WIRELESS NETWORK INTRUSION DETECTION MECHANISM USING ARTIFICIAL NEURAL NETWORKS

BY

ADERIBIGBE TOYESE I. 140407025

KOLAWOLE IKEOLUWA J. 140407007

Presented to

The department of Systems Engineering

University of Lagos

In partial fulfilment for the requirement of the award of the degree of

Bachelor of Science (B.Sc.) In Systems Engineering, Faculty of Engineering,

University of Lagos

Project Supervisor: Dr K.O. Orolu

November, 2019

DECLARATION

We hereby declare that we carried out the work reported in the Department of Systems Engineering, University of Lagos, under the supervision of Dr K.O. Orolu.

We solemnly declare that to the best of our knowledge; no part of this report has been submitted here or elsewhere in a previous application for award of a Degree. All sources of knowledge used have been duly acknowledged.

ADERIBIGBE TOYESE I.

140407025

KOLAWOLE IKEOLUWA J.

140407007

CERTIFICATION

This is to certify that the project "**WIRELESS NETWORK INTRUSION DETECTION MECHANISM USING ARTIFICIAL NEURAL NETWORKS**" submitted to the department of Systems Engineering, University of Lagos for the award of Bachelor of Science (B.Sc. Hons.) in Systems Engineering is a record of original research carried out by **KOLAWOLE I.J.** and **ADERIBIGBE T.I.** of the Department of Systems Engineering, University of Lagos, Akoka, Lagos

Dr. K.O. Orolu	Dr. F.O. Ogunwolu
Project Supervisor	H.O.D.

DEDICATION

We dedicate this project to God Almighty, for always being there for us and for helping us to successfully complete this project. We also will like to dedicate this project to our parents and families for the support and encouragement during the course of our program.

ACKNOWLEGEMENT

We would like to show our foremost appreciation to Almighty God for life, good health, a sound mind, and provisions throughout the duration of this project. We also want to express our utmost gratitude to our parents, Mr. and Mrs. Kolawole, Mr. and Mrs. Aderibigbe, for showing their support and encouragement during the course of this project. We sincerely thank our project supervisor and Project Coordinator, Dr K.O. Orolu, for his excellent guidance and supervision rendered in this project. We also thank the academic and non-academic staff of Systems Engineering Department for their support. We would also like to thank everyone that contributed to the success of this work.

ABSTRACT

The rapid increase in computer, mobile applications and wireless networks has globally changed the features of network security. There have been quite a number of Internet attacks, credential and data theft and also the increased spread of malware, viruses and even ransom ware acts on companies and individual networks which have shown us that computer networks are highly susceptible to intrusion.

The aim of this project is to design and simulate a network authentication mechanism that can detect and prevent cyber-attacks from unauthorised users irrespective of the penetration tool used.

Since the techniques developed on fixed wired networks to detect intruders have been rendered inapplicable in this new environment, the need for ways and methods to develop new architecture and mechanisms to protect mobile computing applications and wireless networks is important.

In conclusion the procedures applied in this project will assist network administrators in making critical decisions as to how to set up a wireless local area network and the security measures to put in place pertaining to the mitigation/reduction of risk posed by hackers and unwanted network users. The results presented in this project are of immense value to everyone who makes use of any wireless technology as a means of communication.

TABLE OF CONTENT

LIST OF ABBREVIATIONS

VPN: Virtual Private Networks

LAN: Local Area Network

MAC Address: Media access control Address

WLAN: Wireless Local Area Network

WPA: Wi-Fi Protected Access

WEP: Wired Equivalent Protocol

DNS: Domain Name Server

O.S.: Operating System

IV: Initiation Vector

DoS: Denial of Service

CRC: Cyclic Redundancy Code

WNIDS: Wireless Network Intrusion Detection System

IDS: Intrusion Detection System

HIDS: Host-based In-between Detection System

WSN: Wireless Sensor Network

IGANN: Improved Generic Algorithm Neural Network

ARM: Advanced RISC machines

RBF: Radial Basis Function

GA: Genetic Algorithms

WNN: Wireless Neural Network

AODV: Adhoc on Demand Vector

OWASP: Open Web Application Security Projector

DC: Dendritic Cells

TC: T-Cells

AIS: Artificial Immune System

DV: Danger Value

MCAV: Mature Context Antigen Value

LLN: Low-power and Lossy Network

WN: Wu- Manber

DoS: Denial of Service

ANN: Artificial Neural Network:

SVM: Support Vector Machine

KPCA: Kernel Principal Component Analysis

RF: Random Forest

RNN: Recurrent Neural Network

LSTM: Long Short-Term Memory

UAA: Universal Administered Address

LAA: Locally Administered Address

SN: Sequence Number

RFF: Ratio Frequency Fingerprints

MSE: Mean Square Error

LM: Levenberg-Marquardt

SCG: Scaled Conjugate Gradient

GDM: Gradient Descent with Momentum

CGP: Conjugate Gradient Back Propagation with polka-riebre update

RP: Resilient Back Propagation

BFGS: Quasi Newton

ROC: Receiver Operating Characteristics

FPR: False Positive Rate

FNR: False Negative Rate

TP: True Positive

FP: False Positive

TN: True Negative

FN: False Negative

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Wireless technology uses radio waves to transmit, without the use of wire or cables. The use of wireless technology started in 1879 (Estes, 2017), and has incredibly increase in functionality and usability over the years. We can see the practical applications in almost all walks of life with the most obvious being the wireless communications in electronic devices such as smartphones, tablets, laptops, smart fridges and so on. Networking technologies that connect multiple computers and devices together without using cables or wired is often referred to as wireless technologies, the popular term Wi-Fi is usually used to classify them.

Wireless solutions range from simple devices to complex ones. Complex systems include Portable Local Area Networks (WLAN) and mobile phones, while simplistic systems include devices such as wireless headphones, microphones and other tools that do not process or store information (Gopalakrishman, 2014). Wireless technologies also include infrared (IR) devices such as remote controls, some cordless computer keyboards and mice, and wireless headsets, of which a lot of these named devices require a direct line of sight between the transmitter and the receiver to close the link.

The importance of wireless technology in the home and business cannot be over emphasised. Wireless technology is used to for inter-connectivity of wireless-enabled devices and connection to the internet. However, with the wide use of wireless technology comes the risk associated with it. Security has been one of the biggest challenges facing the wireless technology enterprise (Fortinet, 2018).

1.2 Wireless Network

Wireless networks allow devices to be moved about with varying degrees of freedom and still maintain communication with other devices on the network (Harold, Kevin , Janne , Damon, & Douglas, 2010). Wireless networks offer greater flexibility than cabled networks and significantly save time, energy and reduce the amount of resources needed to set up a network or extend an existing network and also allows for ad-hoc networks to be easily created, modified or removed.

However, the growing popularity and widespread applications of wireless networks are directly proportionate to their propensity for security exploitation. Besides all of these advantage WLANs are facing the problem of security because many companies are transferring their sensitive data across the WLANs. In 1997, as the name suggests, IEEE (Institute of Electrical & Electronics Engineers) published the 802.11 Wireless Local Area Network (WLAN) protocol, belonging to the community of common IEEE 802.x standards, e.g. IEEE 802.3 Ethernet & IEEE 802.5 token ring (Harold, Kevin , Janne , Damon, & Douglas, 2010). IEEE802.11defines specifications for wireless LANs for media access control (MAC) and physical layer. IEEE introduced two enhanced specifications for the physical layer 802.11a and 802.11b in 1999 (The Government of the Hong Kong Special Administrative Region, 2010) with data transmission rates of up to 11 and 54 Mbps, respectively. 802.11b is also based on DSSS and operates in the 2.4 GHz band and 802.11a is based on OFDM (orthogonal frequency division multiplexing) and operates in the 5GHz band. In 2003, IEEE published 802.11b physical layer to allow data transmission speeds of up to 54mbps in the 2.4GHz band. Wi-Fi is one of the most common mobile technology that uses radio waves to provide high-speed wireless internet and network connectivity. The Wi-Fi Alliance, the company that controls the term Wi-Fi (registered trademark), explicitly describes Wi-Fi as any products based on the IEEE 802.11 standards". Initially, Wi-Fi was used in place of only the 2.4GHz

802.11b standard; however, the Wi-Fi Alliance has expanded the generic use of the Wi-Fi term to include any type of network or WLAN product based on any of the 802.11 standards, including 802.11a, 802.11b, dual-band, and so on, in an attempt to stop confusion about wireless LAN interoperability (Halvorsen & Haugen, 2009).

Wireless networks act as the medium for communication between devices and between devices and conventional wired networks (business networks and the Internet). Wireless networks are categorized into three groups based on their coverage range (Gopalakrishman, 2014):

1. Wireless Wide Area Networks (WWAN)
2. Wireless Personal Area Networks (WPAN).
3. Wireless Local Area Networks (WLAN)

Wireless Wide Area Networks (WWAN): connect individuals and devices over large geographic areas. WWANs are typically used for mobile voice and data communications, as well as satellite communications (Khan, Jonathan, Tahir , & Mohammad, 2008). WWAN includes wide coverage area technologies such as 2G cellular, Global System for Mobile Communications (GSM), and Code Division Multiple Access (CMDA) among several others.

Wireless Personal Area Network (WPAN): a wireless network on a small scale that requires little or no infrastructure and operates within a short range. Typically, a WPAN is used in a single room by a few users instead of attaching the machines to cables. Examples include print services or the possibility of communication with a wireless keyboard or mouse. WPAN represents wireless personal area network technologies such as Bluetooth and infrared (Khan, Jonathan, Tahir , & Mohammad, 2008). All of these technologies receive and transmit information using electromagnetic waves.

Wireless Local Area Networks (WLAN): Are clusters of wireless networking nodes capable of radio communications within a small geographic area, such as an office building or campus. WLANs are typically introduced to provide increased device mobility as enhancements of current wired local area networks. Wired local area networks (LANs) allow greater mobility and portability. In contrast to the wired LAN, which requires an RJ-45 Ethernet cable to connect to an existing port to connect a user's computer to the network, a WLAN connects computers and other components to the network using an access point (Khan, Jonathan, Tahir , & Mohammad, 2008). An access point interfaces to computers that have wireless network adapters; it links via an RJ-45 port to a wired Ethernet LAN. Usually, access point systems have coverage areas of up to 300 feet (about 100 meters) (Gopalakrishman, 2014). This area of coverage is known as a cell or array. Through their computers, tablets, or other network devices, users move freely within the group. Access point cells can be extended to increase the cell broadcast range which allow users to roam within a building or between buildings.

1.2.1 Ad Hoc Networks

Ad hoc networks like Bluetooth are networks designed to connect remote devices like cell phones, laptops, and PDAs dynamically. Thanks to their changing network topologies, these networks are named "ad hoc" (Gopalakrishman, 2014). Although WLANs use a fixed network infrastructure, ad hoc networks maintain arbitrary network configurations based on a wireless-connected master-slave system links to enable devices to communicate. In a Bluetooth network, the master controls the changing network topologies of these networks and also controls the flow of data between devices that are capable of supporting direct links to each other (The Government of the Hong Kong Special Administrative Region, 2010). As devices move about in an unpredictable

fashion, these networks must be reconfigured to handle the dynamic topology. The routing that protocol Bluetooth employs allows the master to establish and also maintain the shifting network.

1.2.2 Security and Privacy

Public Wi-Fi is one of the easiest and cheapest way to connect to the internet, with this convenience comes a greater security threat. According to the Norton Cybersecurity Insight Report, 594 million people around the world were victims of cybercrime in 2015 and majority of this attacks were from free public Wi-Fi. Free public Wi-Fi is a hacker's playground for stealing of personal information (Norton, 2019).

