2015

Novel approach to FM-based device free passive indoor localization through neural networks

Venkata Krishna Sangeetha
Iowa State University

Novel approach to fm-based device free passive indoor localization through neural networks

by

Venkata Sangeetha

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Engineering

Program of Study Committee:

Greg Luecke, Co-Major Professor

Manimaran Govindarasu, Co-Major Professor

Guiping Hu

Iowa State University

Ames, Iowa

2015

DEDICATION

This thesis is dedicated to my father for his unconditional and immense trust in my abilities. He was always there even when I had lost faith in my abilities. An overwhelming feeling of gratitude to my mother for all the sacrifices she made for me to pursue my Masters studies. My brother for his jaunty spirit that helped me survive familial ties half across the world.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I wish to thank my professor and guide Dr. Greg Luecke wholeheartedly for his support throughout my research work both financially and technically. He spent countless hours discussing the finer details of my research and always tackled problems with a smile. His experience added that extra edge that my research work lacked. A special thanks to Dr. Manimaran for having me under his aegis since I joined Iowa State University. The course on Cybersecurity of Smart grids helped me immensely during my internship. The other member of my dissertation committee, Dr. Guiping Hu had generously given her time to better my work.

I am beyond words for the technical help my father Dr. Sai Kumar had helped me with; this research would not have become fruitful if not for his contributing critique.

I would like to acknowledge and thank my friend Akanksha for her countless hours of standing while recording the experiments. My research would not have been possible without her help. I also wish to recognize the emotional support extended by my friends Loukya, Rohini and family, Teja, Pranava and Mani; without them it would not have been an easy ride. Their friendship made my stay in the US a wonderful experience.

I cannot thank Dr. Andrei Popleteev enough for sharing his technical knowledge on my research topic through e-mails and giving me the needed push.

I am grateful to the advices and words of support from the staff of the ECpE department, their services were unflagging.

ABSTRACT

Indoor Localization has been one of the most extensively researched topics for the past couple of years with a recent surge in a specific area of Device-free localization in wireless environments. Particularly FM-radio based technologies are being been preferred over WiFi-based technologies due to better penetration indoors and free availability. The major challenges for obtaining a consistent and highly accurate indoor FM based system are susceptibility to human presence, multipath fading and environmental changes. Our research works around these limitations and utilizes the environment itself to establish stronger fingerprints and thus creating a robust localization system. This novel thesis also investigates the feasibility of using neural networks to solve the problem of accuracy degradation when using a single passive receiver across multiple ambient FM radio stations. The system achieves high fidelity and temporal stability to the tunes of 95% by utilizing pattern recognition techniques for the multiple channel spectra.

CHAPTER 1. INTRODUCTION

There is an increased demand globally for the development of human presence detection methods. To meet these requirements, for the past couple of years researchers have been developing and improving low cost device free passive systems for implementation in smart homes and hospitals.

Another important angle to look for is the concern for energy savings worldwide due to the fast depleting natural resources used for electricity generation. Though alternative methods have been realized, consumers and concerned companies have issues of achieving the same throughput at a considerable expense. These alternative sources of energy and their methods of production are not much in daily use due to high production cost and complex maintenance schedules. Due to the rise of the global energy demands, the electricity price increased leading to installation of power metering devices for conservation of the residential electric energy.

The human factor in any system brings up the maintenance costs due to the misuse and over usage of electricity. Therefore an efficient system is required for automated power management. An interactive energy saving platform was proposed by the name "Ecosystem for Smart Home"(Al-merbawi, Zainal, Mahmud, & Trengganu, 2013) where smart power outlets and smart light switches were installed with human presence detection sensors. The integration of sensors and smart power nodes increases user awareness of the smart home for the advanced automated power management. Though the system achieves modest efficiency, there is a need to shrink the installation complexity and development costs induced by the integration of specific sensors in smart energy environment.

Today, the aging population is progressing rapidly in many countries. Japan, which is one of the most developed countries in the world, is now struggling to deal with falling birth rates and an aging population (Al-merbawi et al., 2013). They coincidentally had only 11.6% of its population over 65 year olds in 1989. The population of 65 year olds had risen to 23.1% and the population of 75 year olds had gone up to 11.4% in 2010. It is highly essential to realize an automatic and accurate human detection system at home for several healthcare services, such as monitoring the elderly, providing remote care, security, and safety management (Li, 2010) .The most serious problem faced is that elderly people have to be left alone at home unchecked due to the high costs of an aid. Elderly patients also have the tendency to go for unannounced walks, which may be a cause of concern for their caregivers. In such circumstances, a cost effective and easy-to-use localization system is the way to go forward.

Accessing information about a user's location has also been a central challenge in large bodies of location based services. Location awareness is important for fields such as ambient intelligence, behavior analysis, social interaction studies, and myriads of other context-aware applications. (Popleteev, 2014)

Therefore there is a strong requirement to develop low cost human presence detection methods to implement them in various global applications. As the requirements increase day by day, any minor improvement in cost and flexibility in usage of these systems makes the research more meaningful.

During the last two decades, localization techniques became more popular due to growing importance of their wide applications. The global positioning system (GPS) and Galileo (Kaplan & Hegarty, 2006) are two well-known techniques, which work on the principle of a direct line of-sight(LOS) to at least four satellites (Kechine, Tiberius, & Marel, 2003). GPS uses the satellites as

reference points to effectively calculate the positions of ground objects. Every object must carry a GPS receiver for accurate estimation of its absolute position in the global coordinate system. Some of the real world applications of GPS include location estimation, tracking, navigation mapping and providing timing services. GPS is widely used to ensure a self-positioning task and this technology is already implemented in many mobile devices.

The widely known and used Global Positioning System practically solved the problem of outdoor localization and has been serving as the defacto standard. However accurate the system might be outdoors, the technology fails in particularly closed environments. However, GPS signals have limited coverage indoors (Jorge Torres-Solis, 2010). In the absence of GPS, fingerprint based indoor localization techniques have been the most accurate approach for indoor environments that need a room level localization.

Accurate indoor positioning information through fingerprinting has the potential to revolutionize the way people search, locate and navigate to points of interest inside buildings, in a similar way that GPS revolutionized the way people navigate outdoors. Indoor areas especially places such as Museums and Hospitals are challenging and appealing for experimentation with novel technologies.

The complex nature of indoor environments challenge the system with smaller dimensions, constantly changing or moving obstacles that either reflect, refract or absorb the radio signals, influence of weather conditions, indoor signal characteristics, nonlinear relation between RSSI levels of signal and distance, and the major Non-Line-of-Sight problem. These dense and information rich environments call for larger data processing and accurate localization systems. Despite substantial amount of research in the area of computing and localization, the existing indoor positioning systems have their own limitations either in the form of limited coverage,

specialized hardware requirements or dedicated infrastructure and thus high establishment and maintenance costs.

The major challenge for fingerprint-based approaches is design of robust and discreet radio signatures. However, the approach of using existing hardware without external deployment has been successful in leveraging already available wireless signals (e.g., Wi-Fi, cellular, FM) to profile a location, usually in the form of received signal strength indicator (RSSI) values (Padmanabhan, 2000) (A. Varshavsky, 2007). Past researchers preferred using RSSI values of WiFi signals as Wi-Fi access points are widely deployed indoors, and every mobile device globally is equipped with a WiFi receiver.

However these WiFi based approaches are brought down by several limitations. Varying significantly spatially, their consistency becomes questionable. They are more susceptible to human presence and fade indoors due to their high operating frequency. To overcome these limitations, in the past few years researchers have proposed FM broadcast radio signals for robust radio fingerprinting. FM signals have several advantages when compared to WiFi for the indoor localization problem like exceptional indoor penetration and low variance over time which will be discussed further in Chapter 2.

The feasibility of device based indoor positioning using FM radio signals, generated either by multiple local transmitters or by broadcasting FM stations had already been proven by (Popleteev, 2013) to outperform both Wi-Fi and GSM indoor localization systems for confidence levels of up to 90%. They also analyzed the performance of the system among the presence of people in low and medium crowd density; and found FM to be relatively more stable than WiFi signals. Another pilot paper on device free passive localization using multiple ambient FM radio stations (Popleteev & Engel, 2014) presented that a single radio receiver could provide sub-meter

median localization accuracy in a small room. However it demonstrated that the performance deteriorated with time.

The thesis is organized as follows.

CHAPTER 2. INDOOR POSITIONING SYSTEMS

CHAPTER 3. DEVICE FREE INDOOR POSITIONING SYSTEMS: A REVIEW

CHAPTER 4. NEURAL NETWORKS: A REVIEW

CHAPTER 6. DATA ANALYSIS STAGE

CHAPTER 7. TRAINING, TESTING, OPTIMIZATION and GENERALIZATION of NEURAL NETWORKS

CHAPTER 8. CONCLUSIONS

Chapter 2.1 introduces the basics of an indoor positioning system and relevant features of FM-based DFP indoor localization. It also explains the challenges and advantages in using FM radio technology and summarizes the related work. A literature survey relevant to the present work is presented in different headings and summarizes the related work.

The core of this thesis is discussed in Chapter 2.2. The over plan is described in detail along with the generalized problem of localization, the reason behind choosing a neural network model, the Levenberg-Marquardt optimization algorithm and Bayesian regularization. The study includes explaining the parameters and mentioning the K-nearest neighbor classification method which has been used as a standard for results comparison.

The instrumentation and experimental set up and data collection methods along with the approach taken were discussed in chapter 2.3.

The experimental results and analysis carried out are given in chapter 4.

Chapter 5 presents the issues with existing solutions, thesis conclusions, major contributions and scope for future work.

The relevant appendices and references are given at the end.

CHAPTER 2. INDOOR POSITIONING SYSTEMS

Location determination is described as a process used to obtain the location information of a device with respect to a set of reference positions within a predefined space (K.Kaemarungi, 2005). It has different terms attached to it in literature as radio locating, position location, geolocation, location establishment, or localization. A system deployed to determine the location of an entity is called a position location system or positioning system. A wireless indoor positioning system provides indoor location information using a wireless network infrastructure; it can be to a user or a system. The entity's location defined by a set of coordinates or reference points within a predefined space is used to indicate the physical location. Location reporting can be physical i.e. local or global, relative (to a local reference point) or symbolic i.e. mapped to a physical identifier in the database. It is classified as physical and virtual based on the existence of place. In the real world, locations are defined based on the meeting places like houses, offices, restaurants, theaters, stadiums etc., whereas the place of meeting when online is defined as virtual location. The physical locations are divided into three sub divisions such as descriptive locations, spatial locations and network locations.

Several different methods are used for location in wireless based localization using different technologies.

2.1 Proximity based

Proximity detection or connectivity based depends on symbolic relative location information. The position of mobile client is determined by the nearest cell of origin (CoO) method with known position and limited range. This method is employed across many technologies in particular Bluetooth (Hallberg J, 2003), IR (Fox D., 2003), RFID (Yang, 2010) and GSM base

stations (K. Laasonen, 2004) (A. LaMarca, 2005) due to its simplicity. The accuracy of the CoO depends on the density of beacon point deployment and signal range which is unfortunately low.

2.2 Triangulation

Using the geometric property of three or more triangles to localize the position of the target is called Triangulation. The two derivations of this method are lateration and angulation. Time based systems such as TOA, RTOF and TDOA, RSS-based and received signal phase based methods come under lateration/trilateration/multilateration. The angle of arrival (AoA) technique determines the angle of arrival of the mobile signal coming from a known location at which it is received at multiple base stations.

Time based methods even though provide good accuracy, (Y. Gu, 2007) require high synchronization and use expensive specialized hardware; their accuracy is limited indoors due to high multipath and NLOS propagation. TDoA technique has no use for a synchronized time source of transmission but synchronized receivers in order to resolve timestamps and locate the node (D.Zhang, 2010). RTOF method uses only one node to record the transmitting and arrival times thus solving problem of synchronization; but the drawback is consecutive range measurement to multiple devices. Angle based techniques are limited in a way that they need additional highly directional antennas increasing the cost of the system. Also indoor environments affect signal propagation affecting angle of arrival and thus degrading the accuracy (Cisco, 2008).

2.3 Signal Property based

In contrast to previous approaches, to avoid using timing or angle based information that are heavily influenced by multipath fading; the method chosen by many is to estimate the distance between known and unknown nodes based on a set of measurements using signal attenuation (Farid, Nordin, & Ismail, 2013). Received signal strength indicator values or its representation in

device-specific units, a property of the radio signal, is the most widely used feature by wireless localization systems. The two general approaches to localization using RSSI employed by a majority of wireless systems are propagation modelling and finger printing.

Propagation modelling leverages the physical laws of signals in order to build a model of the signal propagation between the users and beacons. It requires RSSI values for three or more beacons for the mobile unit to use the propagation model and estimate the distances to each of the beacons, and thus own location. Propagation models are typically expressed in terms of path loss, which shows a signal's attenuation as it propagates through space (W.Barclay, 2002). Total path loss is a complex function of propagation distance, signal properties, terrain data, weather conditions and obstacles around(trees and buildings) (Due & Clouds, 1999; Wang, Du, & Chen, 2015) (ITU, 2007). Indoor environments introduce additional components to this equation such as obstacle interference with signal propagation and layout plans. The loss factors that need to be considered by a good propagation model varies with each scenario and needs to be customized until it reaches the optimum localization accuracy.

A good model requires a considerable effort for evaluation of site-specific characteristics, such as power loss coefficients and floor layout. The advantages with propagation modelling are straight-forward localization phase and good scalability which are best suited for line-of-sight (LOS) and obstacle-free propagation — conditions which are rarely met indoors.

Signal fingerprinting on the other hand relies on a database correlating RSSI measurements with corresponding coordinates which then uses statistics and machine learning algorithms to localize the position of the target among those learned during the training phase. This is a two stage process: calibration or offline stage and localization or online stage. Calibration is the survey of the site to collect the signal fingerprints at predefined points, and building a database which

matches these fingerprints and their locations. During the localization phase, the mobile client's fingerprint which is to be tracked is used by the positioning system along with calibration data and appropriate algorithms to determine the location. The location estimate usually is not perfectly accurate but carries with a certain error margin which all systems aim to minimize.

The algorithms which the systems depend to associate fingerprints with their counterparts in the localization phase are essentially machine learning algorithms and employ various approaches like deterministic and probabilistic (Bahl & Padmanabhan, 2000)(Lin & Lin, 2005; Roos, Myllymäki, Tirri, Misikangas, & Sievänen, 2002), classification and regression (M. Chen, 2006) (B. Ferris, 2006) (A. Schwaighofer, 2004), k-nearest neighbor classification and its variants, artificial neural networks (Wan Mohd, 2012), Bayesian interference, support vector machines or their hybrid combinations (V. Seshadri, 2005). Though signal fingerprinting solely is the most sought after approach, it has certain drawbacks such as laborious fingerprinting in the offline phase which is time consuming and repetitive. The degradation of fingerprints due to changing environments is another issue which can only be solved by repeating the measurements. The other factors that dynamically change include air humidity, snow and rain, doors and windows opening and closing, people and furniture movement (Bulusu, Heidemann, & Estrin, 2000; Chen, Chiang, & Chu, 2005;) (Radim Zemek, 2007) (L. Chan, 2006).

To address the above mentioned problem, RFID sensors were employed to provide updated fingerprints from predefined points (Chen et al., 2005), stationary signal sniffers (de Moraes & Nunes, 2006) and a mobile robot for autonomous signal strength measurements (Ocaña, Bergasa, Sotelo, Nuevo, & Flores, 2005). These methods however required additional hardware and add to the operational and computational cost of the system. To avoid this, spontaneous recalibration technique (Matic, Popleteev, Osmani, & Mayora-Ibarra, 2010) was introduced

without additional hardware. Despite the discussed limitations, signal fingerprinting offers the best accuracy in indoor environments plagued by multipath fading and NLOS reception. The described reasons are why passive fingerprinting was opted as the base for this thesis.

A broad number of fingerprint based technologies mandate prior installation of RF-beacons or require the moving target to be equipped with a mobile device for tracking purposes. (Ni, Liu, Lau, & Patil, 2003) (Want, Hopper, Falcão, & Gibbons, 1992) (de Moraes & Nunes, 2006). In contrast, approaches that do not require additional hardware either in the environment or the target are drawing increasing attention. Device-free localization (DFL) addresses the challenge of locating the user without any wearable devices. The next section is a focused review on the state-of-the-art device free indoor positioning systems and the motivation behind selecting a FM based approach.

CHAPTER 3. DEVICE FREE PASSIVE INDOOR POSITIONING SYSTEMS: A REVIEW

The research on localization and detection has been dedicated for the most part to mobile devices or stationary beacons. However, the person to be localized in a real time scenario cannot and should not be mandated to carry any device. We shall expand upon prior studies on device-free techniques focusing on radio based systems alone.

A device free localization system detects, tracks and identifies any object or human body without the need for any external device attached. The only active participant in this localization technique is the target itself. This works on the concept that the presence and motion of human body changes the pattern of radio frequency signals in a closed or open environment. The movements and sheer presence of a human body inside the monitored area causes a subtle if not huge change in the received signal strength indicator values which are measured with appropriate sensors or antennas depending on the type of DFL system implemented.

Multiple researchers have already studied the use of RSSI in detection and tracking of an entity (Davchev, 2010) and that a moving person caused shadowing of the radio signal leading to multi-path fading (Jeffrey Hightower, 2001).

The importance of RSS information is paramount to RSS-based DFPL techniques. Recent technological achievements have made it easier for the process of location estimation. However, one major disadvantage of this technique is that the RSS measurement involves a high sensitivity to noise and environmental variations, which degrades the performance by quite large amounts.

To overcome this limitation of the RSS links, (Ke, Liu, & Fu, 2014) propose an environmental-adaptive sparsity-based localization method for the DFPL problem. Their approach

is to use a dictionary learning (DL) technique based on quadratic programming so that the location solution can better tune to the changes of the RSS measurements between the node pairs to the spatial location of the target. Another wireless network infrastructure based DFPL technique employing RSS linked nodes (Deak, Curran, & Condell, 2010) deploys a wireless sensor network in the absence of a proper wireless network structure. RASID (Kosba, Saeed, & Youssef, 2012) combines various modules for detecting anomalies statistically when the environment is unstable. The research also uses a lower overhead when detecting human activities using standard Wi-Fi hardware.

To reduce the processing load on the network, researchers introduced Ichnaea (Saeed, Kosba, & Youssef, 2014) that uses a light weight and short training period to profile the network and then applies statistical anomaly detection techniques and particle filtering to successfully adapt to changes in the Wi-Fi environment.

The physical layer information or the channel state information (CSI) of any RSS link (802.11n standard) provides ample details on the current state of the environment owing to the detailed change measurement. PADS, novel scheme uses the amplitude and phase information in the PHY layer to provide reliable detection of humans moving at dynamic speeds. The system employs a fixed transmitter and receiver to detect these changes in the RSS.(Qian, Wu, Yang, Liu, & Zhou, n.d.)

Another paper employing the PHY layer information over a single stream leverages the physical layer information of Wi-Fi networks to provide accurate localization (Abdel-Nasser, Samir, Sabek, & Youssef, 2013). Pilot (Xiao, Wu, Yi, Wang, & Ni, 2013), also a CSI-based DFP indoor localization system monitors the pattern changes in the wide band signal frequency over the physical layer to capture the environment variance.

Crowd sourcing is an active method of research which many have implemented to reduce the load on the builders of the localization system during data collection. The methods using this approach have no need for any manual calibration, external infrastructure or the knowledge of the terrain. The users contribute the data which is used to map the environment and hence aid in target localization. PiLoc, an indoor localization system employing crowd sourcing merges walking segments annotated with displacement and signal strength information obtained opportunistically from users to develop a map of walking paths marked with radio signal strengths (Luo, Hong, & Chan, 2014).

Probabilistic analysis approach was employed by very few researchers in the area of device free passive localization. One such paper improves RF-based localization in cluttered indoor environments using probabilistic classification approaches based on discriminant analysis (Xu et al., n.d.). Another design Nuzzer (Seifeldin, Saeed, Kosba, El-Keyi, & Youssef, 2013) operating over real environment provides device free passive localization using already installed wireless networks to track, monitor and process the changes in the RSS. The access points and monitoring points in the enclose environment measure these changes which are processed by employing probabilistic techniques.

Until now we encountered researchers tracking a single human through an indoor environment, a novel approach SCPL (Xu, Firner, Moore, & Zhang, 2013) counts and localizes multiple subjects in indoor environments by collecting data profiles of each subject. An innovative successive cancellation based approach is later used to determine the number of subjects. Another paper focused on distinguishing multiple human targets by employing Probabilistic Neural Network (PNN), Support Vector Machine (SVM) classifier with kernel functions and Multilayer Perceptron (MLP).

All the research done on device free passive localization concentrated on employing wireless networks or Wi-Fi hardware to detect changes in the RSS links by either using multiple nodes or monitoring points. A study performed to learn the feasibility of using ambient FM-based radio signals to perform device free passive localization concluded that localization was not possible (Shi, Sigg, & Ji, 2012). The most recent paper monitored 11 radio stations with a software define radio and found out that channel diversity not only allowed device free passive localization but also improved the performance of the system when compared to a single frequency based system (Popleteev & Engel, 2014).

