + T_s) = h_1 = |h_1|e^{j\theta_1} \tag{2-24}$$

Where $|h_i|$ and θ_i denote the amplitude gain and phase rotation of i antenna respectively. The received signal then is expressed by

$$y_0 = h_0 x_0 + h_1 x_1 + z_0$$

$$y_1 = -h_0 x_1{}^* + h_1 x_0{}^* + z_1 \tag{2-25}$$

The additive white Gaussian noise at time t and $t + T_s$ is denoted by z_0 and z_1 respectively. The equation in 2-20 can be formed to matrix vector equation as follow

$$\begin{bmatrix} y_0 \\ y_1{}^* \end{bmatrix} = \begin{bmatrix} h_0 & h_1 \\ -h_1{}^* & h_0{}^* \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} + \begin{bmatrix} z_0 \\ z_1{}^* \end{bmatrix} \tag{2-26}$$

In this simulation, $h_1$and h_2 are exactly known. Then, by multiplying both side of 2-21by the Hermitian transpose of the channel matrix, the equation becomes

$$\begin{bmatrix} h_0 & h_1 \\ -h_1^* & h_0^* \end{bmatrix}\begin{bmatrix} y_0 \\ y_1^* \end{bmatrix} = \begin{bmatrix} h_0^* & h_1 \\ h_1^* & -h_0 \end{bmatrix}\begin{bmatrix} h_0 & h_1 \\ h_1^* & -h_0^* \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} + \begin{bmatrix} h_0^* & h_1 \\ h_1^* & -h_0 \end{bmatrix}\begin{bmatrix} z_0 \\ z_0^* \end{bmatrix}$$

(2-27)

$$= (|h_0|^2 + |h_1|^2)\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} + \begin{bmatrix} h_0^* z_0 + h_1 z_0^* \\ h_1^* z_0 - h_0 z_0^* \end{bmatrix}$$

Then, the input-output relations are obtained as follow

$$\begin{bmatrix} \tilde{y}_0 \\ \tilde{y}_1 \end{bmatrix} = (|h_0|^2 + |h_1|^2)\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} + \begin{bmatrix} \tilde{z}_0 \\ \tilde{z}_1 \end{bmatrix} \tag{2-28}$$

where

$$\begin{bmatrix} \tilde{y}_0 \\ \tilde{y}_1 \end{bmatrix} \triangleq \begin{bmatrix} h_0 & h_1 \\ -h_1^* & h_0^* \end{bmatrix}\begin{bmatrix} y_0 \\ y_1^* \end{bmatrix}$$

(2-29)

$$\begin{bmatrix} \tilde{z}_0 \\ \tilde{z}_1 \end{bmatrix} \triangleq \begin{bmatrix} h_0 & h_1 \\ -h_1^* & h_0^* \end{bmatrix}\begin{bmatrix} z_0 \\ z_0^* \end{bmatrix}$$

As the antenna interference does not exist anymore or in other word no existence of unwanted symbol x_1 received at receiver 0, the Maximum Likelihood (ML) receiver structure as follows

$$x_{i,\mathrm{ML}} = \mathrm{Q}\left(\frac{\tilde{y}_i}{|h_0|^2+|h_1|^2}\right), i = 0,1. \tag{2-30}$$

The Q( ) denotes a slicing function determining a transmit symbol for the given constellation set.

2.2.2 Maximum Likelihood (ML) Detector

Maximum Likelihood (ML) detector is one of the most reliable method in spatial multiplexing MIMO communication. It provides minimal probability of error and low complexity in systems with few transmitting antenna (Cho *et* al, 2010). ML detector works by calculating the Euclidean distance between the received signal vector and product of all possible transmitted signal with the given channel $\boldsymbol{H}$, and find the one with the minimum distance. Let C and N_T denote a set of signal constellation symbol points and number of transmit antennas respectively. Then, ML determines the estimate of the transmitted vector $\boldsymbol{X}$ as

$$\hat{X}_{ML} = \underset{x \in C^{N_T}}{\operatorname{argmin}} \|y - Hx\|^2 \tag{2-31}$$

where y is the received signal and $\|y - Hx\|$ corresponds to the ML metrics.

The ML method achieves the optimal performance as the maximum a posteriori (MAP) detection when all the transmitted vectors are equally likely. The drawback of this detection method is that the complexity will exponentially increase as modulation order and/or the number of transmit antennas increase (Cho *et* al, 2010).

Chapter 3

Design of Deep Learning Based Model

This section deeply describes the design of the proposed method in term of the model. Moreover, the differences between the previous research and the proposed research will be clearly addressed. Basically, four models are introduced in this research assigned with the different tasks, 2x2 spatial multiplexing and 2x1 spatial diversity.

3.1 Overview of Proposed Method

Overall, the proposed models consist of several dense and lambda or custom layers representing MIMO communication. All of the proposed models actually follow the autoencoder scheme where model try to replicate its input to its output. In this research, each transmitter was designed to transmit 2 bits, making each antenna has 4 different bit pairs. Therefore, the total of bit pair combinations of each antenna are 16. Instead of expressing them in a one-hot encoding method, in this research, each of bit pair is expressed in an integer number that later be fed into embedding layer. The embedding layer will turn the data indices into vectors in order to save the memory usage. Reshape layer in the transmitter model block has a function to create parallel transmit stream denoted by three dimensional matrix $\mathbb{R}^{2x2xn}$. The first dimension represents the number of transmit antenna, the second dimension represents the complex number consisting of two real numbers, and the last dimension represents n time samples. Layer with linear function will determines the final transmitted symbol that its power will be constrained by BatchNormalization layer. Then, the last layer which has a softmax activation function will decode the message or data transmitted of each antenna. Figure 3-1 shows the Flowchart of the research. All of the models were trained using millions of synthetically generated data with various number of batch size with some hyperparameter tunings that will be explained deeply in the Chapter 4.

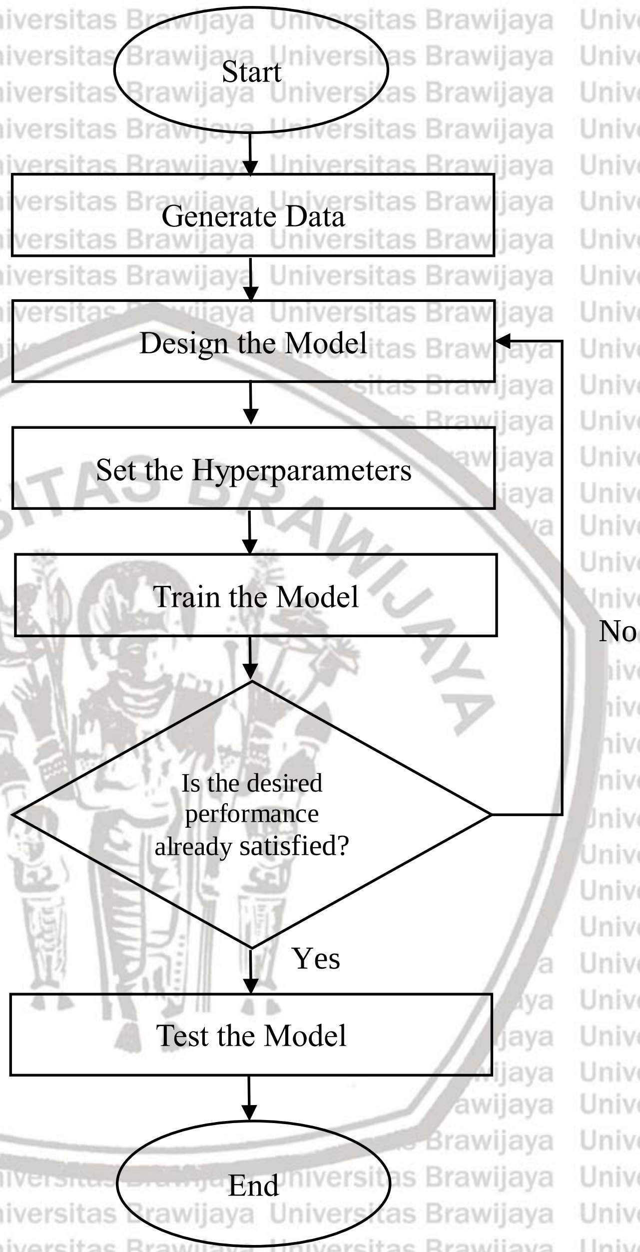


Figure 3-1: Flowchart of the Research

3.2 Spatial Diversity Model

3.2.1 Previous research

Previous research from paper titled "Deep Learning Based MIMO Communication" (O'shea *et al*, 2017) also proposes a model for detection task in spatial diversity MIMO communication system which is shown by Figure 3-2.