The need for improved security standard is more important than ever because there is an increase in internet usage

1.2.2.1 Categories of Security Threats

Network security threats can be subdivided in to four parts based on the type of intruder attempting to breach the network security protocols.

i. **Structured Threats:** Structured threats are threats posed by professional, advanced and sophisticated attackers trying to break into the network with the help of sophisticated tools and such attacks are usually well planned and coordinated with devastating results usually achieved (Khan, Jonathan, Tahir , & Mohammad, 2008). These kinds of threats are usually faced by governments and businesses as the attackers try to steal information which is worth a lot of money nowadays. These kinds of threats are usually difficult to defend against.

ii. **Unstructured Threats:** These threats are often carried out by regular computer users with no technical skill about exploits and little understanding of how network attacks are carried out (Khan, Jonathan, Tahir , &

Mohammad, 2008). These attackers make use of unsophisticated tools and resources that can easily be found on the internet or they even try to guess the passwords to the network. Unstructured threats can be stopped with the implementation of a simple security solution.

iii. **Internal Threats:** These are threats posed by insiders i.e. the network users inside a company, office, school, etc. Internal threats may occur due to an employee leaking information about the security protocols in place to outsiders such as the authentication key and user information (Khan, Jonathan, Tahir , & Mohammad, 2008). Internal threats to the network may also arise due to carelessness of the network administrators, easy to guess passwords, blackmails etc.

iv. **External threats:** These threats frequently occur over the internet and attacks could be perpetrated by both experienced and inexperienced hackers (Khan, Jonathan, Tahir , & Mohammad, 2008). Phishing sites and cloned sites are common tools used on the internet to gain access to computers and then these hacked computers known as zombies can then be manipulated to cause havoc to the local network, they are connected to e.g. a distributed denial of service attack (DDOS).

1.2.2.2 Wireless Network Security Solutions

There are different security solutions for the IEEE 802.11 standard like Wired Equivalent Protocol (WEP), WPA, and WPA2. Each of the following is explained in detail:

I. Wired Equivalent Protocol (WEP)

WEP is the first security technique used in IEEE 802.11 standards and it is the least secure of the available security solutions available. WEP helps to ensure secure communication and provides a hidden encryption system between the access point and the end user. WEP is introduced on the original Wi-Fi networks and without the right password, the user cannot access the device. WEP uses the share key authentication mechanism which requires two things for the user to access the WLANs, the service set identifier (SSID) and the access point generated WEP code.

WEP is considered a weak technique for WLANs security since it use RC4, a stream cipher that simply performs XOR plaintext gives cipher text, so a bit-flipping attack can make cipher text XOR and key give plain text easily and as a result the WEP key can be obtained in minutes using the right tools and techniques. The use of the CRC-32 system used for credibility checking is another controversial aspect of the WEP. Cyclic redundancy code (CRC) is defined as a class of "checksum" algorithms that treat any message as a large binary number and then divide it into binary by a fixed constant without overflow. The remainder is called the “checksum”. Due to the nature of CRC, it fails to provide the required integrity protection and it is not cryptographically strong and as a result of this WEP is no longer recommended as a security measure for wireless networks.

WEP uses the 24-bit long initialization vector (IV), which is a clear text added to the packet, and is then ready for transmission through the air where it can be exposed to attacks. Due to the small size of IV (24 bits), the weak authentication algorithm and the weak data encapsulation method, WEP suffers from lack of mutual authentication and key management. WEP has failed as a wireless security protocol due to its lack of integrity and confidentiality of data (The Government of the Hong Kong Special Administrative Region, 2010).

II. Wi-Fi Protected Access (WPA)/ Temporal Key Integrity Protocol (TKIP)

The WPA-TKIP was developed as a new solution for WLANs security that provides more security than WEP. TKIP is designed on top of WEP to fix all its known weaknesses. To increase the key ability of WEP, TKIP includes four (4) additional algorithms (Halvorsen & Haugen, 2009):

1. A cryptographic integrity test for messages known as Michael Integrity Code (MIC) to secure packets from bit-flipping attacks.
2. Like WEP plain text transfers, an IV encoding system that involves hashing.
3. A mixing feature per packet key to improve cryptographic power.
4. A re-keying mechanism to provide every 10,000 packets with key generation.

TKIP encryption algorithm is used to avoid the problem that may exist in WEP technique by generating a separate key for each packet instead of only one key for all packets in WEP (Halvorsen & Haugen, 2009). TKIP also solves the issue with IVs by increasing the size of IV that will help solve the problems by using a longer packet counter to avoid replay protection and by doing all this, TKIP is able to solve some of the problems in WEP.

III. WPA (Wi-Fi Protected Access) 2 / Advanced Encryption Standard (AES)

AES was established in 2001 by the American National Standards and Technology Institute (NIST) and is considered the best data encryption specification. It is based on the cipher of Rijndael, created by two cryptographers, Joan Daemon, and Vincent Rijmen, who submitted the application that NIST tested during the AES selection process. WPA2 structure is different from WPA and WEP because the ingredients single key management and message integrity, CCMP, based on AES (Vocal Technologies, Ltd., 2003).

The purposes of AES (CCMP) encryption are:

1. Counter mode is used to protect data from unauthorized access.
2. CBC-MAC is used to ensure the integrity of the email to the network.

AES is the best wireless authentication, which relies on the key schedule of Rijndael. (Narasimhan & Padmanabhan, 2013):

1. Initial round: add round key where the round key uses bitwise XOR to combine each byte of the state.
2. Sub bytes: a non-linear substitution step where, according to a lookup table, each byte is replaced by another.
3. Shift rows: a transposition stage in which a certain number of steps are cyclically moved from each row of the state.
4. Mixing columns: a mixing operation which operates on the columns of the state, combining the four bytes in each column.
5. Add round key.
6. At final round doesn't perform a mix column operation.

WPA2 protocol with AES encryption, which performs many rounds to ensure the key is a complex key, which makes it better than WEP that uses RC4 linear expected relation (Narasimhan & Padmanabhan, 2013). WPA2 protocol with AES authentication also differs from WPA / TKIP using RC4 and is known to be a WEP expansion with some enhancements, but TKIP encryption is still soft as WEP. Throughout MATLAB, AES authentication is introduced (Nevon Projects, n.d.).

A Wi-Fi Protected Setup (WPS) pin can also be used to connect users to the access point, but it can be hacked and attacked as well. According to the U.S-CERT, The Wi-Fi Protected Setup (WPS) PIN is susceptible to a brute force attack (Baek, Smith, & Kotz, 2014).

To overcome the dictionary and WPA handshake capture attacks on WPA / WPA2 protocols, many companies recommend using the 802.1x security protocol. This protocol blends the WPA2 with any powerful authentication server that relies on AES encryption. For better protocols such as LEAP (Lightweight EAP), many of these protocols improve EAP authentication, EAP-FAST, EAP-TLS (Transport Layer Security) or EAP-PEAP (Protected EAP), to mitigate the dictionary attack (Gast, 2009).

IV. Media Access Control (MAC) Access list

Access points can be configured to allow MAC address access to the WLAN. The aim of this security mechanism is to deny access to all clients except those expressly allowed to use WLAN i.e. clients with MAC addresses on the whitelist (Baek, Smith, & Kotz, 2014). A blacklist can also be used to secure an access point by placing unauthorized users on the blacklist. It is important to note that a whitelist is more effective than a blacklist because it is easier to place known devices on the whitelist as opposed to blacklisting unknown intruders. There is a great deal of effort to implement and manage permission lists. This method is not well-scale and is suitable only for small

WLANs. An attacker with the right set of tools can easily defeat access lists. It provides no security from the insider, who is an authorized user of the network (Gast, 2009). An outsider who receives a Wireless Network Access Card (WNIC) authorized to enter the WLAN is an insider. An attacker can also sniff the traffic between the AP and the client that collects a valid MAC address during the process and then create packets with a forged MAC address for easy WLAN access (Nevon Projects, n.d.). Although Media access control not a scalable security measure, this mechanism will stop an attacker without any specialized attack tools hence it prevents amateurs and regular PC users from accessing the network.

1.2.2.3 Wireless Network Exploits

Wireless network exploits are the means or techniques used by malicious attackers to get into a network by taking advantage of a flaw or weakness that exists in the setup of the network (Baek, Smith, & Kotz, 2014). The tables below contain a list of attacks, their descriptions and the affected security elements

Table 1.1: The Frame level Wireless attacks (Refaat, Abdelhamid, & Mohamed, 2016).

Attack	Description	Security Element
Man in The Middle attack (MITM)	If data is unprotected, hackers can intercept data.	Confidentiality Integrity
Dictionary attack	Programs that try large passwords from a dictionary to get the correct one.	Authentication Access Control

Stolen Handshakes	The attacker uses the role of the authorized client to steal the handshake between access point and client.	Authentication
Unauthorized client Access	If a network has a weak user authentication, it is very easy for a hacker to achieve access and take information	Access Control

Table 1.2: The RF level Wireless attacks (Refaat, Abdelhamid, & Mohamed, 2016).

Attack	Description	Security Element
DoS (Denial of Service)	Congesting a network resource with more requests to cause disruption of service.	Availability
Rogue Access Points	An unauthorized access point that has been connected to the wired network, which can provide malicious or una33uthorized users with open access to the LAN.	Availability

Table 1.1 shows the frame level group of wireless attacks while Table 1.2 shows the radio frequency level wireless attacks that can be used to gain access to a wireless access point. Each of the attacks is discussed below in detail.

1. **Denial of Service:** Denial of Service (DoS) It is a severe form of active attack on all types of wireless networks, particularly broadband networks, and can target a single node, a portion of the wireless network, a whole wireless network or wireless network resources. DoS attacks will undermine the two essential wireless network features that make them secure, i.e. data integrity and availability of services (Khan, Jonathan, Tahir , & Mohammad, 2008).

Denial of Service in any form against any network is usually regarded as a severe attack. Countermeasures need to be put in place in order to overcome such an attack and other related issues in broadband networks.

The results of different DoS attacks on broadband wireless networks vary with the nature and type of DoS attack.

1) There are Denial of Service attacks of low intensity which are usually launched against a single node either to exhaust its battery or to isolate it from other network devices.
2) Denial of Service attack is of a high intensity if it is launched to make services unavailable for a target area in wireless broadband networks.
3) A Denial of Service attack is said to be of the greatest severity if it is triggered by distributive flooding to cripple the whole wireless broadband network (Gopalakrishman, 2014). For this function, distributed flooding is usually used

to exceed the network's bandwidth or allow the gateway assets to spill.

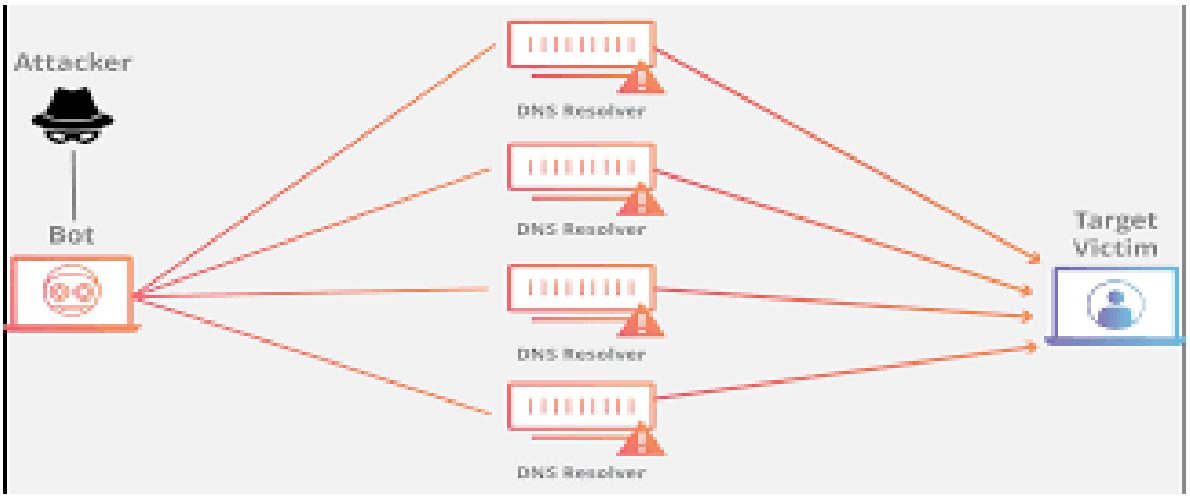


Figure 1.1: Diagrammatic Representation of Denial of Service Attack (www.cloudflare.com, 10-11-19)

2. **Rogue Access Points:** Whenever a user authenticates with a Wireless Local Area Network, the operating system saves the WLAN profile on the device so as to enable the user establish a connection to the wireless LAN whenever the device is within range. Nevertheless, when a wireless client switches to the previously used 802.11 network, the network management systems of the operating system usually try to immediately associate the profiles to one of the networks already protected by successful monitoring, often without the knowledge of the customer. (Nicholson, Chawathe, Chen, Noble, & Wetherall, 2006). Automated re-association raises the risk of evil twin attacks, where an attacker deploys a rogue access point with the same identity as one of a client's previously used and trusted APs in terms of network name (SSID) and MAC address. To make matters worse, jamming and other denial of service tactics can be used by attackers (Bellardo & Stefan, 2003) to prevent targeted clients from associating to secure access points, thereby encouraging clients to associate with an evil twin.