Our thesis motivated from this latest work employs the same off-the-shelf radio receiver and produces considerably better results which will be discussed in the further sections.

CHAPTER 4. NEURAL NETWORKS: A REVIEW

4.1 Artificial Neural Networks in Localization

The functioning prowess of a neuron is the result of collective behavior in a network where all neurons are interconnected. Once connected, the network evolves with the neurons continuously evaluating their output by checking their inputs, calculating the weighted sum and comparing to a threshold to decide if they had to fire. This is a highly complex parallel process whose features cannot be reduced to phenomena taking place with individual neurons.

It is only logical that the evolving process should eventually reach a state where all neurons continue working but no further changes in their state happen. Interestingly a network may possess more than one stable state, and it is obviously determined by the choice of synaptic weights and thresholds for the neurons.

Different types of artificial neural networks can be obtained by varying the activation functions of neurons and structures of weighted interconnections between them. Examples of these classes are the Multi-Layer Perceptrons (MLP), Radial Basis Function (RBF), and the Recurrent Neural Networks (RNN).

4.1.1 MLP Networks

MLP networks are trained by testing the performance of the network with all sets of inputs and trying to minimize the measured error between the desired output and the network output. The weights are modified based on this error until it reaches a minimum tolerance. There are at least five location fingerprinting-based positioning algorithms using pattern recognition technique so far: probabilistic methods (Dawes & Chin, 2011),k-nearest-neighbor (kNN) (Dawes & Chin, 2011), neural networks (Yifeng & Shareef, 2006),support vector machine (SVM) (Brunato &

Battiti, 2005),smallest M-vertexpolygon (SMP) (M.Musavi, 2008)and self-organizing maps(Giorgetti, Gupta, & Manes, 2007).

These probabilistic types of algorithms treat RSS as a statistical random variable and likelihood value for each location is calculated based on estimated probability distributions, typically with Gaussian kernel function. Probabilistic techniques ("Nattapong Swangmuang, 2008) (Ladd, Bekris, Rudys, Kavraki, & Wallach, 2005) store information about signal strength distributions from access points in the radio map and for unknown location, conditional probability of each referent location in radio map is calculated (Din, 2009). The Nibble system uses a Bayesian network approach (Paul Castro, 2001) (P. Castro & Muntz, 2000) to estimate user location. Research shows that the probabilistic technique outperforms the deterministic technique. In the case of probabilistic techniques the location estimation for the same parameters will vary. As for deterministic techniques the location estimation will be a single result. Probabilistic techniques, such as Markov modelling, Kalman filtering and Bayesian analysis (Paul Castro, 2001) were discussed by several authors (Ladd et al., 2005). Probabilistic approaches like Bayesian networks based solutions achieve better performance but they are computationally exhaustive and difficult to scale. As the area and number of target locations and wireless access points increase, the computational complexity of Bayesian structures grows and become computationally hard. Kalman filter techniques depend on knowledge about the environment distribution of noise to estimate the node's location (Yifeng & Shareef, 2006).

Nearest Neighbors based pattern recognition technique and its derivatives have been used traditionally by many researchers (Din, 2009) (Dawes & Chin, 2011). This method requires a database of sample RSS readings at the estimation time for pattern matching. As the area and number of target locations grow, this size of the database dramatically increases and it becomes

impractical to achieve sufficient scalability. On the other hand GPS like triangulation methods provide poor performance due to multi-path propagation effects in indoor environments.

Gogolak et al. (Gogolak, Pletl, & Kukolj, 2011) propose a fingerprint localization methodology while using wireless sensor networks based on ANN which is applied in a real experimental indoor environment. It provides the necessary measurement results to the fingerprint localization (Taok, Kandil, & Affes, 2009).

4.1.2 Radial Basis Function Network (RBFN)

Generally, a radial basis function network can be described as a parameterized model used to approximate an arbitrary function by means of a linear combination of basis functions (Llc, n.d.). RBF networks typically have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer. The weights connecting the basis units to the outputs are used to take linear combinations of the hidden units to produce the final classification or output. The network training is divided into two stages: first the weights from the input to hidden layer are determined, and then the weights from the hidden to output layer. RBF networks have attracted considerable attention and they have been widely applied in many science and engineering fields (Kurban, 2009). RBF networks have been applied in many applications, such as real time approximation (Mekki & Chtourou, 2012), system identification, nonlinear function approximation (Skala, 2013), adaptive control, speech recognition (Oglesby & Mason, 1991). RBF networks also find wide applications in localization. Two positioning systems in real world indoor environment were developed based on WLAN and GSM (Stella, Russo, & Šari, 2014). Radial Basis Function (RBF) network was used to develop fingerprinting positioning system to construct nonlinear relation between RSS and user location. Dynamic neural network were used for localization of a WIFI positioning systems (Fahed & Liu, 2013).

The major drawback of RBF is that it spans the input/output data space of the application affecting memory and computational requirements. Radial basis networks, even when designed efficiently, tend to have more neurons than a comparable feed forward network neurons in the hidden layer. This is because feed forward (sigmoid) neurons can have outputs over a large region of the input space, while RBF neurons only respond to relatively small regions of the input space. The result is that larger the input space (in terms of number of inputs, and the ranges those inputs vary over) the more RBF neurons are required[Matlab9].On the other hand, designing a radial basis network often takes much less time than training a sigmoid/linear network. RBF networks perform exact interpolation, but there are two serious problems with these exact interpolation networks:

1. They perform poorly with noisy data : The network's outputs have to pass through all the data points when the data is noisy, because that will be a highly oscillatory function that will not provide good generalization.

2. They are not computationally efficient: The network requires one hidden unit (i.e. one basis function) for each training data pattern, and so for large data sets the network will become very costly to evaluate. With MLPs generalization can be improved by using more training data – the opposite happens in RBF networks, and they take longer to compute as well

4.1.3 Recurrent Neural Network (RNN)

A recurrent neural network (RNN), is a neural network model proposed in the 80's (Werbos, 1988) (Elman, 1990) (Werbos, 1988) for modeling time series. The structure of the network is similar to that of a standard multilayer perceptron, with the distinction that we allow connections among hidden units associated with a time delay. Through these connections the model can retain information about the past inputs, enabling it to discover temporal correlations

between events that are possibly far away from each other in the data (a crucial property for proper learning of time series). The structure of the MLP network is very similar to the RNN networks. The only difference is that MLP interconnections are straight forward from the inputs to the outputs while in RNN networks it is possible to have feedbacks to previous layers.

While in principle the recurrent network is a simple and powerful model, in practice, it is unfortunately hard to train properly. Among the main reasons why this model is so unwieldy are the vanishing gradient and exploding gradient problems described by Bengio, Y. et al (Bengio, Simard, & Frasconi, 1994). (Neuralnetworks and deep learning, n.d.)

4.1.4 Comparison of Neural Networks with RBF and RNN

MLP and RBF neural networks were compared by Xie et.al. (Xie, Yu, & Wilamowski, 2011) and it is found that for function approximation problems, RBF networks are specially recommended for surface with regular peaks and valleys, since efficient and accurate design can be obtained. While, for surfaces without regular peaks and valleys, traditional neural networks are preferred as a general model. For classification problems, traditional neural networks can get better classification results with much more efficient networks than RBF networks,

RBF's learn faster but do not generalize well. This is due to the fact that in RBF, only a one-layer neuron is trained in a supervised manner. By allowing a good compromise between local and distributed approaches, it was concluded that a structure composed of perceptrons, is a good alternative for difficult classification tasks as it learns fast and generalizes quite well even the input dimensionality is high (Alpaydin & Jordan, 1996).

4.1.5 How NN's were selected for this Thesis

One advantage of using neural networks to perform localization tasks is that it does not require prior knowledge about the noise distribution of the network environment. The noisy

distance measures can be used directly as inputs to train the artificial neural network. ANNs are able to characterize the noise and compensates it for more precision in estimating the location of the node (Rahman, Park, & Kim, 2009) (Shareef, Zhu, & Musavi, 2008)

As shown by Kim et al (Rahman et al., 2009), the use of MLP is more suitable for localization in wireless sensor networks due to its low computational and memory costs. Its accuracy is slightly lower than the RBF (that shown best accuracy, but higher computation resources requirement), but the best trade-off between accuracy and resource requirements is clearly seen.

When the parameters of noise are not changed, the localization using MLP could be a good option. Simulation results show that neural networks can achieve high level of accuracy in estimating the location, and requires fewer anchor nodes. This technique is neither affected by the NLOS (Non-Line-of-Sight) environment nor by the irregularity of the power transmitted by the anchors (Tian & Shi, 2007). The neural network is employed to predict the NLOS error (L. Povescu, 2001).

After critical study of three basic neural networks and considering their weaknesses, it is very important to select appropriate NN for the problem to be modelled. RBF's are very good for simple problems having clean data and deal efficiently with less input data. It is more appropriate for less input space and single output problems. Similarly RNN's are good for the simple problems with time delay. The present research work involve multi input and multi outputs, where MLP's are very strong and they can even handle the data with noise considering all the facts MLP's are selected and implemented for the first time for this type of localization. Levenberg-Marquardt function of MATLAB ("trainlm") is implemented for optimization of NN. As the data is very sensitive to environmental changes, multipath effects and NLOS, it is highly essential to generalize

the model to avoid unwanted training weaknesses. Hence Bayesian networks are implemented by MATLAB function of "trainbr".

4.2 Generalization Methods

Over-fitting problem or poor generalization capability happens when a neural network over learns during the training period. As a result, such a too well-trained model may not perform well on unseen data set due to its lack of generalization capability. Several approaches have been suggested in literature to overcome this problem. The first method is an early stopping mechanism in which the training process is concluded as soon as the overtraining signal appears. The signal can be observed when the prediction accuracy of the trained network applied to a test set, at that stage of training period, gets worsened. The second approach is the Bayesian Regularization. This approach minimizes the over-fitting problem by taking into account the goodness-of-fit as well as the network architecture.

4.2.1 Early Stopping Approach

This approach requires the data set to be divided into three subsets: training, test, and verification sets. The training and the verification sets are the norm in all model training processes. The test set is used to test the trend of the prediction accuracy of the model trained at some stages of the training process. At much later stages of training process, the prediction accuracy of the model may start worsening for the test set. This is the stage when the model should cease training to overcome the over-fitting problem.

4.2.2 Bayesian Regularized Artificial Neural Networks

It minimizes a combination of squared errors and weights, and then determines the correct combination so as to produce a network that generalizes well. The process is called Bayesian regularization (Foresee & Hagan, 1930) (MacKay, 1992a) (MacKay, 1992b). Unlike a standard

feed forward neural network training method where a single set of parameters (weights, biases, etc.) are used, the Bayesian approach to neural network modeling considers all possible values of network parameters weighted by the probability of each set of weights.

Bayesian regularized artificial neural networks (BRANNs) are more robust than standard back-propagation nets and can reduce or eliminate the need for lengthy cross-validation. Bayesian regularization is a mathematical process that converts a nonlinear regression into a "well-posed" statistical problem. The advantage of BRANNs is that the models are robust and the validation process, such as back propagation, is unnecessary. They are difficult to overtrain, since evidence procedures provide an objective Bayesian criterion for stopping training. They are also difficult to overfit, because the BRANN calculates and trains on a number of effective network parameters or weights, effectively turning off those that are not relevant. This effective number is considerably smaller than the number of weights in a standard fully connected back-propagation neural net.

Bayesian regularization minimizes a linear combination of squared errors and weights. It also modifies the linear combination so that at the end of training the resulting network has good generalization qualities. The Bayesian Regularization approach involves modifying the usually used objective function, such as the mean sum of squared network errors (MSE).

$$F = E = \frac{1}{N}\sum_{i=1}^{N} e^2 \tag{1}$$

The modification aims to improve the model's generalization capability. The objective function in Eq. 1 is expanded with the addition of a term, E_w which is the sum of squares of the network weights:

$$F = \beta E_d + \alpha E_w \tag{2}$$

where α and β are parameters, which are to be optimized in Bayesian framework. It is assumed that the weights and biases of the network are random variables following Gaussian distributions and the parameters are related to the unknown variances associated with these distributions. It is a known fact that the optimal regularization technique requires quite costly computation of the Hessian matrix. To overcome this drawback, Gauss-Newton approximation to the Hessian matrix is used.

The Bayesian regularization takes place within the Levenberg-Marquardt algorithm (More, n.d.) and uses back-propagation to minimize the linear combination of squared errors and weights.

4.2.3 Advantages of Bayesian Regularization

The advantages of Bayesian methods are that they produce robust models, well matched to the data, and make optimal predictions. No test or validation sets are involved so that all available training data can be devoted to the model and the potentially lengthy validation process discussed above, is avoided. The Bayesian objective function is not noisy. At the end of training, the Bayesian regularized neural network has optimal generalization qualities. Although there is no need for a test set, since the application of the Bayesian statistics provides a network that has maximum generalization, it is still considered prudent to use a test set. The Bayesian neural net has the potential to give models which are relatively independent of neural network architecture, above a minimum architecture and the Bayesian regularization method estimates the number of effective parameters.

The basic method used in the network training is derived from the Levenberg-Marquardt algorithm (More, n.d.) and the MATLAB implementation of the algorithm uses an automatic adjustment of the learning rate and the Marquardt μ parameter. The training is stopped if

- maximum number of epochs is reached;

- the performance has been minimized to a suitable small goal;
- the performance gradient falls below a suitable target;
- the Marquardt μ parameter exceeds a suitable maximum.

4.2 MATLAB Implementation of Bayesian Neural Networks:

The network training function 'trainbr' updates the weight and bias values according to Levenberg-Marquardt optimization. This Bayesian regularization takes place within the Levenberg-Marquardt algorithm (More, n.d.). Backpropagation is used to calculate the Jacobian jX of performance with respect to the weight and bias variables X. Each variable is adjusted according to Levenberg-Marquardt,

$$jj = jX * jX \tag{3}$$

$$je = jX * E \tag{4}$$

$$dX = - (jj + I* \mu) / je \tag{5}$$

where E is all errors and I is the identity matrix.

The adaptive value μ is increased by μ _inc until the change shown above results in a reduced performance value. The change is then made to the network, and μ is decreased by μ _dec. The parameter mem_reduc indicates how to use memory and speed to calculate the Jacobian jX. If mem_reduc is 1, then trainlm runs the fastest, but can require a lot of memory. Increasing mem_reduc to 2 cuts some of the memory required by a factor of two, but slows trainlm somewhat. Higher values continue to decrease the amount of memory needed and increase the training times.

4.3 Training of Neural Networks:

The basic assumption in Bayesian regularization is that the true underlying function between input-output pairs should be smooth and this smoothness can be obtained by keeping

network weights small and well distributed within the neural network. This is achieved by constraining the size of the network weights which is referred to as regularization which adds additional terms to the objective function

$$F = \beta V + \alpha W \quad (6)$$

where V is the sum of squared errors (performance index), W is the sum of squares of the network weights, α and β are objective function parameters. This modification in performance index will result in a neural network with smaller weights and biases which will force its response to be smoother and decrease the probability of over fitting.

The generalization ability of several approaches on neural network forecasting model was examined on artificial clean real time series data which are contaminated with noises of known levels (Doan & Liong, 2004). Results showed that, in general, the Bayesian Regularization (BR) approach, compared to the early stopping approach, lends the model higher generalization ability. Thus, BR yields higher prediction accuracy than the early stopping approach. Also another advantage of BR over early stopping approach is that, BR does not require a test set. It should be noted that the length of the test set has an impact on the prediction capability of the trained model. The prediction improvement of models trained with generalization approaches over those with standard approach is considerably remarkable when the data is very noisy.

Linear activation functions are used in the output layer. The weights and biases of the network are initialized to small random values to avoid immediate saturation in the activation functions. The data set is divided into two sets as training and test sets. Neural networks are trained by using training data set and their generalization capacity is examined by using test sets. The training data is never used in test data. MATLAB's neural network toolbox is used to train neural

networks. Specifically the MATLAB functions such as “train”, “newff” and “trainbr” are used for training a feed-forward backpropagation network with Bayesian regularization. Simulations with test data is repeated multiple times with different weight and bias initializations.

4.3.1 Backpropagation Network

The back propagation method is part of the parallel distributed processing system (Werbos, 1988). Backpropagation refers to the method that computes the gradient of the case-wise error function with respect to the weights for a feedforward network which is straightforward and an elegant application of the chain rule of elementary calculus. By definition, back propagation or backprop refers to a training method that uses backpropagation to compute the gradient. In other words, a backprop network is a feedforward network trained by backpropagation (Yu & Wilamowski, 2011).

Over the years, different researchers have developed many ANN training algorithms to reduce the execution time and computer storage requirements. There are several different back propagation training algorithms published in literature and no algorithm is best suited to all problems. The basic differences between these algorithms are how they handle the weight up-gradation to reduce error and how they modify the learning rate to reduce the convergence time.

4.4 Objectives and Outline of the Thesis

4.4.1 Gaps identified in the past Research Work

Simplicity, cost and accuracy are the most important requirements of any indoor localization technique. A technique should be easily deployable without requiring huge initial setup, training or environmental adaptation and it should provide sufficient accuracy for the class of applications it is intended for. Finally a low cost solution is essential to make the localization affordable for widespread applications.

Most of the past research work was concentrated on device based systems, which provide high precision but come at an expensive and complex implementation. Such high levels of accuracies would be attractive for applications such as mobile robot navigation and object manipulation tasks in mission critical and strategic areas.

On the other hand, several indoor and outdoor applications may not necessarily require such high levels of sub-meter accuracies. For example, a highly accurate localization system is not required to detect the human presence so as to switch on the power in a smart home. For such applications simpler, low-cost and moderately accurate localization techniques will be more affordable and attractive.

Localization of a human target using signal fingerprinting typically depended on numerical error analysis or an algorithm that used an offline database or a theoretical propagation model. The positioning systems in the past and their algorithms using pattern recognition techniques are deterministic and probabilistic, most of them based on K-nearest-neighbor (KNN), Bayesian graphical models, support vector machine (SVM), or their combinations. They were unsupervised learning methods trying to get a close estimate to the actual position. Some researchers utilized additional equipment to improve the localization accuracy while others depended on other signal quality indicators as well to augment the wireless signature.

4.4.2 Scope of this Thesis

In view of the research gaps presented in the previous section and on basis of feasibility, the following objectives were arrived at:

1. Our goal was to build a positioning system that survived on a single receiver and no additional equipment either in the test environment or on the target to achieve a commendable amount of accuracy. Earlier researchers had employed k-nearest neighbor

classification which was an unsupervised learning method and achieved a modest 85% median accuracy that degraded to 65% after three days of retesting the model. He had considered 11 active stations where all of them had to perform well to record a reasonable amount of accuracy (Popleteev & Engel, 2014). Thus there exists a need for a low-cost system which was simple and easily implementable with a low learning curve.

2. We needed to build an indoor positioning system that produced better efficiency compared to the previous systems even when it relies on just a single frequency.
3. Also the intention of our localization system was to be cost effective, run on algorithms simple enough to be processed by an online system, minimalistic in computational needs and zero responsibility on the human target.

We needed to rethink the positioning system model but not as much as to reinvent the wheel. To understand this subtle research problem at hand, we surveyed the major breakthroughs in the field of indoor localization based on neural networks across multiple domains. The following section will present a brief review of these advancements and explain why our thesis fares better when compared in the areas of cost effectiveness, consistent performance and stability. It also proves the consistency and improved accuracy our method brings to the area of indoor localization compared to the previous results.

4.5 Thesis in a Nutshell

The essence of this thesis lies in modeling the experimental data of received signal strength values across five individual frequencies and five unique scenarios. The datasets from the five days of experimentation after pre-processing present a total of 125 curves varying across 100 samples each. The idea is to develop a neural network model by training it with four days of data and use the other day data for testing. This was not a strict rule on selecting the training and testing

data but just a simple way to explain the general method chosen. The parameters like hidden layer neurons, training algorithm, weights that affected the stability of the neural network were studied and optimal values deduced after rigorous analysis and relentless training. Any machine learning based method aims to optimize its results, we did the same by employing Levenberg-Marquardt optimization algorithm to optimize the neural network.

Training and testing a model do not end with good results alone, generalizability has to be tested to avoid overfitting or underfitting which is of prime importance. This was achieved by implementing early stopping through k-fold cross validation and Bayesian regularization which as the results proved were successful in improving the model. The bias-variance tradeoff was also studied to give a deeper understanding of the model and the appropriate plots were prepared.

To reduce the complexity of the neural network model, parallelization was considered. This proved to shrink not just complexity but also the computation time. This approach is also a simpler way to improve generalization when dealing with noisy datasets and can considerably lower the mean square error. In simpler terms, each individual frequency was trained and tested on an individual neural network and the results were combined rather than depending on just one big neural network. Another motive for this was to realize the discrete contribution of each frequency to the system's performance. In case a frequency decided to go haywire, it would be known immediately.