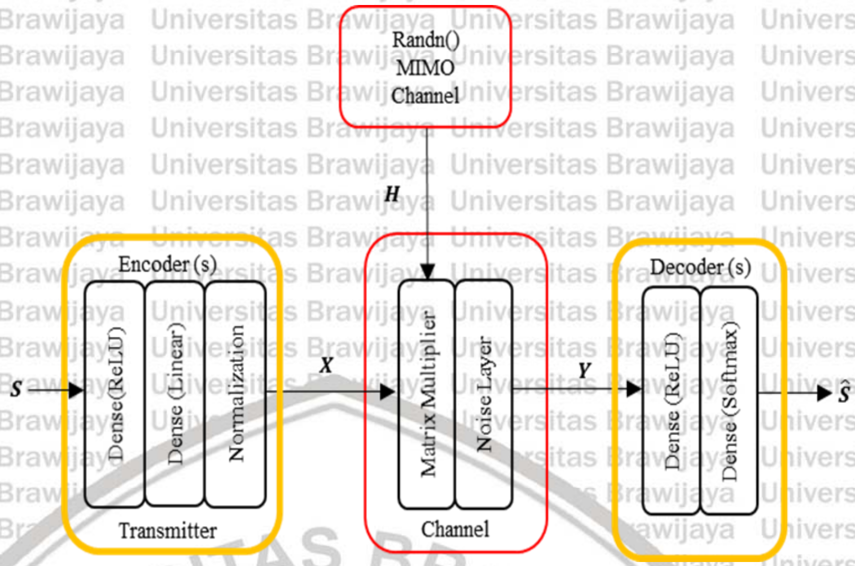


Figure 3-2: Previous Model in Spatial Diversity MIMO Communication

This model actually works identically to the proposed model. As discussed in the overview section, **S** is data which want to be transmitted represented as one-hot vector and $\boldsymbol{X}$ is a three dimensional matrix $\mathbb{R}^{2x2xn}$. However, there are some problems in the previous model. First, channel response $\boldsymbol{H}$ and noise existence are expressed by several custom layers. For the noise, maybe it is not a big problem as Keras already provides Gaussian noise layer as a regularizer but, for the channel response (Rayleigh fading), it brings up a doubt whether the channel response generated by a custom layer is suitable to standard Rayleigh fading or not. The second is no Channel State Information (CSI) in the receiver side which is an uncommon situation in the communication system. Moreover, as this model is compared to the baseline method which perfectly knows CSIR, the provided performance result can be considered as an unfair comparison.

3.2.2 Proposed Model

3.2.2.1 Data Detection with Perfect CSIR

Figure 3-3 shows the model of deep learning based spatial diversity MIMO communication. There are several differences between the previous model and the proposed research beside the depth of network. In this research, there are three input that will be fed to the model, those are data which want to be transmitted ($\boldsymbol{S}$), channel response $\boldsymbol{H}$ (Rayleigh fading) and Additive White Gaussian Noise $\boldsymbol{Z}$ (AWGN). Channel response and noise were generated using random normal function "randn()" from Numpy library. This model also uses perfect CSI in the receiver side, making it fairly compared with the baseline model. The used non-linear activation function is PReLU (He *et al*, 2015) instead of ReLU. One of the advantages of using PReLU is the negative value input will still have output rather than zero. As the data flowing in the model has a range of -∞ to ∞, the PReLU properties is very beneficial for improving the model accuracy. Moreover, we have tried to use ReLU, activation proposed in the previous work, in this model. Unfortunately, the training and validation loss become very high due to zero gradient issue.

After parallel transmitted symbol is formed, the BatchNormalization layer in the end of transmitter model block will performs as a power constraint so that the power of transmitted signal does not exceed the standard power transmission. To obtain the standard power transmission, the hyperparamter of gamma was constrained by setting the maximum value of the maximum-norm constraint to be 0.78. This constraint only takes place on network parameters during optimization. Maximum norm constraint is a regularization used for enforcing the absolute upper bound of neurons' weight vector that eventually being constrained by the calculated gradient descent. Matrix multiplier and noise addiction layer were made using several lambda or custom layers.

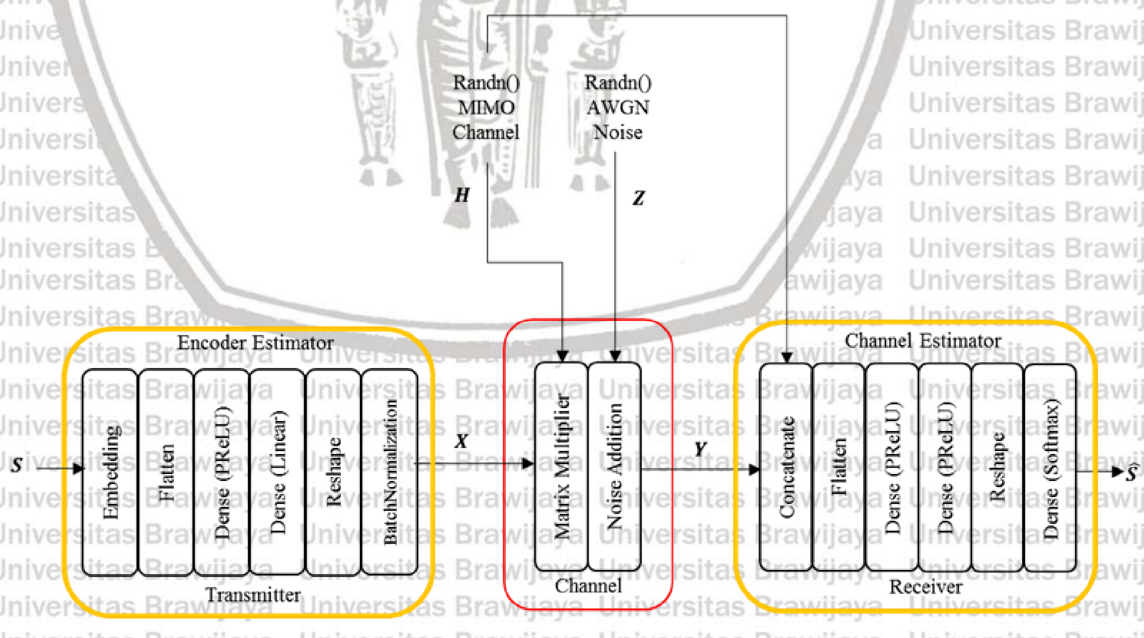
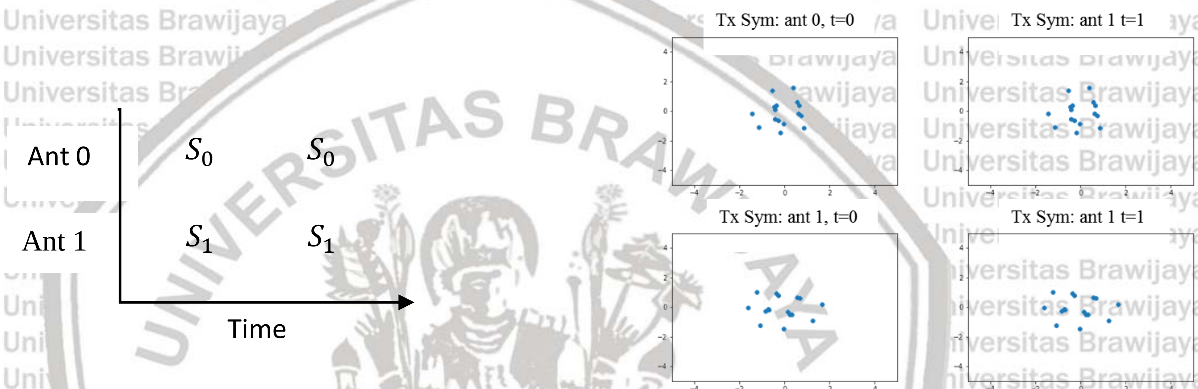


Figure 3-3: Spatial Diversity MIMO Autoencoder Model

Different from standard Alamouti, Symbol transmitted by this NN based model is shown by Figure 3-4 where symbol transmitted by each antenna is identical in every two time slots. Therefore, the equation of received signal becomes

$$y_0 = h_0 x_0 + h_1 x_1 + z_0$$
$$y_2 = h_0 x_0 + h_1 x_1 + z_1 \tag{3-1}$$

This transmission scheme is intended for maintaining diversity in transmission process.



(a) (b)

Figure 3-4: (a) Transmitted Symbol Scheme (b) Constellation Diagram of the Proposed Model

Actually, previous research also implements the aforementioned symbol transmission scheme, but if we observe from the constellation diagram shown by Figure 3-5 the symbols transmitted of each antenna every two time slots are not always identical. In order to give more complete information, Table 3-1 gives information about all of used NN in the proposed model including its parameters.