Once the user is coupled to a rogue or spoofed access point, the attacker can intercept the information of user/victim by transmitting the data or introducing a man-in-middle attack (Vijaya, Scrikanth, & Chandra, 2013).

3. **DNS Spoofing:** Domain Name System (DNS) (Mockapetris, 1987) is one of the key protocols for Internet applications ' efficient functionality. It provides a way to match a domain name with its corresponding IP address, thereby eliminating the need to remember web server IP address. DNS is also said to be a network registry repository because it not only retains the host name for IP routing, but also other records such as name server, mail exchanger, etc.
4. This protocol is vulnerable to attacks such as DNS cache poisoning (Kaminsky, 2008), DNS amplification attacks (Mc Farland, Shue, & Kalafut, 2017) and DNS request flooding (Zargar, Joshi, & Tipper, 2012).

Spoofing is typically a misleading or malicious practice where data is sent from an unknown source defined as the receiver's known source. Spoofing is most prevalent in systems of interaction that lack a high level of safety.

The aim of many attacks in network systems is to hack and spoof servers. We focused on DNS spoofing in this study. The hacker intercepts the address in the DNS spoofing domain sent by the client to the DNS server and is converted into a bogus IP address and then returned to the user. The client supports the attacker based on this situation and builds a TCP connection with the attacker. Finally, the attacker discloses confidential data of the user by exploiting of this connection (Aimasizadeh & Azgomi, 2008).

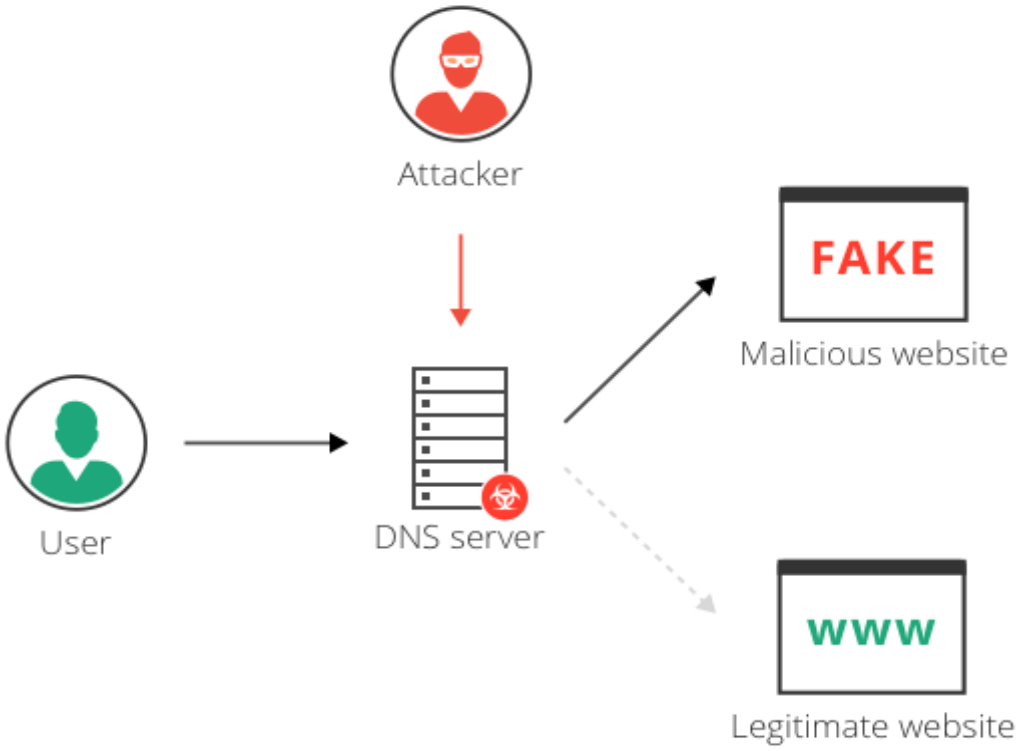


Figure 1.2: Diagrammatic representation of DNS Spoofing (www.imperva.com, 10-11-19)

5. **Brute force and Dictionary attacks:** Brute force and dictionary attacks involve the use of a file containing possible password combinations known as a dictionary. This dictionary can be created for a particular network by using a software like CRUNCH on Linux Operating System platforms to create all possible combinations of letters, numbers and alphanumeric keys (Refaat, Abdelhamid, & Mohamed, 2016).

   This dictionary is then loaded into a program with brute forcing capabilities such as AIRCRACK on Linux. Each password in the dictionary file is then tested based on the processor speed and available memory.

6. **MAC address spoofing**: MAC address spoofing is the process of collecting the MAC address of an authenticated user and then spoofing (changing) the MAC address of the rogue network card to that of a legit wireless network device.

Spoofing is usually used to beat MAC address filters.

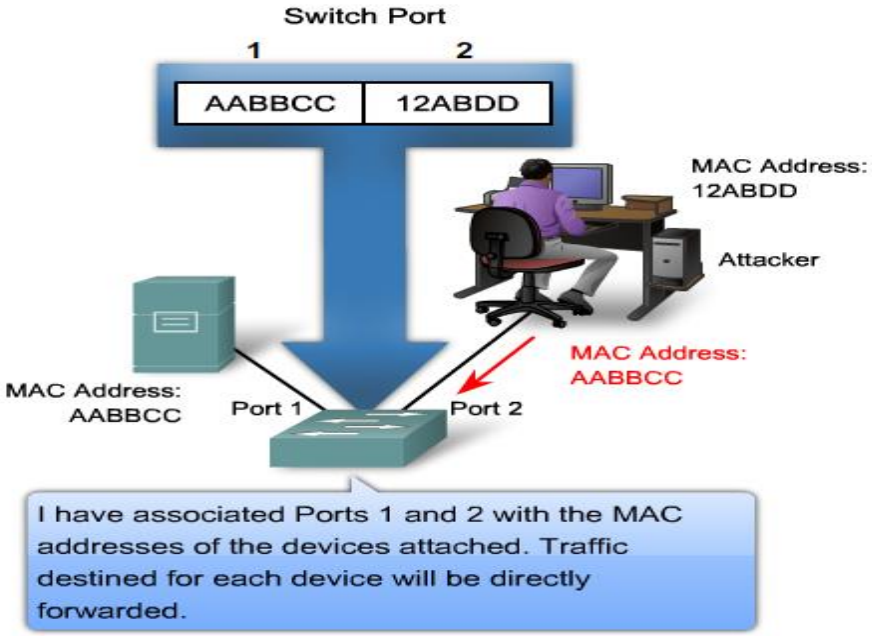


Figure 1.3: Diagrammatic representation of MAC Address spoofing (secureleaves.wordpress.com, 10-11-19)

7. **Man in the middle attacks (MITM):** A man-in - the-middle hacker entices machines to log into a vulnerable AP (Access Point) computer. Once this is finished, the hacker attaches to a real access point via another wireless card which provides a constant flow of traffic to the main network via the clear hacking machine (Choi, Rosslin, Robles, Hong, & Kim , 2008). The data can then be sniffed by the intruder. One method of man-in - the-middle attack depends on security flaws in the protocols of challenge and handshake to execute a "de-authentication attack." This attack forces AP to connect computers to drop their connections and to reconnect with the soft AP of the cracker. Software such as LANjack and AirJack enhances man-in - the-middle attacks (Choi, Rosslin, Robles, Hong, & Kim , 2008). Which automates the process's multiple steps. Now script kiddies can do what once required some skill.

Hotspots are particularly vulnerable to attacks as these networks have little or no security.

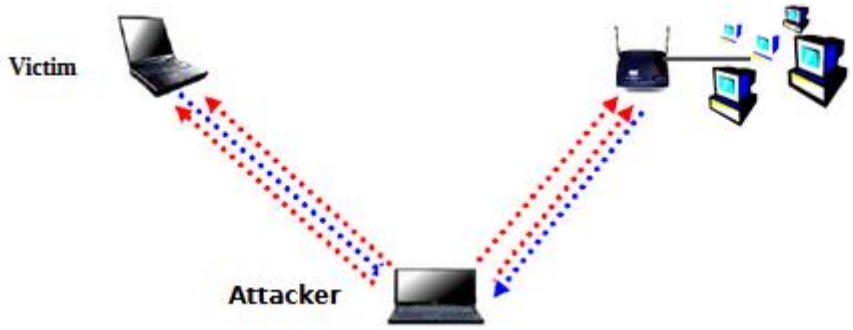


Figure 1.4: Diagrammatic representation of MITM attacks

1.3 Statement of Problem

Wireless is a safe way of communicating and sending information without the wire clusters and the high cost. Due to the increase in the dependency on wireless network, securing wireless networks has been a subject of discussion and this has also brought up certain issues as to the safety of this data. However, such networks are vulnerable to a variety of threats and attacks which are used for a variety of malicious purposes including malware injection or identity theft (Harold, Kevin , Janne , Damon, & Douglas, 2010).

The data gotten from successful hacks and attacks on wireless networks can be sensitive and highly confidential, hence the need to take preventive and defensive security measures to preserve the integrity of such data. The use of updated network security measures will go a long way in ensuring the safety of data transferred over such networks.

The method of intrusion detection using artificial neural network is a developing area of wireless network security, it will protect the network by using algorithm to detect pattern in the MAC Address accessing the network to check for its authenticity.

1.4 Significance of the Project

With the rapid exponential growth in technology, most especially the wireless technology, the need for an impregnable defence system against information theft, cyber bullying and hacking is more important than ever.

The project will be using the Artificial Neural Network technique, a branch of Artificial Intelligence (AI), to provide solution for wireless technology communication.

This will in turn protect sensitive information such as passwords, biometrics, the stock market and so on. The proposed solution will be very effective and it is can be developed across multiple platforms.

1.5 Aim and Objectives of the Project

1.5.1 Aim

Wireless networking is a rapidly expanding and evolving field. It is flexible, mobile and highly productive. The aim of this project is to design and simulate a network authentication mechanism that can detect and prevent cyber-attacks from unauthorised users irrespective of the penetration tool used.

1.5.2 Objectives

1. To study level of threat involved in MAC Address spoofing attack.
2. To train a learning tool to detect patterns between an authorised MAC Address and a fake one.
3. To develop a preventive algorithm to make security mechanism more defensive.
4. To deploy the algorithm on multiple operating platform.

1.6 Scope of Work

1.6.1 In Scope

This study will cover the bounds of wireless network intrusion detection using the MAC Address and will be limited to the neural network toolbox available on MATLAB.

1.6.2 Out Scope

Other forms of wireless network intrusion detection mechanism such as Two Factor Authentication (2FA), Virtual Private Network (VPN) and so on will not be considered within this study.

1.7 Report Outline

The Research work in this thesis has been divided into the following chapters:

Chapter one includes research motivation, significance of this project, project aim and objectives. Also involves the appropriate methodology used during this experiment.

Chapter two shows approach used in this project, review of the practical and theoretical literature on previous works on the project. Problems and limitations of previous works on project.