The performance of the model was evaluated using mean square error and regression analysis. The performance accuracy of this model was also analyzed in depth by testing every parameter that directly or indirectly affected the system. Some of the parameters we considered were the number of channel frequencies, datasets, hidden layer neurons, learning steps, training algorithm, and finally the training, validation and test set sizes.

The next section explains in detail the steps that had to taken right from setting up the test environment, data collection and frequency filtering.

CHAPTER 5. EXPERIMENTAL PROCEDURE

5.1 Measurement Setup

In this chapter, we examine and analyze the overall setup of the experiment, the environmental factors affecting the performance, selecting the positions within the room, fixing the receiver's position and height, FM station filtering, raw sample collection, comments on the hardware and software we relied on.

The variation of radio signals in an indoor environment is based on various factors. When conducting the experiment, it is to be ensured that some of these factors remain constant to study the variation of other factors. Stable weather conditions on sunny days with temperature around 25 degrees Celsius was observed for most of the days of collecting experimental data.

Researchers in the past have already concluded that radio signals vary across different positions of a room. Some approaches have either selected illustrative places or spatial places to categorize the signal space. A target's spatial position signifies a place stated simply by two- or even three-dimensional coordinates within a Euclidean place. In our research we have considered a two dimensional radio environment. Spatial positions would have suited special purposes but the research being the first attempt at signifying particular positions, we selected illustrative positions across the room which offered sufficient details.

The room selected to carry out the experiment was a small room of 10*8 Sq-ft in a two bedroom apartment on the second floor in the heart of the city. The furniture in the room was left intact to mimic a real case scenario. In order to investigate the human impact on the RSS variance as a function of positions, four positions of a human standing or sitting were selected as shown in the figure 1:

Case1 / Position P1: This position involved the human target standing near the door to entrance.

Case2 / Position P2: The position involved the human target standing near the door to exit.

Case3 / Position P3: This position involved the human target sitting beside a wooden shelf in the corner ‘C’.

Case4 / Position P4: This position involved the human target standing halfway P1 and the receiver. This was planned to check the variation of RSS with distance in a closed room.

Case5 / Empty Room: In this position, the measurements were taken without the presence of a human in the room to note the signal variance without any human obstruction and set a base standard to compare other cases with.

Figure1 presents a complete layout of the test environment.

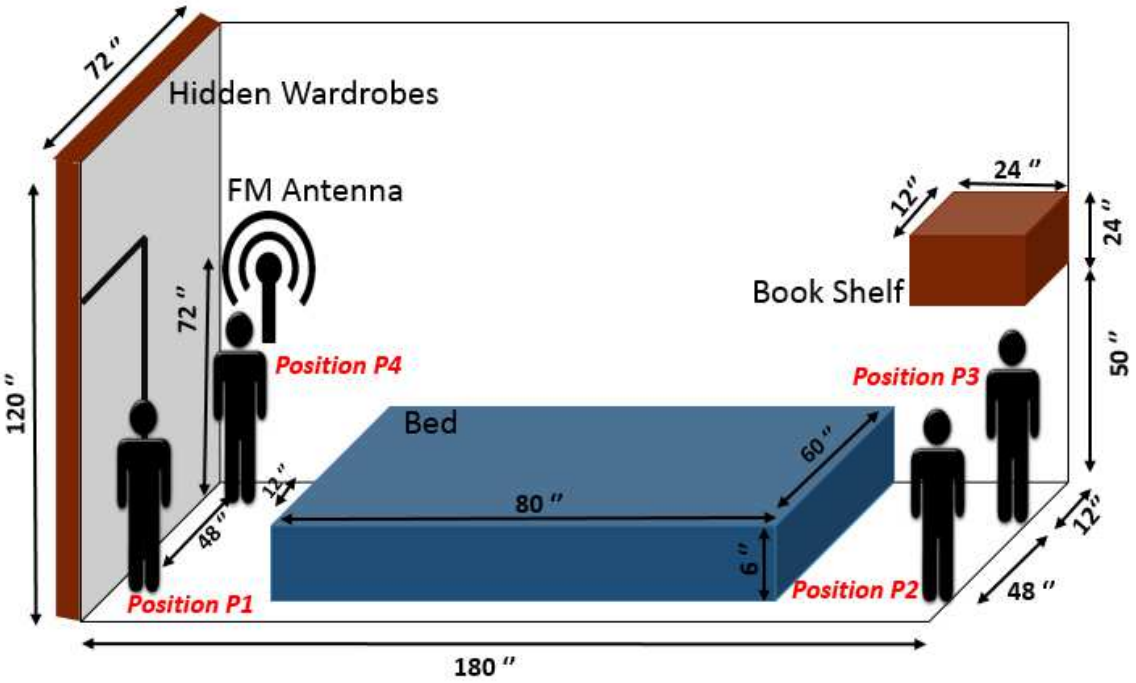

Figure 1Experimental Layout of the Room

5.2 Receiver Antenna Characteristics

The experiment mandated an off-the-shelf antenna. Of all the options that were available, researching the existing options of scientific as well as commercial antennas, a monopole antenna was chosen to be the perfect choice keeping in mind the low cost associated, frequency range to be scanned and small size of the room(a half wave dipole would have occupied larger area). The height of the antenna was selected so that it was possible to scan the FM frequency range from 88 – 108 MHz using the equation provided in (CSGNetwork, n.d.).

The material used to design the receiver's antenna was steel to minimize losses and maximize the reception. The adaptor between the receiver and the antennas was designed with copper to minimize contact losses.

The position of the receiver was finalized after considerable number of tests across the room. The room had four possible positions A, B, C, and D on the four walls of the room to place the antenna. The positions C and D near the window or an open door resulted in aberrations due to external noise and had to be overruled. The other position A near the entrance conflicted with position P1 and had to be rejected thus leaving the one in the corner B to be the ultimate choice for receiver's antenna to be placed.

Several research papers on localization systems tested indoors concluded that the optimum height to place receiver's antenna was 6 feet [Youssef references]. This was to accommodate the average height of people and a better line of sight that could be cut by the motion of the human to cause significant RSS variance.

Several hardware and software options were check listed for FM band monitoring but keeping in mind the low cost design of the entire system, we chose the Realtek RTL2832U. Also

this thesis was aiming to improve a previous research paper results using a single receiver for device free passive localization (Popleteev, 2014).

5.3 FM station filtering

The theoretical advantages of using FM radio signals for indoor positioning has already been mentioned in the previous chapters. The software and hardware utilized for scanning through the FM bandwidth and filtering the required channels will be discussed now.

The number of active FM stations or channels in the bandwidth 88-108 MHz depends on the city entirely. Also the criteria we used to filter the active stations for our final analysis were:

- To check for the 40dbU range within which if a certain station fell, we would classify that as a 'Strong Signal' according to FCC rules. (FCC, n.d.).
- To validate the consistency of the RSS values across time in an empty room; lesser variations meant better fidelity.
- To account for noise characteristics; some signals even though fell in the 'Strong Signal' region exhibited low tolerance to noise and fluctuated highly.
- To select only those stations that indicated significant and consistent variation to human presence in the room across different positions; this was done by running multiple preliminary tests run at different times of the day and validating the data to find RSS variations.

After crosschecking all these preliminary filters, we finalized five unique stations for our final analysis as 91.9, 92.7, 93.5, 98.3 and 101.9 MHz respectively. The experiment was conducted using a Windows OS loaded with a GUI built on the pyFMRadio library. The software and the hardware presented had certain limitations:

Individual Frequency Scanning: To analyze the RSS variations across many channels simultaneously was impossible using a single antenna, so we set filters to scan certain short bandwidths of 100 kHz around the central frequency of each individual station we finalized. Figure 2 is an example plot for the frequency station 101.9 while figure 3 illustrates the full scan of the entire bandwidth between 98 MHz and 108 MHz, the individual frequency bandwidths are highlighted for convenience.

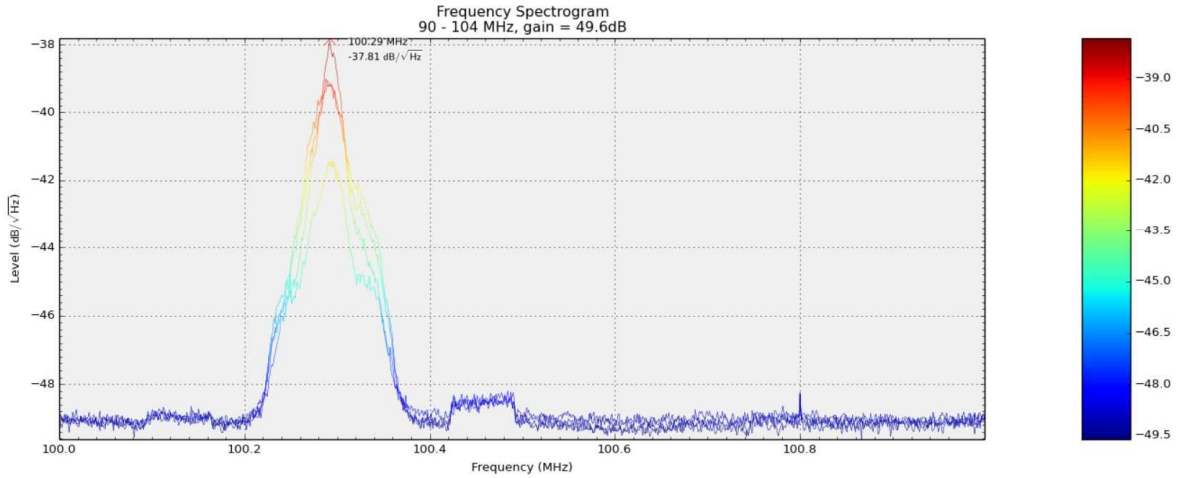

Figure 2 Sample Individual Frequency Spectrogram

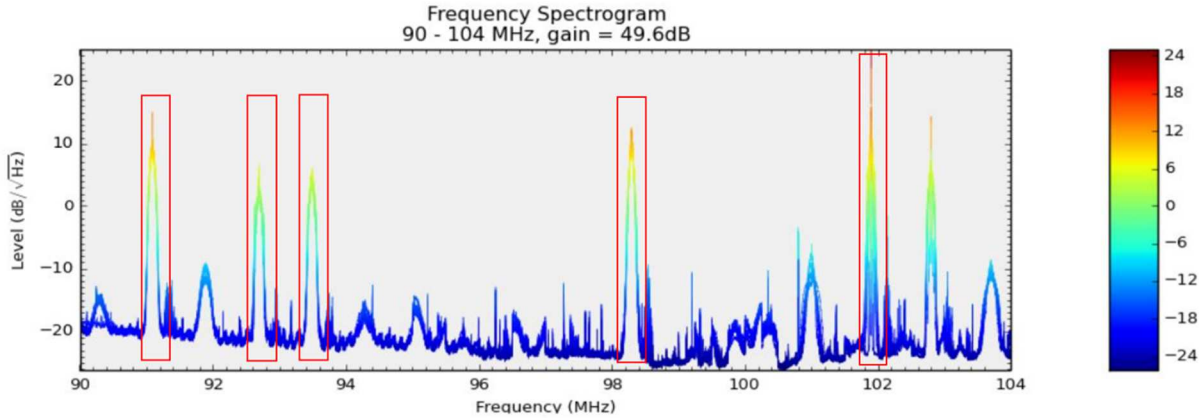

Figure 3 Sample Total Bandwidth Spectrogram

Dwell time selection: Dwell time for the RTL2832U is the time spent sampling at each frequency step. The scanning speed of the receiver depended on this 'dwell time' factor. It had values ranging from 8ms to 2048ms. We tested all these values and found that with increasing dwell time, the potential noise level decreased significantly. The tradeoff was that the total time spent to scan all the frequencies increased. After several tests, we selected 525ms as the optimum dwell time. Figure 4 shows how the amount of noise and minor peaks in the signal significantly drop as dwell time is increased while sampling.

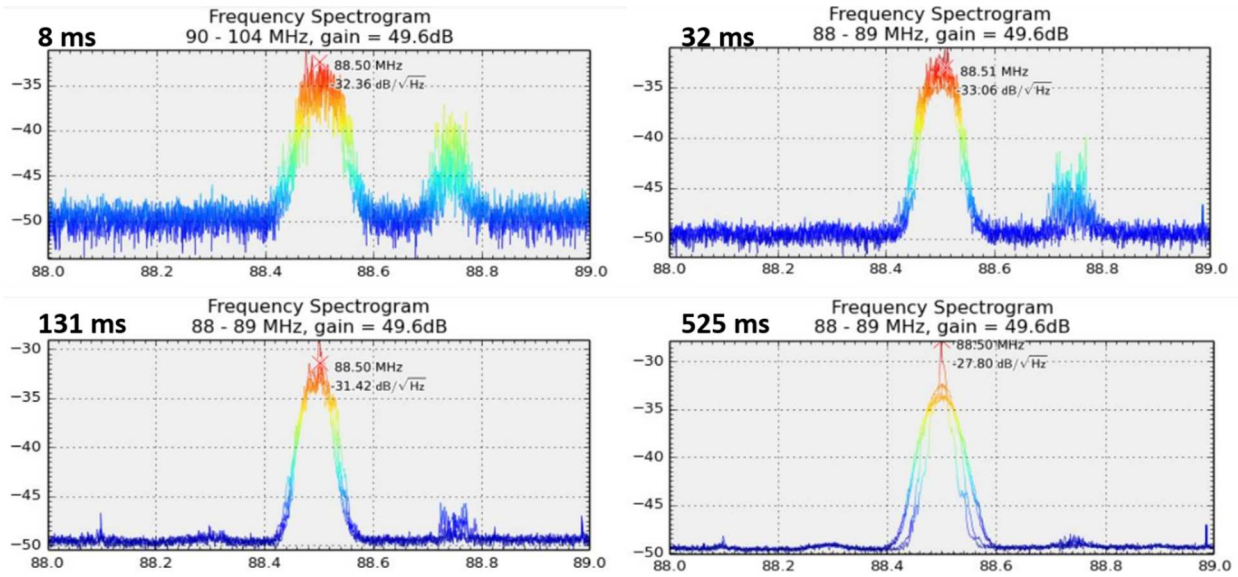

Figure 4 Dwell time comparison of RTLSDR Scanner

5.4 Raw sample collection

The room size, furniture settings and preliminary tests dictated five unique frequency channels and five different scenarios to be taken into consideration. To test the robustness of the system under the influence of heavy external noise, we selected after prior observation, 4pm every day as the optimal time to conduct the experiment.

The entire experiment was stretched to three separate weeks to assess the performance degradation of the system over time. The first three days from Monday to Wednesday on the first week, Monday on the second week and a final Monday on the third week which was to be used for testing the performance of the trained system.

On any single day, each scenario involved taking the RSSI readings for all five channels one after the other which we will refer to as a sweep or run for common understanding. Each scenario (except the empty room scenario) involved two sweeps; one without the test subject followed by another with the test subject in the appropriate position. The empty room scenario was followed by P1, P2, P3 and P4 scenarios.

The RSSI readings acquired when the Realtek RTL2832U (rtl, 2013) was employed to monitor over individual radio spectrum bandwidths was a power spectrum; a frequency versus signal strength curve. Each station with a bandwidth of 200 kHz resulted in 205 raw samples as shown in the figure 5. The frequency curves of 92.7 for all the five positions on the second day are presented here.

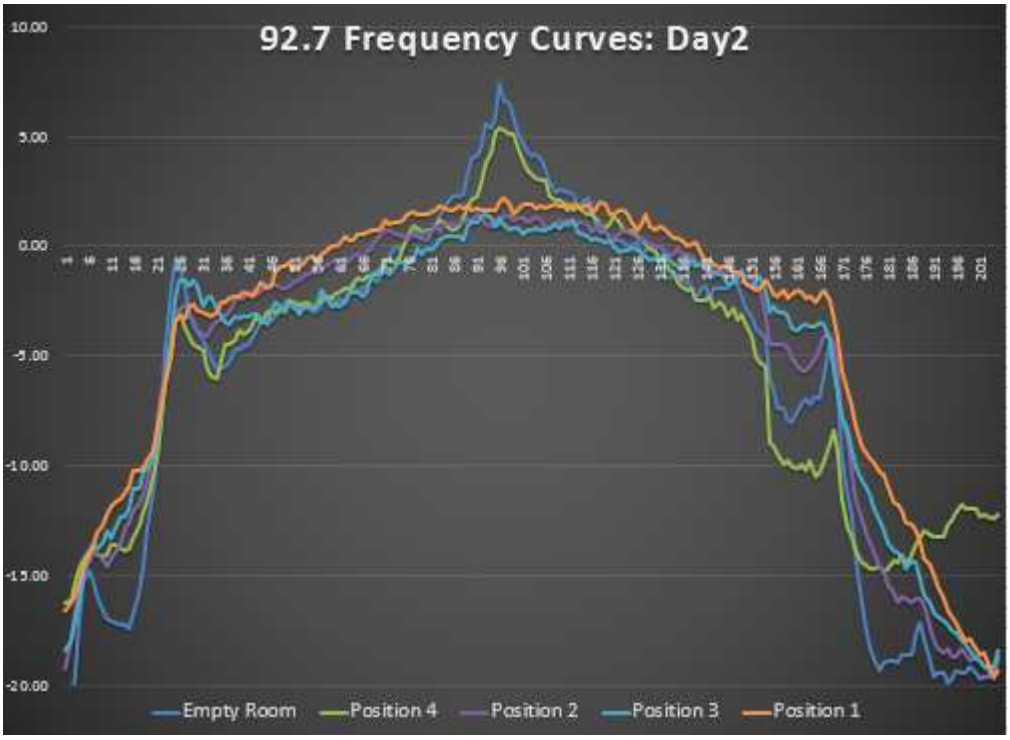

Figure 5 Frequency curves of all positions: 92.7

In total, for each individual frequency, a day concluded with collecting ten sweeps, each containing 205 samples. After the pre-processing stage which will be discussed in further chapters, we finalized five sweeps of 100 samples for each frequency that represent all the five scenarios. That gave us 25 sweeps on any single day encompassing five scenarios and five frequencies. This total dataset over five days of experimentation presented 125 curves which were used for the final analysis of the neural network model.

The next chapter explains in detail the feedforward back propagation based ANN model. The necessity for engaging a data pre-processing stage will be answered along with the core evaluation of the training and testing stages of the model.

CHAPTER 6. DATA ANALYSIS STAGE

In a general positioning system based on fingerprinting, the data analysis needed to associate acquired fingerprints with those already collected in the offline phase was performed in a calibration stage. These methods are ranked and valued according to their error percentages in localizing certain unknown positions. Contrary to routine methods, our fingerprints for each individual position on any day are five signal curves with respect to five different frequency channels. The model is trained with these five fingerprint curves across multiple days to make sure signal fluctuation behaviors are learnt.

This approach is a unique way of fingerprint classification which is not dependent on RSSI variation with respect to distance. The former is flagged by various limiting factors such as multipath fading and NLOS reception. The model prediction percentage was found to be around 99 to 100% when verified with curves that were already part of the training set. The testing curves that were not part of the training set confirmed the same around 80%. These results for a low cost system utilizing just one over-the-shelf receiver and antenna are very encouraging even when compared with a previous research in the same category (Popleteev & Engel, 2014).

The following sections describe the data analysis methods employed and the three stages of the model.

6.1 Data Pre-Processing Stage

Artificial neural networks are a data driven technique whose learning improves as they train with more data. This however is based on a premise that larger training samples contain more events of varied types hence improving the generalization ability of the ANN model. A large amount of input data when given to the models without any pre-processing may lead to larger computational time, more effort and may lead the model to end up a local optima rather than a

global optima (Nawi, Atomi, & Rehman, 2013). Irrelevant data training once when reduced can lead to reduced confusion in the model and thereby benefit faster processing of data (Fazel Famili, W. M. Shen, R. Weber, 1997) . Repetition or longer training with larger datasets may also overfit the series (Gaweda, Zurada, & Setiono, 2001)(Fernando, Maier, & Dandy, 2009). Shailesh Kumar et al in a paper on training ANN using information rich data (Singh, Jain, & Bárdossy, 2014) confirms that ANNs trained with pruned datasets (information rich) after pre-processing gave similar results as the ANN trained using the complete datasets.

From the above discussion, pre-processing of data can be deemed an efficient step if not mandatory when training a data driven model like the ANN. The term "rich" was used to denote very high information content in the original paper (Singh et al., 2014) and will mean the same in our thesis too. Categorizing the raw data to filter out information rich data is highly dependent on the type of data and the model we are dealing with. Critical events mainly influence the training process and calculation of weights in a data driven model and can be the deciding factor in a tradeoff between 'large computational time' and 'higher number parameters/ complexity'.

The raw data that we collected at each position for an individual frequency channel comprised of 205 samples representing the RSSI values spread over a 200 kHz bandwidth. As depicted in the figure 5 each individual sweep represented a bell curve. During experimentation, at each individual position we recorded two sweeps (even though redundant) to account for stabilization of the receiver. This required us to develop filters to select one among the two curves for the final analysis.

Conventional pre-processing techniques such as transformation and normalization of data were unsuitable as we dealt with power signals which were non-linear in nature. Subtle changes distinct to each curve represented the RSS variation during the presence of a human. Each

individual frequency depicted a unique variation in all the five scenarios. Also the vision of this thesis was to build a faster online localization system which does not require complex pre-processing that may hinder the purpose or function of the system.

