Indoor Positioning with Feedforward Neural Network

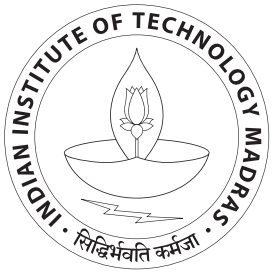
A Project Report

submitted by

PRAGNAYA VENKATA DHARANIKOTA

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

June 2018

THESIS CERTIFICATE

This is to certify that the thesis entitled **Indoor Positioning with Feedforward Neural Network**, submitted by **Pragnaya Venkata Dharanikota**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bonafide record of the research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Dr. Arun Pachai Kannu
Research Guide
Associate Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 16.06.2018

ACKNOWLEDGEMENTS

I would like to express a deep sense of gratitude to my esteemed guide Prof. Dr. Arun Pachai Kannu for giving me an opportunity to work under him. Without his expert guidance and constant support, I would not have reached this far in the project. His valuable and insightful suggestions helped me to narrow down the problem and focus on what is required. I am truly inspired by the patience that he had shown in listening to all my queries and answering them. I take this opportunity to sincerely thank him for permitting me to access the Virgo Super Cluster which helped in expediting the execution of project.

I would like to acknowledge all my friends who lent their helping hand in the project. Finally, I express a whole hearted gratitude to my parents and my brother for their constant encouragement, love, wishes and support.

ABSTRACT

KEYWORDS: Indoor Positioning, Received Signal Strength Indicator, Access Points, Feed Forward Neural Network

Indoor Positioning is a technique that helps us to get the knowledge about the location of a person or an object in indoor environments. This project is dedicated towards the implementation of one such Indoor Positioning technique which makes the use of Received Signal Strength Indicator values from different Access Points (Wi-Fi routers) and a Feed Forward Neural Network to predict the location of a person. As part of the project, the location of the person in one room and in three adjacent rooms is predicted. The error in the prediction is plotted against various positions of the Access Points. Considering any one of these positions of the Access Point, the variation in the error is observed for different number of layers and for different number of neurons per layer in the neural network, for different number of indoor locations, for different standard deviations of the shadowing noise etc. Finally, necessary conclusions are derived from the plots to see what can be an optimal prediction when Indoor Positioning is implemented in bigger indoor environments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IP	Indoor Positioning
IPS	Indoor Positioning System
MU	Mobile User
RSS	Received Signal Strength
FNN	Feedforward Neural Network
AP	Access Point
ADE	Average Distance Error
SD	Standard Deviation

CHAPTER 1

Introduction

1.1 Indoor Positioning and its applications

Indoor Positioning (IP) is a technique used for tracking the objects or the people in indoor environments with the help of radio waves, magnetic fields, acoustic signals and other sensory information collected by the mobile devices. The existing Global Positioning System (GPS) technology is generally not suitable to establish the indoor locations because the microwaves coming from the satellite get attenuated and scattered by roofs, walls and other objects. Therefore, Indoor Positioning System (IPS) has become an important technology to rely on indoors. With the help of IPS, a position can accurately be identified inside an enclosed building to within a few feet. It gained tremendous popularity in the recent times because of its myriad applications. Some of its applications include navigation in big indoor environments (shopping malls, hospitals, airports, university campuses and museusms), asset tracking, augmented/virtual reality, rescue operations, autonomous robot navigation, warehouse/-workforce monitoring and proximity marketing.

1.2 Different technologies used for Indoor Positioning

RFID (Radio Frequency Identification) is an emerging Indoor Positioning technology that allows for the mobile tracking of objects or people. It contains transponders (attached to objects/people) and a reader (Infsoft Locator Node). In a passive RFID system, the locator node functions as a power source and transfers radio frequency energy to the transponder at short distance (remote coupling). ID and data from the transponder are then captured by the locator node and forwarded to the Infsoft

LocAware Platform where the data is processed. As RFID offers a limited range of less than a meter, it is not suitable for area wide positioning but rather useful for providing selective object identification and location. It is easily maintainable and cost effective. All these features make localization through RFID particularly suitable for tracking in industrial environments (e.g. asset management).

Bluetooth is a good alternative technology for IP and indoor navigation. In client based applications, positioning happens via an app on the smartphones. The bluetooth beacons send out unique signals with which the app determines the position by means of fingerprinting. Bluetooth beacons are not capable of receiving the signals. They are relatively cheap. They can run on button cells up to two years and have a maximum range of 30 meters indoors. Accuracy is up to one meter.

With the booming development of mobile devices, commercial off-the-shelf (COTS) smartphones are equipped with various built-in inertial measurement unit (IMU) sensors (accelerometers,gyroscope and compass). This facilitated the Pedestrian Dead Reckoning (PDR) approach on smart phones. PDR approach estimates the current position of user based on the previous position, the current walking length and the direction of user. Although it is accurate in short range, the bottleneck of PDR is the drifting problem with walking distance.

Wi-Fi based IPS is recognized as the primary alternative to GPS for indoor localization because Wi-Fi network infrastructures are widely available in indoor and nearly every COTS smartphone is Wi-Fi enabled. There are three basic methods to compute the location of a Mobile User (MU) in indoor envrionment (a)Time of Arrival (ToA)/Time Difference of Arrival (TDoA) (b) Angle of Arrival (AoA) and (c) Received Signal Strength (RSS). ToA/TDoA is based on the measurement of the time flight of the signal. The implementation of this technique requires clock synchronization between the Wireless Access Points (WAP). AoA is also based on the time of flight of electromagnetic waves. It requires very accurate timing measurements and thus, the implementation of it requires additional hardware. RSS uses

pattern matching algorithms used to compute the location.

RSS fingerprinting is one of the most frequently used techniques for indoor positioning which consists of two phases. First phase is called the offline phase in which the RSS from all the APs is recorded at number of locations in the building i.e., generating a location matrix with the fingerprint of all the APs at each specific location. The next phase is the online phase where pattern matching algorithms are used to solve the location matrix and calculate the location of the MU by comparing the RSS being observed by the MU with the one recorded in the location matrix.

The method used in the current implementation involves a Feedforward Neural Network (FNN) which uses optimization algorithm and back propagation technique to estimate the location of the MU.

1.3 Merits and Demerits of a Neural Network used for Indoor Positioning

When there is a noise in the indoor environment, it causes an error in the measurement of RSS. This affects the accuracy of the positioning performed by the FNN. On the other side, in a noiseless environment, a FNN for IPS is credited for the following:

1. Powerful learning and adaptive capabilities
2. Ability to map the non linearity between RSS signals and the position coordinates of an object
3. Parallel distributed processing
4. Data fusion
5. Multivariable structure
6. Less time and low cost for laying out the location system and

7. Less storage cost for radio map establishment.

CHAPTER 2

Data representation for Indoor Positioning

2.1 When there is a single room

2.1.1 Setup environment

Consider a cubical room with one of its edges as 10 units. There are no restrictions in selecting the shape of the room and the room of any shape is perfectly suitable for carrying down the simulation. It is just that the cube has uniform dimensions and perfect symmetry in all the directions that led us to narrow down to this shape. The cubical room with different faces of it marked from 1 to 6 is shown below.

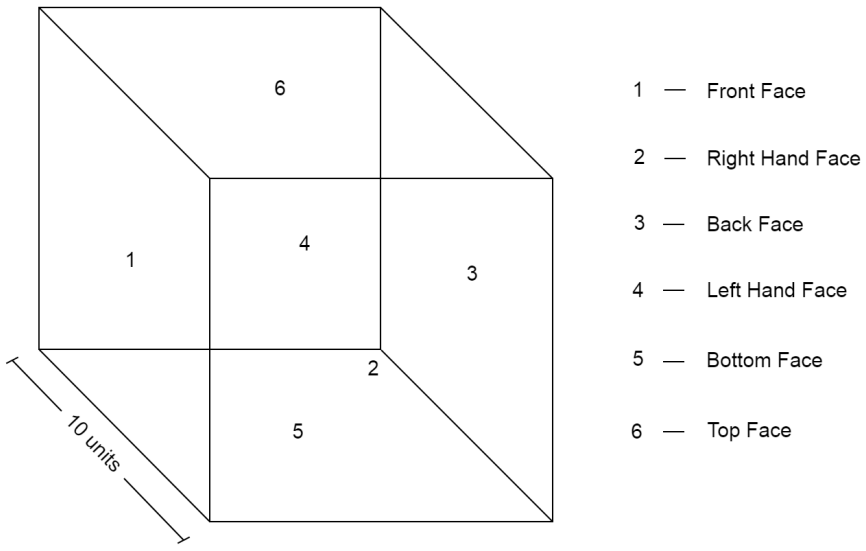


Figure 2.1: A 10 unit cubical room with its numbered faces

An Access Point (AP) which is nothing but a Wi-Fi router is a device that keeps transmitting radio waves from its antenna when it is in ON state. These APs are placed at different positions inside the room. Based on the accuracy of the tracking, the number of routers deployed and the positions at which they are deployed can be varied. The specifications of all the APs that are used in the setup are

Power of the transmitting antenna = 24dbm

Gain of the transmitting antenna = 6dbi (The AP is assumed to have a Yagi-uda antenna)

Frequency of the radio signal emitted from the antenna = 2.4 GHz

A Mobile User (MU) is the one who is inside the room and carries a device that is capable of receiving the radio signals transmitted by the AP. The specification of the device carried by the MU in the setup is

Gain of the receiving antenna in the device = 6dbi (The device with the MU is assumed to have an antenna which is similar to that of the transmitter)

At each location inside the room , the power received is different. This emphasizes the fact that location is one of the most important aspect for sensing the power variation which helps us get some useful information for indoor tracking. Therefore, for the better processing of the locations, each location is assigned an ordered triple from the real space (note that the values in the ordered triple range only from 0 to 10 as the dimensions of the room restrict so). Cartesian Coordinate System is employed for assigning the ordered triples to all the interested locations inside the room. The origin of the coordinate system is taken at one of the corners of the room, preferably at the intersection of 1st, 4th and 5th faces of the cube. X axis is taken along the intersection of first face and fifth face, Y axis is taken along the intersection of fourth and fifth face and Z axis is taken along the intersection of first face and fourth face.

A sample setup with APs placed on the edges of the three axes is shown below.

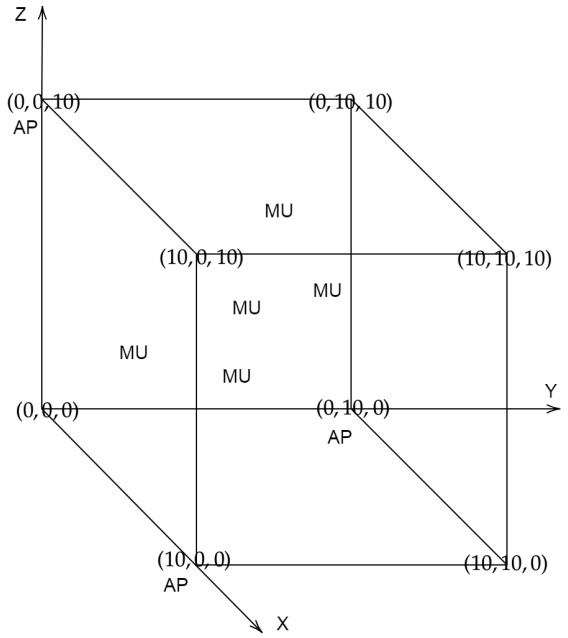


Figure 2.2: Setup for tracking the MU when there is a single room

Even though only five MU locations were shown in the setup, for the development of indoor positioning technique thousands of MU locations are required. All these MU locations are the points at which the power is received from different APs.