Chapter three discusses various kinds of spoofing attacks and their types, MAC address spoofing attacks, Artificial Neural Network and Feed Forward Neural Network algorithms. Simulation is done with methods of Back Propagation Neural Network that are: Gradient Descent Algorithm, Conjugate Gradient Algorithm and Quasi-Newton Algorithm. we analyse the behaviour of hidden neurons based on Epochs, MSE and Time.

Chapter four provides documentation on deployment of Intrusion Detection Mechanism on multiple operating platforms.

Chapter five summarizes research contribution with research findings and significance.

Finally, this chapter provides the recommendation on the current study along with future ideas.

CHAPTER 2

LITERATURE REVIEW

2.1 Wireless Network Intrusion Detection

A major component in Wireless Network Intrusion Detection (WNID) is called Deep packet inspection, in this process incoming data stream packets are to be compared with a set of patterns in a database, using string matching or regular expression matching.

Regular expression matching has a lot of flexibility and efficiency in terms of identifying various attacks; it also requires high computation and storage resources for Wireless Network Intrusion Detection Systems (WNIDS), which in turn creates a challenge in line-rate packet processing.

Detection of intrusion is the process of detecting hacks (actions) against information systems. To obtain unauthorized access to any computer system, these hacks are commonly known as intrusions. People who perform these hacks are called Intruders, and may be found externally or internally. Internally found intruders are network users with some degree of legitimate access attempting to increase their access privileges in order to gain unauthorized privileges. Outside the target network, outside intruders are commonly found to achieve unauthorized access to device data (Vacca, 2013). Intrusions are unwanted activities in our systems and they cause harm to computer systems. The WNIDS provides protection against unauthorized activities in the system. Intrusion Detection Systems are always required to provide protection against both insider and outsider intrusion (Safiqul *et al*., 2010). We are less successful against interference from within. An example is a cryptographic system that protects against intrusion by outsiders, but is not effective against intrusion by insiders. (Megha, 2015) declared that the importance of intrusion detection process is to represent, control and

analyse reports from the system and network activities. A standard WNIDS includes sensors, an analytics engine, and a reporting system that deploys sensors on a network or hosts at different locations. Its primary task is to collect data from the network or host, such as traffic reports, packet headers, connection requests, operating system calls, file system updates. Sensors send the collected data to the analysis engine, which investigates the collected data and detect ongoing intrusions. When there is an intrusion, the reporting system generates an alert to the network administrator.

WNIDS can be divided into different ways, which includes Network and Host based IDSs, active and passive based IDSs.

A. Network Based and Host Based IDSs

Network-based IDSs are put on networks primarily to identify attacks on the network of hosts. It must observe all the information that is passed on the network and analyse the traffic. Network-based IDSs consist of modules of hardware mounted on a separate system. Host-based IDSs are mounted directly on a network to control traffic to the site. They monitor all users' activities; this cannot be done on a network based IDSs. Trojan and backdoors are examples of threats that can also be stopped by the host based IDSs.

What is being implied from the previous paragraph is that, Network-based IDS (NIDS) can connect to one or various segments on a network and monitor network traffic for malicious activities. Host-based IDS (HIDS) is paired with a desktop interface for tracking malicious program operations, HIDS analyses not only network traffic, but also user calls, operating procedures, inter-process interaction, file system changes, and request logs. Signature-based, anomaly-based or specification-based approaches to IDS can be classified.

B. Passive based and Active based IDSs

Passive attacks do not change existing data and are not as effective as aggressive attacks; they are commonly referred to as an implicit attack. Passive IDSs simply alerts a user to suspected system threats while active attack alters existing data called a direct attack (Sheeba *et al*). The Active IDSs automatically blocks a threat without any intervention from a user.

The efficacy of an Intrusion Detection System can be measured using two well-known metrics. These well-known tests are false positives and false negatives. A false positive happens when an unwanted and false negative traffic is identified when an illegal activity (invalid traffic) is not observed. (Tavallaee *et al*, 2009) proposed that because of unavailable data sets for IDS; the efficacy used to measure the performance of any IDS is contentious. Various techniques have been brought up in literature for the construction of different types of IDS. Majority of them are predominantly resource intensive because of the size of both signature-based databases and anomaly models. In order to keep the databases and models up to date, constant updates are required (Milenkoski *et al*, 2015). As a result of the integrally heavy resource, both methods of IDS are not suitable for the constrained resources of IoT embedded devices (Gendreau and Moorman, 2016). It is very evident that these days there are wide varieties of literature(reviews) on different attack detection methods.

2.2 Wireless Network Intrusion Detection Techniques

Intrusion Detection Systems serve as a form of alert system in case of any unauthorized activity, and also protects against external and internal intrusions or attacks. In this review there are three major techniques amongst others that used in WNIDS.

A. Misuse Detection Technique.

These types of detection technique use a database of already discovered attacks. Network traffic, system-level activities, and other activities are usually compared to various signatures stored in the database. When there is a match the specific activity is tagged as suspicious. Repeatedly testing for open ports is an example of such activity, another example is when a shell code is detected in network packets. The Misuse detection technique is known for its success rate at identifying attacks that are measured as low false positives (known), but are ineffective at detecting attacks measured as high false negatives(unknown). This is as a result of the absence of signature for original attacks. Furthermore, when a device is constrained it becomes impossible to store and update the databases containing these signatures (Abduvaliyev *et al*, 2013). (Manali *et al*, 2014) proposed a Misuse Detection Technique known to be one of the most used intrusion detection method. It is also referred to as signature-based technique and can be very effective for known attacks or intrusions.

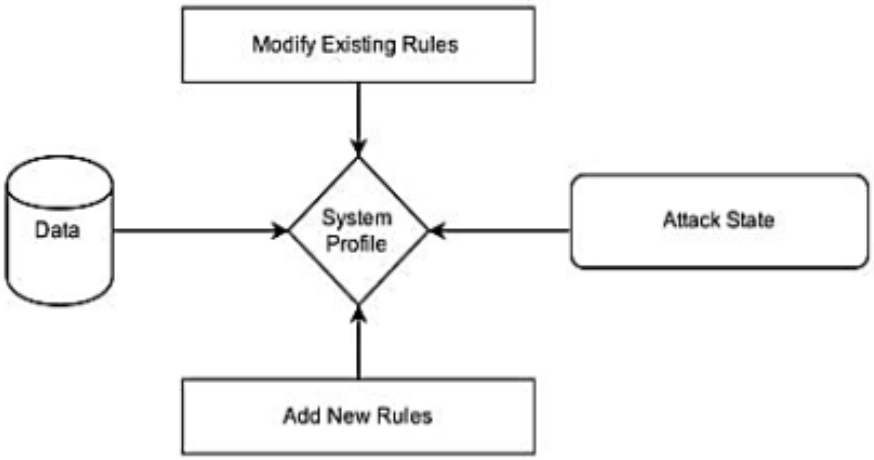


Figure 2.1 Representation of Misuse Detection Technique **(source: Bhupinder and Sumit, 2017)**

B. Anomaly Detection Technique.

The Anomaly detection techniques uses a different approach where a model representing a distinctive activity is developed which allows comparison between a current activity and the model, where any inconsistencies are tagged as suspicious. Consider an example where a model is built to monitor the active time and usage of various applications on a device and whenever an application is used outside working hours then that strange activity will be tagged. In activity models that are network related, we consider an example of a server gaining access to an unfamiliar address or service then it is flagged as a malicious activity. Anomaly detection techniques are very efficient at recognizing fresh attacks that misuse detection methods fail to identify and therefore their measure of efficacy shows a low positive rate. They are also exposed to high rate of false positives when their models are not frequently updated. False positives are also generated as a result of the constantly changing nature of wireless communications and technology associated with them. (Butun *et al*, 2014) proposed that constantly updating the models have proven to be very resource intensive and can be a form of complexity for resource-constrained devices.

Anomaly Detection techniques do not have the limitations found in Misuse detection techniques. They are generally used against unknown attacks (Jyothsna and Rama, 2011). These types of detection techniques raise an alert or alarm when there is a variation in the network traffic patterns from the usual traffic patterns. The rate of false positive is considered as one of the main disadvantages of this method of detection.

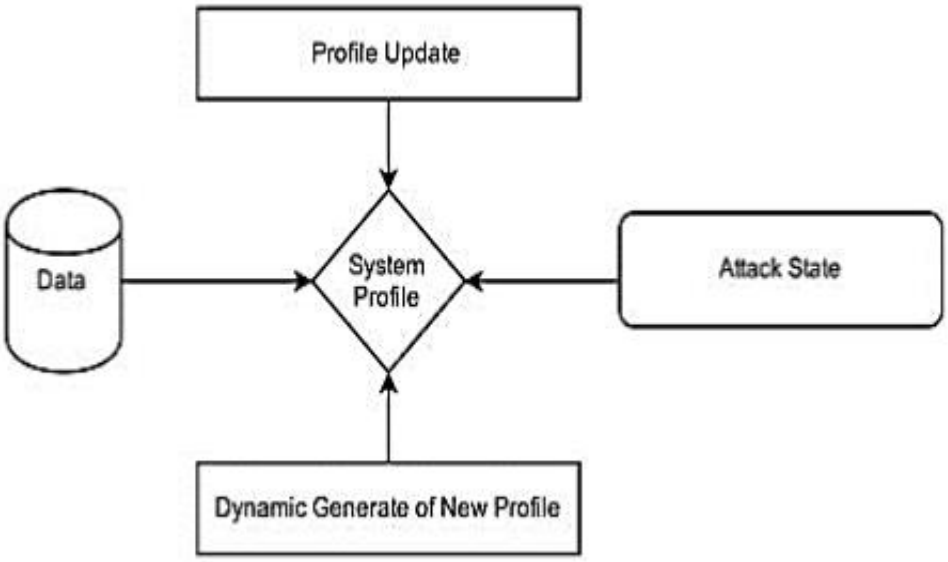


Figure 2.2 Representation of Anomaly Detection Technique (source: Bhupinder and Sumit, 2017)

C. Hybrid Detection Technique

They can also be referred to as Specification Based Technique. This method combines both Misuse Detection Technique and Anomaly Detection Technique, also these techniques have various advantages and disadvantages. The technique comprises of two detection elements, the first aids in identifying the known attack and the second is used for detecting unknown intrusions (Megha, 2015). Wireless sensor network (WSN) consume a lot of resources and energy, hence implementing hybrid technique is usually not favourable. This method has its modules classified into Analysis modules, Monitoring modules and Response modules. Hybrid detection technique also involves detecting strange activity from a model but this activity has to be confirmed as malicious by a user. One of the advantages of this method is its high accuracy, but this is also hindered by the timely user interaction when creating a signature (Butun *et al*, 2014).

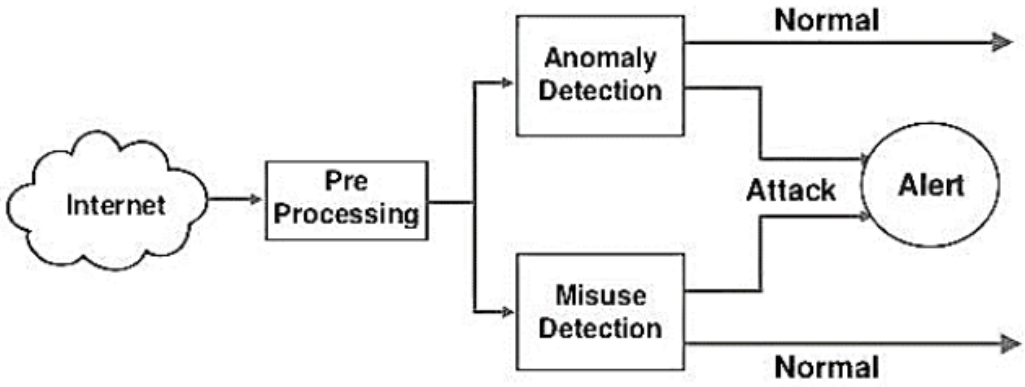


Figure 2.3 Representation of Hybrid Detection Technique (source: Bhupinder and Sumit, 2017)

2.3 Other Wireless Network Intrusion Detection Techniques

A. Genetic Algorithm Based IDS.

These days Artificial Intelligence in its own definition has created a certain awareness for Fuzzy logic-based methods associated with specific Artificial Intelligence techniques. Fuzzy logic is used to generate models of fuzzy association used for data classification and these can be applied with certain techniques associated with Genetic Algorithms. An example is applying these models with genetic programming to achieve better results. A genetics-based fuzzy algorithm which was devised by (Wang, 2009) and used to analyse current IDSs. The algorithm has two stages, the first involves generates initial rules from fuzzy algorithm, the second involves the construction of a fuzzy system with high performance, as a result optimized member functions used in GA in combination with fuzzy algorithm. It is then attached to IDSs for better performance. Another example is the IDS based on Improved Genetic Algorithm Neural Network (IGANN) which uses advanced RISC machines (ARM) wireless mesh network introduced by (Meijuan, 2009). The process behind this scheme was that data was acquired from sensors and processed using an algorithm for image detection, the processed data is sent to a control centre using wireless network. If unusual activities are detected, IGANN uses a camera to detect face image. Floating point code provides

system with high accuracy and speed. The system provides a form of self-relief for workers. (Yonghui Shi *et al*, 2011) proposed that to overcome the issue of precision in these IDSs a method of detection using Chaos optimization and Radial Basis Function (RBF) is sufficient.