To accommodate all these requirements, visual analysis and simpler mathematical cues were considered when building the three filter technique. Before we divulge in detail into the filtering process, the signal process under visual analysis showed that the signal was most turbulent or erratic in the starting and the ending stages as evident from figure 5.

The contribution from the turbulent starting and ending samples to the pattern recognition model were very minimal. The central portion was found to be stable after thorough and critical analysis of the curves. After repetitive tests with various dataset sizes, the first 45 samples and the last 60 samples were truncated from the original data before being inserted into the data pre-processing stage. The pre-processed signal curves of those represented in the figure 5 are displayed in the figure 6.

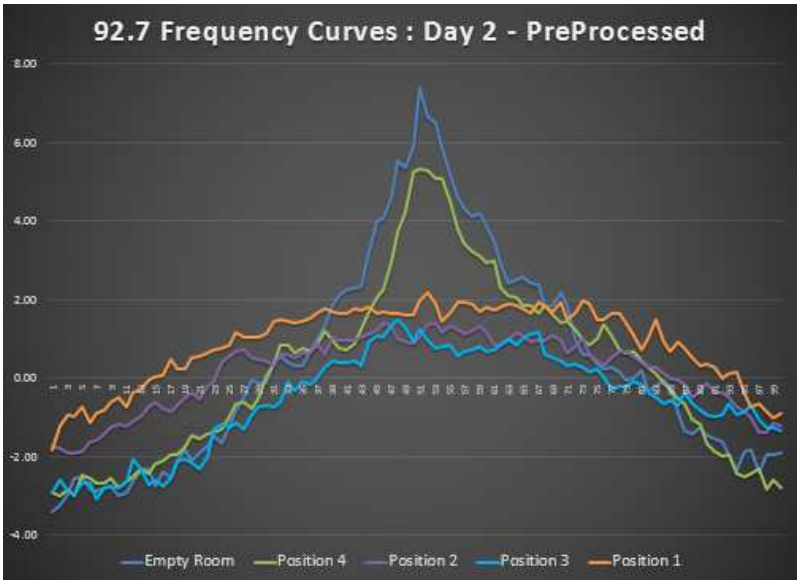

Figure 6 Signal Curves after Pre-Processing: 92.7

This was the preliminary stage where we established raw information rich data that helped us consolidate the model's performance. However we compared the performance analysis of the model against the truncated and actual dataset which lead to almost identical results.

The base signal however when comparing the two signal curves/ sweeps was taken as their average curve since each day's data was independent and the average curve was a standard that was employed by researchers in the past (Popleteev, Osmani, & Mayora, 2012).

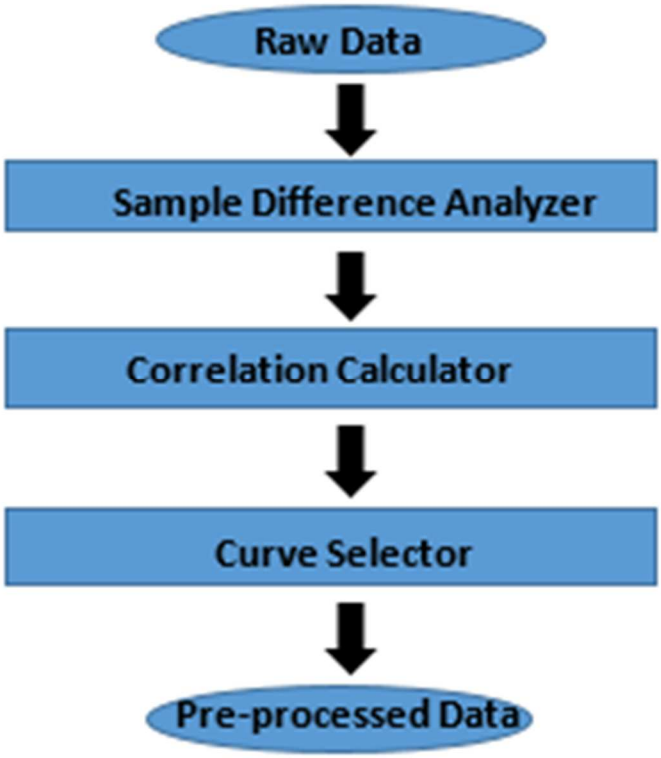

Figure 7Pre-Processing Stage Flow

Sample Difference Analyzer: The first filter was intended to find curves that were fundamentally close in the signal space to the average curve. The process involved finding the average of total sum of sample differences between two curves under consideration.

Correlation Calculator: Pearson product-moment correlation coefficient (sometimes referred to as the PPMCC or PCC or Pearson's r) is a measure of the linear correlation (dependence) between two variables X and Y, giving a value in the range [-1,1]. The results are interpreted as follows.

- 1 is total positive correlation
- 0 is no correlation
- −1 is total negative correlation.

The coefficient when applied to a sample is referred to as the 'Sample Pearson correlation coefficient'. It is widely used as a measure of the degree of linear dependence between two variables. This filter extended the same concept to verify how close the individual signal curves were to the base curve.

Curve Selector: The two filters when applied to the curves under observation give individual results which have to be jointly analyzed to filter the final curve for that particular frequency channel. Common logic dictates that the curve with the lowest sample difference and a confidence correlation coefficient value of 95% to 99% (or the one closest to 1) is the one to be selected.

The result of the data pre-processing stage are 100 sample curves that closely reflect the RSS variations observed on each of the five frequency channels at every position. In total we have five signal curves that act as a diverse fingerprint for each position on any day. The total dataset of each day amounts to 125 curves which will be used for training the ANN model in the next stage.

6.2 Setting the Target Outputs

The targets of a supervised learning based model have to be set prior to the training stage. The experimental data that we worked with were signal curves representing radio signal strengths

observed across respective bandwidths. The aim of our model was to predict the position of the human target based on a finger print that contained five unique signal frequency curves. Every dataset representing a day brought in a different pattern for each position and the job of the model was to learn these variations. The model's accuracy was tested on how it would predict new untested data which was close to the training data.

To summarize, the target outputs we selected had to satisfy these criteria:

- The output target which we selected would have to represent how each of the 100 samples varied individually for every frequency.
- They should in essence capture the range within which the training curves of individual frequencies vary. This would then set limitations within which the test data had to fall to be predicted by our model.

The optimal target output after analyzing the available methods was Cross Correlation. The average curve set was first formed by averaging individual frequency curves of the training set. Cross Correlation between each individual curve and the average curve served as the target output that the model had to achieve.

The following figure 7 explains how we arrived at the target outputs for each neural network. Each frequency had four days of data for the training set, the average curves of each position were calculated.

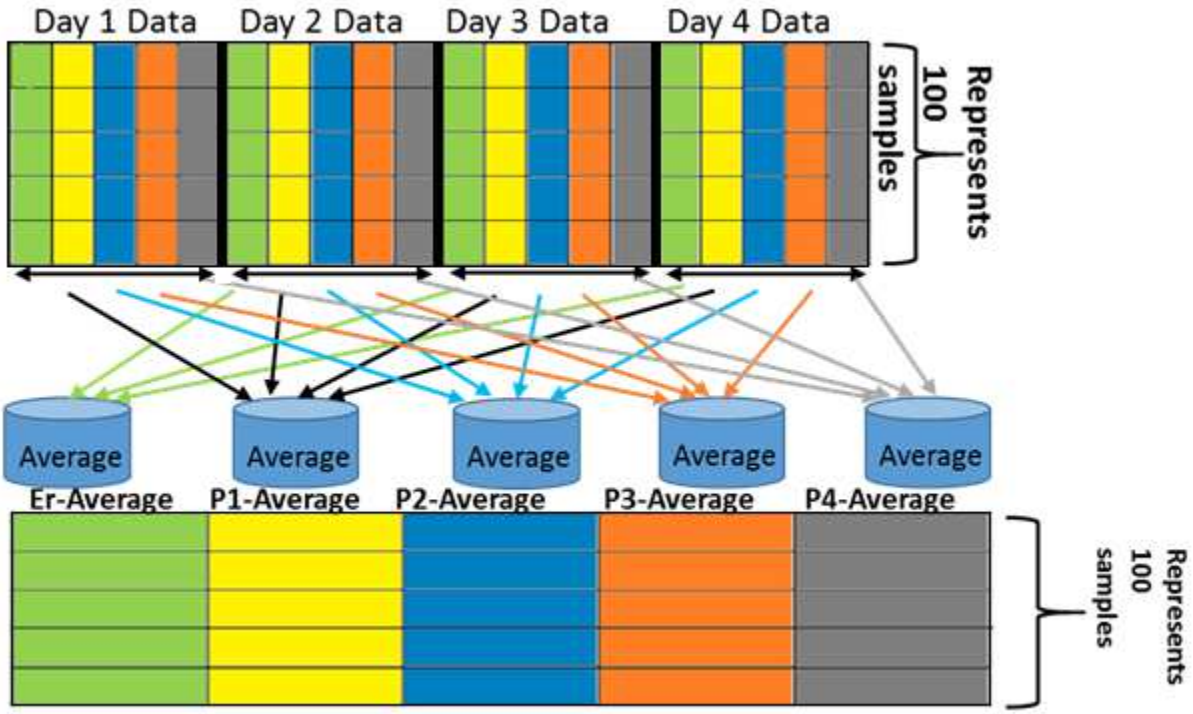

Figure 8 Data Analysis Representation

From figure 8, the average curve set was used to feed the next step where we found the cross correlation values of each individual curve across the five positions (ER, P1, P2, P3 and P4). This process is depicted in the next figure 9 for easier understanding.

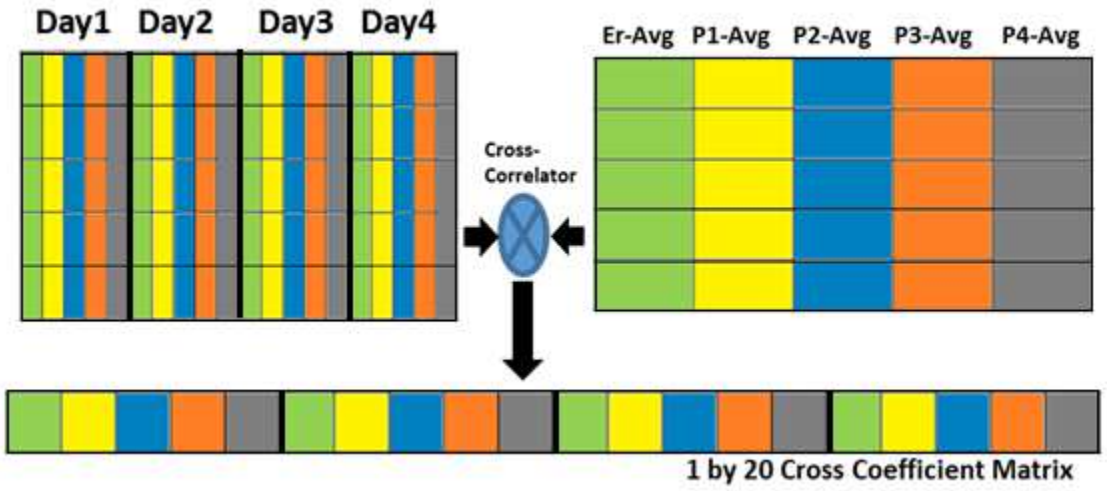

Figure 9 Cross-Coefficient Matrix formation

The 1-by-20 cross coefficient matrix was denoted by a diagonal matrix with the diagonal elements representing the cross coefficients for the respective signal curves and the other elements by a zero. This was done to match the usual style of target outputs of a neural network model. The next figure is an actual target output for the frequency 91.9 where the first four days were used for the training set.

Empty Room				Position-1				Position-2				Position-3				Position-4			
0.967	0.954	0.951	0.948	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0.912	0.863	0.89	0.929	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0.915	0.915	0.966	0.906	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0.926	0.952	0.925	0.953	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.875	0.943	0.944	0.957

Figure 10 Sample Output for a Training Set

This approach was used to form the outputs based on all our respective training datasets. The next chapter deals with tuning parameters for the neural network while training and the actual steps taken to achieve generalization.

CHAPTER 7. TRAINING, TESTING, OPTIMIZATION AND GENERALIZATION OF NEURAL NETWORKS

7.1 Neural Network Training Stage

This section explains in detail the training process that the model goes through. The question of how and when the parameters of the neural network were tuned to achieve the required targets will be discussed. Any neural network's performance is defined by the way it chooses the initial weights, number of hidden layers, number of neurons in each layer, the training, transfer and performance functions, learning rate and network architecture. There is no general consensus or a universal rule which defines a perfect combination of these parameters for the best performance. However certain thumb rules exist which are usually followed to reduce the 'trial and error' runs that lead us to the final optimal combination of parameters. These are not always fruitful considering the diversity of applications neural networks are used for.

When analyzing the effect of numerous parameters on the efficiency of a model, it is common logic to assume the least affecting factors as constants or unchanging to make the task and analysis of the model better. In this thesis, some of the parameters were left to assume their default values so that the major parameters could gain the limelight.

7.1.1 Initial Weights

The first task during any neural network modelling is to initialize the weights with which the network would start. In the case of prior information being available, it is usually advised to use that information to guess the initial values. Choosing uniform weights bodes ill for the network as the error is back-propagated in the network through the weights in proportion to the value of weights. This leads to every hidden unit connected to the outputs receiving identical error signals and them transferring back the same. This is called the symmetry-breaking problem (Bashiri &

Farshbaf Geranmayeh, 2011). Internal symmetry also lead to multiple temporary minima or flat valleys and cost function landscape periodicities. In short the performance of the model gets affected.

How to avoid these problems? The customary practice is to set the weights of the network to random numbers that are uniformly distributed inside a small range of values. Having larger weights saturates the sigmoids from the very beginning of training and the system gets stuck in a saddle point. This phenomenon is called premature saturation (Bashiri & Farshbaf Geranmayeh, 2011). This is avoided by choosing smaller initial weights and distributed inside a small range of values.

In this thesis, as discussed the training started with random weights which is the default method available in MATLAB while creating a feedforward network. After the model reaches its optimal MSE value, the network as a whole is saved. This stores in memory the particular network weights that resulted in the best performance which can be used to test the generalization capability of the model.

7.1.2 Number of layers

The design of the neural network architecture is the selection of the number of layers and the number of nodes in each of those.

Every neural network in general has exactly one input layer, no exceptions howsoever. The number of neurons however vary according to the number of features in the training data. In the training data for each of the neural networks, four days of data typically were used as the training set. This resulted in 20 columns. The number of samples considered were 100, which were set as the number of neurons in the input layer.

Similar to the input layer, every neural network had exactly one output layer. The size of this layer depended solely on the chosen model configuration. Our targets had to be classified into five different positions and this lead to the setting of the output layer size as 5.

In the literature it was mentioned that MLPs with any of a wide range of continuous nonlinear hidden-layer activation functions, one hidden layer with a large neuron level suffices as an universal approximator (J. L. Castro, Mantas, & Benítez, 2000; Hornik, Stinchcombe, & White, n.d.). Hence in this thesis the training of the model was limited to one hidden layer.

7.1.3 Number of hidden neurons

The number of hidden layer neurons however has had much research done by the NN community to get a fix or an estimate on the optimum number of neurons. But there is no theory that can confirm a fixed number of hidden neurons that are needed to achieve the best performance in any NN model. (faqs.org, n.d.). This is because hidden neurons can greatly influence the network output.

Quite a few researchers have come up with thumb rules to determine the optimal number of neurons in the hidden layers (Lahiri & Ghanta, 2010), such as the following:

- The number of hidden neurons should be somewhere between the input layer size and that of the output layer.
- The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer.
- The hidden neuron number should be less than twice the size of the input layer.

These three rules are just a rough estimate to starting the training process. Ultimately, the selection of an optimal architecture for the neural network is determined by the problem at hand and by using trial and error.

The stability of the network is estimated by the error the network produces which is affected by the hidden neuron level. In most situations, there is no right way to determine this number without training several networks and calculating the generalization error of each case. If the model is trained with too few hidden units, it results in high training error and high generalization error due to underfitting and high statistical bias. Excess number of these cause an overfitting problem which greatly degrades the generalization capability of the network, this will be discussed further into the thesis (Sheela & Deepa, 2013). The complete list of factors which in a complex way affect the best number of neurons required are:

- the input and output numbers
- the training cases
- the noise in the targets
- the complexity of the classification to be learned
- the network architecture
- the type of hidden unit activation function
- the training algorithm
- regularization

In this thesis, various sizes of the hidden layer were tested for their error values to find the optimum number of hidden layer neurons. The default hidden layer size was 10 for the feed forward backpropagation network.

7.1.4 Transfer Functions

The ability or the success of a neural network lies in its ability to transform the transfer level of summing the weighted inputs of neurons into outputs. This is the sole purpose of a transfer function. The transfer functions of a neural network are classified into three types – threshold, linear and nonlinear. Based on the layer, the functions used vary. The hidden layer usually uses a non-linear function to map the complex relations between the inputs whereas an output layer can employ either linear or nonlinear functions.

MLPLs are characterized by their transfer functions. MLPs use sigmoidal transfer functions which are universal approximators. The speed of learning and final network complexity varies between networks based on the choice of their transfer functions. In real applications, it is advised to zero-in on an appropriate model before starting the tiresome task of selecting the best architecture and network training.

Since MATLAB was being used, the available built-in transfer functions were preferred. Four functions namely Linear (purelin), Hyperbolic tangent sigmoid (tansig), Logistic sigmoid (logsig) and Gaussian RBF were available, of which only the first three were suitable to be used by a MLP network. The default functions that were used by a three layer network were tansig for the hidden layer and purelin for the output layer which were sufficient for the classification problem we were trying to solve.

7.1.5 Training Algorithm

Of all the additions to neural network training algorithms, the most significant achievement was the introduction of steepest descent algorithm or error backpropagation algorithm (EBP) (D. E. Rumelhart). Many improvements though made to the EBP were clouded by the path breaking Levenberg-Marquardt algorithm (Yu & Wilamowski, 2011). The slow convergence of the EBP

due to the step size being adequate to the gradients was one reason for its inefficiency. The other being the classic "error valley" problem that might be a result of its slow convergence.

The slow convergence of the steepest descent method was greatly improved by the Gauss–Newton algorithm (Osborne, 1992) but only when the quadratic approximation of the error function was reasonable. LM algorithm blended both these methods and thereby inherited the speed of the Gauss- Newton and the stability of the steepest descent algorithm. It is a combined training process which switches to the steepest descent until the curvature is appropriate to make the quadratic approximation and then back to Gauss-Newton to speed up the convergence. This is one of the fastest backpropagation algorithms in the MATLAB neural network toolbox and hence was chosen to supervise model training.

The steepest descent algorithm is a first order algorithm which uses the first-order derivative of the total error function to find the minima in the error space. This is defined as the gradient *g*.

The Newton's method assumes that all gradient components are functions of weights and the weights are linearly independent. Comparing with the steepest descent method, here the second-order derivatives need to be calculated which is called the Hessian Matrix. Hessian Matrix gives us a proper evaluation of the change in gradient.

However calculating the second order derivatives can get very complicated. To simplify this process, Gauss-Newton method introduced the Jacobian Matrix *J which is defined by the following equation.*

$$H = J^T J \qquad (7)$$

The Levenberg-Marquardt algorithm introduced another approximation to the Hessian matrix by adding a component μI.

$$H = J^T J + \mu I \qquad (8)$$

Where μ is always positive, called combination coefficient and I is an identity matrix.

When the value of μ is very small, the LM algorithm switches to the Gauss-Newton method and when it is very large, it tends to the steepest descent method. The inverse of the combination coefficient μ is the learning coefficient α or the step size used in the steepest descent algorithm.

Of all the training algorithms that MATLAB provides, LM has the fastest convergence for typically the majority of the problems. The advantage is very noticeable when accurate training or low mean square errors are the requirement. Since the performance function that was considered involved MSE and also the dataset being noisy, LM was relied on for training the model.

However one of the disadvantages this algorithm faces is high storage requirements which can be traded off for an increased execution time. The algorithm loses its advantage as the number of weights in the network increases.

Every training algorithm requires to set itself a stopping criteria. The training process stops when certain limits are usually reached as shown below.

- The model reached the maximum number of epochs set.
- The maximum amount of time when exceeded.
- Performance had reached the set goal.
- The performance gradient fell below the minimum gradient (min_grad).
- The value of mu surpassed mu_max (maximum learning rate, mu).

- During validation, if the model's performance had increased more than max_fail times since the last time it decreased.

7.1.6 Network Training-Stopping Functions

The training stops once the NN work meets the conditions mentioned in section 4.2.3 are met. The most profitable and widely used performance functions preferred by neural network researchers for a research problem of our type are:

MSE: Mean squared error (MSE) of an estimator is a quantification of the average measures of the squares of errors, which is the difference between the estimators and predicted values. It indicates the closeness of the fit of the model to the data. It is a comparison between the target and the predicted values. MSE is an absolute measure of the fit when compared to R-squared value which is more relative. MSE can be interpreted as the standard deviation of the unexplained variance, and has similar units as the response variable. Lower the value of MSE, better the fit.