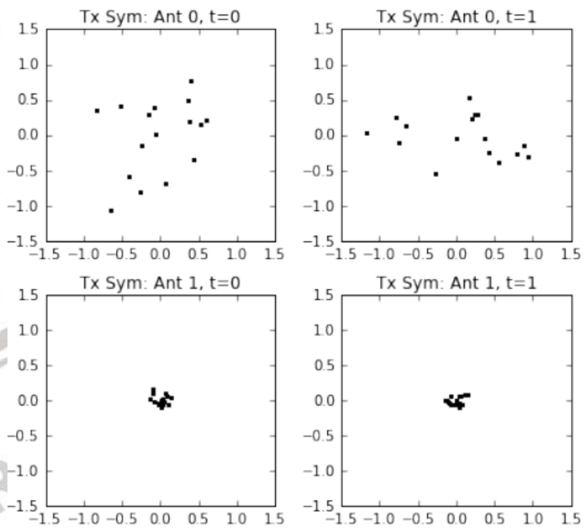


Figure 3-5: Constellation Diagram of the Previous Research

Table 3-1: Layout of all used NNs (2x1 Scheme)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,2,1
Embedding	128	None,2,1,8
Flatten	0	None,16
Dense + PReLU	272+16	None, 16
Dense (Linear)	68	None,4
Reshape	0	None,2,2,1
Normalization	4	None,2,2,1

Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	1664 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Reshape	0	None,2,16
Dense (softmax)	68	None,2,4

3.2.2.2 Channel Estimation

This research also extends the idea of the previous research to the channel estimation in spatial diversity MIMO communication. The method consists of two model, first is channel estimator model which results in pilot, and the second is data transmission model. Figure 3-6 shows the channel estimator model.

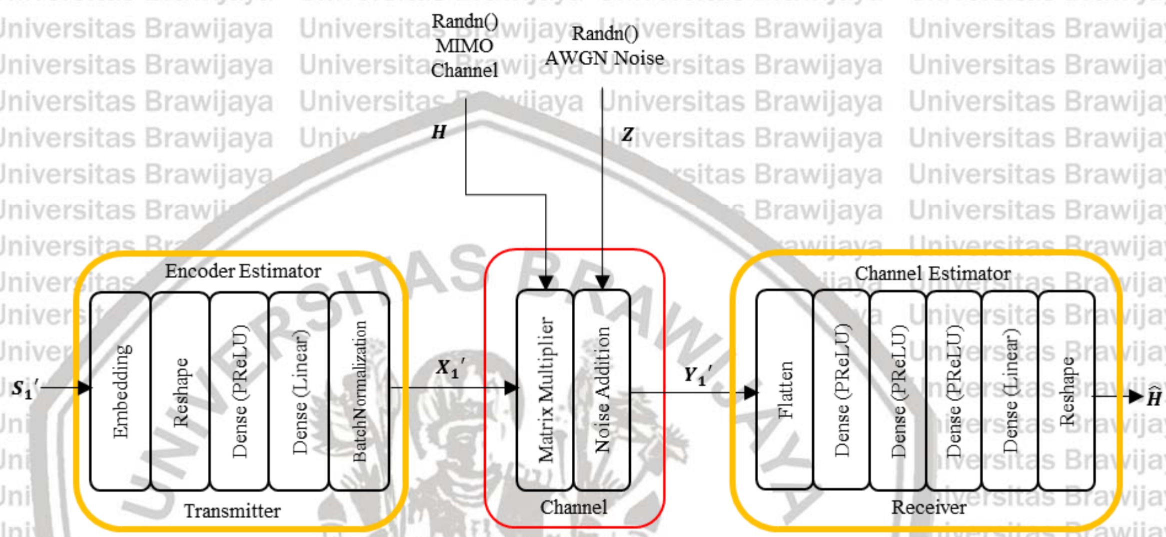


Figure 3-6: Model for Generating 1 Pilot (2x1 Scheme)

$\boldsymbol{S_1}'$ is a fixed 16 data stream that used for generating parallel transmit stream pilot $\boldsymbol{X_1}'$. Channel response $\boldsymbol{H}$ and noise $\boldsymbol{Z}$ were also generated using "randn()" function from Numpy library. The most important hyperparameter tuning among other parameters is that in BatchNormalization layer that we must set the maximum-norm constraint of the beta constraint and gamma constraint to be 0.05 and 0.9 respectively. If we desire to use more than one pilot, then we just need to add more encoder estimator model block to the system as depicted by Figure 3-7. However, the gamma constraint must be set differently in different scheme and will be deeply explained in the next chapter. After a good estimator model is obtained which is indicated by low training and validation loss, we then put the encoder and channel estimator model block to the data transmission model as a non-trainable layers as depicted by Figure 3-8. Table 3-2 shows the layout of all NN used for generating pilot and data transmission.

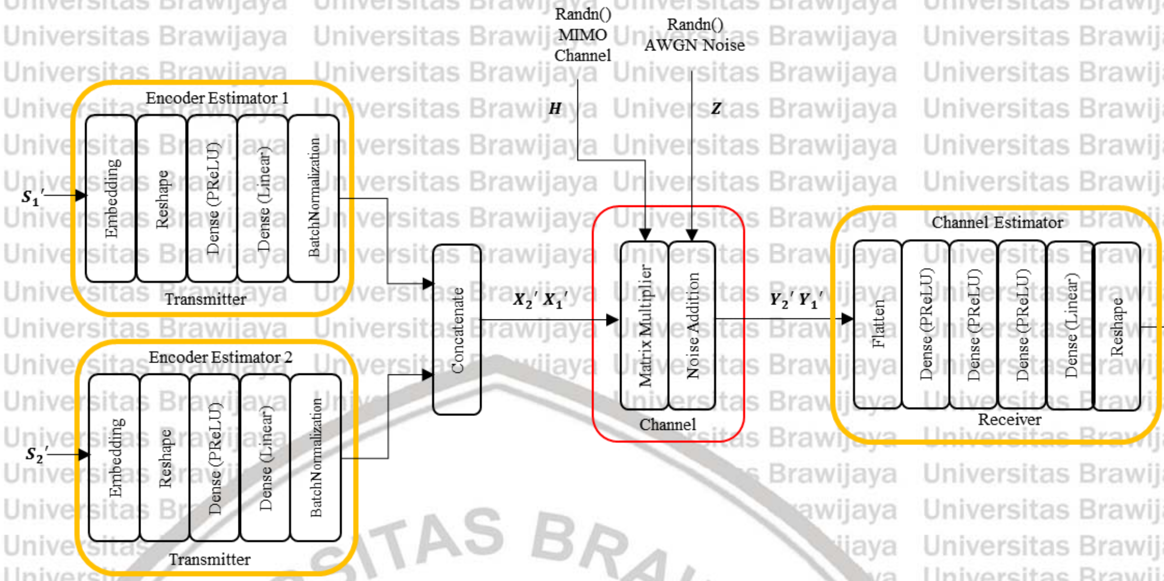


Figure 3-7: Model for Generating 2 Pilots (2x1 Scheme)

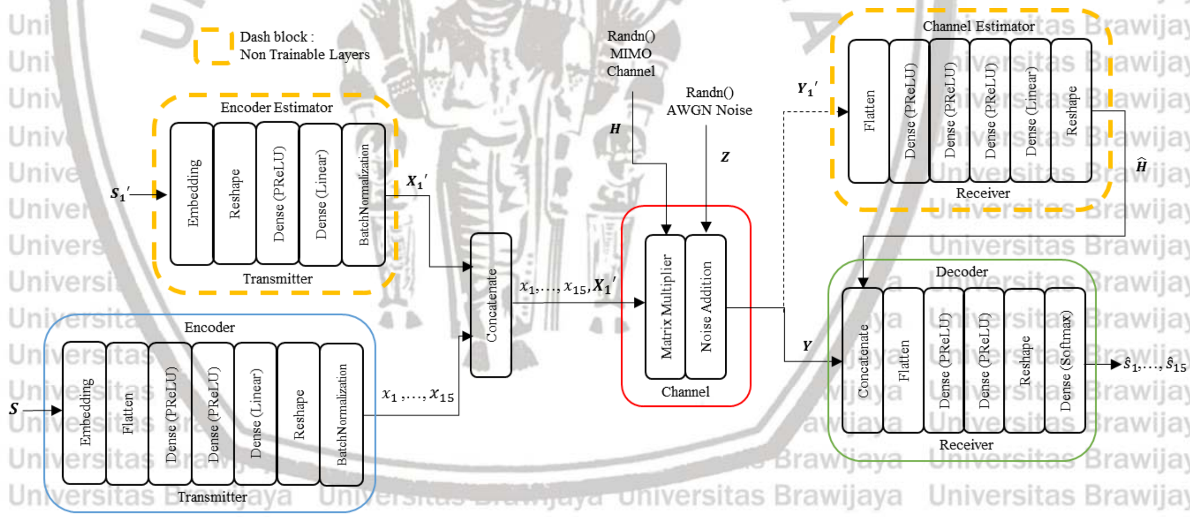


Figure 3-8: Data Transmission Model (2x1 Scheme)