Datasets containing intrusion models are trained using the generalization ability of RBF detected model. Standard GA methods of intrusion detection lack the detection rate and calculation speed derived from the system discussed above. One of the most recent findings on fuzzy based GA methods is detection of data from a network database introduced by (Jabez and Mala, 2012). Recent research plan to use Wavelet Neural Network (WNN) in Intrusion Detection Systems but have been found to be inefficient in detection. GA was combined with WNN to improve its network performance (Hai-Yan *et al*, 2009). Attacks can be directed to the AODV (Adhoc On Demand Vector), to prevent these attacks (Sujatha *et al*, 2012) introduced a hybrid IDS using GA. The process involved analysis of nodes and their behaviours and providing detail of attacks. Features like request forwarding rate and reply receive rates were considered when developing set of rules that governs this process.

(Torkaman *et al*, 2013) proposed an approach for Hybrid Intrusion Detection Systems, which was constructed to have two-layer GA and Neural network, it used data mining techniques to evaluate traffic from web applications. This new model created awareness to liabilities based on Open Web Application Security Project (OWASP). Genetic Algorithm provides the model discussed above with the ability to improve individual detectors algorithm and reduced redundancy as a result of GA characteristics like automatic optimization and many more.

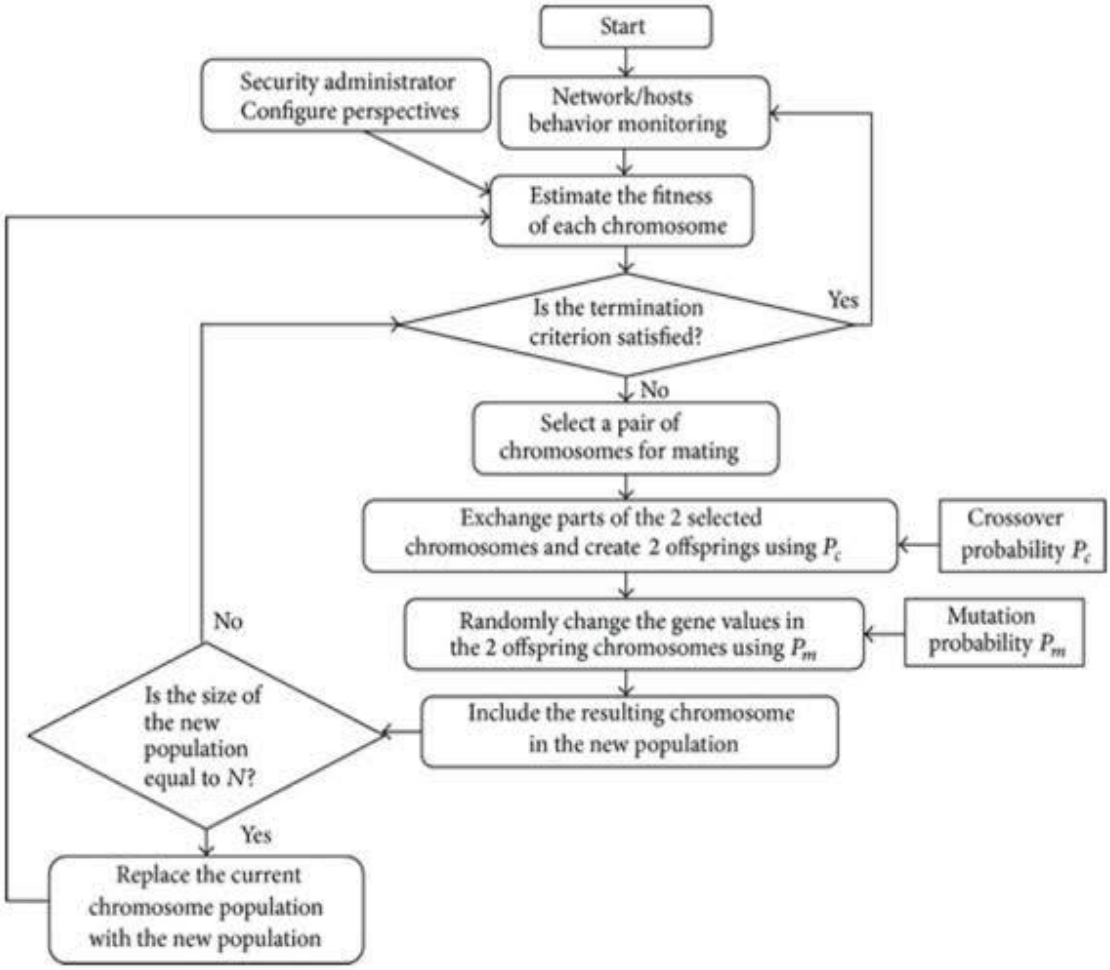


Figure 2.7 Implementing Genetic Algorithm for IDSs (source: www.researchgate.net, 10-11-19)

B. Artificial Immune System (AIS) Based IDS.

AIS based IDS use agents and were developed with the concept of the human immune system and its danger theories (Chung-Ming *et al*, 2011). Innate immune subsystem was represented using Dendritic Cells (DC agent), and the adaptive immune subsystem was modelled after artificial T-Cell agents (TC agent). The AIS Based IDS represents system calls by using the concept of Antigens and the behaviours are used as signals.

Detection in AIS based IDS has a duality property, that is, detection of DC agents for signals and TC agents for detecting antigen, and both agents resolve the Danger Value (DV). When these agents are used in AIS based IDS, it provides the system with

memory and the ability to learn. Malicious activities are provided immune response from a host according to their DVs. Security responses are as a result of agents deriving a Mature Context Antigen Value (MCAV) for updating activation threshold. Intrusions on host networks are detected with the help of input signals and temporary output signals, danger or safe signals. A big issue in this method of network intrusion detection is how detection accuracy and efficiency can be improved. The concept of e-learning was one amongst others that were developed by (Zhou, 2010) from which he developed a structure for IDS model. E-learning is a concept on which the architecture of AIS based IDS was built. AIS has unique features that inspired researchers to find ways of implementing this concept into various forms of application which includes detection of intrusion. (Hosseinpour *et al*, 2010) proposed that the performance of IDSs could be improved by using a distributed multi-layered framework, the design uses a central engine, sensors, and agents that identify intrusions. The design was to decrease the time used during detection for connections by giving each host a detector. A detection method based on Hollands classifier introduced by (Randrianasolo and Pyeatt, 2012) provided more knowledge on the possibility of combining AIS and Hollands classifier for identifying intrusions on computer network. It achieved more than 90% detections and the percentage of false negatives was not more than 10%. An experiment was performed using a network with regular nodes and intruders, giving a result of low false positive rates. An observation that involves setting tolerization and parameters differently gave the optimum performance, the more parameters involved the more the number of experiments needed to further analyse the IDS. Generating detectors can be achieved with a kernel algorithm in AIS, the algorithm is divided into processes such as; gene library, negative selection and clone selection.

A simulation, modelling intrusion detection on a network using AIS principles and structures was introduced by (Yu Jing and Feng, 2010) to have distributed nodes which were not centrally controlled at various locations, for the analysis of invalid behaviour and valid behaviour. When an anomaly is detected an alarm is raised by the model. Intrusion detection has gained awareness over the years, due to increase in cyber-attacks on a global scale. This resulted into the need for more accurate IDSs and researchers use technologies like Neural network, Fuzzy logic and others in modern IDSs to achieve reasonable results. Finally, an AIS based IDS that involves optimizing feature selection and quantization algorithms was introduced by (Junyuan *et al*, 2011), the design was used to test KDD CUP 99 dataset and it gave a result that was shown to have performed better than other schemes in terms of detection accuracy.

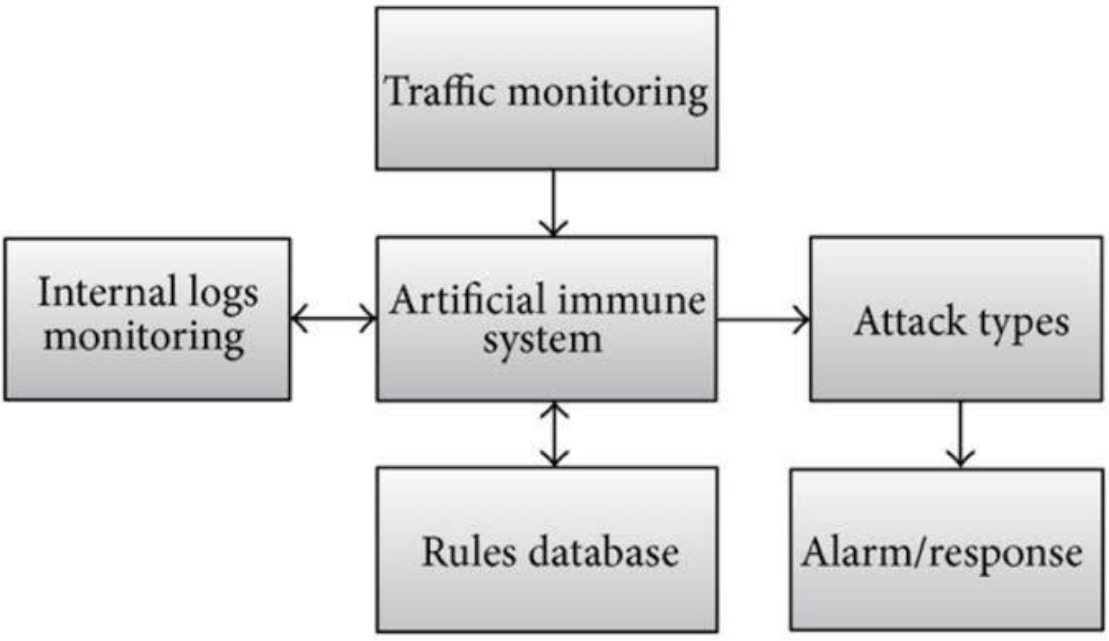


Figure 2.8. Architecture of AIS based IDS (source: www.researchgate.net, 10-11-19)

2.4 Placement Strategies Used for Intrusion Detection

Intrusion Detection Systems can be located in routers, or in dedicated hosts. An advantage of having IDSs in routers is that it helps prevent attacks from the internet from causing harm to objects in a physical domain. IDS located in routers can cause a bridge in communication between the Low power and Lossy Network (LLN) nodes and the router itself, as a result of IDS frequently inspecting the state of the network. When IDSs are located in LLN nodes, there is a possibility of a reduction in communication overhead related to network monitoring, the process requires processing, storage and energy resources (Wallgren *et al*, 2013). LLN nodes have some resource restrictions that may generate issues.

Another challenge that may be encountered is that for more processing capacity and less monitoring traffic to be achieved, allocating IDSs agents on some dedicated nodes serve as a solution but requires organizing the network into various regions which might be a complex process. In this section we have suggested feasible placement strategies used for IDSs and possible drawbacks, as well as various advantages.