R-Squared: This is a measure of how well the variations in the outputs are explained by the targets or the goodness of fit. R-squared ranges from zero to one, with zero indicating no improvement over the mean model in terms of prediction whereas 1 denotes perfect prediction. As obvious as it is, larger the value, better the fit.

Network Training Time: The training time is denoted by the time spent between the start and end of a training process. This measures the performance of how quickly the particular parameters have helped in reaching the desired fit. Lesser the training time, better the model.

In this section the training process which included the tuning of network parameters, selection of the best training algorithms and performance functions were overviewed thoroughly.

The next section shall deal with the issues faced while training a neural network model and ways to improve the generalization capability.

7.2 Improving NN Generalization

Once the learning process of a neural network is completed and a model is obtained, the model then is supposed to have the general ability to predict the accurate output when tested with data that was not used in the training process. This ability as explained throughout this thesis is referred to as the generalization capability of the model.

Generalization is not a given property and is usually learned by the network through a practical knowledge of what the relevant inputs might be, an adequate approximation of the function that is to be learned and the cases we need to generalize have to bear some sort of resemblance to the training cases. These are some rough conditions which are necessary but not necessarily sufficient to be met to achieve good generalization.

Neural networks are usually plagued by two phenomena which affect the learning and generalization capabilities.

7.2.1 Overfitting

An overfitted model has an excess of parameters. The added complexity often aids the model in performing well on a set of training data but inhibits any future prediction. This occurs when the neural network had been over trained with noisy or imprecise data during the learning phase. Thus, the model could achieve 100% accuracy to classify a given data set; however, it will not have a strong generalization ability of new input data (Srihari, n.d.). To add to this problem, datasets usually have some amount of inbuilt noise.

7.2.2 Underfitting

This phenomenon is the opposite of overfitting. This means that the model is incapable of capturing the variability of the data. This can be explained with an example. When training a linear (y=ax+b, where a and b are constants) classifier on a data set that is nonlinear like a parabola (Yarkony, 2008). The resultant classifier shall have no predictive power over the training data. The result is underfitted model, or a model that is too simple to describe the data.

The number of neurons in the hidden layers is a very important part of deciding the overall neural network architecture. Even though the hidden layers do not directly interact with the external environment, they have tremendous impact on the final output. Choosing the number of hidden layers and number of hidden neurons must be carefully done.

Underfitting is a result of having too few neurons in the hidden layers to effectively detect the signals in a complicated data set. Using too many neurons in the hidden layers can result in several problems. First, too many neurons in the hidden layers may result in over fitting. An exceedingly large number of neurons in the hidden layers can increase the time it takes to train the network. This training time can increase to the point that it is impossible to adequately train the network. A logical equilibrium point has to be reached when choosing between too many and too few neurons in the hidden layers.

Underfitting and overfitting identify the bias and variance dilemma in any model. To understand these concepts, we need to understand the concept of test error and training error and how they are related. These in turn explain the bias and variance tradeoff.

7.2.3 Training error versus Test error

The training error is the one we calculate by applying the model to the observations or data that had already been used in the training. In contrast, the test error is the average error that results from using the trained model to predict the response on data that has never been part of the training.

It has to be understood that both these errors are quite different even though they both explain the fit of the model. The training error can dramatically underestimate the test error. Typically, the training error is quite low compared to the test error as the former is computed on data that the model has already seen, so it fits the training set with a lower error than it was going to with the test set. Retraining the network with more reruns will only make the training error lower and might not necessarily lower the test error (Casella, Fienberg, & Olkin, 2006). This is because the model fits the training data perfectly but might not do the same with data it never worked with.

Figure 11 summarizes these concepts. As the model complexity or the number of features increases from low to high, the training error decreases. The test error on the other hand decreases for a while to a moderate level of model complexity for most of the models and goes up again. This is because, on the left due to addition of certain features that add to the complexity; we lower the error. But after a point any more added features are just noise and lead to an increase in the test error and an obvious decrease in the training error.

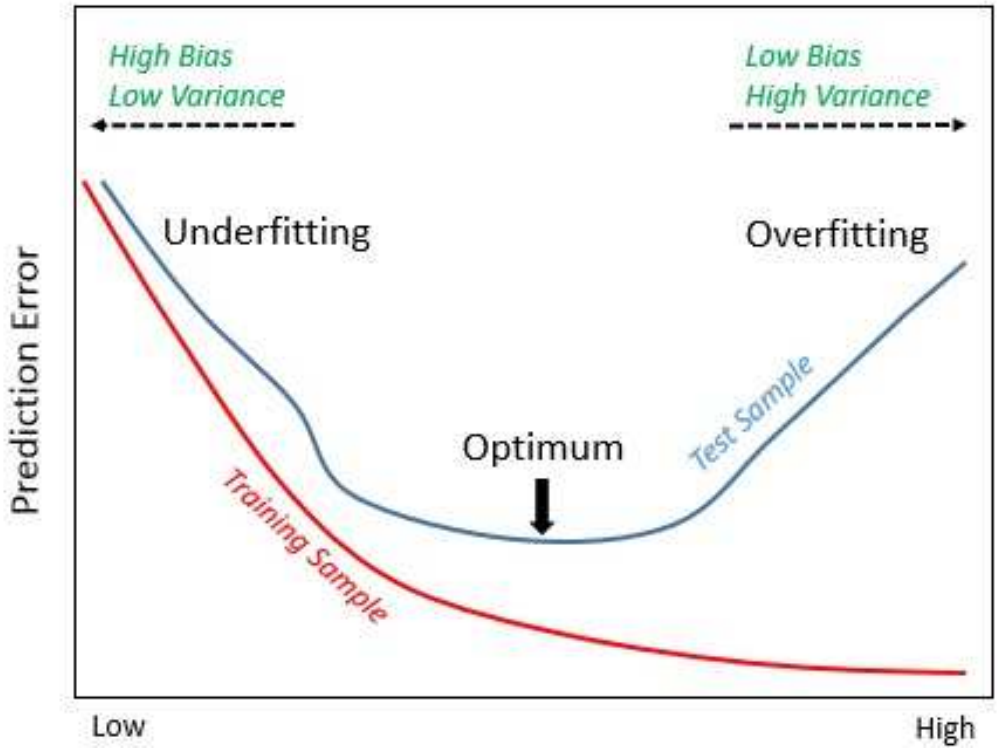

Figure 11 Complexity of a Neural Network Model

7.2.4 Bias-Variance Tradeoff

The ingredients of the overall generalization error is the result of the two concepts bias and variance (Revisited & Under-fitting, 2004). Bias occurs when the network tries to fit all data points including the noises. In contrary, variance problem addresses the smoothness of an approximated model in comparison with the actual underlying function that generated the training data.

When the model doesn't fit very hard, the bias is high and the variance is low as we observed at the start of the training process due to very few parameters being fit. As the complexity increases, the model can fit to more and more subtleties in the data, so the bias decreases. But the variance increases as there are more and more parameters to be estimated from the same amount of data. As model becomes overfit, it reaches a very low bias and large variance.

To reach a good tradeoff between bias and variance and also to better training and test errors, the complexity of the model was slowly increased either by adding more epochs (or runs) or increasing the number of hidden layers, neurons to a point where the test error stops decreasing and starts increasing. The training error at such a point will not be the least, the model can achieve but we would have reached the optimum generalization capacity of the model.

One major way to improve network generalization is to use a network that is just large enough to offer the perfect fit. The larger the networks that are employed, more complex the functions that can be created. Unfortunately, the requirements of each network vary and it is difficult to predict it beforehand. The best method proven to generalize neural networks is the Bayesian regularization.

7.2.5 Cross Validation

One widely employed method called early stopping through k-fold cross validation produces good results and shall be discussed in detail. This approach is a noticeable improvement on the split-sample method which allows you to use all the data for training. The approach divides the data into k subsets of equal size. The network is trained k times, each time leaving one subset from training which is used to test the network and compute the error. If k equals the sample size, this is called the leave-one-out cross validation. The total error of the network is later computed by averaging the individual errors using k. 5-fold and 10-fold cross validation are the most popular forms used in data mining and machine learning fields. MATLAB framework provides four functions for dividing data into training, validation and test sets. They are dividerand (the default), divideblock, divideint, and divideind. In this thesis, all the random selection of data was done using dividerand as it usually provides the best data selection possible.

However the disadvantage of employing cross-validation is that there is a need to retrain the net many times.

7.2.6 Regularization

This concept which was first introduced by Tikhonov (Linear, Rls, & Theorem, n.d.) involves modifying the performance function, which is usually chosen to be the sum of squares of the network errors on the training set.

The typical performance function used for training feedforward neural networks is the mean sum of squares of the network errors. To improve generalization of the model, the performance function has to be modified by adding a term that consists of the mean of the sum of squares of the network weights and biases:

$$msereg = \gamma mse + (1 - \gamma)\, msw \qquad (9)$$

where γ is the performance ratio.

This modified performance function leads to smaller weights and biases forcing the network response to be smoother and thus making it less likely to overfit.

However the problem with regularization is the difficulty in determining the optimum value for the performance ratio parameter. If this parameter is set too large, it might result in overfitting and if the ratio is too small, the network does not adequately fit the training data. To avoid this confusion, automatic regularization is preferred (MATLAB, Bayesian Regularization, n.d.).

One of the approaches to automatic regularization is the Bayesian framework. The network weights and biases are assumed to be random variables with specified distributions in this framework. The regularization parameters are related to the unfamiliar variances associated with

these distributions. The parameters are estimated later using statistical techniques. A detailed discussion of Bayesian Regularization is beyond the scope of this thesis. This regularization was implemented with the help of the trainbr function available as part of the neural network toolbox in MATLAB (MATLAB, Bayesian Regularization, n.d.).

The performance function that is employed when using trainbr is the sum of squared errors (SSE) which as the term defines is the sum of the squares of the differences between the predicted values and target outputs. To know when the network had reached its optimum performance, it is important to let it run until the effective number of parameters have converged. The training might stop by displaying a message "Maximum MU reached" which is typical and a good indication that the algorithm has truly converged. Another indicator is the sum squared error (SSE) and sum squared weights (SSW) becoming relatively constant over several iterations. This dictates us to stop the training by clicking the **Stop Training** button in the training window.

In summary regularization usually ensure good network generalization when they are applied properly. Faster algorithms like trainlm when used have to be set to converge slowly. With both cross validation and regularization, it is always advised to start training the network from several different initial conditions. Though nothing is given; by checking the performance of each condition it is possible to achieve robust network performance. Bayesian regularization provides better generalization performance than cross validation for small datasets and when training function approximation networks. This is because Bayesian regularization uses all the data and requires no separate validation and training data set.

The next section describes the post-training analysis of the neural network. We preferred to elucidate the process of training, validating and testing a NN model using the MATLAB

framework. The section following that is dedicated to performance evaluation using regression analysis between the network response and corresponding targets.

7.3 Performance Analysis

7.3.1 Development of Neural Network Model

MATLAB (MATLAB, mathworks, n.d.) is a numerical computing environment and programming language which allows easy matrix manipulation, plotting of functions and data, implementation of algorithms, creating user interfaces and interfacing with programs in other languages (MATLAB, mathworks, n.d.). Our concern is the Neural Network Toolbox provided by MATLAB for designing, implementing, visualizing and simulating neural networks. It enables users to design and manage neural networks in simple ways with graphical user interfaces (GUIs) to support the programs.

In our thesis, MATLAB (R2009a) (MATLAB, mathworks, n.d.) was used to write script files to filter the raw data and pre-process it, developing the MLP neural network models, evaluate their performance using statistics such as R-Squared, MSE and Regression analysis and plot the results using the nntraintool GUI (MATLAB, Improve Neural Network Generalization and Avoid Overfitting, n.d.). Figure 12 lists the procedural steps in developing the NN mode.

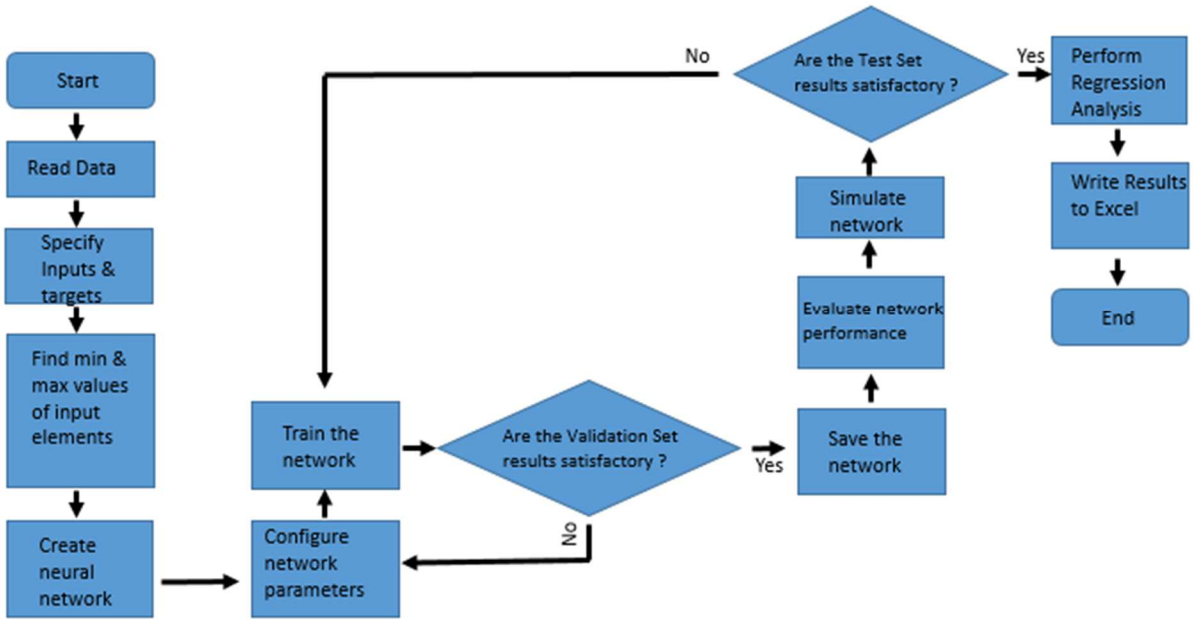

Figure 12 Neural Network Lifecycle Flowchart

The MLP program starts by reading data that can be imported from an excel file. The inputs and targets are stored as variables in the MATLAB workspace which makes it easier and faster for the MATLAB program to fetch. The minimum and maximum values of the input data are calculated for use in the "newff" function (ucebna, n.d.).

The MLP feedforward NN is created using the newff function which takes a R*2 matrix of maximum and minimum values, matrix containing the layer sizes, the transfer functions used in the layers which in our thesis are "tansig" and "purelin", the training algorithms which in our case are either "trainlm" or "trainbr". Figure 13 is a screenshot while the network trains to reach the goal set, the algorithm used is 'trainlm' and the performance function is 'MSE'.

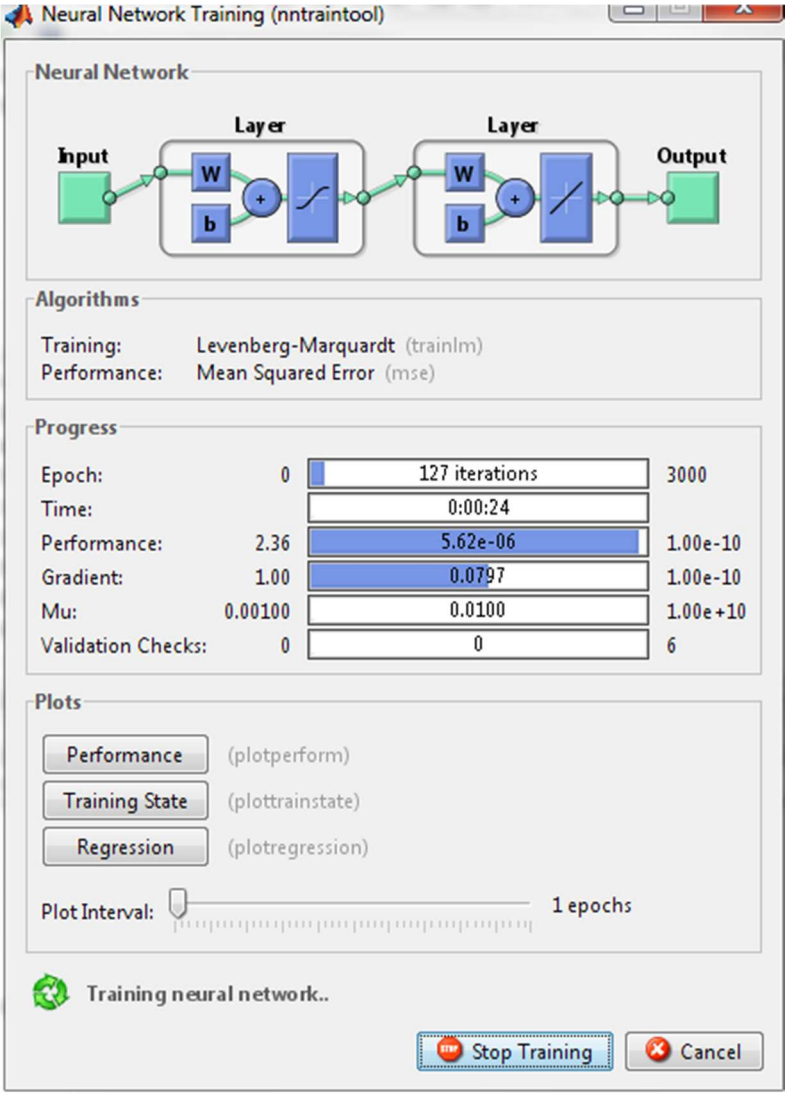

Figure 13 Training a Neural Network Example

The corresponding network parameters are chosen and varied according to the performance of the model. The network is then trained until it reaches the set goal or one of the stopping conditions as discussed force it.

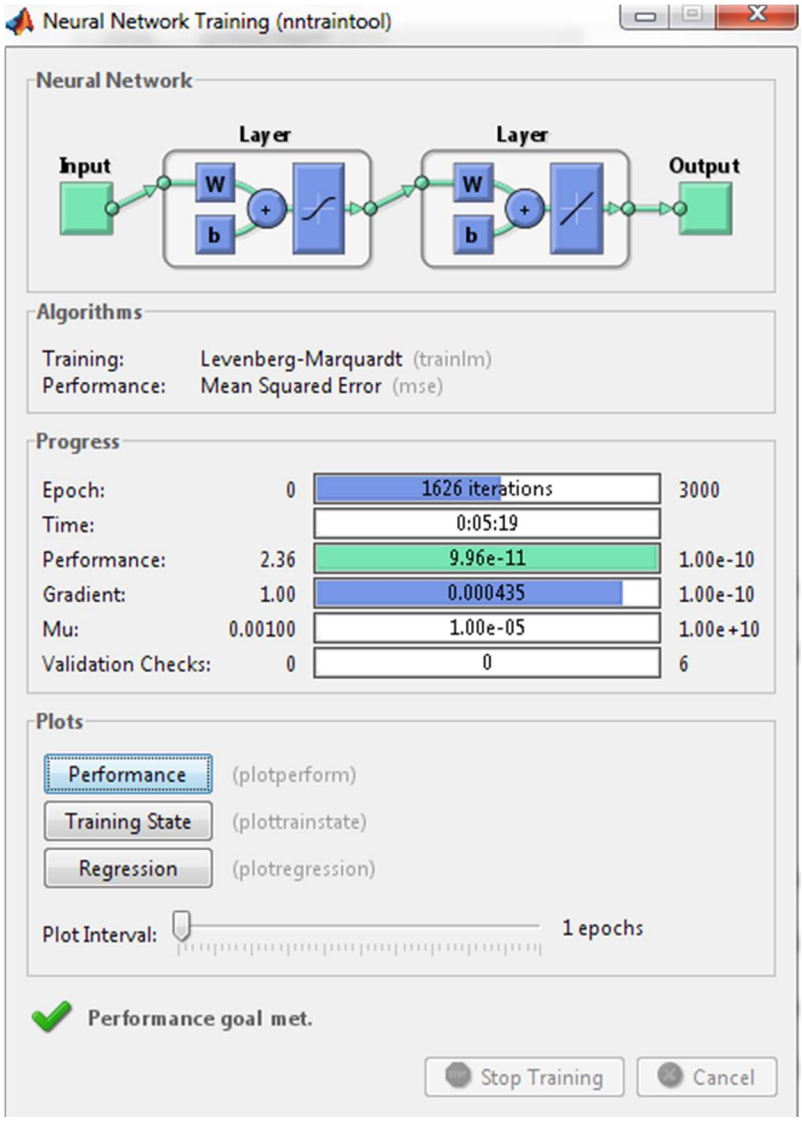

Figure 14 A Trained Neural Network Example

After the model reaches a satisfactory level, it is saved to store the weights that resulted in the best results. The training performance function used is MSE while the post-training analysis is done using regression. The network is then simulated to check the validation or test data performance while the results are written onto an excel sheet for further analysis. The network training state, network performance and regression plots are also made available all along the training process which follow the training process in real time as shown in the figure 15, 16 and 17 respectively.