Table 3-2: Layout of all used NNs (Channel Estimation 2x1 Scheme), (a) Channel Estimator Model (1 Pilot), (b) Data Transmission Model

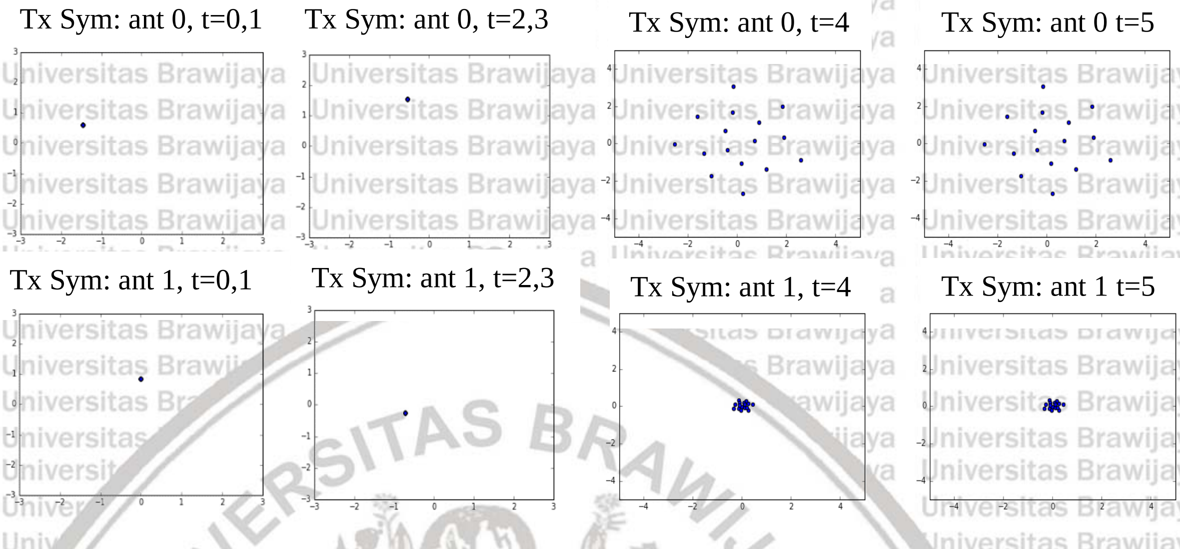
(a)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,1
Embedding	128	None,1,8
Reshape	0	None,2,2,2
Dense + PReLU	48 + 64	None,2,2,16
Dense (Linear)	17	None,2,2,1
Normalization	4	None,2,2,1
Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	640 + 128	None,128
Dense + PReLU	16512 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Dense (Linear)	264	None,8
Reshape	0	None,2,2,2

(b)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,2,1
Embedding	128	None,2,1,8
Flatten	0	None,16
Dense + PReLU	2176+128	None,128
Dense + PReLU	2064+16	None, 16
Dense (Linear)	68	None,4
Reshape	0	None,2,2,1
Normalization	4	None,2,2,1
Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	1664 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Reshape	0	None,2,16
Dense (softmax)	68	None,2,4

While constellation diagram of pilot and the symbol transmitted for data transmission are shown by Figure 3-9.



(a) (b)

Figure 3-9: Constellation Diagram of the Proposed Model for Spatial Diversity Channel Estimation, (a) pilot, (b) Symbols from Data Transmission Model

3.3 Spatial Multiplexing Model

2x2 spatial multiplexing MIMO communication with knowledge of CSIR is a new topic that has not been addressed yet in the previous research. This section is also divided into two part, data detection and channel estimator model. Overall, how the models working are just similar to the proposed model in the spatial diversity case.

3.3.1 Data Detection with Perfect CSIR

Figure 3-10 shows the model of data detection in 2x2 spatial multiplexing MIMO communication with perfect CSIR. $\boldsymbol{S}$ 16 combination of data stream that used for generating parallel transmit stream pilot $\boldsymbol{X}, \mathbb{R}^{2x2x1}$. Channel response $\boldsymbol{H}$ and noise $\boldsymbol{Z}$ were also generated using “randn()” function from Numpy library The difference of this model to the spatial diversity communication model are the location of the reshape layer in the end of the transmitter model block, the hyparparameter of BatchNormalization layer was set to 1 for the max norm of gamma constraint. Constellation diagram of the model transmitted symbols is shown by Figure 3-11 and the layout of all used NN is depicted in Table 3-3

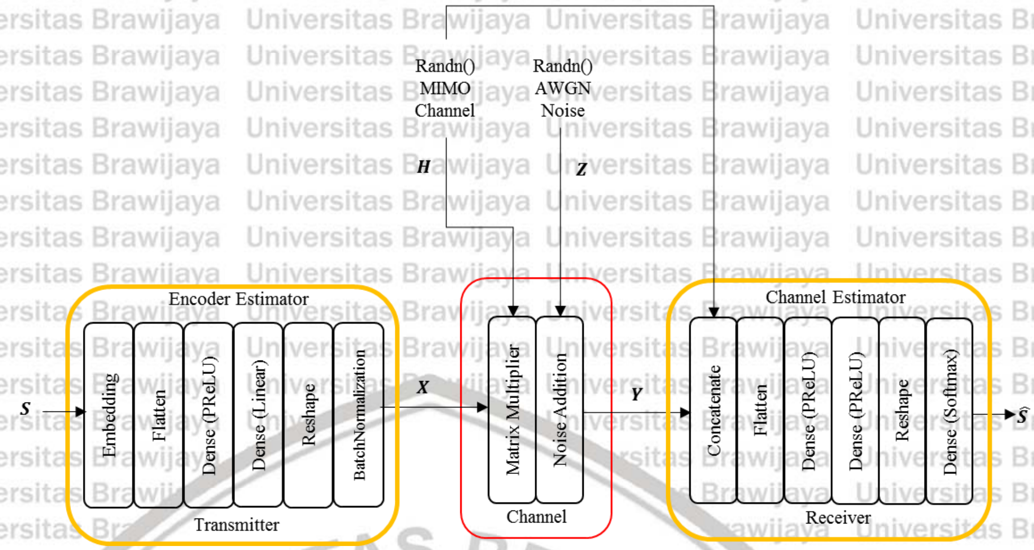


Figure 3-10: Spatial Multiplexing MIMO Autoencoder

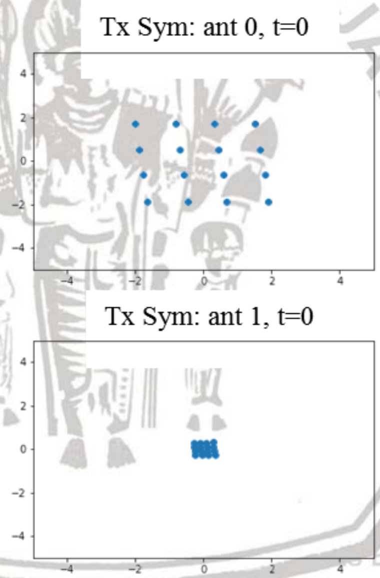


Figure 3-11: Constellation Diagram of Transmitted Symbols of 2x2 NN Based Model

Table 3-3: Layout of all used NNs (2x2 Scheme)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,2,1
Embedding	128	None,2,1,8
Flatten	0	None,16
Dense + PReLU	272+16	None, 16
Reshape	0	None,2,2,1
Dense (Linear)	5	None,2,2,1
Normalization	4	None,2,2,1
Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	1664 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Reshape	0	None,2,16
Dense (softmax)	68	None,2,4

3.3.2 Channel Estimation

The idea of data detection with perfect CSIR is also extended to the channel estimation case. As mentioned before, the main idea of channel estimation in the spatial multiplexing MIMO communication is just similar with the previous channel estimation model in the spatial diversity MIMO communication. Beside of the hyperparameter tuning in the BatchNormalization layer, the other differences are the depth of layer where in this case, only one dense layer with PReLU activation function is required in the transmitter model block and batch size in training stage that will be clearly discussed in the next chapter. Figure 3-12 shows the model for generating pilot in spatial multiplexing MIMO communication. Similar with the spatial diversity case, if we want to add more pilots to the communication scheme, then we just

need to add more encoder estimator model block to the network as shown by Figure 3-13.