2.1.2 Structure of the output data

Different MU locations are randomly selected inside the room. The selection is in such a way that the MU locations spread over the entire room. Depending upon the dimensions of the room, number of MU locations selected can be varied i.e. if a larger room is considered, number of MU locations selected can be increased. A matrix is formed wherein each row corresponds to a 3-D MU location inside the room. We call this as the Coordinate Matrix and is denoted by C. This also forms our output data matrix. The first, second and third elements of each row in this matrix correspond to x, y and z coordinates of the MU location respectively.

A general Coordinate Matrix is shown below.

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

where N_{MUL} is the number of MU locations.

In the above matrix, the first row i.e. $[c_{11}\ c_{12}\ c_{13}]$ corresponds to the first MU location with c_{11} being its x coordinate, c_{12} being its y coordinate and c_{13} being its z coordinate, the second row i.e. $[c_{21}\ c_{22}\ c_{23}]$ corresponds to the second MU location with c_{21} being its x coordinate, c_{22} being its y coordinate and c_{23} being its z coordinate ... Eg: For the setup discussed in the above section, a sample Coordinate Matrix for 1000 MU locations is shown below.

$$C = \begin{bmatrix} 6.28858583 & 5.95822446 & 3.35333601 \\ 5.23703063 & 0.41524339 & 3.90588532 \\ 8.78011019 & 4.91266992 & 7.04965456 \\ \vdots & \vdots & \vdots \\ 4.12338296 & 4.05170427 & 7.51914268 \\ 6.98194044 & 2.14159756 & 3.69641588 \\ 5.16197382 & 7.77066492 & 2.57788852 \end{bmatrix}_{1000 \times 3}$$

Note that all the values in the above matrix lie only between 0 and 10.

2.1.3 Structure of the input data

A matrix is formed wherein each row corresponds to a 3-D point at which the AP is located inside the room. We call this as the Access Point Location matrix and is

denoted by APL. A general Access Point Location matrix matrix is shown below.

$$APL = \begin{bmatrix} ap_{1x} & ap_{1y} & ap_{1z} \\ ap_{2x} & ap_{2y} & ap_{2z} \\ ap_{3x} & ap_{3y} & ap_{3z} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{AP} \times 3}$$

where N_{AP} is the number of APs

In the above matrix, the first row i.e. $[ap_{1x}\ ap_{1y}\ ap_{1z}]$ corresponds to the location of the first AP with ap_{1x} being its x coordinate, ap_{1y} being its y coordinate and ap_{1z} being its z coordinate, the second row i.e. $[ap_{2x}\ ap_{2y}\ ap_{2z}]$ corresponds to the location of the second AP with ap_{2x} being its x coordinate, ap_{2y} being its y coordinate and ap_{2z} being its z coordinate ... Eg: For the sample setup shown in the section 2.1, the Access Point Location matrix is constructed as follows:

$$APL = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}_{3 \times 3}$$

From each MU location in the Coordinate Matrix, we calculate the Euclidean distance to all the AP locations in the Access Point Location matrix to get different distance vectors. These different distance vectors are stacked together to get a new matrix which we call as the Distance Matrix and is denoted by D. A generalized Distance Matrix is shown below.

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & \dots & \dots \\ d_{21} & d_{22} & d_{23} & \dots & \dots & \dots \\ d_{31} & d_{32} & p_{33} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{AP}}$$

In the above matrix, d_{11} corresponds to the distance between the first MU loca-

tion and the first AP, d_{12} corresponds to the distance between the first MU location and the second AP, d_{13} corresponds to the distance between the first MU location and the third AP ... . Similarly, d_{21} corresponds to the distance between the second MU location and the first AP, d_{22} corresponds to the distance between the second MUL and the second AP, d_{23} corresponds to the distance between the second MU location and the third AP ... . Eg: The distance matrix obtained for the above mentioned examples is as follows:

$$D = \begin{bmatrix} 7.77945347 & 8.1931144 & 10.91901519 \\ 6.1736735 & 11.59956849 & 8.04594001 \\ 8.67871456 & 12.35592537 & 10.48471263 \\ \vdots & \vdots & \vdots \\ 10.36766328 & 10.43657108 & 6.29072711 \\ 5.23054624 & 11.14295614 & 9.64723303 \\ 9.50974447 & 6.18558145 & 11.92128121 \end{bmatrix}_{1000\times 3}$$

$$P_r = P_t G_t G_r \left(\frac{\lambda}{4\pi d_0}\right)^2 \times \left(\frac{d_0}{d}\right)^{\eta}$$

where

- P_r is the received power at the location of the MU
- P_t is the transmitted power from the AP
- G_t is the gain of the transmitting antenna
- G_r is the gain of the receiving antenna
- λ is the wavelength of the radiation from the AP
- d_0 is the close in reference distance from the AP
- d is the distance between the MU location and the AP
- η is the path loss exponent for the selected indoor environment

The term $(\lambda/4\pi d_0)^2$ in the above equation indicates the path loss occured during the propagation. The close in reference distance d_0 is always chosen such that it is

in the far field region of the transmitting antenna and is smaller than any practical distance used in the system.

The above equation tells that for any distance d, there is a received power P_r that can be obtained. So it is obvious that for each distance d in the distance matrix, there is a received power p_r associated in some other matrix. This matrix is called as the Power Matrix and is denoted by P_r. A generalized power matrix is shown below.

$$P_r = \begin{bmatrix} p_{r11} & p_{r12} & p_{r13} & \cdots & \cdots & \cdots \\ p_{r21} & p_{r22} & p_{r23} & \cdots & \cdots & \cdots \\ p_{r31} & p_{r32} & p_{33} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{AP}}$$

In the above matrix, the first row constitutes the received powers from all the APs at the first MU location, the second row constitutes the received powers from all the APs at the second MU location ... . Putting it in a different way, any element p_{rmn} (where $1 \leq m \leq N_{sp}$ and $1 \leq n \leq N_r$) in the above matrix corresponds to the power received from the n^{th} AP at the m^{th} MUL. Eg:Assuming η = 1.8, the power matrix calculated for the corresponding distance matrix in the above example is as follows:

$$P_r = \begin{bmatrix} 1.10352768e-05 & 1.00527147e-05 & 5.99461566e-06 \\ 1.67306574e-05 & 5.37645840e-06 & 1.03861208e-05 \\ 9.06299753e-06 & 4.79861705e-06 & 6.44896223e-06 \\ \vdots & \vdots & \vdots \\ 6.58060767e-06 & 6.50260669e-06 & 1.61744703e-05 \\ 2.25479419e-05 & 5.77950750e-06 & 7.49146243e-06 \\ 7.68754489e-06 & 1.66727269e-05 & 5.11811820e-06 \end{bmatrix}_{1000 \times 3}$$

2.2 When there are three adjacent rooms

2.2.1 Setup environment

Consider three adjacent cubical rooms with each room having one of its edges as 10 units. The setup environment is almost similar to that of a single room. The only difference in the setup is that now the y coordinates range from 0 to 30 instead of 0 to 10 because of the adjaceny in the y direction. There is no restriction in selecting the adjacency of the rooms and the adjacency can be along any direction. It is just for the better visualisation that the adjaceny is taken along Y axis. A sample setup with APs placed at all the eight external corners of the room is shown below

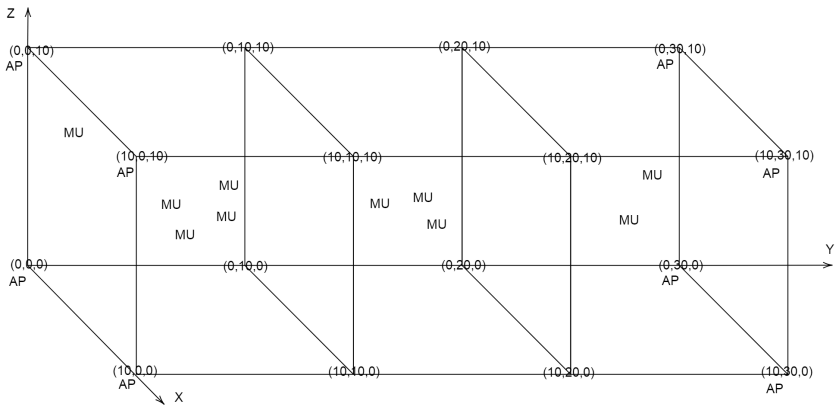


Figure 2.3: Setup for tracking the MU when there are three adjacent rooms.

2.2.2 Structure of the output data

The structure of the output matrix is similar to that of for a single room. The only difference is that now the range of the y coordinates is increased because of the adjacency whereas the x and z coordinates still lie between 0 and 10. Eg: For the setup discussed in the above section, a sample Coordinate Matrix for 1000 MU locations

is shown below.

$$C = \begin{bmatrix} 1.4719672 & 7.07248715 & 0.04105686 \\ 2.29014842 & 21.60984909 & 6.55266723 \\ 7.46336366 & 19.25884311 & 1.46398148 \\ \vdots & \vdots & \vdots \\ 1.77031805 & 5.13503939 & 9.15663144 \\ 6.54307613 & 5.34979062 & 0.85382079 \\ 1.72185728 & 17.20492605 & 0.43128565 \end{bmatrix}_{3000\times 3}$$

2.2.3 Shadowing and its effect on the signal strength

When there is an obstruction between a transmitter and a receiver, the signal at the receiver gets attenuated. This phenomenon is known as Shadowing. The obstruction can be anything like a wall, a building, a tree etc. The path loss term $(\lambda/4\pi d_0)^2$ used in the simplified path loss model in section 2.1.3 represents an average path loss. Empirical measurements show that the difference between the actual path loss and the average path loss is random in nature and exhibits log normal distribution with zero mean. A random variable is log normally distributed if the probability density function of the logarithm is Gaussian in nature. Therefore, the above simplified path loss model in shadowing gets modified to the following form:

$$P_{r(indB)} = 10log_{10}\left(P_t G_t G_r \left(\frac{\lambda}{4\pi d_0}\right)^2 \times \left(\frac{d_0}{d}\right)^{\eta}\right) - X_\sigma$$

where X_σ is log normally distributed random variable with mean 0 and standard deviation σ. A typical value of 7 dB is considered for the standard deviation in the log normal shadowing noise.

2.2.4 Structure of the input data

As there are three rooms separated by walls in between them, shadowing occurs when there is a propagation of the signal from one room to the other.The shadowing noise is to be properly accounted when the input data is formed. The structure of the input data when there are three adjacent rooms is almost similar to that of when there is a single room except for the accounting of the log normal shadowing noise. The accounting of the log normal shadowing and forming the input data is as follows:

1. Convert the received power at each location into dB with the help of the following expression.

$$10log_{10}\left(P_tG_tG_r\left(\frac{\lambda}{4\pi d_0}\right)^2 \times \left(\frac{d_0}{d}\right)^{\eta}\right)$$

2. For each AP, identify the number of walls obstructing the direct propagation of the signal from it to a location.

3. If there is only one wall, reduce the received power corresponding to that location by one random variable. If there are two walls, reduce the received power corresponding to that location by two random variables. Similarly if there are n walls in between, the reduction has to happen by n random variables. It is very important to note that the reduction is happening only in the dB scale.

After properly accounting for the log normal shadowing noise, the powers which are in dB scale are converted back to the normal scale with the expression $10^{\left(\frac{x}{10}\right)}$

CHAPTER 3

Feedforward Neural Network for Indoor Positioning

3.1 Architecture of the Feedforward Neural Network

An Artificial Neural Network is a system made up of a number of simple and highly interconnected computing elements which process information by their dynamic state response to the external inputs. If the interconnection is directed from input to output and acyclic (meaning that there are no feedback connections or loops), then it is called a Feedforward Neural Network (FNN). The computing elements in the FNN are called nodes. These nodes are stacked to form columns which are called layers. The FNN has one input layer, one output layer and one or more hidden layers. The number of nodes in a layer is called the size of the layer. Each node in a layer (except the output layer) is connected to every other node in the next layer and these connections are assigned some values (real numbers) which are called Weights. There are some special nodes present at almost each and every layer except the output layer in the FNN. They are special in the sense that they are connected only to the nodes that are present in the immediate next layer of it but not to that of present in the previous layer of it. They are called bias nodes. The value of the bias nodes is always 1. The bias nodes are represented as $bn1$, $bn2$, $bn3$ ... which are stacked under input layer, first hidden layer, second hidden layer ... respectively. The entire data processing in the FNN takes place in the form of weight manipulations. The processing of data and the role of bias are discussed in detail in the following sections.