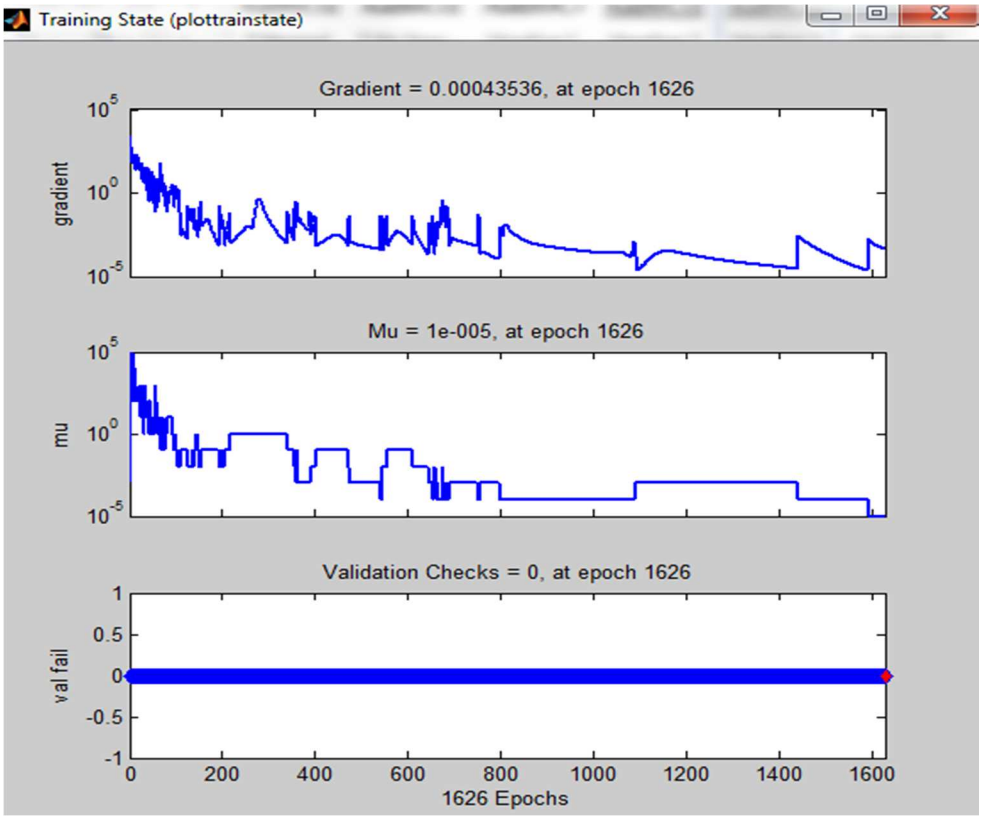

Figure 15 Training States of a Neural Network Example

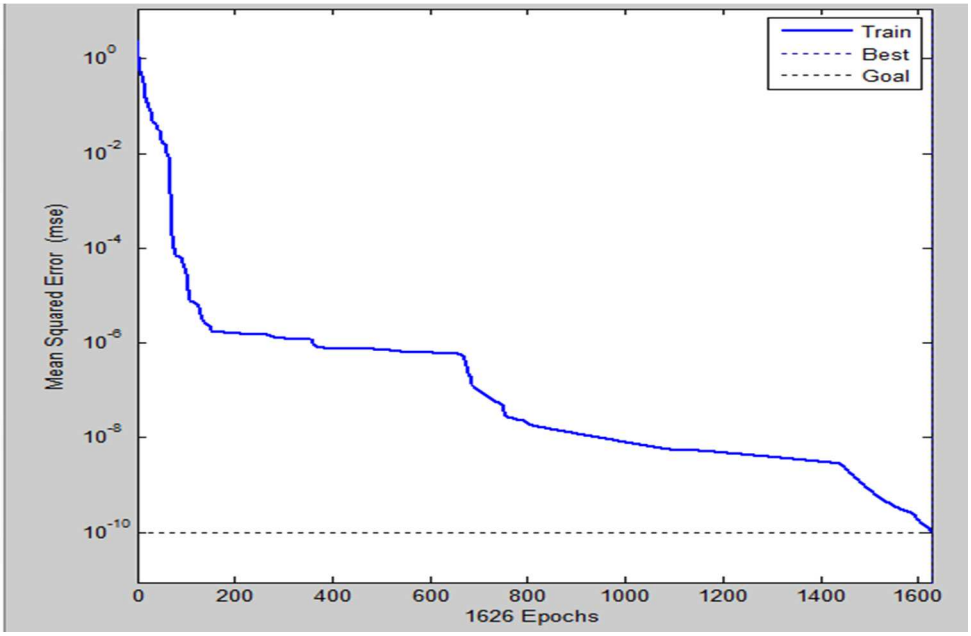

Figure 16 Performance Plot of a Neural Network Example

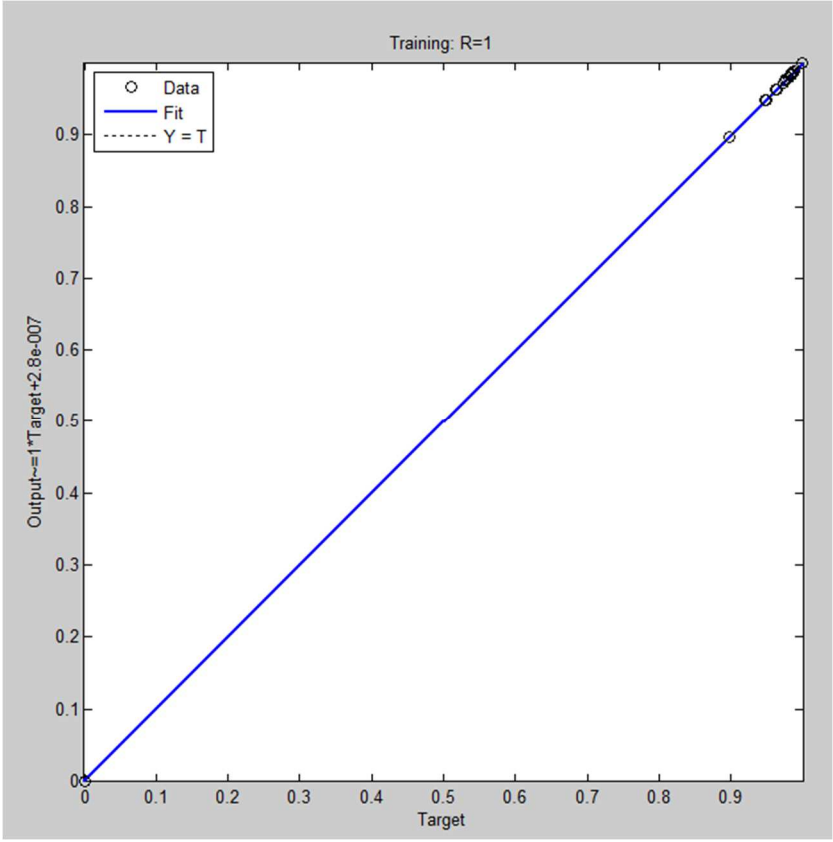

Figure 17 Regression Plot of a Neural Network Example

7.3.2 Post-training Performance Analysis

This section presents the best achieved results for the neural network model across all the five frequencies. The performance of the model was tested using three approaches: optimization without generalizing, early stopping through cross validation and then Bayesian Regularization. The first method used to train and test the model was the random sampling method which depicts the network's performance without any generalization.

For the random sampling and bayesian regularization approaches, the total dataset of each frequency was broken into two sets of ratio 80:20 where 80% corresponded to the training set while the other 20% to the test set. The model was trained using the 80% set by varying the parameters until a good prediction percentage on the training set was achieved. The training error was computed by testing the trained model with data which it had already seen. This was done to

check how well the model performed on known patterns and obviously the model reported excellent prediction percentages in all the approaches. The test error on the other hand was performed on the 20% data that was set aside and never used for testing. This was done to check the performance of the model on untrained data. The train-test process was repeated thrice with fresh data to remove any anomalies during data selection thereby helping us evaluate the average network performance.

7.3.2.1 What the Error Value Signifies?

The purpose of this thesis revolves around predicting the presence of a person in any particular position based on frequency patterns and hence it only makes sense to analyze the performance of the neural networks in terms of prediction percentage rather than prediction error itself. The values depicted in all the results are percentages of prediction as calculated for each individual neural network. The prediction error was first calculated from the results of each neural network's performance against the test data.

Prediction Error = Target Value – Predicted Value. (10)

The error percentage was then computed.

Error Percentage = (1- Target Value / Predicted Value) *100. (11)

The prediction percentage was then calculated by subtracting error percentage from 100.

It has to be clearly understood here what the error value signifies in this model. The target value set for each sample is a cross correlation coefficient for the average curve of that particular position and frequency. The neural networks aim to reach this or predict how close the test sample is to the target value. A value close to the target value implies very good prediction performance

of the system. A value between 0.5 and 0.75 implies average prediction and anything less than that is bad prediction.

The neural network results in all the cases were achieved after continuous runs and testing the model with numerous configurations until we reached the best performance. In further sections, the effect of using excessive neurons and overrunning the neural network on the performance of the model will be analyzed.

The network's performance is minimized using the MSE function in all cases of the 'trainlm' algorithm and using the SSE function when implementing the 'trainbr' algorithm. Lower values of MSE and SSE are preferred which represent good performance of neural network. The network training time however was observed to depend on a lot many factors like the number of reruns and complexity of the neural network itself and could not be used for performance comparison. The R-Squared value which usually depicted a good fit of the model was misleading too in many cases.

7.3.3 Random Sampling Results

This method is one of the most common and efficient ways to test the performance of a neural network. This gives a fair chance for every sample to be selected. Another key feature of simple random sampling is its representativeness of the population. Theoretically, the only thing that can compromise this method's representativeness is luck. If the sample is not representative of the population, the random variation is called sampling error.

An unbiased random selection and a representative sample is imperative in drawing definite conclusions from the results of any test. For this approach, the dataset of each neural network was randomly divided in the ratio of 80:20 as already explained. The results displayed here were averaged over three different tests as mentioned in the previous sections.

It was observed in our analysis that the sampling error was very high when the training and test sets were improperly selected while implementing the 'trainlm' algorithm. The best approach in selecting a test set would be to ensure that it contained all the patterns that the training set already had been trained. This simply means that the test set in our case should have all the five frequency curves because it is utterly impossible for any neural network to predict patterns it was never trained on. In any case of the training data missing out on one or more of the five frequencies skewed the performance by a huge value.

Two experiments were conducted which displayed this behavior. The first experiment ensured that the training and test sets both had similar patterns of curves but just that the test data was never a part of the training data while the second experiment was totally random. The neural networks were trained using the 'trainlm' method and the following results were observed.

Table 1Random Sampling: Test set Variation Analysis

Freq / Position	91.9	92.7	94	98	102	Avg NNet Performance
Complete Testset	88.4047	86.073	87.1	98.4	95.5	91.09754
Incomplete Testset	8.46735	86.075	68.2	75	12.2	49.97858667

The difference in the average network performance of 40% clearly shows a wide demarcation between the two experiments and stresses the importance of selecting good sets. This observation was crucial in giving us a good indication on how to pick the training and test sets and avoiding the pitfalls of a huge sampling error.

To better understand the performance achieved by implementing random sampling via 'trainlm' method, the prediction accuracy of each network needs to be analyzed which was the primary concern. The localization accuracy of all the positions across each of the five neural

networks are computed and displayed in table 1. The dataset was divided as mentioned in the ratio 80:20.

Table 2 Location Accuracy of 'trainlm' algorithm

Freq / Position	ER	P1	P2	P3	P4	Avg NNet Performance	
91.9	89.766	96.232	82.33	94.02	79.67	88.4047	
92.7	97.54	95.878	69.72	68.44	98.79	86.0728	
93.5	95.72	93.47	85.16	81.09	80.23	87.134	
98.3	96.19	99.84	98.49	97.86	99.72	98.4202	
101.9	89.04	98.12	99.5	91	99.62	95.456	
Avg Pos. Accuracy	93.6512	96.708	87.04	86.48	91.61	91.09754	**System Perf**

The average efficiency of the system hovers at 91% which considering the random samples used for the network is a good score. However the model as such is unfinished, it needs to reach a state where it can predict general patterns of the five frequencies. To achieve this capability, generalization methods were employed which will be discussed in further sections.

7.3.4 Early Stopping via Cross Validation Results

The ambition of any neural network modelling researcher is to make the model more robust to untested data that it might see in the future. This implies that it has to be generalized for which the methods of early stopping and bayesian regularization were sought.

The five-fold cross validation approach was implemented which meant that the dataset had to be randomly divided into five subsets and the network trained 5 times, each time leaving one subset from training which was used to test the network and compute the error. The total error of the network was later computed by averaging the individual errors over the five sets. The 'trainlm' algorithm was applied and the data sets were selected randomly without taking into consideration the completeness of the test set.

There was a considerable improvement in the average performance of all the five neural networks which is displayed in table 2.

Table 3 Localization Accuracy of Cross-Validation Approach

`

Freq / Position	ER	P1	P2	P3	P4	Avg NNet Performance	
91.9	59.49733	63.8	19.1	42.22	79.67	52.85715067	
92.7	93.1545	89.354	78.64	88.19	88.08	87.4819	
93.5	96.44	79.685	86.07	59.93	89.67	82.35866667	
98.3	94.04	96.885	98.1	76.02	73.34	87.67773333	
101.9	37.83333	35.755	50.7	52.17	81.22	51.53416667	
Avg Pos. Accuracy	76.19303	73.0958	66.52	63.7	82.39	72.38192347	System Perf

The overall average system accuracy hovers around 72% due to two bad neural network performances. The best-of-the-lot approach or median accuracy if taken, will improve it to a considerable 82%. The probable reason for the average performance would be the smaller datasets which were used for the entire analysis and hence did not aid the network training to accommodate all the patterns. This issue is usually rectified by training the data with larger datasets and varied patterns that can help the model generalize. But it is to be understood that larger datasets are not always the norm and difficult to obtain.

However this method is usually applied to problems of the multiple-input-single-output type and considering our case which is a multiple input-multiple-output case, this method proved to display decent performance but is still plagued by two main shortcomings.

The first would be the just near enough prediction precision imparted by this approach. Three frequency networks out of the available five perform with an average 85% prediction accuracy here which would be just sufficient for the lossy environment and frequency curves we deal with. Also the prediction accuracies of the five positions is to be noted here. Three of the five

positions have been localized with an average of 75% which may be good news but still doubt exists for the other two which only have a 65% average.

The other disadvantage would be the extensive amount of training and computational work that is involved. To overcome these, bayesian regularization approach had been chosen to generalize the network with minimal training.

On a general note; as a neural network model approaches to being generalized, the drop in prediction accuracy is expected. The model achieves the generalization capability with an adjusted tradeoff in accuracy. Individual networks based solely on their respective data patterns react differently to this method as can be seen from the results.

7.3.5 Bayesian Regularization Results

The most widely used method for neural network generalization is bayesian regularization. As expected, this method proved more successful in generalizing our model. Similar to the previous methods, we implemented random data selection to form the 80% training data and 20% test data without any consideration for the completeness of the test set. This method remarkably produced a more generalized model with a very low tradeoff of prediction accuracy, which one can gather from the results. The training algorithm 'trainlm' that was employed in both the previous methods was replaced by 'trainbr' here.

Table 4 Bayesian Reg. Results: Localization Accuracy & NNet. Performance

Freq / Position	ER	P1	P2	P3	P4	Avg NNet Performance	
91.9	84.73367	66.34	86.3	71.72	51.74	72.16648333	
92.7	59.7	88.5085	88.56	92.39	94.16	84.6642	
93.5	67.975	87.99	75.81	38.39	84.27	70.88566667	
98.3	83.32	97.7165	97.09	93.52	93.43	93.01672667	
101.9	80.93525	94.7	99.41	89.45	32.25	79.34838333	
Avg Pos. Accuracy	75.33278	87.051	89.43	77.09	71.17	80.016292	System Perf

The system performance impressively moves up to 80% from the previous 72% observed in the cross validation approach. However the significant improvement would be the four positions that were localized with a prediction percentage over 75% that is more than sufficient for the problem at hand. This implies that on any day, four out of the five networks will be able to localize a position accurately.

Although 'trainbr' algorithm trained with a low learning rate comparative to the 'trainlm' algorithm, the results depict an improved generalization capability of the model. The exact analysis of the training time will be analyzed in the next section.

Another interesting feature would be improvement in the network performance of the frequency 91.9 which improved from a meagre 37% in the first approach to a considerable doubled value of 72%. Another network which showed remarkable improvement would be the 101.9 frequency. These results prove that bayesian regularization proved to be the best method for training and regularizing our data of frequency curves. The next section will compare the results from all the generalization approaches to show the clear winner.

7.3.6 Performance Comparison of Generalization Approaches

To gain a better understanding of the model's training and testing prowess, the two approaches are compared in table 5.

Table 5 Localization Accuracy of Generalization Approaches

Generalization / Position	ER	P1	P2	P3	P4
Cross Validation	76.2	73.1	66.5	63.7	82.4
Bayesian Regularization	75.3	87.1	89.4	77.1	71.2

Bayesian regularized model is consistent in predicting all the five positions as can be observed from the compared results. The individual network performances are also categorized in table 6.

Table 6 System Performance of Generalization Approaches

Generalization / Position	91.9	92.7	93.5	98.3	102	Perf. Average
Cross Validation	52.9	87.5	82.4	87.7	51.5	72.38192347
Bayesian Regularization	72.2	84.7	70.9	93	79.3	80.016292

The system performance analysis of both the approaches have proven that generalization of the model though resulted in a declined prediction accuracy in network models, it produces a visibly better model that can perform well on untrained data. Bayesian Regularization resulted in the best results when trained and tested on the available data on hand.

This improved overall performance has its considerable roots in owing to the fact that five neural networks were utilized. This is where the advantage lies in this positioning system. Under circumstances of one or two of the frequency networks failing to provide a credible result, the performance of the system does not fail completely as the other networks chip in equally. This will be better understood when analyzed in detail the support and stability provided by employing multiple frequency networks.

7.3.7 Localization Accuracy of the System

The localization accuracy of the system depended on how well it predicted each individual position. The performance of our neural networks depended solely on the performance of the training algorithms and it is only fair to evaluate them against each other.

Both the algorithms were tested on a similar dataset that was picked up in a totally random fashion. To give them an equal edge, the number of neurons in each layer were also equally maintained at 9 along with the other general parameters such as the learning rate, step size increase and decrease values, epochs and the minimum gradient. The results of the performance functions for both the algorithms are displayed in table 7. These results of the performance functions were

only utilized to optimize the error function and in no way explains a method's accuracy or performance.

Table 7 Performance Comparison: Trainlm VS Trainbr

	Trainlm Parameters			Trainbr Parameters		
Freq/ Param	MSE-Te	SSE-Te	Tra.Time	MSE-Te	SSE-Te	Tra.Time
91.9	0.4595	11.4877	32.434349	0.4424	11.06	112.141006
92.7	0.0223	0.5577	0.418837	0.0223	0.5577	993.353805
93.5	0.0425	1.0632	23.743966	0.0228	0.5704	12.460388
98.3	0.3093	7.7336	35.176166	0.0602	1.5042	2271.80219
101.9	0.0255	0.6375	40.958782	0.0421	1.0521	135.889504
Average	0.17182	4.29594	26.54642	0.11796	2.94888	705.1293786

These results corroborate the need to analyze the individual performances and value contributed by each neural network before finalizing the system for external testing. The training times for both the algorithms confirm to the theoretical slow learning rate of the 'trainbr' algorithm and relatively faster 'trainlm'.

The overall system performance was charted in table 9 to complete the analysis for the best algorithm among the two most suited for our research problem.

Table 8 Network Performance Comparison: Trainlm VS Trainbr

Algorithm / Position	91.9	92.7	93.5	98.3	101.9	System Perf. Avg
Trainlm	39.76	85.35	79.58	81.46	64.34	70.09
Trainbr	84.21	83.86	72.12	94.01	81.47	83.13

These results successfully conclude that 'trainbr' provides the best results for our research problem with consistent efficiency across all the frequency networks when tested against random datasets.

7.3.8 Assessing the Multi-frequency Approach

It is common knowledge that an uncertainty exists in the field of wireless systems as the radio signals cannot always be consistent and add to that the effect of noise and random variations. To accommodate this limitation in our research, we needed to understand the value that each of the five individual frequencies and their corresponding networks bring to the system. Having said that, it would be appropriate to analyze if using all the five neural networks for the analysis always is a wise option. The comprehensive results of all the frequency neural networks have already been analyzed which allows to learn the value added by each of these individual neural networks to the system's performance.

To estimate the added value that each frequency brings to the system, the system's performance was evaluated with one frequency neural network to start with. Each of the other frequency networks were added sequentially to learn the actual increase in the prediction percentage and also the added robustness that the system gained. The reason to choose all the methods was to realize the effect of generalization on the value that each frequency added to the system. The summarized results are depicted in table 9.