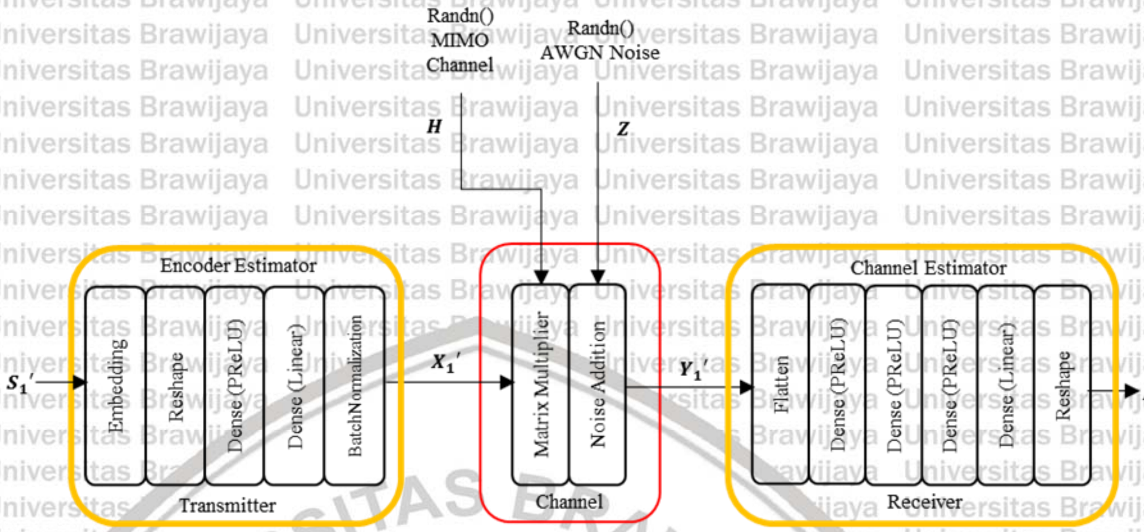


Figure 3-12: Model for Generating 1 Pilot (2x2 Scheme)

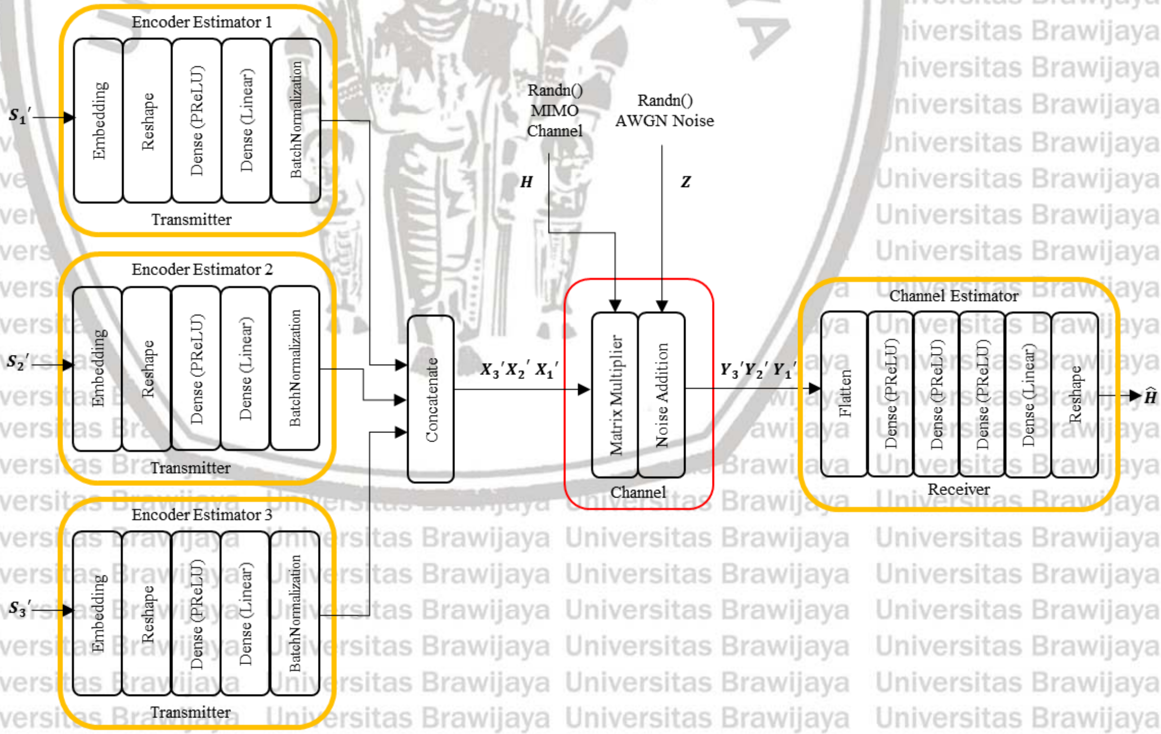


Figure 3-13 Model for Generating 3 Pilots (2x2 Scheme)

After good estimator model is obtained, then we fix the model as a non-trainable model and put it as a part of data transmission model which is depicted in Figure 3-14. Table 3-4 gives information about layout of all NN used in the estimator model and data transmission model respectively.

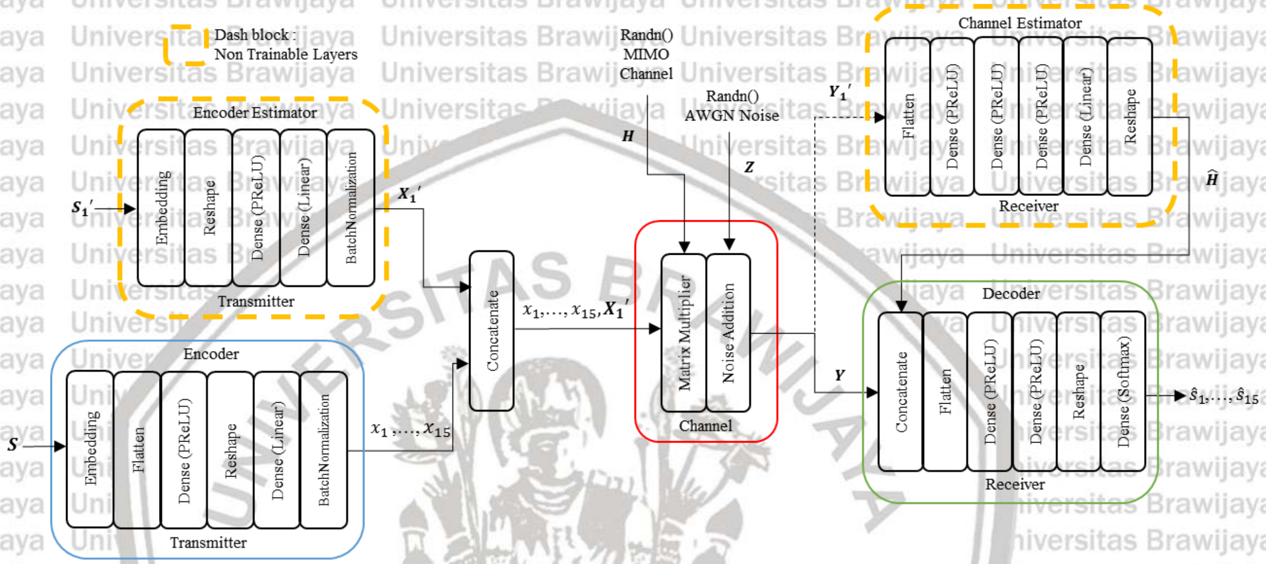


Figure 3-14: Data Transmission Model (2x2 Scheme)

Table 3-4: Layout of all used NNs (Channel Estimation 2x2 Scheme), (a) Channel Estimator Model (1 Pilot), (b) Data Transmission Model

(a)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,2,1
Embedding	128	None,2,1,8
Flatten	0	None,16
Dense + PReLU	272+16	None, 16
Reshape	0	None,2,2,1
Dense (Linear)	5	None,2,2,1
Normalization	4	None,2,2,1
Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	1664 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Reshape	0	None,2,16
Dense (softmax)	68	None,2,4

(b)

Encoder Estimator :	Parameters	Output Dimension
Input	0	None,1
Embedding	128	None,1,8
Reshape	0	None,2,2,2
Dense + PReLU	48 + 64	None,2,2,16
Dense (Linear)	17	None,2,2,1
Normalization	4	None,2,2,1
Estimator :	Parameters	Output Dimension
Flatten	0	None,4
Dense + PReLU	640 + 128	None,128
Dense + PReLU	16512 + 128	None,128
Dense + PReLU	4128 + 32	None,32
Dense (Linear)	264	None,8
Reshape	0	None,2,2,2

Chapter 4
Result and Discussion

This chapter discusses about the result of the proposed method in term of Bit Error Rate (BER) over a range of Signal to Noise Ratio (SNR) and several hyperparameters tuning to obtain the mentioned result. The proposed methods which implement deep learning method were fairly compared with the baseline or conventional methods. All of the models were trained using Adam optimizer (Kingma *et al*, 2014) with learning rate of 0.01 and sparse categorical cross-entropy and logcosh loss function for data detection and channel estimation task (only for pilot model) respectively. All of the obtained result in deep learning field were obtained from simulation using Keras with tensorflow backend (Abadi *et al*, 2016), while the baseline results were obtained through simulation using Matlab. As a reminder, in data detection task, the channel response and noise are fluctuate changed every data transmission. On the other hand, in channel estimation task, the channel responses are identical in every 16 data transmission while the noise are varied in every data transmission.