2.4.1 Distributed IDS placement

Every physical object of the LLN in this placement technique or strategy have IDS located in them. Due to the nature of the resource constrained nodes, there is a need to optimize IDSs located in each node. This issue was addressed by the use of light-weight IDSs (Oh *et al*, 2014) and (Lee *et al*, 2014). A light-weight algorithm for identifying signature attacks and packets payload was developed (Oh *et al*, 2014). Using auxiliary shifting and early decision techniques to achieve a reduction in the number of needed matches to detect attacks. This approach was compared with Wu-Manber (WM) pattern matching algorithm and was observed to be faster while running on a platform that is resource constrained. (Lee *et al*, 2014) proposed

another approach to analyse and monitor the energy consumed by nodes used to detect intrusion. The process behind this setup was to reduce the amount of computing resource needed for detecting intrusion, by concentrating on a single node parameter.

This placement strategy allows the nodes to take responsibility of monitoring their neighbours, and they can be referred to as watchdogs. Watchdogs can be combined with the concept of trust and reputation as a measure for detecting attacks (Cervantes

et al, 2015). Nodes are classified into various groups (leaders, member nodes) with ranked structure. After an attack or reconfiguration of a specific network, the role of each node may differ from their initial assignment. Now each node can observe a higher node by approximating inbound and outbound traffic. The attacker is then isolated if an attack is detected by a node, by alerting other nearby nodes.

2.4.2 Centralized IDS placement

The centralized placement strategy involves a centralized component, the IDS is located on the border router or dedicated host. Data collected by LLN nodes to be transmitted to the internet pass the router along with internet client request transmitted to the LLN nodes. (Raza *et al*, 2013) proposed a method of placing the IDS in a border router to monitor the traffic between the LLN nodes and the internet. Analysing traffic crossing the router has limitations in detecting attacks associated with nodes in LLN. Therefore, there is a need for IDS that monitor traffic between LLN nodes while considering the impact this has on nodes operating at low capacity. Compromising attacks to part of a network makes a centralized IDS ineffective at monitoring nodes during the attack. A solution for analysing packets

that transverse between physical and network domain in a router was introduced by (Cho *et al*, 2009), basing the work on botnet attacks.

Another approach used this placement strategy while considering DoS (Denial of Service) attacks (Kasinathan *et al*, 2013). The process involves a powerful dedicated host with an IDS analysis engine and report system. IDS sensors are deployed in LLN, to sniff the network traffic and deliver the data collected to the analysis engine. To avoid transmitting the data from both the network and IDS in the same wireless network the host is connected to the sensors in the IDS with a wire. This is a form of preventive technique for when a DoS attack damages the wireless transmission quality, the data being transmitted by the IDS won't be affected.

Another approach looked at locating the IDS in the border router to detect attacks targeted at the physical domain, this time a protocol called the heartbeat protocol was introduced instead of sniffing border router traffic (Wallgren *et al*, 2013). The setup behind this protocol is that the router sends ICMPv6 requests to all LLN nodes at equal intervals and a response is expected to detect any attack or availability problems. This process produces additional network traffic, but according to results from experiments the LLN nodes does not require additional memory for the heartbeat algorithm, it also required minimal energy.

2.4.3 Hybrid IDS placement

In Hybrid IDS placement strategy, the main concepts behind this method are centralized and distributed placement strategies. It takes advantage of their best features and minimizes their drawbacks. The network is grouped into clusters (regions) where only the main node hosts an IDS. This is now the monitoring node; its job is to monitor the other nodes in each region. Only appointed leader nodes that are robust can host the IDS. This placement strategy consumes much more resources

when compared to an example of a distributed placement IDS (Amaral *et al*, 2014). Appointed nodes sniff the exchange packets around them and infer when a node is compromised by following a set of rules. Different behaviour shown by each component on the network requires the appointed node to have particular sets of rules. This approach gives the advantage of developing multiple rules for each network region.

(Le *et al*, 2011) proposed an approach developed to use hybrid placement with a backbone of nodes for monitoring. It required a lesser number of monitor nodes to observe the network, to detect an intrusion the monitor node observes if neighbouring nodes have been compromised. Its major advantage was producing additional communication overhead, as a result of observing transmission from neighbouring nodes.

Recent works show the network being organized into small regions with the same number of nodes (Le *et al*, 2016). A node referred to as a cluster head having direct contact with all the member nodes in the cluster. The IDS is located in each cluster head that monitors member nodes by sniffing their communication. Cluster members serve as a means of reporting information about nodes in the cluster to the cluster head.

IDS can be located in the border router and also on the other network node. This approach involves a central component. When an IDS is located in the router it becomes associated with tasks that demand resource capacity, but the one located in the normal nodes are light-weight. SVELTE Hybrid placement IDS was introduced by (Raza *et al*, 2013), the hosts obtain IDS elements needed for intensive processes like detection of intrusion by monitoring RPL network data. Network nodes are used for light-weight tasks like reporting RPL network data and malicious traffic found.

Network nodes can be tasked with identifying changes that occur in their neighbourhood and report information to the centralized component in the border router (Pongle and Chavan, 2015). Storage and analysis of data acquired for detection of attacks and intrusions occurs in the centralized component. This method of placement was observed to have performed above standards for an environment with constrained nodes.

Finally, considering the IDS designed to appoint different tasks to the router and nodes on the network, allowing them to support each other (Thanigaivelan *et al*, 2016). The setup uses network nodes to detect attacks around their neighbourhood and reports to the IDS component on the router. Where the data is pulled from various nodes to take required actions about the intrusion. This method can be categorized as an example of the distributed placement strategy but the router acting as a central component for making decisions regarding the intrusions makes it an IDS using Hybrid placement strategy.

2.5 Validation Strategies Used for Intrusion Detection

Validation helps confirm the developed model or system works within the scope of our study objectives. Validation techniques are categorized by these two sources which are experts and data. The experts are tasked with providing a particular and qualitative validation of the model, while data serves to generate a quantitative and objective validation of the model (Chrun, 2011). The validation of strategies used in the intrusion detection techniques have been classified using (Verendel, 2009) validation methods:

1. Hypothetical: When the relationship between the actual phenomena and the degree of realism is unclear.

2. Empirical: When data is collected in a systematic way from doing experiments with operational settings.
3. Simulation: When different scenarios are simulated for data acquisition.
4. Theoretical: Providing precise theoretical findings or arguments to support results.
5. None: When there is no form of validation method used.

Independent validation and comparing results by recurrent large-sample tests is a major reason scientific development rely on the ability to re-reproduce results from experiments (Mahoney and Chan, 2003). Most of the traditional IDSs have based their evaluations on data collected from experiments conducted at Lincoln Laboratory (1998 and 1999). This research is considered as one of the most complete assessment of the research work conducted on IDSs (Shiravi *et al*, 2012). Several researches still conclude that the data set used are outdated and do not specify the latest trends of attacks, and the data used is very important in the study of the precision of a model (Nehinbe, 2011). It can be inferred that validation techniques used in IDSs are of no certain standard.

2.6 Artificial Neural Network

2.6.1 Artificial Neural Network (ANN) and Wireless Security

Many researchers have applied ANN to solve various related wireless security issues. (Kuang *et al*, 2014) proposed a support vector machine (SVM) that combines kernel principal component analysis (KPCA) and genetic algorithm (GA) as a method of intrusion detection. The approach was developed by using the popular KDD Cup99 dataset (Hettich *et al*, 1999). Another approach also developed an intrusion detection system using SVM and adopting effective discriminant function for the improvement of accuracy (Reddy *et al*, 2016). (Ingre *et al*, 2015) proposed a method to analyse NSL-

KDD performance using neural networks. (Farnaaz *et al*, 2016) went ahead and advanced this method by implementing Random Forest (RF) and NSL-KDD datasets to access the performance of this model. An intrusion detection system, that uses RF and KDD Cup99 to review the success of their model was developed by (Zhang *et al*, 2008), the development of this method gave rise to some complications which involved the low performance rate when selecting or extracting features from the big data, this also led to the decrease in accuracy of the method (Niyaz *et al*, 2015).

Recent researchers have now diverted more attention to deep learning a branch of artificial intelligence under machine learning, since the methods discussed so far do not generate better results. (Yin *et al*, 2017) proposed a method to classify intrusion detection using deep learning techniques like Recurrent Neural Networks (RNN). The approach has been developed for classification processes on NSL-KDD dataset using deep learning techniques that generated better results, compared to the initial method used. Therefore, RNN can be successfully used for detection of intrusion and to generate promising results for classification of intrusion. (Hochreiter and Schmidhuber, 1997) proposed a method of Long Short-Term Memory (LSTM) for intrusion detection, this method is reviewed using NSL-KDD dataset and is extensively compared with RNN based intrusion detection systems.

2.7 MAC Address Spoofing

The MAC address plays an important role in us accomplishing this objective, the MAC address is a 48-bit address linked to hardware component of network adapters. The MAC address can be permanent and can be altered in some situations. We have considered two certain types which are the Universally administered address (UAA) and the Locally administered address (LAA). UAA are usually assigned upon manufacture (Hatkar *et al*, 2012). 24-bit define the manufacturer while the next set of 24-bit

identifies the individual adapters. LAA changes an adapter's MAC address. MAC address spoofing has become a prominent attack for hackers. Various methods have been introduced by researchers to prevent and detect MAC spoofing attacks (Babu *et al*, 2010). An approach termed sequence number analysis involves MAC headers with sequence number (SN). SN fields are analysed in MAC headers for spoofing detection. Hackers can manipulate the MAC address but not a sequence number, as a result of this the attacker's frames would have a sequence number not incremented by one. The mismatch between sequence number with the same MAC address serves as a hint that something is wrong (Wright, 2003). Another approach was to use signal print, as a result of attacks being able to change MAC addresses but not physical location. Packets are tagged with signal points and a matching rule is generated to enable wireless networks detect DoS attacks based on MAC address spoofing (Faria and Cheriton, 2006).

An algorithm to detect and prevent 802.11 WLAN environments from MAC layer DoS attack was introduced by (Arockiam and Vani, 2013). Passkeys are exchanged between a user and an access point with a time stamp. NS2 simulator tool validates this algorithm. MAC address spoofing can be identified by using Radio Frequency Fingerprints (RFF). To enhance the performance of this IDS a transceiver is linked with the MAC address, the RFF process detects a transceiver based on the signal portion it generates (Hall *et al*, 2012). Finally, another algorithm based on smart antenna with shared key exchange method of detecting and preventing MAC spoofing (Hegde, 2016).

2.8 Limitations and Future Directions

In this research we have chosen the approach of implementing intrusion detection systems to device a mechanism that follows principles of artificial neural network. In this section, we reviewed various issues and concerns in the implementation of Wireless

Network Intrusion Detection Systems, and the possible directions for future research. To overcome limitations in WNIDS, experiments should be carried out to establish pros and cons of detection methods and placement strategies. Another cause to the limitations is that the range of attacks being researched on can be somewhat constrained to a particular set and this creates a problem when implementing the IDSs in the real world, as the number of attacks increase with each passing day. Improving the validation strategies used in WNIDS can go a long way to providing more reliable systems for detecting network intrusion. Also, effective management of traffic monitoring and reporting techniques should provide solutions to the limitations on WNIDS. WNIDS cannot depend on human interaction forever, hence further studies should be directed at making autonomous IDSs. Where these systems would be able to configure, adjust and reconstruct functions with little to no human intervention.

CHAPTER 3

METHODOLOGY

3.1 Artificial Neural Network

Artificial neural network is built on the model of the human brain and is a form of handling system that adapt and recognize patterns. ANN aims to perform functions faster and considerably provide a different approach, compared to the outdated approaches. The outdated approach required a system programmer to translate problems into code and process it using the computer system while generating result with respect

to the situation. ANN learns patterns, maps problem and generates result without coding the problem. ANN can be applied in fields such as engineering, physics, biology etc. Computers find it difficult to perform some ANN task such as; optimizing function, vector quantization, data clustering or grouping and matching patterns. Human brain can collect information about past incidents and apply the data in future scenarios. It consists of cells termed as neurons that are interconnected. Neurons can be grouped into four regions: Dendrites, axon, synapses and Soma. Soma (Cell body) has a nucleus and trees in form of networks made up of fibre connected to the dendrites. Connections from the soma to other cells is referred to as axon. The synapse transmits signals to neighbouring neurons and it is biochemical in nature. Hence, the functions of the four regions can be described as follows:

1. Soma: A cell body that adds up all the incoming signals.
2. Dendrites: Receives incoming signals as a result of nerves attached to the soma.
3. Axon: stores impulses of neurons and releases when enough input is received.
4. Synapse: it changes pre-synaptic electrical signals to chemical signals.