As mentioned in section 2.3, the generalized power matrix

$$P_r = \begin{bmatrix} p_{r11} & p_{r12} & p_{r13} & \cdots & \cdots & \cdots \\ p_{r21} & p_{r22} & p_{r23} & \cdots & \cdots & \cdots \\ p_{r31} & p_{r32} & p_{33} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{AP}}$$

Each element in the first row of the power matrix is fed to a neuron in the input layer of the FNN. Therefore, the size of the input layer in the FNN is equal to the number of columns in the power matrix room i.e. N_{AP}. Based on the computational circuitry and the accuracy of the tracking, the number of hidden layers and the size of each hidden layer can be varied. Say N_{H1}, N_{H2}, N_{H3} ... are the sizes of the first hidden layer, second hidden layer, third hidden layer ... respectively. After the size of the each hidden layer is decided, the weights are generated in the form of matrices. Let the first weight matrix which is in between the input layer and the first hidden layer be denoted as $W^{(1)}$, the second weight matrix which is in between the first hidden layer and the second hidden layer be denoted as $W^{(2)}$, the third weight matrix which is in between the second hidden layer and the third hidden layer be denoted as $W^{(3)}$ ... . The orders of the weight matrices $W^{(1)}$, $W^{(2)}$, $W^{(3)}$ ... are $N_{AP} \times N_{H1}$, $N_{H1} \times N_{H2}$, $N_{H2} \times N_{H3}$ ... .

The generalized first, second and third weight matrices are shown below.

$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} & \cdots & \cdots & \cdots \\ w_{12}^{(1)} & w_{22}^{(1)} & w_{32}^{(1)} & \cdots & \cdots & \cdots \\ w_{13}^{(1)} & w_{23}^{(1)} & w_{33}^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{AP} \times N_{H1}} \qquad W^{(2)} = \begin{bmatrix} w_{11}^{(2)} & w_{21}^{(2)} & \cdots & \cdots & \cdots \\ w_{12}^{(2)} & w_{22}^{(2)} & \cdots & \cdots & \cdots \\ w_{13}^{(2)} & w_{23}^{(2)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{H1} \times N_{H2}}$$

$$W^{(3)} = \begin{bmatrix} w_{11}^{(3)} & w_{21}^{(3)} & w_{31}^{(3)} & \dots & \dots & \dots \\ w_{12}^{(3)} & w_{22}^{(3)} & w_{32}^{(3)} & \dots & \dots & \dots \\ w_{13}^{(3)} & w_{23}^{(3)} & w_{33}^{(3)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{H2} \times N_{H3}}$$

The main purpose of the FNN is to take the input power vector, process it and generate the location of the MU inside the room. As the output is a location which is an ordered triple, it is obvious that the number of nodes in the output layer is equal to three. The first node is for determining the x coordinate, the second node is for determining the y coordinate and the third node is for determining the z coordinate of the MU location. The number of rows and the number of columns in the last weight matrix which is in between the last hidden layer and the output layer are equal to the size of the last hidden layer and the size of the output layer respectively.

The architecture of the FNN with one hidden layer and two hidden layers is shown in the figures 3.1 and 3.2 respectively

3.2 Forwarding operation

Forwarding operation is the process in the FNN by which an output is generated for a given input. Except the nodes in the input layer, all the other nodes participate in the forwarding operation.

The forwarding operation can be explained in three steps.

Step 1: Vector multiplication

Each node participating in the forward operation does a dot product between the

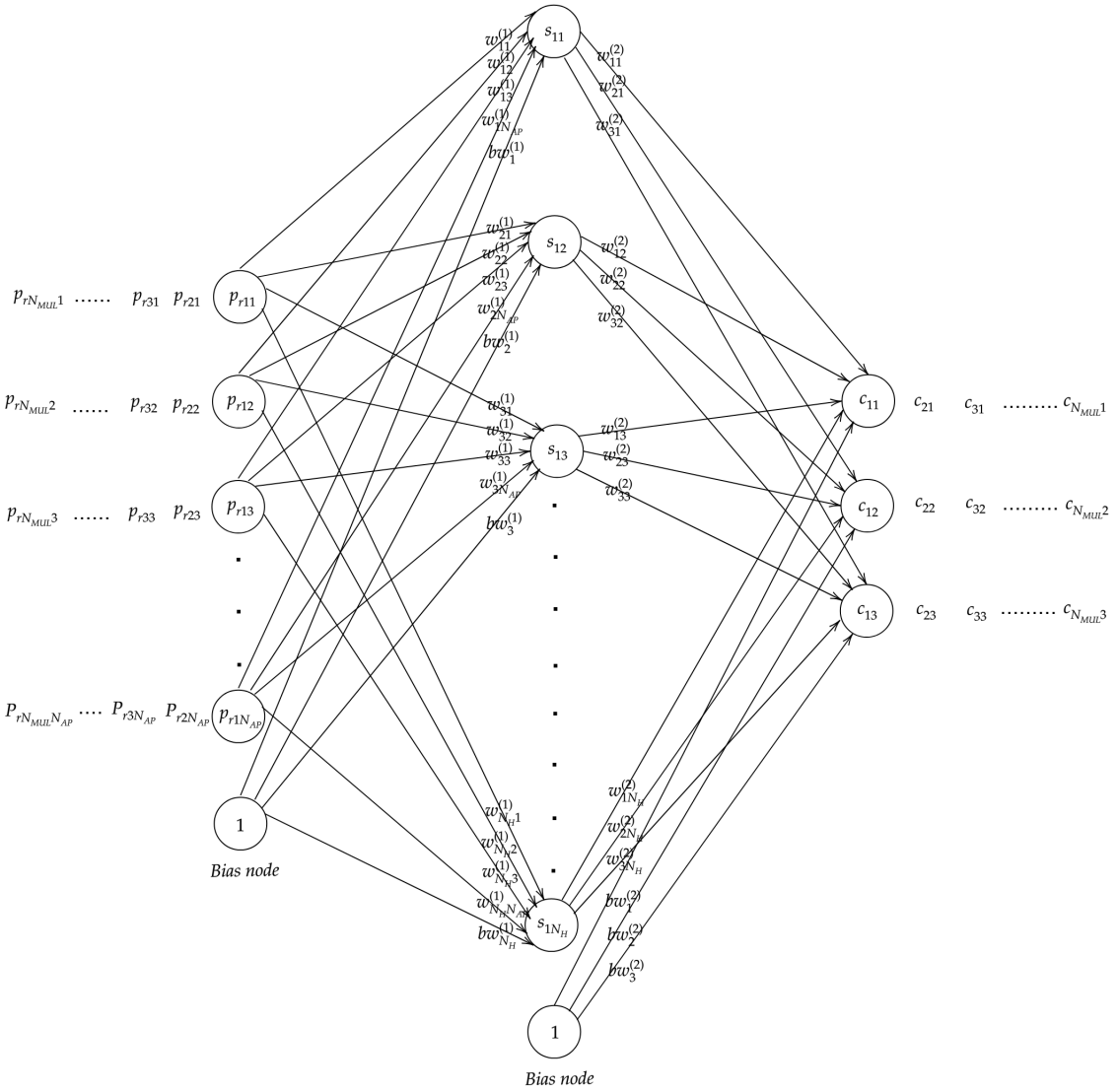


Figure 3.1: A Feedforward Neural Network with one hidden layer

nodes in the previous layer and a selected column from the corresponding weight matrix. The weight matrix and the column selected in the weight matrix for the dot product depends upon the position of the hidden layer and the position of the node in that hidden layer. For example, the first node in the first hidden layer performs a dot product between the input nodes and the first column of the first weight matrix, the second node in the first hidden layer performs a dot product between the input nodes and the second column of the first weight matrix, the third node in the first hidden layer performs a dot product between the input nodes and the third column of the first weight matrix, the first node in the second hidden layer performs a dot product

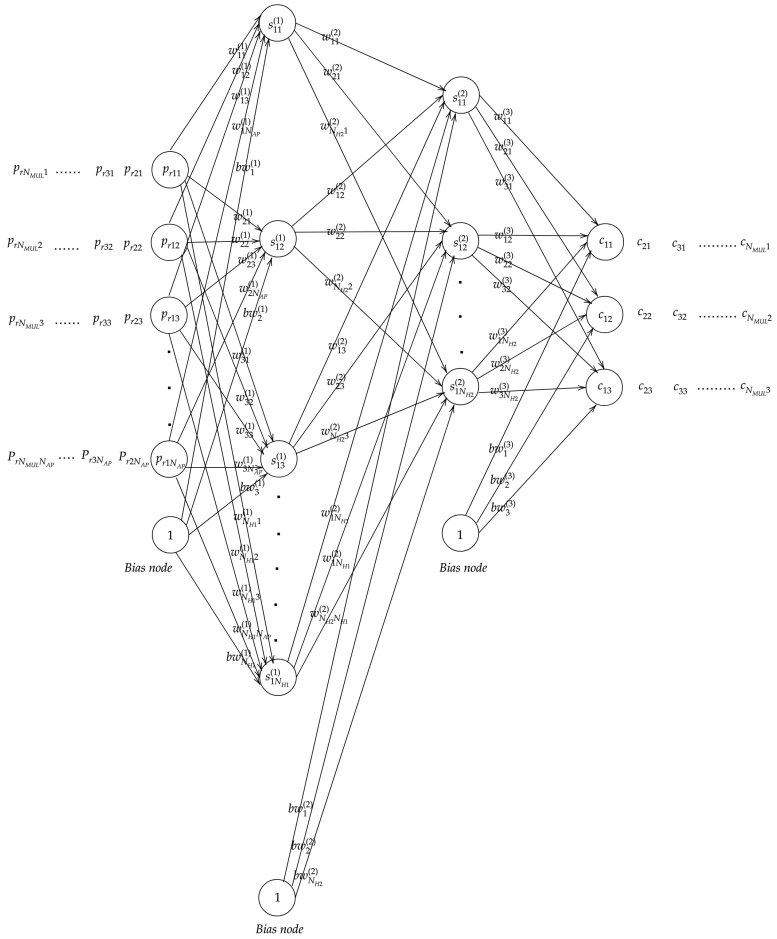


Figure 3.2: A Feedforward Neural Network with two hidden layers

between the nodes of the first hidden layer and the first column of the second weight matrix... .

The whole process of the predicting the location of a MU can be visualized as the surface fitting between the input power vector and the output coordinate matrix. The best fit is obtained when the standardized error (Mean Square Error) from the surface to all the points is minimum. Therefore for the best fit, it is required that the surface is oriented in the right direction. The vector multiplication exactly does this! It orients the surface in the best possible direction for minimum error.

Step 2: Bias addition

The resultant dot product obtained at each node is added with the bias value multiplied by the weight over the connection between the respective node and the bias node. For example, the dot product that is obtained at the third node in the fourth

hidden layer is added with the bias value that is multiplied by the weight that is there over the connection between the third node in the fourth hidden layer and the fourth bias. As the values of the bias nodes are always 1, the weights that are there over their connections directly get added to the calculated dot products.

Sometimes with just the orientation, the fitting may be incomplete as the surface also needs the right position among the data points. The addition of the weighted bias node value exactly does this! It helps to locate the curve in the right position. It is important here to emphasize that positioning the surface and orienting the surface cannot be two independent tasks. They are interdependent and they keep taking place simultaneously until the best fit is obtained.