4.1 Spatial Diversity MIMO Communication

4.1.1 Data Detection with Perfect CSIR

In spatial diversity MIMO communication, the end-to-end learning based model was compared with the standard Alamouti system (Alamouti, 1998) over 1000000 bits. The proposed model was trained with millions of data (4000000 bits) and batch size of 500 data over 50 epochs. The NN based model was also trained in a fixed $E_b/N_0 = 21\text{dB}$. We set the hyperparameter in BatchNormalization layer, gamma constraint, to be max_norm(max_value=0.78). Figure 4-1 shows the performance of the NN based model compared to standard Alamouti scheme. From Figure 4-1, over the range of SNR, the NN based model shows promising result by outperform the standard Alamouti performance. Moreover, as SNR becomes higher, the gap performance between the proposed model and the baseline model also becomes bigger.

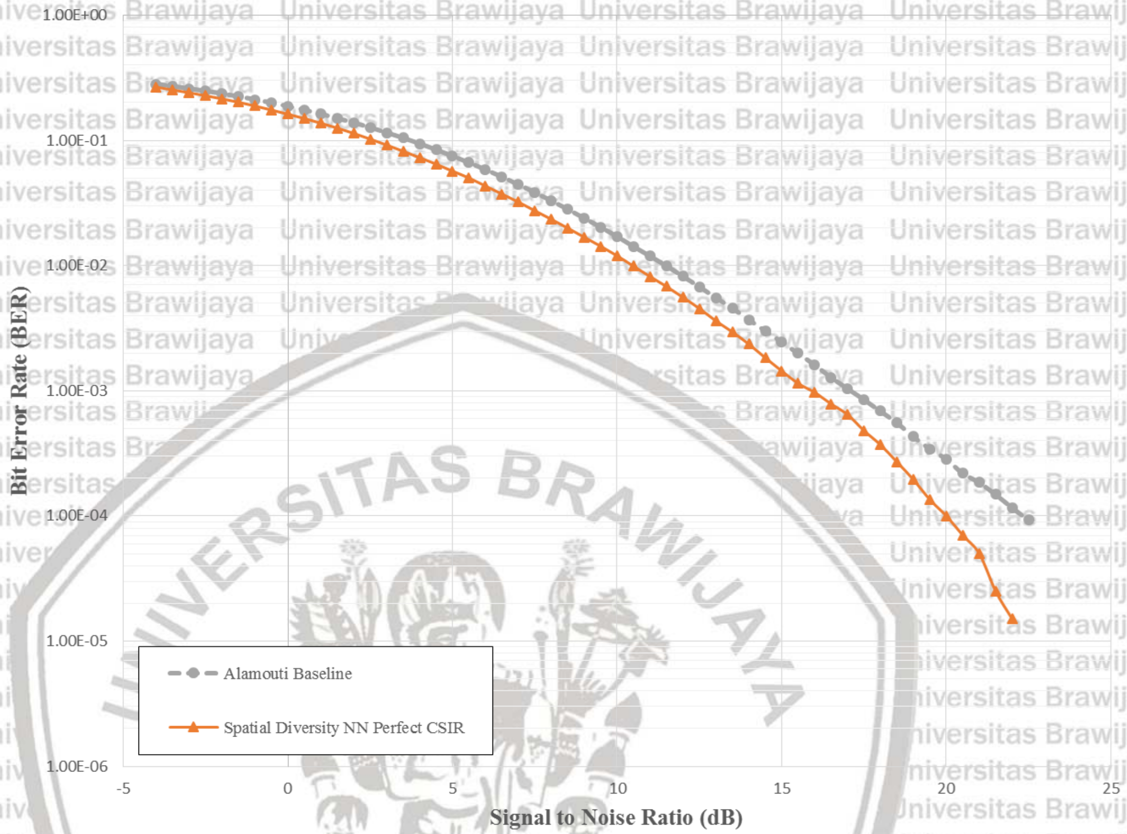


Figure 4-1: Bit Error Rate Performance of Learned Diversity Scheme (Perfect CSI)

We also tried to compare the proposed model with the previous model with the assumption that the reshape layer position, batch size and epoch are just identical with the proposed model, the model use perfect CSIR. Moreover, the number of neurons was assumed to be identical to the proposed model except with the last dense layer in the decoder block model as we only used one dense layer in the decoder. Figure 4-2 shows the performance of the aforementioned case. The result shows that the performance of the system become worst that the deep learning based model performance cannot outperform the baseline model. It indicates that the depth of layer and PReLU activation function has significant impact of the model accuracy.

4.1.2 Channel Estimation

In the case of channel estimation, we have generated five transmission scheme that are system using 1 pilot, 2 pilots, 3 pilots, 4 pilots and perfect CSIR. All models were trained by 4000000 bits and tested by 1000000 bits. Training process in each data transmission model is different to each other because of the difference in transmission scheme, but the value of E_b/N_0 are identical (21 dB). Table 4-1 shows the difference of hyperparameter value among each models.

Figure 4-2 shows the result of the NN based channel estimation model in term of BER over a range of SNR. The result shows that the increase of pilot number will improve the system performance. Moreover, the proposed models which use imperfect CSI show an outstanding performance as this model is able to outperform the baseline model which perfectly knows the CSI in the receiver side after transmitting at least 3 pilots.

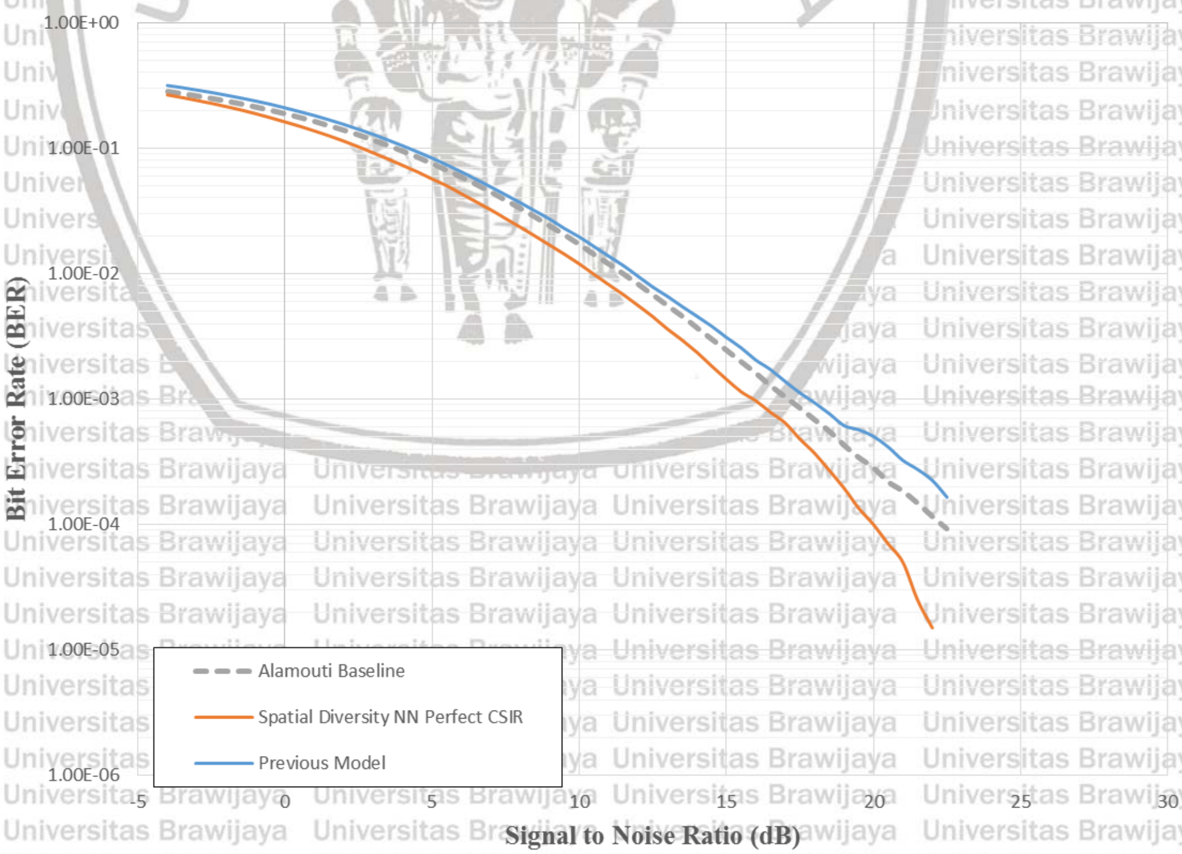


Figure 4-2: Bit Error Rate Performance of Learned Diversity Scheme Compared with Previous Model