ANN properties can be associated with neurons in biology, neurons are treated like cells in neural network. Dendrites serve as accepted input of weights and interconnection, soma helps convert input while axon generates output from input (Haykin, 1998).

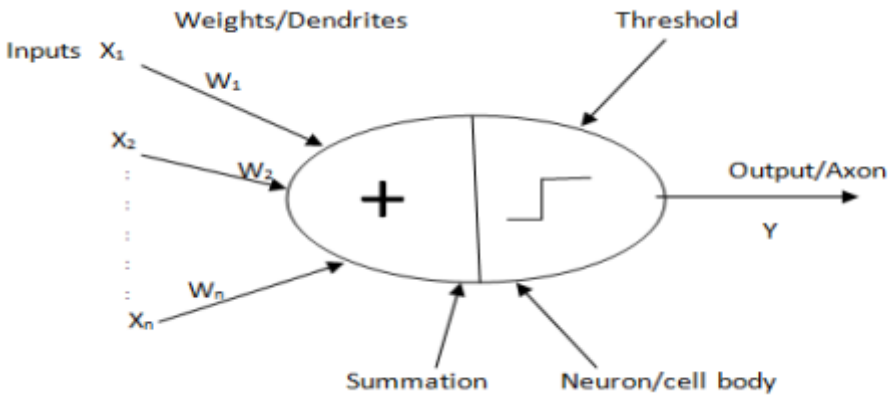


Figure 3.1. Association of biological neural network to ANN (source www.slideshare.net 11/11/19)

Figure 3.1, illustrates the association of biological and artificial neural network. Sender release a certain type of transmitter element while processing elements to receive signal. Elements result to modify incoming signals by adding up the weighted inputs. When the potential gets to the threshold the neurons send output to the synaptic junction of further cells through the axon. Neurotransmitters tend to be either inhibited or excitatory depending on how they hinder or effect the release of the receiving cell. From figure 3.1 the net input can be shown as:

$$Y_{in} = X_1W_1 + X_2W_2 + \ldots\ldots + X_nW_n \quad 1$$

$$\sum_{i=1}^{n} x_i w_i \quad 2$$

From equation 2, i represents the ith element while weight is represented by the synapse strength (Sivanandam and Deepa, 2011). Positive weight results into an excitatory synapse, negative weight produces an inhibitory synapse.

3.1.1 Back Propagation Neural Network

A neural network has basically two types of networks which are called feed forward and feedback neural network. Feedback NN generates output value by reverting input

values. Feed forward networks involve the movement of information in the forward direction (YuLiy, 2002). The most common development in NNs is the Back-propagation algorithm; it is also the most used algorithm for experiments. The developers of the algorithm were G.E. Hinton, Rumelhart and R. O. Williams (1986). This algorithm uses multi-layered feed forward network, with the idea of gradient descent approach where propagated errors are sent back to previous units. This network is used to achieve equilibrium between the input patterns memorized by the networks and networks identical to the ones used for training.

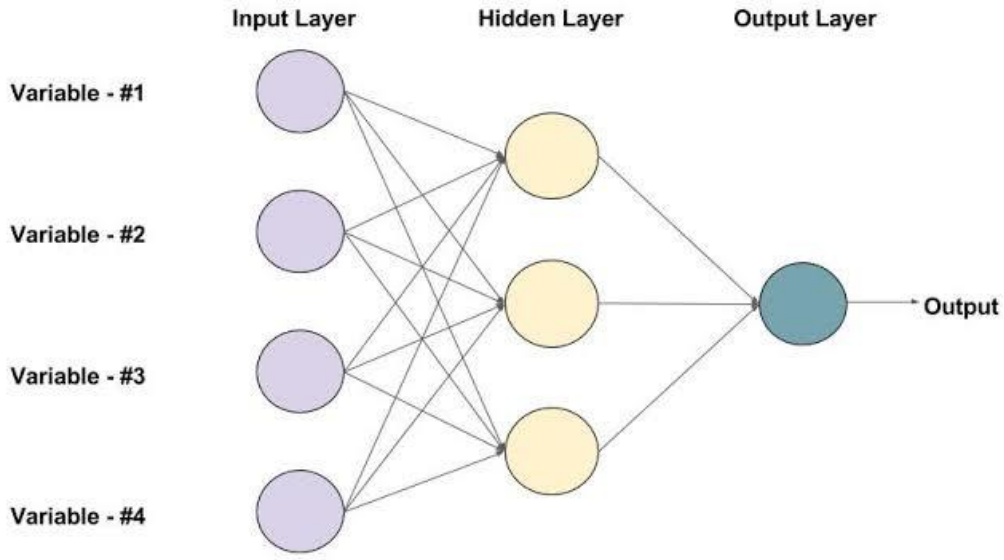


Figure 3.2 Feed Forward Neural Network (source: www.learnopencv.com 11/11/2019)

Calculating weights of hidden layers to achieve zero output error can be difficult. As a result of the complexity of the network increasing as the number of hidden layers increases as well. There are three training steps involved in Back propagation: feed forward of the input units, calculating errors and updating the weights. The architecture behind back propagation is made up of input, hidden and an output layer. The hidden layer is not fixed in number. From figure 3.3, the input later is represented by x, hidden

by z and output by y and can be referred to as the target output vector. Also, the forward movement of information can be observed but the signals move in the reverse direction as well. Output and input layer comprise of biases with activation as one and act as weights (Zweiri *et al*, 2005). Results of input and output can either be in binary or bipolar form, while the activation function increases monotonically.

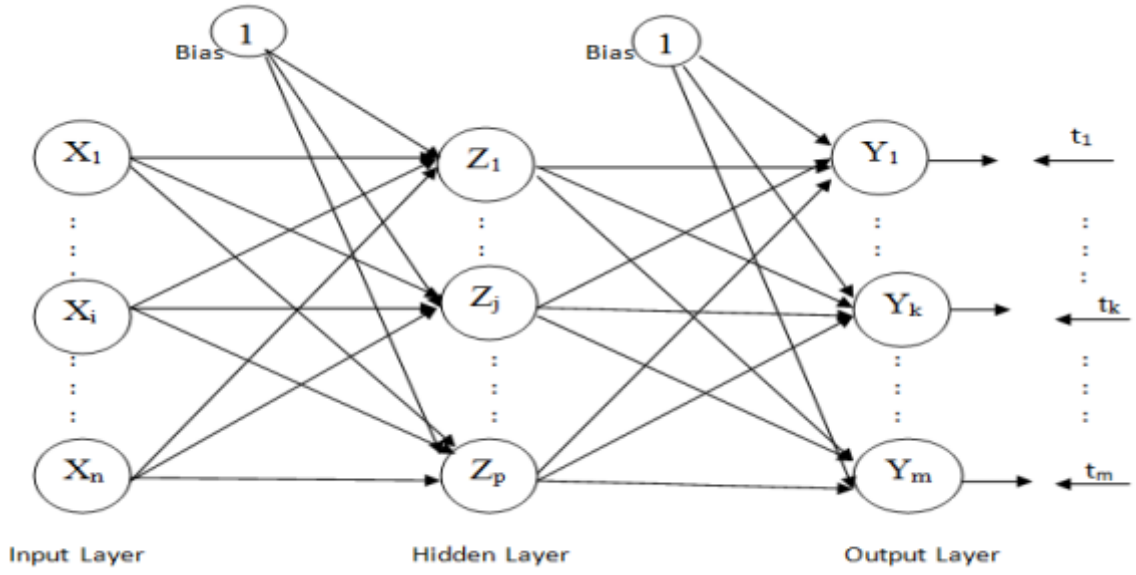


Figure 3.3 Architecture of Back Propagation Network (source: www.researchgate.net 11/11/19)

The steps behind the back-propagation algorithm can be classified into four steps (Sivanandam and Deepa, 2011):

1. Initialization of weights: Minimal odd values are assigned and we initialize learning rate.
2. Feed Forward: All layers are connected and input units transmit input signals to next layer (hidden layer). Activation process is applied on the weighted input signals from hidden layers, the signals are transmitted to next units. Output unit then activates weighted input signals to determine the output signals.
3. Back propagation of errors: Output units acquire a desired pattern that corresponds to the input pattern to calculate error, after which the weights are

updated and delivered back to the hidden layer. Each hidden unit adds up the data from output unit and calculate the error.

4. Updating weights and biases: Output units update their respective weights and biases; each hidden unit does the same then checks for a stopping condition. The stopping condition could be number of epochs achieved or minimal error derived.

There are important factors that affect the performance of the back-propagation algorithm, the following parameters are to be chosen appropriately to achieve an effective back propagation network.

1. Initial weights: They affect the output of a feed forward network. This parameter is usually initialized using a small random value, and this value determines the convergence speed of the network. The network becomes dormant in the local minima if initial weight values are very large, when the value is very small then the system becomes slow at learning with the net input to output unit being zero (Hunter *et al*, 2012). Primary weights values are fixed between ranges -0.5 to 0.5 or -1 to 1 for an optimum result.
2. Learning rate: This affects network convergence, higher learning rates improves learning, lower learning rates causes the learning process to be slow. Several back-propagation algorithms use the range of 10^{-3} to 10 for their learning rate.
3. Learning in back propagation: Involves two types, sequential technique and the batch learning technique. The first technique the first input is transmitted and error is calculated and transmitted backwards while simultaneously updating the weights. Batch learning updates weights after the network has been completely trained.

4. Number of training data: Training pairs that are best suited for a network should be decided by the network user before any form of network training is done, because of the importance of this parameter during training of nets. Training data are always set to cover the input space and training pairs are randomly selected from the set (Sheela and Deepa, 2011).
5. Generalization: A generalized state of a network can be achieved when input networks is interpolated with one that exists in the network already. Generalization is often used to solve the issue of over fitting. Back propagation is the most efficient network for generalization as it maintains stability between generalization and memorization.
6. Number of Hidden Layers: The selection of hidden layers and nodes in the hidden layer is very important in neural network. The increase in hidden layer increases the complexity of a network. An n-input, m-output function uses not more than 2n + 1hidden units. In the presence of more than one hidden layer, process in a single layer are repeated in all hidden layers and are summed up at the end.
7. Number of neurons in the hidden layer: This determines the final output generated from hidden layers, it is an important feature as it determines the overall architecture of the network. Hidden layer nodes are always 2/3 of the sum of input and output layer. The number of hidden neurons is also below two times the size of input layer.
8. Momentum Factor: Momentum factors improves the rate of learning. It speeds up the descent and reduces error surface while reducing the impact of local minima of the error surface. It can also increase the convergence of the algorithm.