Step 3: Transformation using an activation function

At each node in the FNN, the result obtained from the above two steps is transformed to generate a new value. The function through which the transformation takes place is called an activation function. For the present implementation of IP, sigmoid is used as the activation function. The mathematical form of the sigmoid function is shown below.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.1}$$

Activation functions are really important for a FNN to learn and make sense of something really complicated. They introduce non-linear properties to the network. If an activation function is not applied, then the output signal would simply be a simple linear function. A linear function is just a polynomial of one degree. It is easy to solve but it is limited in its complexity and has less power to learn complex functional mappings from the data. We want the FNN to not just learn and compute a linear function but something more complicated than that. The main reason for using sigmoid as the activation function is because it exists only between 0 and 1. It is useful for the current model wherein the data that is processed is also normalized between 0 and 1.

From the entire discussion in this section, it is clear that processing takes place

for each input power vector separately. But when it comes to the representation, we always show it in batch i.e in matrices. The operations like determining the error cost function, extracting the gradients, updating the weights can be very easily done with matrices... . Moreover, representation in matrices helps us to get a better overall perspective of what is happening inside the FNN. One such representation for a single hidden layer FNN is as follows.

The step 1 in the forwarding operation i.e vector multiplication between the input layer and the hidden layer is shown below.

$$\underbrace{\begin{bmatrix} p_{r11} & p_{r12} & p_{r13} & \dots & \dots & \dots \\ p_{r21} & p_{r22} & p_{r23} & \dots & \dots & \dots \\ p_{r31} & p_{r32} & p_{33} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{AP}}}_{P_r} \underbrace{\begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} & \dots & \dots & \dots \\ w_{12}^{(1)} & w_{22}^{(1)} & w_{32}^{(1)} & \dots & \dots & \dots \\ w_{13}^{(1)} & w_{23}^{(1)} & w_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{AP} \times N_H}}_{W^{(1)}}$$

$$= \underbrace{\begin{bmatrix} i_{11}^{(1)} & i_{12}^{(1)} & i_{13}^{(1)} & \dots & \dots & \dots \\ i_{21}^{(1)} & i_{22}^{(1)} & i_{23}^{(1)} & \dots & \dots & \dots \\ i_{31}^{(1)} & i_{32}^{(1)} & i_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H}}_{I^{(1)}}$$

In the above operation, note that each element $i_{mn}^{(1)}$ in the matrix $I^{(1)}$ denotes the dot product obtained at the nth node in the first hidden layer.

The step 2 in the forwarding operation i.e bias addition between the input layer and the hidden layer is shown below

$$\underbrace{\begin{bmatrix} i_{11}^{(1)} & i_{12}^{(1)} & i_{13}^{(1)} & \dots & \dots & \dots \\ i_{21}^{(1)} & i_{22}^{(1)} & i_{23}^{(1)} & \dots & \dots & \dots \\ i_{31}^{(1)} & i_{32}^{(1)} & i_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H}}_{I^{(1)}} + \underbrace{\begin{bmatrix} bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \dots & \dots & \dots \\ bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \dots & \dots & \dots \\ bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H}}_{BW^{(1)}}$$

$$= \underbrace{\begin{bmatrix} i_{11}^{(1)} + bw_1^{(1)} & i_{12}^{(1)} + bw_2^{(1)} & i_{13}^{(1)} + bw_3^{(1)} & \dots & \dots & \dots \\ i_{21}^{(1)} + bw_1^{(1)} & i_{22}^{(1)} + bw_2^{(1)} & i_{23}^{(1)} + bw_3^{(1)} & \dots & \dots & \dots \\ i_{31}^{(1)} + bw_1^{(1)} & i_{32}^{(1)} + bw_2^{(1)} & i_{33}^{(1)} + bw_3^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H}}_{I^{(1)}}$$

where $BW_1^{(1)}$ denotes the first Bias-Weight matrix which is nothing but a collection of the weighted bias node values between the input layer and the first hidden layer. The number of different weighted bias nodes are equal to the number of nodes in the first hidden layer. Note that each row in BW_1 is same which tells that the same weighted bias node values are used for all the MU locations

The step 3 in the forwarding operation i.e Transformation using an activation function between the input layer and the hidden layer is shown below

$$S^{(1)} = \begin{bmatrix} f(i_{11}^{(1)} + bw_1^{(1)}) & f(i_{12}^{(1)} + bw_2^{(1)}) & f(i_{13}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ f(i_{21}^{(1)} + bw_1^{(1)}) & f(i_{22}^{(1)} + bw_2^{(1)}) & f(i_{23}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ f(i_{31}^{(1)} + bw_1^{(1)}) & f(i_{32}^{(1)} + bw_2^{(1)}) & f(i_{33}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_h}$$

$$= \begin{bmatrix} s_{11}^{(1)} & s_{12}^{(1)} & s_{13}^{(1)} & \dots & \dots & \dots \\ s_{21}^{(1)} & s_{22}^{(1)} & s_{23}^{(1)} & \dots & \dots & \dots \\ s_{31}^{(1)} & s_{32}^{(1)} & s_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H}$$

where $f(.)$ is the sigmoid activation function.

The step 1 in the forwarding operation i.e vector multiplication between the hidden layer and the output layer is shown below

$$\begin{bmatrix} s_{11}^{(1)} & s_{12}^{(1)} & s_{13}^{(1)} & \dots & \dots & \dots \\ s_{21}^{(1)} & s_{22}^{(1)} & s_{23}^{(1)} & \dots & \dots & \dots \\ s_{31}^{(1)} & s_{32}^{(1)} & s_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_H} \begin{bmatrix} w_{11}^{(2)} & w_{21}^{(2)} & w_{31}^{(2)} \\ w_{12}^{(2)} & w_{22}^{(2)} & w_{32}^{(2)} \\ w_{13}^{(2)} & w_{23}^{(2)} & w^{(}2)_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_H \times 3} = \begin{bmatrix} i_{11}^{(2)} & i_{12}^{(2)} & i_{13}^{(2)} \\ i_{21}^{(2)} & i_{22}^{(2)} & i_{23}^{(2)} \\ i_{31}^{(2)} & i_{32}^{(2)} & i_{33}^{(2)} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$S^{(1)} \qquad\qquad W^{(2)} \qquad\qquad I^{(2)}$$

In the above operation, note that each element $i_{mn}^{(2)}$ in the matrix $I^{(2)}$ denotes the dot product obtained at the nth node in the output layer.

The step 2 in the forwarding operation i.e bias addition between the hidden layer and the output layer is shown below

$$\begin{bmatrix} i_{11}^{(2)} & i_{12}^{(2)} & i_{13}^{(2)} & \dots & \dots & \dots \\ i_{21}^{(2)} & i_{22}^{(2)} & i_{23}^{(2)} & \dots & \dots & \dots \\ i_{31}^{(2)} & i_{32}^{(2)} & i_{33}^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3} + \begin{bmatrix} bw_1^{(2)} & bw_2^{(2)} & bw_3^{(2)} \\ bw_1^{(2)} & bw_2^{(1)} & bw_3^{(2)} \\ bw_1^{(2)} & bw_2^{(2)} & bw_3^{(2)} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$I^{(2)} \qquad\qquad BW^{(2)}$$

$$= \begin{bmatrix} i_{11}^{(2)} + bw_1^{(2)} & i_{12}^{(2)} + bw_2^{(2)} & i_{13}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ i_{21}^{(2)} + bw_1^{(2)} & i_{22}^{(2)} + bw_2^{(2)} & i_{23}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ i_{31}^{(2)} + bw_1^{(2)} & i_{32}^{(2)} + bw_2^{(2)} & i_{33}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$I^{(2)}$$

where $BW^{(2)}$ denotes the second Bias-Weight matrix which is nothing but a collection of the weighted bias node values between the hidden layer and the output layer. The number of different weighted bias node values are equal to the number of nodes in the output layer. Note that each row in BW_1 is the same which tells that the same weighted bias node values are used for all the samples.

The step 3 in the forwarding operation i.e Transformation using an activation function is shown below

$$\hat{C} = \begin{bmatrix} f(i_{11}^{(2)} + bw_1) & f(i_{12}^{(2)} + bw_2) & f(i_{13}^{(2)} + bw_3) \\ f(i_{11}^{(2)} + bw_1) & f(i_{12}^{(2)} + bw_2) & f(i_{13}^{(2)} + bw_3) \\ f(i_{11}^{(2)} + bw_1) & f(i_{12}^{(2)} + bw_2) & f(i_{13}^{(2)} + bw_3) \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3} = \begin{bmatrix} \hat{c}_{11} & \hat{c}_{12} & \hat{c}_{13} \\ \hat{c}_{21} & \hat{c}_{22} & \hat{c}_{23} \\ \hat{c}_{31} & \hat{c}_{32} & \hat{c}_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

where $f(.)$ is a sigmoid activation function.

Similarly, the forwarding operation in a network with two hidden layers is shown below.

$$\begin{bmatrix} p_{r11} & p_{r12} & p_{r13} & \cdots & \cdots & \cdots \\ p_{r21} & p_{r22} & p_{r23} & \cdots & \cdots & \cdots \\ p_{r31} & p_{r32} & p_{33} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{AP}} \begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} & \cdots & \cdots & \cdots \\ w_{12}^{(1)} & w_{22}^{(1)} & w_{32}^{(1)} & \cdots & \cdots & \cdots \\ w_{13}^{(1)} & w_{23}^{(1)} & w_{33}^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{AP} \times N_{H1}}$$

$$P_r \qquad\qquad W^{(1)}$$

$$= \begin{bmatrix} i_{11}^{(1)} & i_{12}^{(1)} & i_{13}^{(1)} & \cdots & \cdots & \cdots \\ i_{21}^{(1)} & i_{22}^{(1)} & i_{23}^{(1)} & \cdots & \cdots & \cdots \\ i_{31}^{(1)} & i_{32}^{(1)} & i_{33}^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}}$$

$$I^{(1)}$$

$$\begin{bmatrix} i_{11}^{(1)} & i_{12}^{(1)} & i_{13}^{(1)} & \cdots & \cdots & \cdots \\ i_{21}^{(1)} & i_{22}^{(1)} & i_{23}^{(1)} & \cdots & \cdots & \cdots \\ i_{31}^{(1)} & i_{32}^{(1)} & i_{33}^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}} + \begin{bmatrix} bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \cdots & \cdots & \cdots \\ bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \cdots & \cdots & \cdots \\ bw_1^{(1)} & bw_2^{(1)} & bw_3^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}}$$

$$I^{(1)} \qquad\qquad BW_1^{(1)}$$

$$= \begin{bmatrix} i_{11}^{(1)} + bw_1^{(1)} & i_{12}^{(1)} + bw_2^{(1)} & i_{13}^{(1)} + bw_3^{(1)} & \cdots & \cdots & \cdots \\ i_{21}^{(1)} + bw_1^{(1)} & i_{22}^{(1)} + bw_2^{(1)} & i_{23}^{(1)} + bw_3^{(1)} & \cdots & \cdots & \cdots \\ i_{31}^{(1)} + bw_1^{(1)} & i_{32}^{(1)} + bw_2^{(1)} & i_{33}^{(1)} + bw_3^{(1)} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}}$$

$$I^{(1)}$$

$$S^{(1)} = \begin{bmatrix} f(i_{11}^{(1)} + bw_1^{(1)}) & f(i_{12}^{(1)} + bw_2^{(1)}) & f(i_{13}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ f(i_{21}^{(1)} + bw_1^{(1)}) & f(i_{22}^{(1)} + bw_2^{(1)}) & f(i_{23}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ f(i_{31}^{(1)} + bw_1^{(1)}) & f(i_{32}^{(1)} + bw_2^{(1)}) & f(i_{33}^{(1)} + bw_3^{(1)}) & \dots & \dots & \dots \\ \vdots & & \vdots & & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}}$$

$$= \begin{bmatrix} s_{11}^{(1)} & s_{12}^{(1)} & s_{13}^{(1)} & \dots & \dots & \dots \\ s_{21}^{(1)} & s_{22}^{(1)} & s_{23}^{(1)} & \dots & \dots & \dots \\ s_{31}^{(1)} & s_{32}^{(1)} & s_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}}$$

$$\begin{bmatrix} s_{11}^{(1)} & s_{12}^{(1)} & s_{13}^{(1)} & \dots & \dots & \dots \\ s_{21}^{(1)} & s_{22}^{(1)} & s_{23}^{(1)} & \dots & \dots & \dots \\ s_{31}^{(1)} & s_{32}^{(1)} & s_{33}^{(1)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H1}} \begin{bmatrix} w_{11}^{(2)} & w_{21}^{(2)} & w_{31}^{(2)} & \dots & \dots & \dots \\ w_{12}^{(2)} & w_{22}^{(2)} & w_{32}^{(2)} & \dots & \dots & \dots \\ w_{13}^{(2)} & w_{23}^{(2)} & w(2)_{33} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{H1} \times N_{H2}}$$