Scheme	Gamma Constraint	Batch Size
1 Pilot	max_norm(max_value=1)	800
2 Pilots	max_norm(max_value=1)	400
3 Pilots	max_norm(max_value=0.9)	500
4 Pilots	max_norm(max_value=0.9)	250
Perfect CSIR	max_norm(max_value=0.8)	2000

Table 4-1: Hyperparameters Tuning for Channel Estimation (2x1 Scheme)

4.2 Spatial Multiplexing MIMO Communication

4.2.1 Data Detection with Perfect CSIR

Identical with the previous case in section 4.1.1, this model also trained using 4000000 bits input data and tested by 1000000 data bits. However, the hyperparameters were set differently. Batch size were set to 2000 data over 50 epochs and the value of E_b/N_0 are 22 dB. We set the maximum normalization of the gamma constraint in BatchNormalization layer to have maximum value equal to 1. Table 4-2 shows the comparison of hyperparameter value between spatial diversity model and spatial multiplexing model.

Scheme	Gamma Constraint	Batch Size	E_b/N_0
Spatial Diversity	max_norm(max_value=0.78)	500	21 dB
Spatial Multiplexing	max_norm(max_value=1)	2000	22 dB

Table 4-2: Comparison of Hyperparameter Tuning between 2 Different Schemes

Figure 4-3 shows the performance of end-to-end training based model compared to the conventional spatial multiplexing scheme using Maximum Likelihood (ML) detector. From Figure 4-3, it is clear the NN based model performs much better than the traditional method. In order to obtain this result, batch size take a significant impact to the system.

Difference batch size will cause another hyperparameters tuning, and if the hyperparameters are not well tuned then the performance will not good and sometimes result in uncommon constellation diagram. Position of the reshape layer also have a significant impact to the BER result and determine the shape of constellation diagram as well.

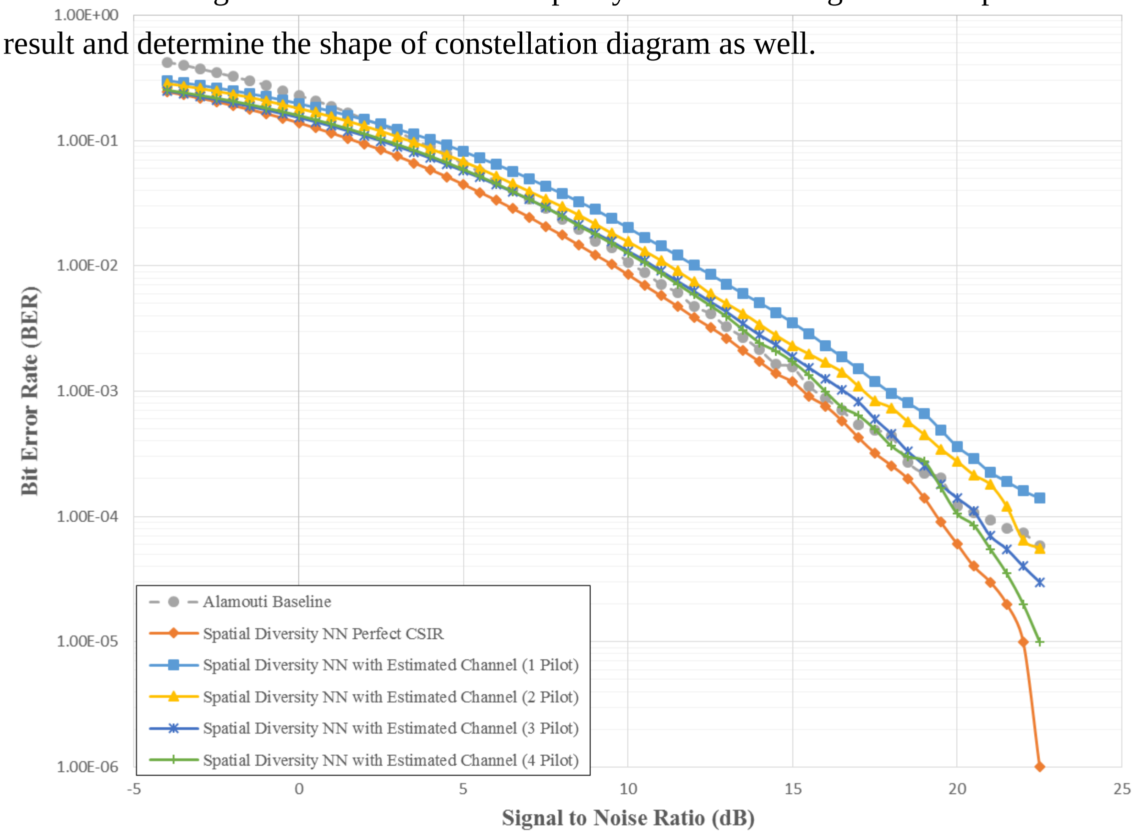


Figure 4-3: Bit Error Rate Performance of Learned Diversity Scheme (Channel Estimation)

4.2.2 Channel Estimation

In the case of channel estimation, we have generated five transmission scheme that are system using 1 pilot, 2 pilots, 3 pilots, 4 pilots and perfect CSIR. Different from the previous case in section 4.1.2 where training process in each data transmission model is different to each other, in the case of spatial multiplexing all of the schemes' batch size were set to 2000 data over 50 epochs, the value of E_b/N_0 are 22 dB and we set the maximum-norm constraint of the beta constraint and gamma constraint in BatchNormalization layer to be 0.05 and 0.9 respectively.

Figure 4-4 and Figure 4-5 shows the result of the NN based channel estimation model in term of BER over a range of SNR. The result shows that the increase of pilot number will improve the system performance. Moreover, the proposed model which use imperfect CSI shows a promising performance as these models outperform the baseline model which perfectly knows the CSI in the receiver side even by only transmitting 1 pilot.