Table 9 System Stability Analysis: Multiple Frequency Support

Frequency / Loc. Method	Random Sampling	Cross Validation	Bayesian Reg.
91.9	88.40	52.86	72.17
91.9 + 92.7	87.24	70.17	78.42
91.9 + 92.7 + 93.5	87.20	74.23	75.91
91.9 + 92.7 + 93.5 + 98.3	90.01	77.59	80.18
91.9 + 92.7 + 93.5 + 98.3 + 101.9	91.10	72.38	80.02

Except for a few unusual bad performances by networks like 91.9 in all the three generalization approaches and one out of the box hiccup by the 101.9 frequency, most of the

systems showed a steady increase in their performance as the frequency networks were added sequentially.

To improve the average performance of the system and increase its fidelity, we can pick just the top performing neural networks out of the total five. The tables 10 and 11 show what the best average performance would be if four or three of them are selected.

Table 80 Performance Comparison: Three Best Networks

Approach	Four Best performing Freqs	System Perf.
Random Sampling	98.3 + 101.9 + 91.9 + 93.5	92.35
Cross Validation	98.3 + 92.7 + 93.5 + 91.9	77.59
Bayesian Reg.	98.3 + 101.9 + 92.7 + 91.9	82.30

Table 91 Performance Comparison: Four Best Networks

Approach	Three Best performing Freqs	System Perf.
Random Sampling	98.3 + 101.9 + 91.9	94.09
Cross Validation	98.3 + 92.7 + 93.5	85.84
Bayesian Reg.	98.3 + 101.9 + 92.7	85.68

By limiting the list to just the best performing frequencies, the overall system performance increased evidently.

This improved performance by multiple frequency implementation is what strengthens the localization system as a whole in contingencies. This is in spite of being able to achieve decent localization performance through one or two frequency networks. Under worse circumstances, either the median accuracy or the best accuracy provided by the networks can be used, as understood from the tables 10 and 11. It should be noted that relying on multiple frequencies adds

credibility to the system as a whole where the results of one network are substantiated by the others as well.

7.3.9 Effect of Excessive Neurons and Network Overruns

During the process of achieving the optimum results for each of the methods, multiple runs on the same algorithm had to be done by varying the parameters. This is a necessary step during any neural network training as there exists no fixed values for a certain type of problem. The efficiency of a neural network model in predicting the pattern or producing a good fit to the training data depends on its complexity.

The complexity of any model varies between an underfit, good fit and an overfit. Striking the right cord takes more than just random guessing and so it was realized. The complexity of a model is a complex factor that depended on a various number of factors but most importantly on three major factors: number of hidden layers, number of neurons and the number of network runs. Our thesis assumption was to use the universal network of 3 layers which could be used for 99% of pattern recognition or neural network fitting. As evident from the results, a single hidden layer did the job perfectly.

The other two factors were varied within logical limits to check how the efficiency of the model varied. The results we obtained matched the theoretical predictions on the result of utilizing excessive neurons or running the network too many times.

The table 12 shows the MSE of the neural network corresponding to 91.9 and how it varies as we keep adding neurons to the network. The MSE of the test set reaches a minimum at 12.

Table 102 Effect of Excessive Neurons on Mean Square Error: Freq. 91.9

Run #	Neurons	MSE-Te	MSE-Tr
1	1	0.5525	0.0943
2	2	0.1661	0.1054
3	3	0.8138	0.06
4	5	0.1648	0.0965
5	7	1.3348	0.0398
6	8	4.0507	0.0039
7	10	2.5685	4.75E-13
8	12	1.0196	2.39E-10
9	14	1.0705	1.65E-16
10	18	1.7291	1.59E-14
11	22	1.3642	1.56E-13
12	26	1.3583	6.70E-11

As the data suggests, the performance of the network reaches a minima as the neuron number reaches 12 and then after this point, the performance goes uphill. Figure 18 explains the same in a better way. The reason behind this behavior is due to the model's complexity increasing gradually. The model starts from being an underfit and sharpens the fit as the number of neurons are added until it reaches a 'good fit' point. This is observed by a reduced or minimal test error. After this, any number of neurons added will only make things worse as it becomes overfit observed as an increase in the test error rather than a decrease. In figure 19, the validation set MSE is plotted against the number of neurons. It reaches a minima around 14 neurons and adding number of neurons added after that only minimizes the train error but not the test error. This

behavior was observed in all the networks cited by this example. Here the network should have stopped with a value between 12 and 14.

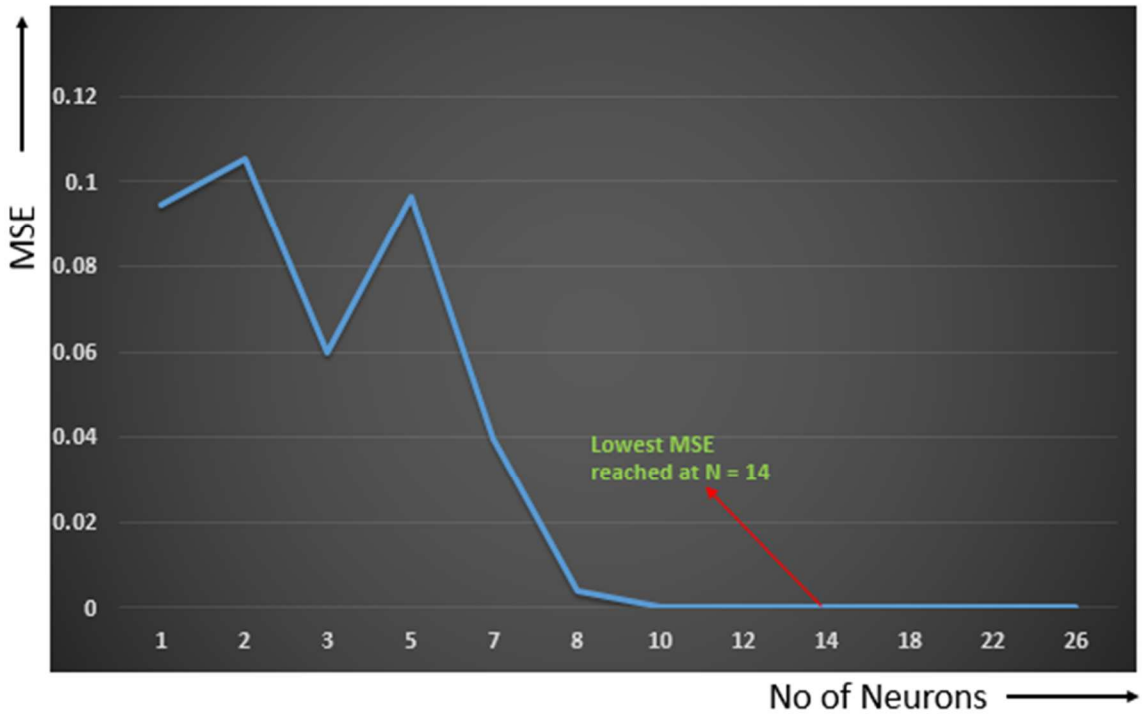

Figure 18 Validation Set MSE VS Number of Neurons Plot

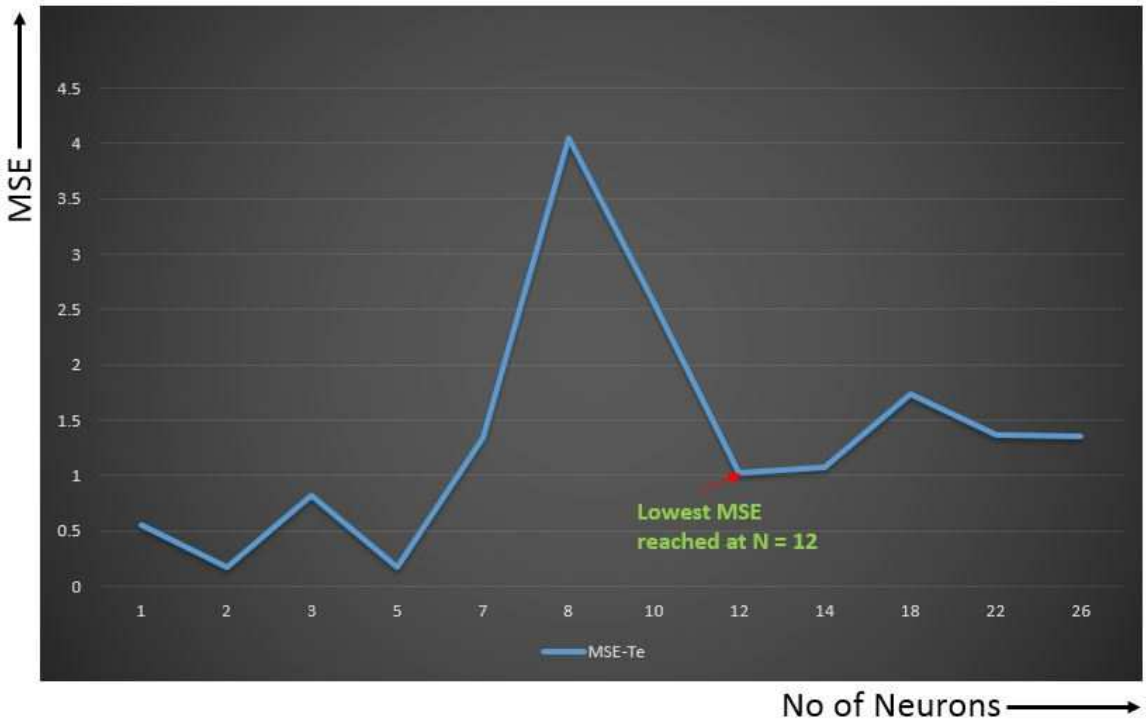

Figure 19 Test Set MSE VS Number of Neurons Plot

The other factor which affects the complexity of a neural network model is the number of reruns on a model. The data from one of the networks corresponding to the frequency 93.5 is shown in the table 13 with its MSE.

Table 13 Effect of Network Overruns on MSE

Run #	MSE-Te	MSE-Tr	Neurons
1	0.0195	4.27E-05	9
2	0.0083	4.27E-05	9
3	0.0523	4.27E-05	9
4	0.0053	4.27E-05	9
5	0.0212	4.27E-05	9
6	0.0237	4.27E-05	9
7	0.2623	4.27E-05	9

To understand how this factor caused a change in complexity, the number of neurons were fixed at 9 which had resulted in a good MSE on most of our results. Initially, the model started with a random set of network weights which were changed after each run to accommodate the optimization of the error function or simply a reduced MSE. Similar to the above case, the network shifted towards a global minima after each run but once it reached that point at run 4, any more reruns on the network only worsened its performance as mentioned in the table and also evident from the figure 20.

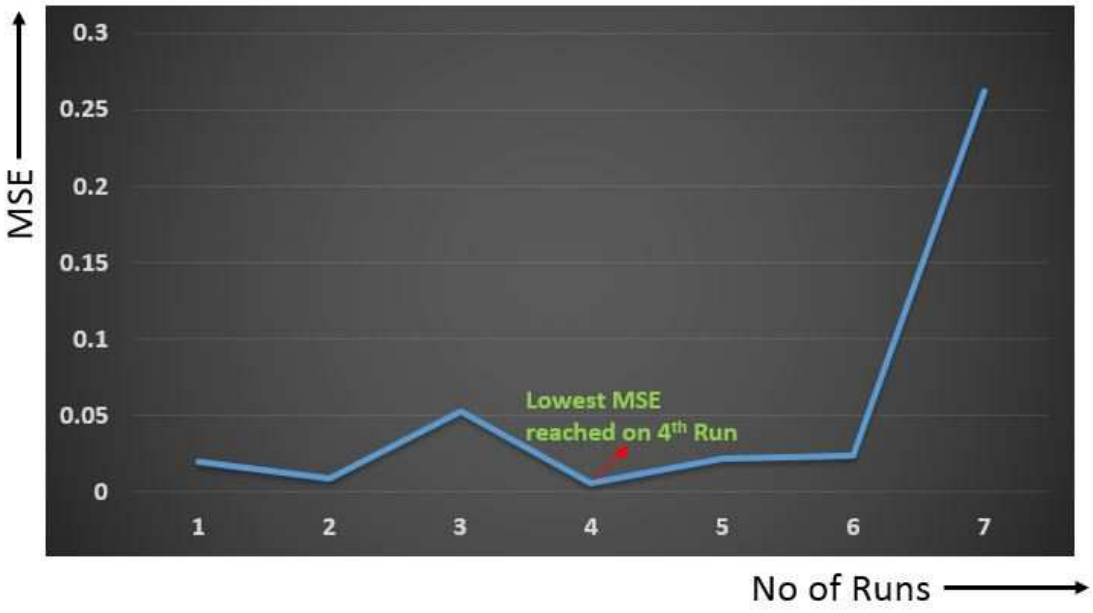

Figure 20 Test set Mean Square Error VS No. of Network Reruns

Constant vigilance and care had to be taken to ensure that the networks did not become overfit or remained underfit. It has to be understood that each network had its own set of parameters that resulted in a good fit and could not be generalized according to the results of others.

7.3.10 Bettering the Industrial Standards

Our research was motivated in part by a pilot research project by using a single passive off-the-shelf receiver and ambient FM radio signals. The authors evaluated fine-grained sub-room level localization performance of the system and its temporal stability. (Popleteev, 2014). Their research demonstrated that channel diversity improved the localization performance. This research being the most current and the only one in the field of device free passive localization using a single receiver to date helped set a standard to improve upon. The idea of relying on multi-channel frequency scanning was borrowed from their research and improved by implementing the idea of neural networks for pattern prediction.

The previous research paper (Popleteev, 2014) had few limitations like the accuracy degradation with time and the inaccuracy concerning two positions. Even though both localization approaches heavily depend on test environment conditions, we achieved better localization and accurate position prediction with our system. Their results had a drop in 20% over three days of testing which shows a heavy inconsistency in the approach as their model relied on RSSI values solely and the k-nearest neighbor approach. This model as already explained in the previous sections improved from a meagre 63% to 72% and then consistently performed at an average of 80% as the training set size increased even when tested with randomly selected data. Moreover the datasets used were spanned across three weeks to test the temporal performance of the network which proved to be very efficient.

Also the results from previous paper (Popleteev, 2014) only managed a meagre 0.5 recognition accuracy with 5 channels; our results however had a success of 0.8 on an average and just regarding the best case alone pushed it to 0.95.

Accuracy as understood from our analysis of the results was not always a reliable metric for the real performance of the neural networks as the datasets were not all equally accurate in their prediction. A confusion matrix on the other hand allows for a more detailed approach than the usual proportion of correct guesses. In the final conclusive analysis, the confusion matrices will be plotted for both the algorithms that were used to train the data with. These results are then compared with the best performance results from Popleteev's research (Popleteev, 2014).

Table 14 Confusion Matrix of 'trainlm' algorithm

Confusion Matrix: Trainlm Algorithm					
Positions	**Empty Room**	**Pos. 1**	**Pos. 2**	**Pos. 3**	**Pos. 4**
Empty Room	0.95	0.00	0.00	0.00	0.00
Pos. 1	-0.02	0.98	-0.01	0.00	0.00
Pos. 2	0.01	0.02	0.97	0.00	0.02
Pos. 3	0.02	-0.01	0.00	0.92	0.00
Pos. 4	0.00	-0.02	0.00	-0.01	0.96

Table 14 constitutes the resultant confusion matrix when testing the figth day data with respect to the network trained on the fist 4 days, a week apart using the 'trainlm' algorithm. The next table summarizes the resultant confusion matrix plotted for the 'trainbr' algorithm.

Table 15 Confusion Matrix of 'trainbr' algorithm

Confusion Matrix: Trainbr Algorithm					
Positions	**Empty Room**	**Pos. 1**	**Pos. 2**	**Pos. 3**	**Pos. 4**
Empty Room	0.98	0.00	0.00	0.05	0.00
Pos. 1	0.00	0.97	0.05	-0.02	0.02
Pos. 2	0.00	0.00	0.98	-0.02	-0.01
Pos. 3	0.00	0.00	0.00	0.99	-0.02
Pos. 4	0.00	0.00	0.02	-0.04	1.01

These results when compared to the confusion matrix of the previous paper (Popleteev & Engel, 2014) are a significant improvement as can be seen from the data in the table 16.

Table 16 Confusion Matrix from Popleteev's Paper

	Ground truth				
Classifier result ↓	Empty	Desk	Bed-lying	Bed-sitting	Ward-robe
Empty	**0.66**	0.08	0.10	0	0.02
Desk	0.01	**0.90**	0.02	0.01	0
Bed-lying	0.08	0.02	**0.68**	0.04	0.04
Bed-sitting	0.11	0	0.10	**0.85**	0.11
Wardrobe	0.14	0	0.10	0.10	**0.83**

These results confirm that this work was a consistent and significant improvement over the industrial best standards in device free passive localization using radio signals and a single receiver.

CHAPTER 8. CONCLUSIONS AND FUTURE WORK

Overall, the model finished with a total system performance of 90% over the five neural networks when employing MLP neural network with Levenberg-Marquardt algorithm. The generalized performance of 80% was achieved with the implementation of Bayesian regularization. These results when compared to the k-NN results of Popleteev's research show a considerable improvement. The developed localization system substantiated that one individual frequency would be sufficient to achieve commendable accuracy but adding many will enhance the performance, efficiency, reliability and fidelity. There was no performance degradation as observed from the tests we conducted with a test set apart by two weeks from our initial training set proving the extreme temporal stability of the system. We also proved that the addition of a pre-processing stage did not vary the results much from a system without the same.

The neural network models were also tested with random datasets that were crossovers of each day's total datasets to analyze the performance of the models on dealing with days when data is muddled or a certain neural network receives another network's data.

The 'trainlm' algorithm fared poorly over such sets while the 'trainbr' algorithm showed a consistent performance over 80% across all the networks.

8.1 Major Contributions towards Research

The main objective of this thesis was to improve and stabilize the room-level localization accuracy when using an off the shelf radio receiver and multiple ambient FM radio stations over time. This research involved the localization of four different locations a person moves inside an average sized room of an apartment. The room in which the experiment was conducted included the usual wardrobes, book shelves, ceiling fan and a wall mounted AC system. The idea was to include furniture and other environmental variables in the localization process to represent a real

time environment. It was observed that the furniture in the room aided the process of localization by producing strong fingerprints that did not vary with time. The analysis involved a neural network model trained to learn the different signals captured at each of the positions.

I consider this contribution novel and a significant improvement over past research due to the following reasons:

1. Device free passive indoor localization using a single off-the-shelf receiver and multiple FM radio stations is in itself a niche area of research and the pilot experimental results observed by Popleteev et al (Popleteev, 2014) struggled due to accuracy degradation. The methodology used in this thesis simplifies the problem by directly interacting with the signals and developing individual Neural Network models for each FM frequency. This approach is the first one to employ neural networks to solve localization in device free environments using FM radio signals and just a single passive receiver.

2. Unlike the past research work involving indoor localization using FM signals which utilized radio fingerprinting with RSSI values that depended on signal strength variation with distance, the present localization model ignores the relationship with distance to avoid complexities.

3. Compared to the previous research work which produced a very low recognition accuracy of 0.4 when using 5 channels (Popleteev, 2014), in our case for the same we produced an average of 0.8 and a best of 0.95.

4. There was utterly no complex post processing of data required as the model delivers the required localization accuracy taking in pre-processed signal data.

5. The model is highly scalable and works better with more data as the network would get trained to recognize signal variations.

6. The preprocessing of data to filter out noise is primitive enough to run on simple online systems and aids in producing better and faster runs on the algorithm.

7. The median accuracy of the latest research paper was shown to be 85% which degrades to 65% after three days of testing. The model proposed in this thesis succeeded in providing an accuracy of 98% without temporal degradation.

8. The proposed method implements automated power management in residential power saving systems. Motivation comes from a need for decreasing the installation complexity and development costs induced by the integration of specific human presence detection sensors

8.2 Future Work

Future researchers could advance the proposed method to profile the environment based on the weather conditions and the person being profiled (Each person will have a unique fingerprint due to his/her shape and height). An online web based system when fed the raw data can be trained to recognize the weather patterns and subtle changes in the RSS variations across various channels. Once this system is implemented online, the efficiency can be improved by continuous training of the neural network at odd conditions in all seasons so as to meet all environmental conditions in a year. This feature that the thesis put forward brings multidimensionality of neural networks imparting stability and performance which can always be open for further improvement.

Also the performance of the model depended on the pre-processing stage where unnecessary data was cropped from the signal curves. These numbers however are relevant only to the present test conditions and the data that were generated with this experimental setup. To follow or not to follow this truncation process should be decided by practical observation of their own data by future researchers.

This stage of further improving the neural network model however shall involve extensive data collection over a wide range of days for better performance of the system. If aided by a stronger receiver that can scan faster with more frequency and supported by a better data processor, the system can also predict and plot the movement of the human target in the room. Researchers can also study the other parameters which indirectly affect the performance of a system like adding obstacles and also presence of multiple people inside a room which we had not considered.