$S^{(1)}$ $W^{(2)}$

$$= \begin{bmatrix} i_{11}^{(2)} & i_{12}^{(2)} & i_{13}^{(2)} & \dots & \dots & \dots \\ i_{21}^{(2)} & i_{22}^{(2)} & i_{23}^{(2)} & \dots & \dots & \dots \\ i_{31}^{(2)} & i_{32}^{(2)} & i_{33}^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}}$$

$I^{(2)}$

$$\begin{bmatrix} i_{11}^{(2)} & i_{12}^{(2)} & i_{13}^{(2)} & \dots & \dots & \dots \\ i_{21}^{(2)} & i_{22}^{(2)} & i_{23}^{(2)} & \dots & \dots & \dots \\ i_{31}^{(2)} & i_{32}^{(2)} & i_{33}^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}} + \begin{bmatrix} bw_1^{(2)} & bw_2^{(2)} & bw_3^{(2)} & \dots & \dots & \dots \\ bw_1^{(2)} & bw_2^{(2)} & bw_3^{(2)} & \dots & \dots & \dots \\ bw_1^{(2)} & bw_2^{(2)} & bw_3^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}}$$

$$I^{(2)} \qquad\qquad BW_1^{(2)}$$

$$= \begin{bmatrix} i_{11}^{(2)} + bw_1^{(2)} & i_{12}^{(2)} + bw_2^{(2)} & i_{13}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ i_{21}^{(2)} + bw_1^{(2)} & i_{22}^{(2)} + bw_2^{(2)} & i_{23}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ i_{31}^{(2)} + bw_1^{(2)} & i_{32}^{(2)} + bw_2^{(2)} & i_{33}^{(2)} + bw_3^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}}$$

$$I^{(2)}$$

$$S^{(2)} = \begin{bmatrix} f(i_{11}^{(2)} + bw_1^{(2)}) & f(i_{12}^{(2)} + bw_2^{(2)}) & f(i_{13}^{(2)} + bw_3^{(2)}) & \dots & \dots & \dots \\ f(i_{21}^{(2)} + bw_1^{(2)}) & f(i_{22}^{(2)} + bw_2^{(2)}) & f(i_{23}^{(2)} + bw_3^{(2)}) & \dots & \dots & \dots \\ f(i_{31}^{(2)} + bw_1^{(2)}) & f(i_{32+bw_2^{(2)}}^{(2)}) & f(i_{33}^{(2)} + bw_3^{(2)}) & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}}$$

$$= \begin{bmatrix} s_{11}^{(2)} & s_{12}^{(2)} & s_{13}^{(2)} & \dots & \dots & \dots \\ s_{21}^{(2)} & s_{22}^{(2)} & s_{23}^{(2)} & \dots & \dots & \dots \\ s_{31}^{(2)} & s_{32}^{(2)} & s_{33}^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}}$$

$$\begin{bmatrix} s_{11}^{(2)} & s_{12}^{(2)} & s_{13}^{(2)} & \dots & \dots & \dots \\ s_{21}^{(2)} & s_{22}^{(2)} & s_{23}^{(2)} & \dots & \dots & \dots \\ s_{31}^{(2)} & s_{32}^{(2)} & s_{33}^{(2)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times N_{H2}} \begin{bmatrix} w_{11}^{(3)} & w_{21}^{(3)} & w_{31}^{(3)} \\ w_{12}^{(3)} & w_{22}^{(3)} & w_{32}^{(3)} \\ w_{13}^{(3)} & w_{23}^{(3)} & w(2)_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{H2} \times 3} = \begin{bmatrix} i_{11}^{(3)} & i_{12}^{(3)} & i_{13}^{(3)} \\ i_{21}^{(3)} & i_{22}^{(3)} & i_{23}^{(3)} \\ i_{31}^{(3)} & i_{32}^{(3)} & i_{33}^{(3)} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$S^{(2)} \qquad\qquad W^{(3)} \qquad\qquad I^{(3)}$$

$$\begin{bmatrix} i_{11}^{(3)} & i_{12}^{(3)} & i_{13}^{(3)} \\ i_{21}^{(3)} & i_{22}^{(3)} & i_{23}^{(3)} \\ i_{31}^{(3)} & i_{32}^{(3)} & i_{33}^{(3)} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3} + \begin{bmatrix} bw_1^{(3)} & bw_2^{(3)} & bw_3^{(3)} \\ bw_1^{(3)} & bw_2^{(3)} & bw_3^{(3)} \\ bw_1^{(3)} & bw_2^{(3)} & bw_3^{(3)} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$I^{(3)} \qquad\qquad BW^{(3)}$$

$$= \begin{bmatrix} i_{11}^{(3)} + bw_1^{(3)} & i_{12}^{(3)} + bw_2^{(3)} & i_{13}^{(3)} + bw_3^{(3)} & \dots & \dots & \dots \\ i_{21}^{(3)} + bw_1^{(3)} & i_{22}^{(3)} + bw_2^{(3)} & i_{23}^{(3)} + bw_3^{(3)} & \dots & \dots & \dots \\ i_{31}^{(3)} + bw_1^{(3)} & i_{32}^{(3)} + bw_2^{(3)} & i_{33}^{(3)} + bw_3^{(3)} & \dots & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

$$I^{(3)}$$

$$\hat{C} = \begin{bmatrix} f(i_{11}^{(3)}) & f(i_{12}^{(3)}) & f(i_{13}^{(3)}) \\ f(i_{21}^{(3)}) & f(i_{22}^{(3)}) & f(i_{23}^{(3)}) \\ f(i_{31}^{(3)}) & f(i_{32}^{(3)}) & f(i_{33}^{(3)}) \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3} = \begin{bmatrix} \hat{c}_{11} & \hat{c}_{12} & \hat{c}_{13} \\ \hat{c}_{21} & \hat{c}_{22} & \hat{c}_{23} \\ \hat{c}_{31} & \hat{c}_{32} & \hat{c}_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL} \times 3}$$

The two important points that can be inferred from this section are:

1. The processing happens with input power vectors sequentially i.e. after the first power vector is processed, the second power vector is processed, after the second power vector is processed, the third power vector is processed ...
2. The dimensions of the weight matrices do not depend on the number of the MU locations inside the room.

3.3 Error cost function

The predetermined Coordinate Matrix $C = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL}\times 3}$ and the output matrix that is generated from the neural network is $\hat{C} = \begin{bmatrix} \hat{c}_{11} & \hat{c}_{12} & \hat{c}_{13} \\ \hat{c}_{21} & \hat{c}_{22} & \hat{c}_{23} \\ \hat{c}_{31} & \hat{c}_{32} & \hat{c}_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL}\times 3}$

Let the difference of the two matrices be the Error Matrix denoted by E.

$$E = C - \hat{C} = \begin{bmatrix} c_{11} - \hat{c}_{11} & c_{12} - \hat{c}_{12} & c_{13} - \hat{c}_{13} \\ c_{21} - \hat{c}_{21} & c_{22} - \hat{c}_{22} & c_{23} - \hat{c}_{23} \\ c_{31} - \hat{c}_{31} & c_{32} - \hat{c}_{32} & c_{33} - \hat{c}_{33} \\ \vdots & \vdots & \vdots \end{bmatrix}_{N_{MUL}\times 3}$$

The cost function denoted by J is obtained by summing up all the squared elements of the error matrix and dividing it by the number of MU locations considered for processing (or the number of input power vectors processed) The equation of the cost function is as follows:

$$J = \frac{1}{3MUL_P} \sum_{i=1}^{N_{MUL_P}} \sum_{j=1}^{3} (c_{ij} - \hat{c}_{ij})^2$$

where MUL_P is the number of MU locations considered for processing ($MUL_P \leq MUL$)

c_{ij} belongs to the coordinate set that is predetermined as a part of the output data

$\hat{c}_{ij}$ belongs to the coordinate set that is generated from the neural network. One of the most common standards that is used for obtaining the error functions is Mean Square Error (MSE) and the same is used in the present implementation too.The error cost function as the name suggests is the measure of the total error occured during the indoor tracking process. It should be as mimimum as possible (ideally zero) for a better tracking.

3.4 Optimization algorithm

From the forwarding operation discussed above, it is to be inferred that the error cost function depends on both the weight matrices and the weighted bias matrices $BW^{(1)}$. The main objective of the AFNN is to minimize the cost function with respect to these matrices. When the cost function is plotted against the weight matrices and the weighted bias matrices, different 3-D surfaces are obtained. The minimization of the error cost function is done by descending to a point on each of these 3-D surfaces at which the gradients(slopes) with respect to weight matrices and weighted bias matrices become approximately zero. The descent of the point is not a one step process but rather happens through several iterations. In each iteration, we find the gradients(slopes) with respect to weight matrices and weighted bias matrices and reduce the old weights by the scaled versions of these differentials to get the new weights. With these new weights, the forwarding operation takes place again to generate a error cost function which in turn give new gradients. This process is repeated until the gradients tend to zero. The updation of the weights and bias weights take place through the following equations: For a single hidden layer AFNN,

$$W_{i+1}^{(2)} = W_i^{(2)} - step * \frac{dJ}{dW_i^{(2)}}$$

$$W_{i+1}^{(1)} = W_i^{(1)} - step * \frac{dJ}{dW_i^{(1)}}$$

$$BW_{i+1}^{(2)} = BW_i^{(2)} - step * \frac{dJ}{dBW_i^{(2)}}$$

$$BW_{i+1}^{(1)} = BW_i^{(1)} - step * \frac{dJ}{dBW_i^{(1)}}$$

where $W_{i+1}^{(1)}$, $W_{i+1}^{(2)}$ are the new weight matrices after the updation
$BW_{i+1}^{(2)}$, $BW_{i+1}^{(1)}$ are the new bias weight matrices after the updation
$dJ/dW_i^{(1)}$ and $dJ/dW_i^{(2)}$ are the weight differentials evaluated during the iteraion $dJ/dBW_i^{(1)}$ and $dJ/dBW_i^{(2)}$ are the bas weight differentials evaluated during the iteration. The step value is appropriately selected such that the cost function descends at a decent rate.

For a two hidden layer AFNN,

$$W_{i+1}^{(3)} = W_i^{(3)} - step * \frac{dJ}{dW_i^{(3)}}$$

$$W_{i+1}^{(2)} = W_i^{(2)} - step * \frac{dJ}{dW_i^{(2)}}$$

$$W_{i+1}^{(1)} = W_i^{(1)} - step * \frac{dJ}{dW_i^{(1)}}$$

$$BW_{i+1}^{(3)} = BW_i^{(3)} - step * \frac{dJ}{dBW_i^{(3)}}$$

$$BW_{i+1}^{(2)} = BW_i^{(2)} - step * \frac{dJ}{dBW_i^{(2)}}$$

$$BW_{i+1}^{(1)} = BW_i^{(1)} - step * \frac{dJ}{dBW_i^{(1)}}$$

As the computation of the gradient descents involve the entire data set(input and output matrices) which we will see in the next section, the optimization algorithm is called batch gradient descent algorithm.The batch gradient descent algorithm converges to the global minimum of the function if the function is convex. As the MSE cost function is always convex, it is guaranteed that a global minimum can be ob-

tained if enough training is performed.