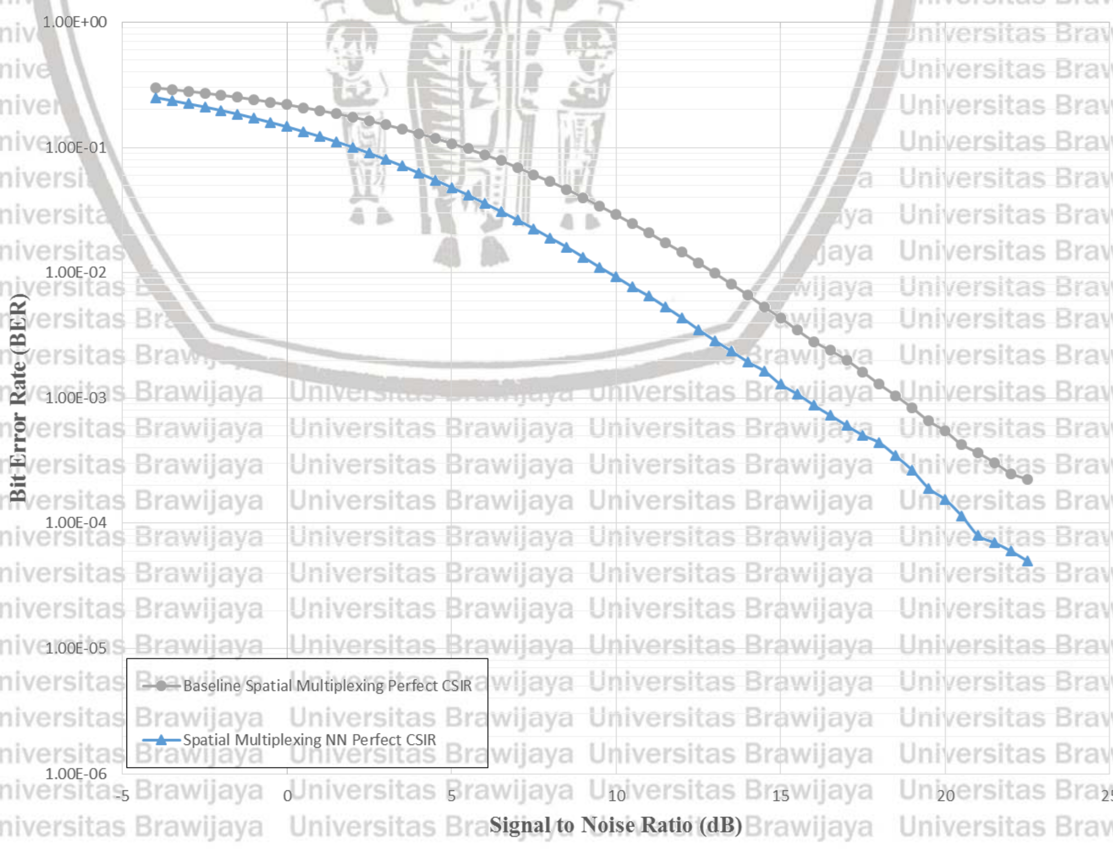


Figure 4-4: Bit Error Rate Performance of Learned 2x2 Scheme (Perfect CSI)

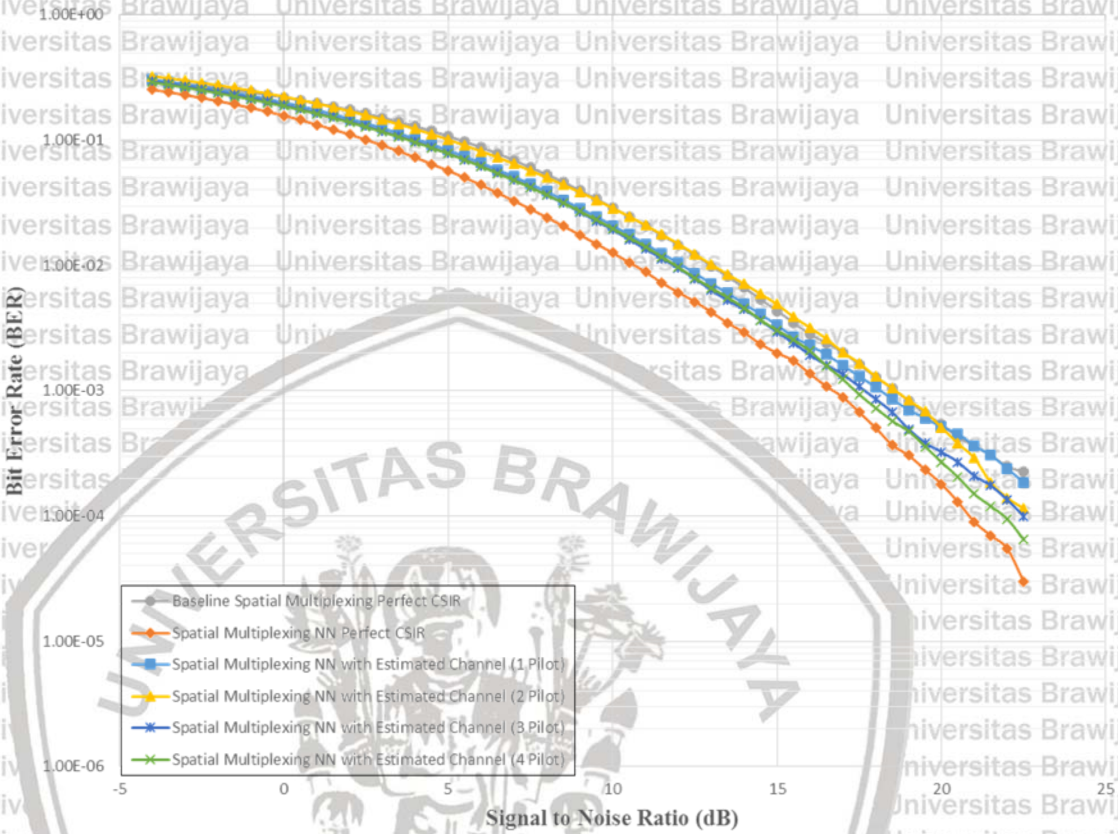


Figure 4-5: Bit Error Rate Performance of Learned 2x2 Scheme (Channel Estimation)

Chapter 5
Conclusion

Trade-off phenomenon between system performance and computational complexity always become the biggest consideration in developing performance of MIMO communication. Based on that problem, this thesis proposes deep learning based methods for optimizing the performance in both spatial diversity and spatial multiplexing MIMO communication. This research proposes solutions from deep learning field because it has been proven to research very well in several domain especially image. Moreover, computational complexity is only suffered in training stage. Once we obtain the well trained weights, we just need to load them and pass the data for testing stage.

There are four different models in this research which each two of them handle data detection and channel estimation task. Those models are fairly compared to the baseline methods. Every hyperparameter of each model was differently tuned in order to obtain the best result, especially in BatchNormalization layer and batch size for training the models. The obtained results show that NN based methods show promising performance by outperforming the baseline performance in a predetermined range of SNR (-4 dB until 22.5 dB). In perfect CSIR (Channel State Information in Receiver side) case, the proposed models achieve BER nearly 10^{-5} at SNR 22.5 dB. While in channel estimation case, the proposed models can exceed the baseline performance even by only transmitting 2 or 3 pilots.

These promising results were obtained due to appropriate hyperparameters tuning that eventually result in promising model accuracy. We believe that the obtained result can be improved by doing several hyperparameter tunings and/or even by building a new model with different algorithm.

References

Oestges, C., & Clerckx, B. (2010). *MIMO wireless communications: from real-world propagation to space-time code design*. Academic Press.

Biglieri, E., Calderbank, R., Constantinides, A., Goldsmith, A., Paulraj, A., & Poor, H. V. (2007). *MIMO wireless communications*. Cambridge university press.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, *1*(4), 541-551.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672-2680).

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural computation, 18*(7), 1527-1554.

O'Shea, T., & Hoydis, J. (2017). An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking*, *3*(4), 563-575.

Dörner, S., Cammerer, S., Hoydis, J., & ten Brink, S. (2018). Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing*, *12*(1), 132-143.

Baldi, P. (2012, June). Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning* (pp. 37-49).

Gruber, T., Cammerer, S., Hoydis, J., & ten Brink, S. (2017, March). On deep learning-based channel decoding. In *Information Sciences and Systems (CISS), 2017 51st Annual Conference on* (pp. 1-6). IEEE.

Ye, H., Li, G. Y., & Juang, B. H. (2018). Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters*, *7*(1), 114-117.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). Cambridge: MIT press.

Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026-1034).

Duan, K., Keerthi, S. S., Chu, W., Shevade, S. K., & Poo, A. N. (2003, June). Multi-category classification by soft-max combination of binary classifiers. In *International Workshop on Multiple Classifier Systems* (pp. 125-134). Springer, Berlin, Heidelberg.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*(Jul), 2121-2159.

Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, *4*(2), 26-31.

Neuneier, R., & Zimmermann, H. G. (1998). How to train neural networks. In *Neural networks: tricks of the trade* (pp. 373-423). Springer, Berlin, Heidelberg.

Yong, S. C., Jaekwon, K., Won, Y. Y., & Chung, G. K. (2010). MIMO-OFDM wireless communications with MATLAB. In *Singapore: John Wiley & Sons (Asia) Pte Ltd* (p. 544).

Alamouti, S. M. (1998). A simple transmit diversity technique for wireless communications. *IEEE Journal on selected areas in communications*, *16*(8), 1451-1458.

Chollet, F. (2015). Keras. https://github.com/fchollet/keras, 2015

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016, November). Tensorflow: a system for large-scale machine learning. In *OSDI* (Vol. 16, pp. 265-283).

Ng, A. (2012). Introduction to Machine Learning. *Machine learning Course, Coursera based into Standford University. org Backpropagation Lecture, https://www.coursera.org/learn/machine-learning/lecture/1z9WW/backpropagation-algorithm.*

Fausett, L. V. (1994). *Fundamentals of neural networks: architectures, algorithms, and applications* (Vol. 3). Englewood Cliffs: Prentice-Hall.

O'Shea, T. J., Erpek, T., & Clancy, T. C. (2017). Deep learning based MIMO communications. *arXiv preprint arXiv:1707.07980*.