Device free passive indoor localization using FM radio signals with just a single receiver aided by neural networks has worked out to be a precise technique irrespective of the noisy data. Current research trends and applying hybrid technologies can make it better.

REFERENCES

A. LaMarca, Y. C. (2005). Place Lab: Device Positioning Using Radio Beacons in the Wild.

A. Schwaighofer, M. G. (2004). GPPS: A Gaussian process positioning system for cellular networks.

A. Varshavsky, E. d. (2007). Gsm indoor localization. ACM Digital Library.

AboutGPS. (2001). Retrieved from GARMIN: www.garmin.com

B. Ferris, D. H. (2006). In Proc. of Robotics: Science and Systems.

Cisco. (2008). Wi-Fi-Based Services 4.1 Design Guide. Cisco System.

CSGNetwork. (n.d.). Whip Antenna Length and Frequency Calculator. Retrieved from CSGNetwork: http://www.csgnetwork.com/antennagenericfreqlencalc.html

D. E. Rumelhart, G. E. (n.d.). Learning representations by back-propagating errors. Nature, 323, 533–536, 1986.

D.Zhang, F. L. (2010). Localization technologies for indoor human tracking. Proceedings of the 5th International Conference on Future Information Technology. FutureTech.

Davchev, S. a. (2010). Localization in Wireless Sensor Networks. Sensors.

Din, C. W. (2009). Improvement of indoor location sensing algorithm using wireless local area network (WLAN) IEEE 802.11b. IEEE.

faqs.org. (n.d.). Retrieved from faqs.org: http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html

Fox D., H. J. (2003). Bayesian Filtering for Location Estimation. IEEE Pervasive Computing.

Hallberg J, N. M. (2003). Positoning with Bluetooth. ICT, (pp. 954-958).

ITU. (2007, October). Recommendation ITU-R. Retrieved from ITU.int: http://www.itu.int/

Jorge Torres-Solis, T. H. (2010). A review of indoor localization technologies: towards navigational assistance for topographical disorientation. INTECH OPEN.

K. Laasonen, M. a. (2004). Adaptive On-Device Location Recognition. Volume notes in Computer Science (pp. 287-304). Springer.

K.Kaemarungi. (2005). Design of Indoor Positioning Systems based on Location Fingerprinting Technique. Pittsburgh: Univ. of Pittsburgh.

Kurban, T. E. (2009). A comparison of RBF neural network training algorithms for inertial sensor based terrain classification.

L. Chan, J. C. (2006). Enhancing WiFi-based position estimation with neighborhood links in clusters. Proceedings of the 4th Conference.

L. Povescu, I. N. (2001). Neural networks applications for the prediction of propagation path loss in urban environments. IET Conference on Wireless, mobile and sensor networks.

M. Chen, T. S. (2006). Practical metropolitan-scale positioning for GSM phones. Orange County, CA: Springer Berlin.

M.Musavi, A. S. (2008). Localization using neural networks in wireless sensor networks`. Mobiware.

MATLAB. (n.d.). Bayesian Regularization. Retrieved from mathworks: http://www.mathworks.com/help/nnet/ref/trainbr.html

MATLAB. (n.d.). Improve Neural Network Generalization and Avoid Overfitting. Retrieved from MATHWORKS: http://www.mathworks.com/help/nnet/ug/improve-neural-network-generalization-and-avoid-overfitting.html

MATLAB. (n.d.). mathworks. Retrieved from MATHWORKS: http://www.mathworks.com/products/matlab/

Nations, U. (2013). World Population Ageing 2013. United Nations.

Osborne, M. R. (1992). Fisher's method of scoring. International Statistical Review.

Padmanabhan, P. B. (2000). RADAR. INFOCOM. INFOCOM.

Popleteev, A. (2014). Device-Free Indoor Localization Based on Ambient FM Radio Signals. Interdisciplinary Centre for Security, Reliability and Trust (SnT).

Radim Zemek, M. T. (2007). Effect of Walking People on Target Location Estimation Performance in an IEEE 802.15.4 Wireless Sensor Network. IEICE.

Shehadi Dayekh, S. A. (2010). Cooperative Localization in Mines Using Fingerprinting and Neural Networks`. IEEE.

ucebna. (n.d.). MATLAB TOOLBOX. Retrieved from http://dali.feld.cvut.cz/ucebna/matlab/toolbox/nnet/newff.html

V. Seshadri, G. Z. (2005). A bayesian sampling approach to indoor localization if wireless devices using received signal strength indication. Pervasive Computing and Communications.

W.Barclay, L. (2002). Propagation of Radiowaves. IET.

Y. Gu, A. L. (2007). Survey of wireless Indoor positoning techniques and systems. Man and Cybernetics Part C, IEEE Transactions on Systems, 1067-1080.

Yang, H. L. (2010). Location, Localization, and Localizability. Location-awareness Technology, Springer.

Performance Evaluation of Mobile U-Navigation based on GPS/WLAN Hybridization. Journal of Convergence Information Technology, 7(12), 235–246.

http://doi.org/10.4156/jcit.vol7.issue12.27

A Location Fingerprint Framework Towards Efficient Wireless Indoor by Nattapong Swangmuang B . Eng ., Chulalongkorn University , Thailand , 1998 M . S ., Telecommunications , University of Pittsburgh , 2002 Submitted to the Graduate Faculty of the School . (2008).

Abdel-Nasser, H., Samir, R., Sabek, I., & Youssef, M. (2013). MonoPHY: Mono-stream-based device-free WLAN localization via physical layer information. IEEE Wireless Communications and Networking Conference, WCNC, 4546–4551. http://doi.org/10.1109/WCNC.2013.6555311

Al-merbawi, A. M. M., Zainal, S., Mahmud, J. S., & Trengganu, K. (2013). Ageing Population in Developed Countries : A Study of Process and Phenomena, 17(2), 219–225. http://doi.org/10.5829/idosi.mejsr.2013.17.02.12178

Alpaydin, E., & Jordan, M. I. (1996). Local linear perceptrons for classification. IEEE Transactions on Neural Networks. http://doi.org/10.1109/72.501737

Bahl, P., & Padmanabhan, V. N. (2000). RADAR: an in-building RF-based user location and tracking system. Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064), 2. http://doi.org/10.1109/INFCOM.2000.832252

Bashiri, M., & Farshbaf Geranmayeh, a. (2011). Tuning the parameters of an artificial neural network using central composite design and genetic algorithm. Scientia Iranica, 18(6), 1600–1608. http://doi.org/10.1016/j.scient.2011.08.031

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2), 157–166. http://doi.org/10.1109/72.279181

Brunato, M., & Battiti, R. (2005). Statistical learning theory for location fingerprinting in wireless LANs. Computer Networks, 47(6), 825–845. http://doi.org/10.1016/j.comnet.2004.09.004

Bulusu, N., Heidemann, J., & Estrin, D. (2000). GPS-less low-cost outdoor localization for very small devices. IEEE Personal Communications, 7(5), 28–34. http://doi.org/10.1109/98.878533

Casella, G., Fienberg, S., & Olkin, I. (2006). Springer Texts in Statistics. Design (Vol. 102). http://doi.org/10.1016/j.peva.2007.06.006

Castro, J. L., Mantas, C. J., & Benítez, J. M. (2000). Neural networks with a continuous squashing function in the output are universal approximators. Neural Networks, 13(6), 561–563. http://doi.org/10.1016/S0893-6080 (00)00031-9

Castro, P., & Muntz, R. (2000). Managing context data for smart spaces. IEEE Personal Communications, 7(5), 44–46. http://doi.org/10.1109/98.878537

Chen, Y., Chiang, J., & Chu, H. (2005). Sensor-assisted wi-fi indoor location system for adapting to environmental dynamics and Mobile Systems, 118–125. http://doi.org/10.1145/1089444.1089466

Dawes, B., & Chin, K.-W. (2011). A comparison of deterministic and probabilistic methods for indoor localization. Journal of Systems and Software, 84(3), 442–451. http://doi.org/10.1016/j.jss.2010.11.888

De Moraes, L. F. M., & Nunes, B. A. a. (2006). Calibration-free WLAN location system based on dynamic mapping of signal strength. Proceedings of the International Workshop on Mobility Management and Wireless Access - MobiWac '06, 92–99. http://doi.org/10.1145/1164783.1164799

Due, A., & Clouds, T. O. (1999). Specific attenuation coefficient, (2).

Elman, J. L. (1990). Finding structure in time* 1. Cognitive Science, 14(2), 179–211. http://doi.org/10.1207/s15516709cog1402_1

Fahed, D., & Liu, R. (2013). Wi-Fi-Based Localization in Dynamic Indoor Environment Using a Dynamic Neural Network. International Journal of Machine Learning and Computing, 3(1), 127–131. http://doi.org/10.7763/IJMLC.2013.V3.286

Farid, Z., Nordin, R., & Ismail, M. (2013). Recent Advances in Wireless Indoor Localization Techniques and System, 2013. http://doi.org/10.1155/2013/185138

Fazel Famili, W. M. Shen, R. Weber, E. S. (1997). Data Pre-Processing and Intelligent Data Analysis. International Journal on Intelligent Data Analysis, 18(6), 1–29.

Fernando, T. M. K. G., Maier, H. R., & Dandy, G. C. (2009). Selection of input variables for data driven models: An average shifted histogram partial mutual information estimator approach. Journal of Hydrology, 367(3-4), 165–176. http://doi.org/10.1016/j.jhydrol.2008.10.019

Gaweda, a. E., Zurada, J. M., & Setiono, R. (2001). Input selection in data-driven fuzzy modeling. 10th IEEE International Conference on Fuzzy Systems. (Cat. No.01CH37297), 3, 1251–1254. http://doi.org/10.1109/FUZZ.2001.1008885

Giorgetti, G., Gupta, S. K. S., & Manes, G. (2007). Wireless localization using self-organizing maps. Proceedings of the 6th International Conference on Information Processing in Sensor Networks - IPSN '07, 293. http://doi.org/10.1145/1236360.1236399

Gogolak, L., Pletl, S., & Kukolj, D. (2011). Indoor fingerprint localization in WSN environment based on neural network. SISY 2011 - 9th International Symposium on Intelligent Systems and Informatics, Proceedings, 293–296. http://doi.org/10.1109/SISY.2011.6034340

Hornik, K., Stinchcombe, M., & White, H. (n.d.). Kornick_et_al.

Jeffrey Hightower, G. B. (2001). A Survey and Taxonomy of Location Systems for Ubiquitous Computing. IEEE Computer, 34, 57–66. http://doi.org/10.1109/2.940014

Kaplan, E. D., & Hegarty, C. J. (2006). Understanding GPS.

Ke, W., Liu, G., & Fu, T. (2014). Robust Sparsity-Based Device-Free Passive Localization in Wireless Networks, 46(December 2013), 63–73.

Kechine, M. O., Tiberius, C. C. J. M., & Marel, H. Van Der. (2003). Network Differential GPS: Kinematic Positioning with NASA's Internet-Based Global Differential GPS. Journal of Global Positioning Systems, 2(2), 139–143. http://doi.org/10.5081/jgps.2.2.139

Kosba, A. E., Saeed, A., & Youssef, M. (2012). RASID: A RobustWLAN Device-free Passive Motion Detection System. Pervasive Computing and …, (March), 180–189. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6199865

Ladd, A. M., Bekris, K. E., Rudys, A., Kavraki, L. E., & Wallach, D. S. (2005). Robotics-based location sensing using wireless ethernet. Wireless Networks, 11(1-2), 189–204. http://doi.org/10.1007/s11276-004-4755-8

Lahiri, S. K., & Ghanta, K. C. (2010). Artificial neural network model with parameter tuning assisted by genetic algorithm technique: Study of critical velocity of slurry flow in pipeline. Asia-Pacific Journal of Chemical Engineering, 5(5), 763–777. http://doi.org/10.1002/apj.403

Li, C. (2010). Recent advances in Doppler radar sensors for pervasive healthcare monitoring. Microwave Conference Proceedings (APMC), 283–290. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5728701

Lin, T. N., & Lin, P. C. (2005). Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks. Wireless Networks, Communications and Mobile Computing, 2005 International Conference on, 2, 1569–1574. http://doi.org/10.1109/WIRLES.2005.1549647

Linear, T. R., Rls, T., & Theorem, R. (n.d.). Tikhonov Regularization and ERM, (8), 1–8.

Llc, C. R. C. P. (n.d.). Handbook of Neural Network Signal Processing. IEEE Transactions on Neural Networks (Vol. 16). http://doi.org/10.1109/TNN.2005.848997

Luo, C., Hong, H., & Chan, M. C. (2014). PiLoc: A self-calibrating participatory indoor localization system. IPSN 2014 - Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (Part of CPS Week), 143–153. http://doi.org/10.1109/IPSN.2014.6846748

Matic, A., Popleteev, A., Osmani, V., & Mayora-Ibarra, O. (2010). FM radio for indoor localization with spontaneous recalibration. Pervasive and Mobile Computing, 6(6), 642–656. http://doi.org/10.1016/j.pmcj.2010.08.005

Mekki, H., & Chtourou, M. (2012). Variable structure neural networks for online identification of continuous-time dynamical systems using evolutionary artificial potential fields. International Multi-Conference on Systems, Sygnals & Devices, 11(2), 1–6. http://doi.org/10.1109/SSD.2012.6197954

Nawi, N. M., Atomi, W. H., & Rehman, M. Z. (2013). The Effect of Data Pre-processing on Optimized Training of Artificial Neural Networks. Procedia Technology, 11(Iceei), 32–39. http://doi.org/10.1016/j.protcy.2013.12.159

Ni, L. M., Liu, Y., Lau, Y. C., & Patil, a P. (2003). LANDMARC: indoor location sensing using active RFID. Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on, 407–415. http://doi.org/10.1109/PERCOM.2003.1192765

Ocaña, M., Bergasa, L. M., Sotelo, M. a., Nuevo, J., & Flores, R. (2005). Indoor robot localization system using WiFi signal measure and minimizing calibration effort. IEEE International Symposium on Industrial Electronics, IV, 1545–1550. http://doi.org/10.1109/ISIE.2005.1529162

Oglesby, J., & Mason, J. S. (1991). Radial basis function networks for speaker recognition. [Proceedings] ICASSP 91: 1991 International Conference on Acoustics, Speech, and Signal Processing, 393–396. http://doi.org/10.1109/ICASSP.1991.150359

Paul Castro, P. C. (2001). A Probabilistic Room Location Service for Wireless Networked Environments. UbiComp '01 3rd International Conference on Ubiquitous Computing, 18–34. http://doi.org/10.1.1.3.6233

Popleteev, A. (2013). Device-free indoor localization using ambient radio signals. Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication - UbiComp '13 Adjunct, 549–552. http://doi.org/10.1145/2494091.2497613

Popleteev, A., & Engel, T. (2014). Device-Free Indoor Localization Based on Ambient FM Radio Signals. International Journal of Ambient Computing and Intelligence, 6(1), 35–44. http://doi.org/10.4018/ijaci.2014010103

Popleteev, A., Osmani, V., & Mayora, O. (2012). Investigation of indoor localization with ambient FM radio stations. 2012 IEEE International Conference on Pervasive Computing and Communications, PerCom 2012, (March), 171–179. http://doi.org/10.1109/PerCom.2012.6199864

Rahman, M. S., Park, Y. P. Y., & Kim, K.-D. K. K.-D. (2009). Localization of Wireless Sensor Network using artificial neural network. 2009 9th International Symposium on Communications and Information Technology, 639–642. http://doi.org/10.1109/ISCIT.2009.5341165

Revisited, G., & Under-fitting, P. (2004). Computational Power of MLPs. In Practice, 1–12.

Roos, T., Myllymäki, P., Tirri, H., Misikangas, P., & Sievänen, J. (2002). A Probabilistic Approach to WLAN User Location Estimation. International Journal of Wireless Information Networks, 9(3), 155–164. http://doi.org/10.1023/A:1016003126882

Saeed, A., Kosba, A. E., & Youssef, M. (2014). Ichnaea: A low-overhead robust WLAN device-free passive localization system. IEEE Journal on Selected Topics in Signal Processing, 8(1), 5–15. http://doi.org/10.1109/JSTSP.2013.2287480

Seifeldin, M., Saeed, A., Kosba, A. E., El-Keyi, A., & Youssef, M. (2013). Nuzzer: A large-scale device-free passive localization system for wireless environments. IEEE Transactions on Mobile Computing, 12(7), 1321–1334. http://doi.org/10.1109/TMC.2012.106

Sheela, K. G., & Deepa, S. N. (2013). Review on methods to fix number of hidden neurons in neural networks. Mathematical Problems in Engineering, 2013. http://doi.org/10.1155/2013/425740

Shi, S., Sigg, S., & Ji, Y. (2012). Passive detection of situations from ambient FM-radio signals. Proceedings of the 2012 ACM Conference on Ubiquitous Computing - UbiComp '12, 1049. http://doi.org/10.1145/2370216.2370440

Singh, S., Jain, S., & Bárdossy, A. (2014). Training of Artificial Neural Networks Using Information-Rich Data. Hydrology, 1(1), 40–62. http://doi.org/10.3390/hydrology1010040

Skala, V. (2013). Fast Interpolation and Approximation of Scattered Multidimensional and Dynamic Data Using Radial Basis Functions, 12(5).

Srihari, S. (n.d.). Neural Network Training !

Stella, M., Russo, M., & Šari, M. (2014). RBF Network Design for Indoor Positioning based on WLAN and GSM, 8, 116–122.

Taok, A., Kandil, N., & Affes, S. (2009). Neural networks for fingerprinting-based indoor localization using ultra-wideband. Journal of Communications, 4(4), 267–275. http://doi.org/10.4304/jcm.4.4.267-275

Tian, J., & Shi, H. (2007). Study of localization scheme base on neural network for wireless sensor networks. IET Conference on Wireless, Mobile and Sensor Networks 2007 (CCWMSN07), 2007, 64–67. http://doi.org/10.1049/cp:20070084

Wang, H., Du, W., & Chen, X. (2015). Evaluation of Radio over Sea Propagation Based ITU-R Recommendation P. 1546-5, 10(4). http://doi.org/10.12720/jcm.10.4.231-237

Want, R., Hopper, A., Falco, V., & Gibbons, J. (1992). The active badge location system. ACM Transactions on Information Systems, 10(1), 91–102. http://doi.org/10.1145/128756.128759

Webs, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. Neural Networks, 1(4), 339–356. http://doi.org/10.1016/0893-6080 (88)90007-X

Xiao, J., Wu, K., Yi, Y., Wang, L., & Ni, L. M. (2013). Pilot: Passive Device-Free Indoor Localization Using Channel State Information. 2013 IEEE 33rd International Conference on Distributed Computing Systems, 236–245. http://doi.org/10.1109/ICDCS.2013.49

Xin, T., Yu, H., & Wilamowski, B. (2011). Comparison between traditional neural networks and radial basis function networks. 2011 IEEE International Symposium on Industrial Electronics, 1194–1199. http://doi.org/10.1109/ISIE.2011.5984328

Xu, C., Firner, B., Moore, R., & Zhang, Y. (2013). Scpl: indoor device-free multi-subject counting and localization using radio signal strength. Proceedings of the 12th …, 79–90. http://doi.org/10.1145/2461381.2461394

Xu, C., South, R., Brunswick, N., Firner, B., Howard, R., Lin, X., & Hall, J. H. L. (n.d.). Improving RF-Based Device-Free Passive Localization in Cluttered Indoor Environments through Probabilistic Classification Methods.

Harmony, S. J. (2008). Lecture 1 — Feb 25 Chertoff bound Empirical risk minimization Shattering, 1–8.

Yifeng, Z., & Shareef, A. (2006). Comparisons of three Kalman filter tracking algorithms in sensor network. Proceedings - 2006 International Workshop on Networking, Architecture, and Storages, NAS'06, 2006(2), 61–62. http://doi.org/10.1109/IWNAS.2006.22

Yu, H., & Wilamowski, B. M. (2011). Levenberg-Marquardt training. Industrial Electronics Handbook, Vol. 5 – Intelligent Systems, 12–1 to 12–18. Retrieved from http://www.eng.auburn.edu/~wilambm/pap/2011/K10149_C012.pdf

Foresee, F.D. and Hagan, M.T., Gauss-Newton Approximation to Bayesian Regularization. Proceedings of the 1997 International Joint Conference on Neural Networks, pp. 1930-1935, 1997.

MacKay, D.J.C., A Practical Bayesian Framework for Backpropagation Networks. Neural Computation, Vol. 4(3), pp. 448-472, 1992

MacKay, D.J.C., Bayesian Interpolation. Neural Computation, Vol. 4(3), pp. 415-447, 1992.

More, J.J., The Levenberg-Marquardt Algorithm: Implementation and Theory. Numerical Analysis, edited by. G. A. Watson, Lecture Notes in Mathematics 630, Springer Verlag, pp. 105-116, 1977.

DOAN,C.D., LIONG, S.Y., "Generalization For Multilayer Neural Network Bayesian Regularization or Early Stopping", In Proceedings of Asia Pacific Association of Hydrology and Water Resources, 2nd Conference(2004)

http://neuralnetworksanddeeplearning.com/chap5.html