3.5 Backpropagation

The most important thing in the entire implementation of the FNN is the evaluation of the gradients. For a single hidden layer FNN, the evaluation of gradients is explained through the following equations:

$$\begin{aligned}\frac{dJ}{dW^{(2)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dW^{(2)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dW^{(2)}} \\ &= -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(I^{(2)})}{dW^{(2)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}W^{(2)})}{dW^{(2)}} \\ &= (S^{(1)})^T - k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}} = (S^{(1)})^T - k(C-\hat{C})f'(I^{(2)}+BW^{(2)})\end{aligned}$$

$$\frac{dJ}{dBW^{(2)}} = -k(C-\hat{C})\frac{d\hat{C}}{dBW^{(2)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dBW^{(2)}} = -k(C-\hat{C})f'(I^{(2)}+BW^{(2)})$$

$$\begin{aligned}\frac{dJ}{dW^{(1)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dW^{(1)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dW^{(1)}} \\ &= -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(I^{(2)})}{dW^{(1)}} \\ &= -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}.W^{(2)})}{dW^{(1)}} \\ &= -k(C-\hat{C})fracdf(I^{(2)}+BW^{(2)})dI^{(2)}(W^{(2)})^T\frac{d(S^{(1)})}{dW^{(1)}} \\ &= -k(C-\hat{C})fracdf(I^{(2)}+BW^{(2)})dI^{(2)}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dW^{(1)}} \\ &= -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}}\frac{dI^{(1)}}{dW^{(1)}} \\ &= -k(C-\hat{C})\frac{df(I^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}}\frac{d(P_r.W^{(1)})}{dW^{(1)}} \\ &= (P_r)^T. - k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}} \\ &= (P_r)^T. - k(C-\hat{C})f'(I^{(2)}+BW^{(2)})(W^{(2)})^T.f'(I^{(1)}+BW^{(1)})\end{aligned}$$

$$\frac{dJ}{dBW^{(1)}} = -k(C-\hat{C})\frac{d\hat{C}}{dBW^{(1)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dBW^{(1)}}$$
$$= -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(I^{(2)})}{dBW^{(1)}} = -k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}.W^{(2)})}{dBW^{(1)}}$$
$$= -k(C-\hat{C})fracdf(I^{(2)}+BW^{(2)})dI^{(2)}(W^{(2)})^T\frac{d(S^{(1)})}{dBW^{(1)}}$$
$$= -k(C-\hat{C})fracdf(I^{(2)}+BW^{(2)})dI^{(2)}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dBW^{(1)}}$$
$$= -k(C-\hat{C})f'(I^{(2)}+BW^{(2)})(W^{(2)})^T f'(I^{(1)}+BW^{(1)})$$

In the above four equations, the term $-k(C-\hat{C})\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}$ is named as the error term. This error term is involved even in the evaluation of the the gradient $dJ/dW^{(1)}$ which can be thought as an error propagation from the last layer to the first layer. As the error seems propagated from back to the first layer, the method of evaluation is termed as backpropagation algorithm. The evaluation of gradients for an AFNN with two hidden layers is explained through the following equations:

$$\frac{dJ}{dW^{(3)}} = -k(C-\hat{C})\frac{d\hat{C}}{dW^{(3)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dW^{(3)}}$$
$$= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(I^{(3)})}{dW^{(3)}}$$
$$= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(S^{(2)}.W^{(3)})}{dW^{(3)}} = (S^{(2)})^T - k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}$$
$$= (S^{(2)})^T. - k(C-\hat{C})f'(I^{(3)}+BW^{(3)})$$

$$\frac{dJ}{dBW^{(3)}} = -k(C-\hat{C})\frac{d\hat{C}}{dBW^{(3)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dBW^{(3)}}$$
$$= -k(C-\hat{C})f'(I^{(3)}+BW^{(3)})$$

$$
\begin{aligned}
\frac{dJ}{dW^{(2)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dW^{(2)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(I^{(3)})}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(S^{(2)}.W^{(3)})}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{d(S^{(2)})}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{df(I^{(2)}+BW^{(2)})}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{dI^{(2)}}{dW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}.W^{(2)})}{dW^{(2)}} \\
&= (S^{(1)})^{T}.k-(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}} \\
&= (S^{(1)})^{T}.k-(C-\hat{C})f(I^{(3)}+BW^{(3)})(W^{(3)})^{T}f'(I^{(2)}+BW^{(2)})
\end{aligned}
$$

$$
\begin{aligned}
\frac{dJ}{dBW^{(2)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dBW^{(2)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dBW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(I^{(3)})}{dBW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(S^{(2)}.W^{(3)})}{dBW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{d(S^{(2)})}{dBW^{(2)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^{T}\frac{df(I^{(2)}+BW^{(2)})}{dBW^{(2)}} \\
&= -k(C-\hat{C})f(I^{(3)}+BW^{(3)})(W^{(3)})^{T}f'(I^{(2)}+BW^{(2)})
\end{aligned}
$$

$$
\begin{aligned}
\frac{dJ}{dW^{(1)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dW^{(1)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(I^{(3)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(S^{(2)}.W^{(3)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{d(S^{(2)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{dI^{(2)}}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}.W^{(2)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{dS^{(1)}}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}}\frac{dI^{(1)}}{dW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T \\
&\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}}\frac{d(P_r.W^{(1)})}{dW^{(1)}} \\
&= (P_r)^T. - k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dI^{(1)}} \\
&= (P_r)^T. - k(C-\hat{C})f'(I^{(3)}+BW^{(3)})(W^{(3)})^Tf'(I^{(2)}+BW^{(2)})(W^{(2)})^Tf'(I^{(1)}+BW^{(1)})
\end{aligned}
$$

$$
\begin{aligned}
\frac{dJ}{dBW^{(1)}} &= -k(C-\hat{C})\frac{d\hat{C}}{dBW^{(1)}} = -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(I^{(3)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}\frac{d(S^{(2)}.W^{(3)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{d(S^{(2)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{dI^{(2)}}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}\frac{d(S^{(1)}.W^{(2)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{dS^{(1)}}{dBW^{(1)}} \\
&= -k(C-\hat{C})\frac{df(I^{(3)}+BW^{(3)})}{dI^{(3)}}(W^{(3)})^T\frac{df(I^{(2)}+BW^{(2)})}{dI^{(2)}}(W^{(2)})^T\frac{df(I^{(1)}+BW^{(1)})}{dBW^{(1)}} \\
&= -k(C-\hat{C})f'(I^{(3)}+BW^{(3)})(W^{(3)})^T f'(I^{(2)}+BW^{(2)})(W^{(2)})^T f'(I^{(1)}+BW^{(1)})
\end{aligned}
$$

Note that the constant term k in all the above equations comes from the initial derivative of the error cost function and is equal to $k = \frac{2}{3MUL_p}$

The three most important and fundamental things to remember while evaluating the gradients are:

1. The differentiation of a scalar with respect to a matrix is a matrix with order same as that of the first matrix.
2. The summation is not necessary during the evaluation of the gradient as ultimately the matrix multiplication takes care of summing the appropriate terms.
3. If A = f(C) and B are two matrices, then the product of A and B differentiated with respect to C always results in the transpose of B and this transposed matrix has to be placed in the proper place such that the matrix order properties get satisfied.

CHAPTER 4

Mobile User location for a given input power vector

4.1 Training the Feedforward Neural Network

Normalization is the process of modifying the range of a data set (generally to 0 and 1 or to -1 and +1) by appropriately scaling the elements in the data set. There are different techniques through which normalization can be achieved. For the current implementation of IPS, at the input side min-max normalization technique is used whereas at the output side, simple scaling technique is used.
The min-max normalization is obtained through the following equation.

$$output = \frac{input - min(thewholedataset)}{max(thewholedataset) - min(thewholedataset)}$$

Similarly, the scaling operation is performed with the following equation.

$$output = \frac{input}{maximumlengthintheparticulardirection}$$

The outputs of both the above techniques range from 0 to 1. It is important to note that these normalization techniques are performed over the columns of the data sets. So at the input side, the normalization is performed over the input powers received at different locations from a single router and at the output side, the normalization is performed over the output coordinates at different locations in a single direction.

Normalization is required when there is a huge difference in the ranges of different features. For example in the case where there are three adjacent rooms, the x, y and z coordinates differ in their maximum values. The difference might be very significant if the rooms are really big and unsymmetrical. So we cast all the three

in one unified range i.e 0 to 1. It becomes even more important when the activation function is Sigmoid. At very high values and at very low values, the function saturates i.e., its gradient becomes close to zero. The learning of the network and updation of the weights during back propagation solely depends on the gradients of output with respect to the weights. A very small gradient results in an insignificant update in weights making the whole learning process slow.

The step size for the training process is chosen appropriately by trial and error based on the descending rate of the cost function. Generally, the step size is chosen such that at each iteration there is a reduction of the gradient scaled by 0.01.

The traininig process of the neural network is explained through the following flowchart diagram

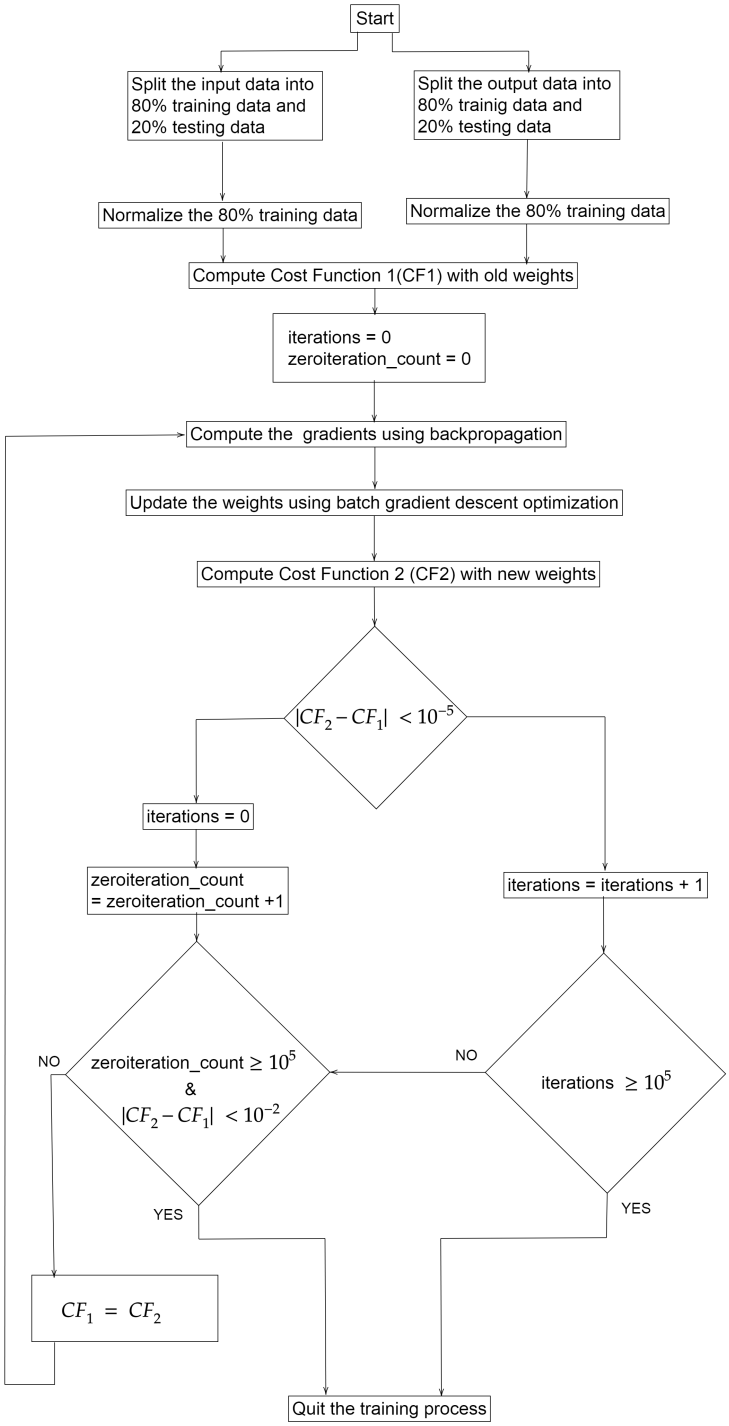


Figure 4.1: Training procedure for the Feedforward Neural Network

4.2 Testing the Feedforward Neural Network

The testing of the neural network is explained through the following flowchart diagram

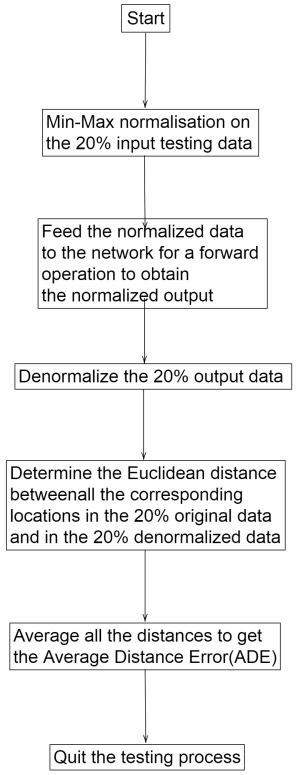


Figure 4.2: Testing procedure for the Feedforward Neural Network

4.3 Determination of the output for a single input power vector

The prediction of the MU location for a single input power vector is explained through the following flow chart.

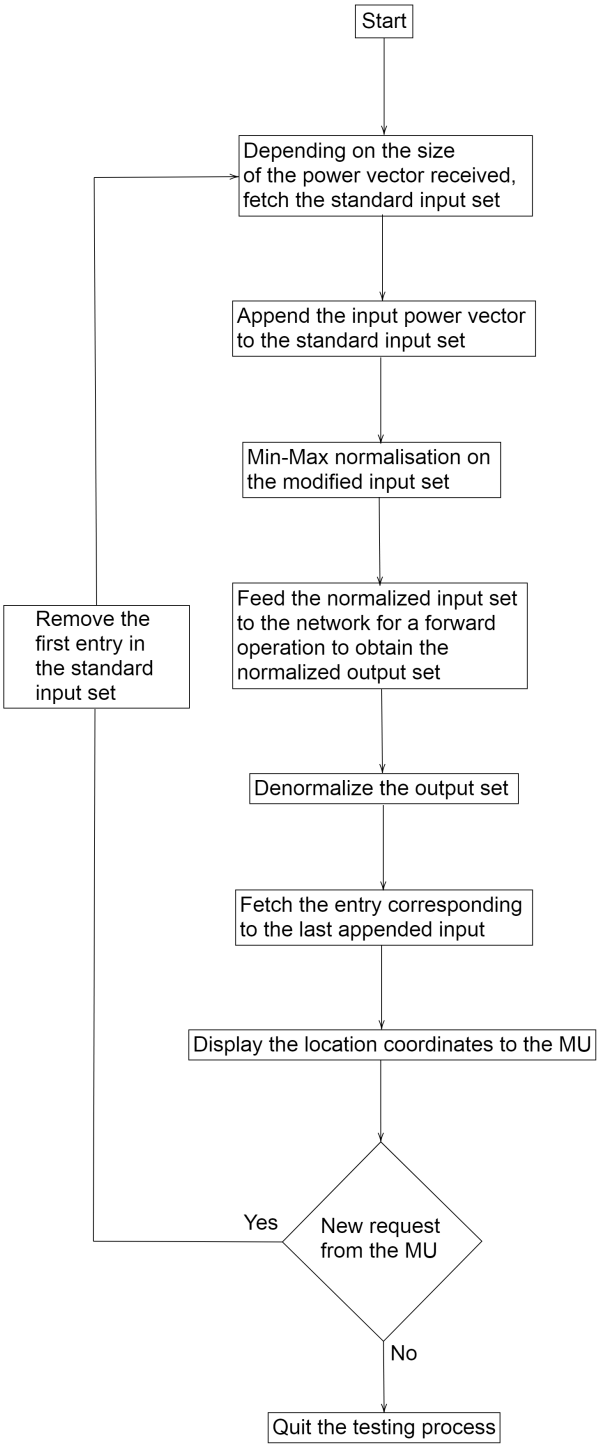


Figure 4.3: Procedure for fetching the output to a single input power vector

The standard input sets mentioned in the above flowchart are nothing but the matrices which have fixed number of rows and varying number of columns. The

column number of the matrices depends on the number of routers placed in the room. The main purpose of these standard input sets is to help in the normalization of the input vector. As the first row entries of the standard input sets are removed after getting the output, the sizes of the standard input remain constant thereby avoiding memory related issues.

CHAPTER 5

Prediction error plots

5.1 When there is a single room

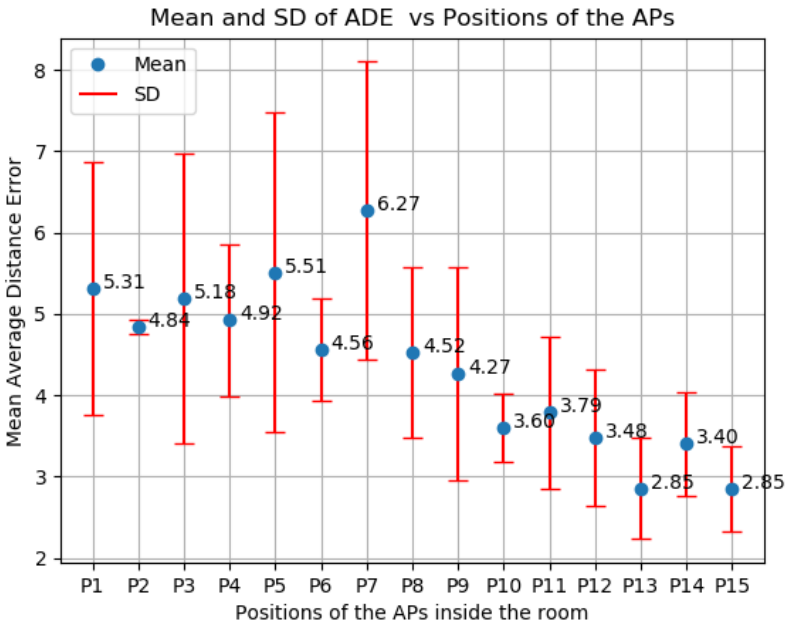


Figure 5.1: The graph showing the variation of mean ADE for different positions of APs
P1 - 1 AP at 1 corner (origin) of the room, P2 - 1 AP at the center of the room, P3 - 2 APs on 1 edge (X-axis) of the room, P4 - 2 APs at 2 diagonally opposite corners of the room, P5 - 2 APs at 2 opposite top corners of the room, P6 - 2 APs on the centre line along Y axis, P7 - 3 APs at 3 corners (i.e. on X,on Y and on Z), P8 - 4 APs at 4 alternate corners of the room, P9 - 4 APs at 4 bottom corners of the room, P10 - 6 APs at 6 faces of the room, P11 - 6 APs on along 3 axes and inside the room, P12 - 8 APs at 8 corners of the room, P13 - 8 APs in 8 quadrants of the room, P14 - 15 APs i.e. 1 AP at the center, 6 APs at the 6 faces and 8 APs at the 8 corners of the room, P15 - 16 APs i.e. 8 APs on the top edges and 8 APs on the bottom edges

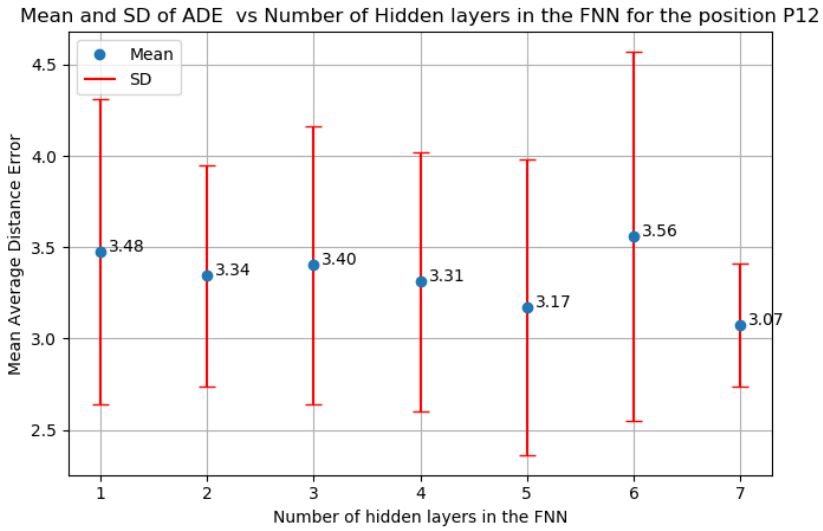


Figure 5.2: The graph showing the variation of mean ADE for different number of hidden layers in the FNN

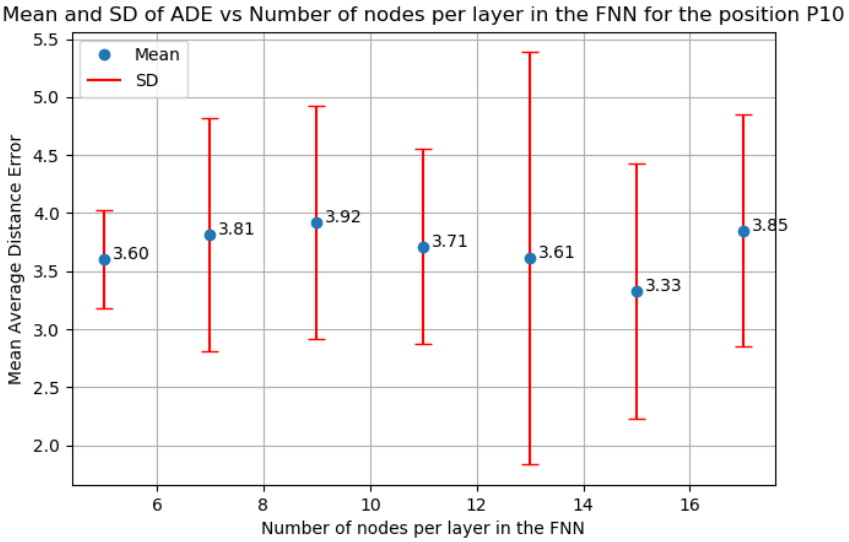


Figure 5.3: The graph showing the variation of mean ADE for different number of nodes per a layer in the FNN

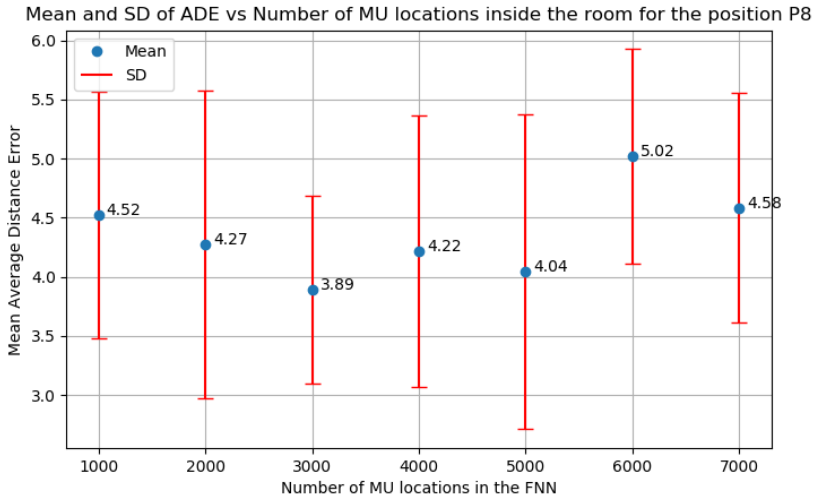


Figure 5.4: The graph showing the variation of mean ADE for different number of MU locations inside the room

5.2 When there are three adjacent rooms

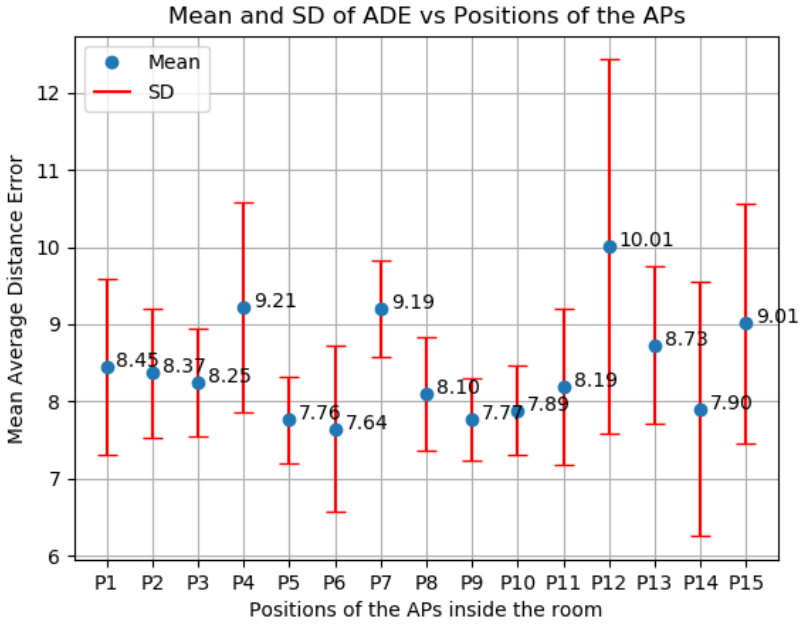


Figure 5.5: The graph showing the variation of mean ADE for different positions of APs
P1 - 3 APs at 3 corners on X ,Y and Z, P2 - 4 APs at 4 diagonally opposite corners, P3 - 4 APs on Y-axis, P4 - 6 APs at 6 faces of the center room, P5 - 6 APs at 6 external faces, P6 - 4 APs on top face and 4 APs on bottom face across the 3 rooms, P7 - 8 APs at 8 corners of the center room, P8 - 8 APs at 8 external corners of the room, P9 - 8 APs on diagonals of the 4 center faces, P10 - 8 APs on alternate edges (along x-axis), P11 - 2 APs on each edge(along X-axis) in the outer rooms and 4 APs on the top face of the center room, P12 - 3 APs on X,Y and Z in each room, P13 - 2 APs on edges and 1 AP at the corner on each center face, P14 - 14 APs at the centers of all the faces except the intermediate walls, P15 - 8 APs on the top edges and 8 APs on the botom edges

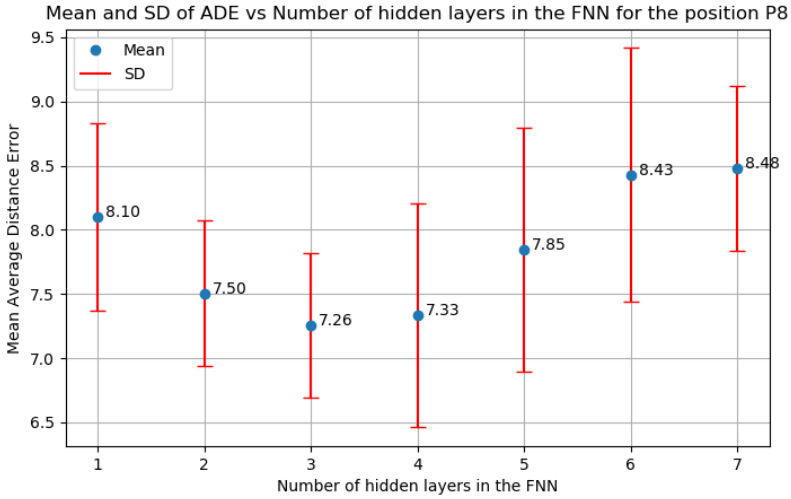


Figure 5.6: The graph showing the variation of mean ADE for different number of hidden layers in the FNN

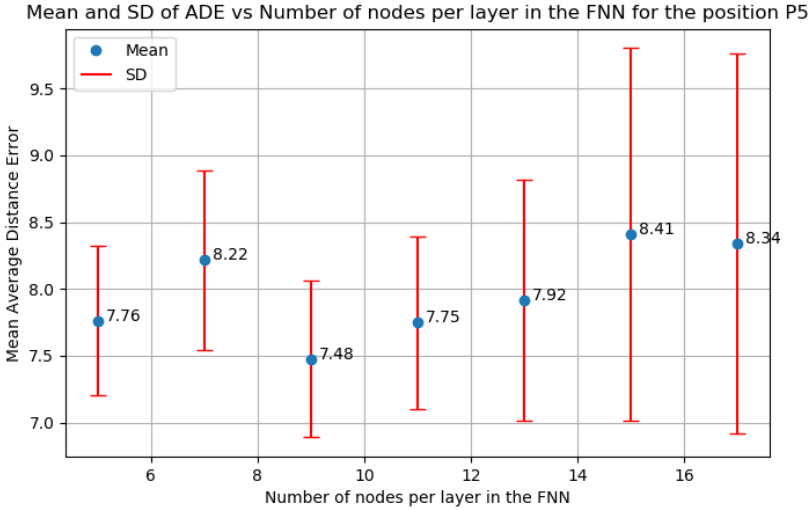


Figure 5.7: The graph showing the variation of mean ADE for different number of nodes per a layer in the FNN

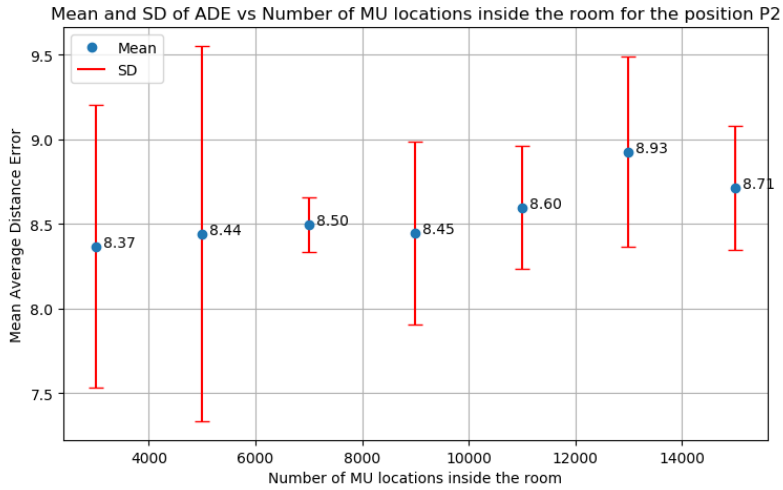


Figure 5.8: The graph showing the variation of mean ADE for different number of MU locations inside the room

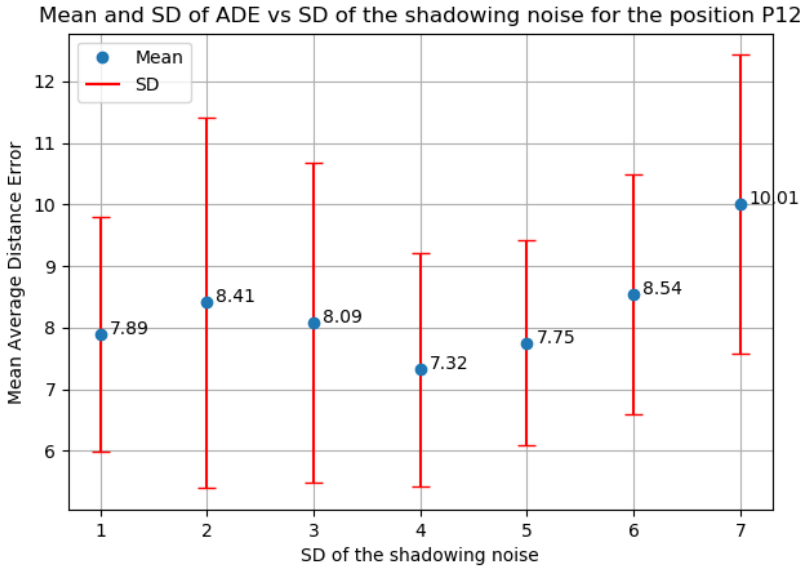


Figure 5.9: The graph showing the variation of mean ADE for different standard deviations of shadowing noise

Few more observations to consolidate the results:

Different scenarios considered	ADE
Position P13, 1 room, 7 hidden layers, 17 nodes per hidden layer	2.22
Position P8, 1 room, 1 hidden layer, 5 nodes per hidden layer, 6000 MU locations, step =24	3.61
Position P12, 3 rooms, SD = 1 and SD = 2, Equal Shadowing across all the walls	4.63 and 4.66
Position P8, 3 rooms, 7 hidden Layers, 17 nodes per hidden layer, SD = 7, Equal shadowing across all the walls	4.92

Table 5.1: A table showing the reduction of the ADE for various scenarios

5.3 Conclusions

In the single room scenario, it is observed that the key factors for getting minimum ADE are the positioning of the AP, number of neurons per layer in the FNN and the number of hidden layers in the FNN. The number of MU locations can impact the ADE if the step size is chosen proportional to the number of MU locations . Similarly, in the three adjacent room scenario, positioning of the AP, number of neurons per layer in the FNN, number of hidden layers in the FNN and the SD of the shadowing noise are the key factors for the minimization of ADE. As in the case of a single room, the number of MU locations can impact the ADE only if the step size is proportional to the number of MU locations. Also the random behaviour in the variation of ADE in the three room scenario can be minimized by having approximately equal shadowing effects for all the walls. We conclude by saying that in both the single room and the three room cases, when all the optimal cases (the cases for which the error was found to be less) are applied together, a significant reduction in the ADE is guaranteed and the prediction results get better.

As part of the future work, we would like to implement the IP for bigger and real indoor environments by including the effects like fading, interference ... besides shadowing. Various other types of neural networks like Radial Basis Function (RBF) Neural Network, Kohonen Self Organizing Neural Network, Modular Neural Network etc., with different optimization algorithms like Newton's method, Conjugate Gradient algorithm, Quasi-Newton method, Levenberg-Marquardt algorithm etc., will be used.

REFERENCES

[Kim *et al.*(2012)Kim, Park, and Lee] **Kim, W.**, **M. Park**, and **S.-J. Lee**, Effects of shadow fading in indoor rssi ranging. *In 2012 14th International Conference on Advanced Communication Technology (ICACT)*. 2012. ISSN 1738-9445.

[Rojas(1996)] **Rojas, R.**, *Neural Networks - A Systematic Introduction*. Springer-Verlag, Berlin, New-York, 1996.

[Zou *et al.*(2017)Zou, Chen, Jiang, Xie, and Spanos] **Zou, H.**, **Z. Chen**, **H. Jiang**, **L. Xie**, and **C. Spanos**, Accurate indoor localization and tracking using mobile phone inertial sensors, wifi and ibeacon. *In 2017 IEEE International Symposium on Inertial Sensors and Systems (INERTIAL)*. 2017.

[Zou *et al.*(2015)Zou, Lu, Jiang, and Xie] **Zou, H.**, **X. Lu**, **H. Jiang**, and **L. Xie** (2015). A fast and precise indoor localization algorithm based on an online sequential extreme learning machine. **15**, 1804–24.