The basic system architecture is a multilayer feed forward network using back propagation model. The proposed and achieved defined as x_i and s_k for the i^{th} neurons in the input layer with the h^{th} neuron in the output layer, including w_{ih} is the weight that link the input of i^{th} neuron to the output of h^{th} neuron.

$$s_h = f\,[y_i^h] \tag{3}$$

where y_i^h are the activation vectors of any layer j and likely as

$$\frac{\partial E^p}{\partial w_{kj}} = \frac{\partial E^p}{\partial y_j^k}.\frac{\partial y_1^h}{\partial w_{hj}} = \frac{\partial E^p}{\partial y_j^h}.[s_h(y_1^h)] = \frac{\partial E^p}{\partial s_j(y_1^h)}.\frac{\partial s_j(y_j^h)}{\partial y_j^h}[s_h(y^h)] \tag{4}$$

$$\sin ces_j(y_j^h) = \frac{\partial s_j(y_j^h)}{\partial y_j^h} \tag{5}$$

$$\Delta w_{ih} = -\eta \frac{\partial E^p}{\partial s_j(y_j^h)}.s_j(y_j^h).s_k(y_j^h) = -\eta \sum_{j=1}^{J}(d_j - y_j).s_j(y_j^h).s_h(y_j^h) \tag{6}$$

$$\Delta w_{hj}(t+1) = w_{hj}(t) + n\sum_{j=1}^{J}(d_j - y_j).s_j(y_j^h).s_h(y_j^h) \tag{7}$$

$$\Delta w_{ih} = \eta \sum_{j=1}^{N}(d_j - y_j)y_j^p.w_{hj}.s_h(y_h^j)x_i^j \tag{8}$$

The error formulation initiated for a single perceptron are normalized to accommodate all squared errors of the achieved results j = 1,2…J for a particular design p= 1, 2,3…P that is at the input as well as comes across the output error.

$$E^p = -\eta \frac{\partial E^p}{\partial w_{k\cdot j}} \tag{9}$$

Where, η stands for back propagation learning rate

Formation adopted as below:

$$\frac{\partial E^p}{\partial w_{kj}} = \frac{\partial E^p}{\partial y_j^k}.\frac{\partial y_1^h}{\partial w_{hj}} = \frac{\partial E^p}{\partial y_j^h}.[s_h(y_1^h)] = \frac{\partial E^p}{\partial s_j(y_1^h)}.\frac{\partial s_j(y_j^h)}{\partial y_j^h}[s_h(y^h)] \tag{10}$$

Since $\frac{\partial s_j(y_j^h)}{\partial y_j^h} = s_j(y_j^h)$ 11

Therefore, weight adaptations are presently designated as below:

$$\Delta w_{ih} = -\eta \frac{\partial E^p}{\partial s_j(y_j^h)}.s_j(y_j^h).s_k(y_j^h) = -\eta \sum_{j=1}^{J}(d_j - y_j).s_j(y_j^h).s_h(y_j^h) \quad 12$$

The final adjusted weight is given by:

$$\Delta w_{hj}(t+1) = w_{hj}(t) + n\sum_{j=1}^{J}(d_j - y_j)s_j(y_j^h).s_h(y_j^h) \quad 13$$

$$\Delta w_{ih} = \eta \sum_{j=1}^{N}(d_j - y_j)y_j^p.w_{hj}.s_h(y_h^j)x_i^j \quad 14$$

3.1.2 Components of Artificial Neural Network.

1. Network architecture: This involves the arrangement of neurons in respective layers, these layers are input, output and hidden layer. Neurons between layers re connected through weights and sometime are not connected at all. It is important to observe the connection points between neurons. We have different types of architecture used in ANN. Examples are Feed forward network which can be Single or Multi-layered (Bose and Liang, 1996). Recurrent Networks that have at least one feedback loop (Basheer and Hajmeer, 2000).
2. Learning rules: ANN have the ability to learn, which is one of its common reason for being used in many research works. Training involves modifying parameters in the network layers in order to achieve a desired output, using Supervised, Unsupervised and Reinforcement learning methods. For supervised learning the learning process has a guide that serves as a supervisor, its role is to reduce error. An example of supervised learning method is the back propagation neural network; networks used in pattern classification use this method of learning to train (Lee *et al*, 2005). For unsupervised method the learning process

is done without a helping guide, they can also be referred to as self-organizing networks are complex to implement. In reinforcement learning an exact target value is unknown but relative information is given (Simpson, 1990). Using this information increases performance but does not give accurate results.

3. Activation Function: They are functions that accelerate the network's efficiency to achieve optimal results (output). Responses are obtained from weighted inputs being combined with activation functions. We have linear and non-linear types of activation functions. Linear functions are used on single layered networks while non-linear functions are used on multi-layered networks (Sivanandam and Deepa, 2011). Examples of activation functions are Identity function, binary step, Sigmoidal and bipolar sigmoidal function.

3.1.3 Fundamental terms used in Artificial Neural Network

1. Weights: these are parts of input associated with each input unit in a communication link. They are expressed in matrix form and can be calculated using matrix multiplication (Leshno *et al*, 1993). Final input to output unit Y_m given by:

$$Y_m = X_i W_m \quad (15)$$

$$Y_m = \sum_{i=1}^{l} x_i w_{im} \quad (16)$$

Where X (input vector) = ($x_1 \ldots x_i$), W_m (weight matrix of mth column).

2. Bias: This serves as an increasing or decreasing factor when calculating net input. It adds 1 to input unit and acts as a weight. Net input to output neuron is described as;

$$Net = B + \sum_{i=1}^{l} x_i w_i \quad (17)$$

Based on the neural network being considered the bias can be given any particular value or can be adjusted to zero.

3. Threshold: This term is used to calculate final output, and is always the maximum set value. It can be denoted by θ used in activation function. Threshold values are used to calculate output, based on threshold limits (Sivanandam and Deepa, 2011).

   Considering the activation function:

   F(net) == {1 if ne t≥ θ 18

   {-1 if net < θ

   Where θ denotes the fixed threshold value.

3.2 Neural Network Architecture

Neural network architecture is the arrangement of neurons into layers and connections pattern between layers, functions and learning method. (Kalogirou, 2009)

A typical neural network consists of the input layer, the hidden layer, and the output layer. It is usually represented with nodes and arrows.

Input Layer: This is the layer that receivers input from the user or device. The input nodes of the input layer are passive, that is they do not modify data, they receive a single value on their input and it's duplicate before sending it to straight the hidden node. This is called a fully interconnected structure.

The input layer used for the training consist of 48-bit neurons, as shown below. The input is gotten from the MAC addresses, the MAC address which is in hexadecimal is converted into binary number (1 and 0).

Hidden Layer: The hidden layer is responsible for the training of the input; it consists of the weight and biases used in the training of the network. The value gotten from the input is multiplied by weights, a set of pre-programmed number stored on the program. The hidden layer is an active node, that is they can modify data they receive.

The hidden layers used consist of 24-bit neurons which are user-defined, and it's used to generate the weight, which in turn was applied in the testing of the algorithm.

Output Layer: This is responsible for the results for the given input neuron. The output layer collects and transfer information as pre-programmed. The number of neurons in the output layer is usually proportional to the neurons in the input layer.

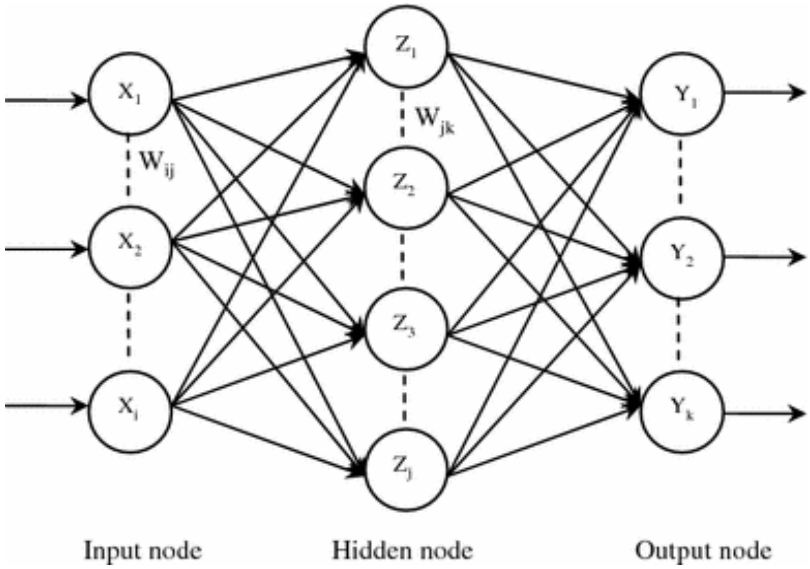


Figure 3.4: A Neural Network Architecture

Where

X_i = Input Neurons

Z_i = Hidden Neurons

Input 1

Input 2

Input 48

Output 1

Output 2

Output 24

Y_i = Output Neurons

3.3 Experimental Setup

3.3.1 Training Phase

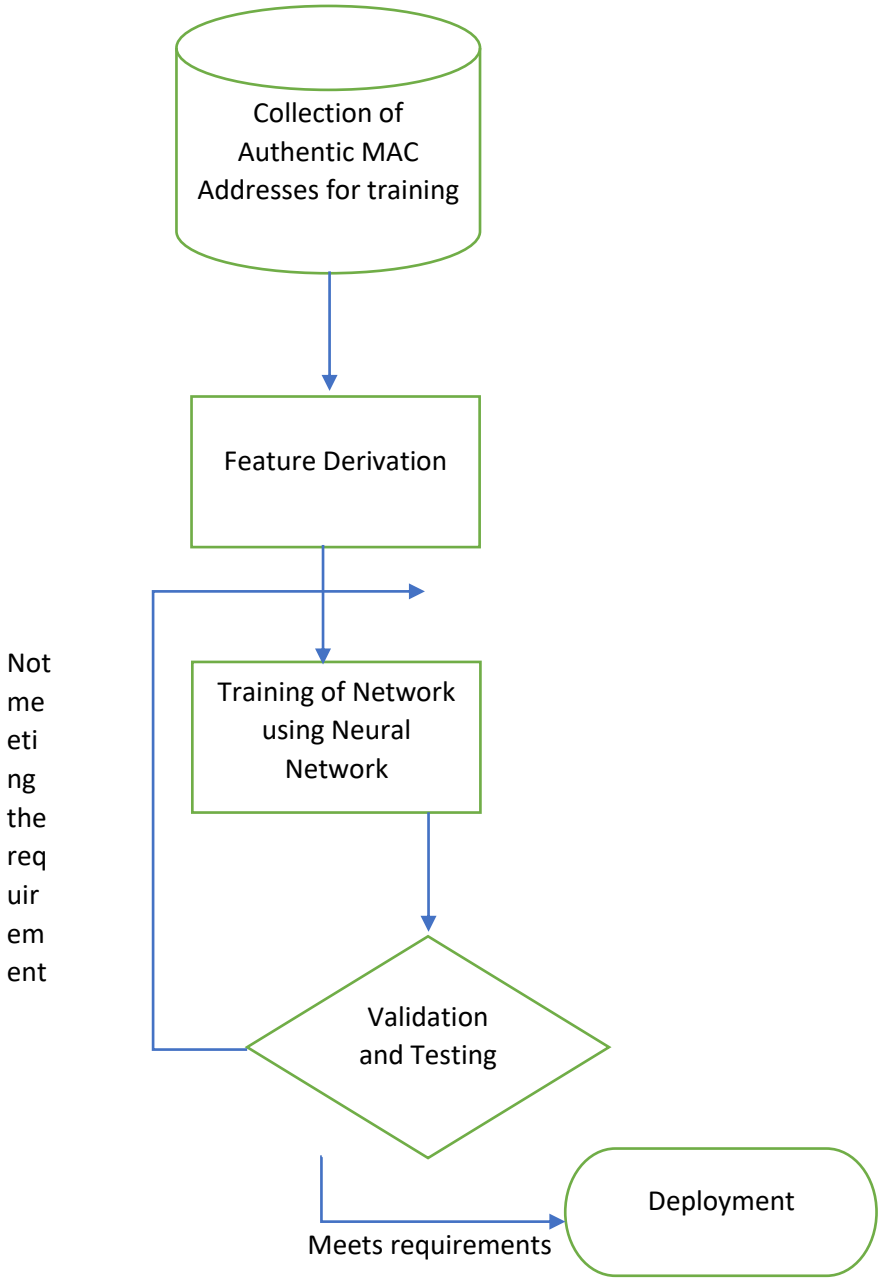
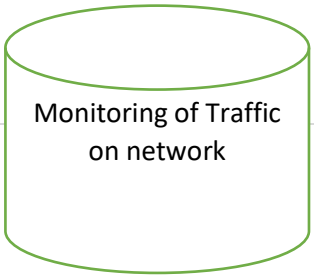


Figure 3.5: A Flowchart explaining the Training Phase

3.3.2 Operation Phase



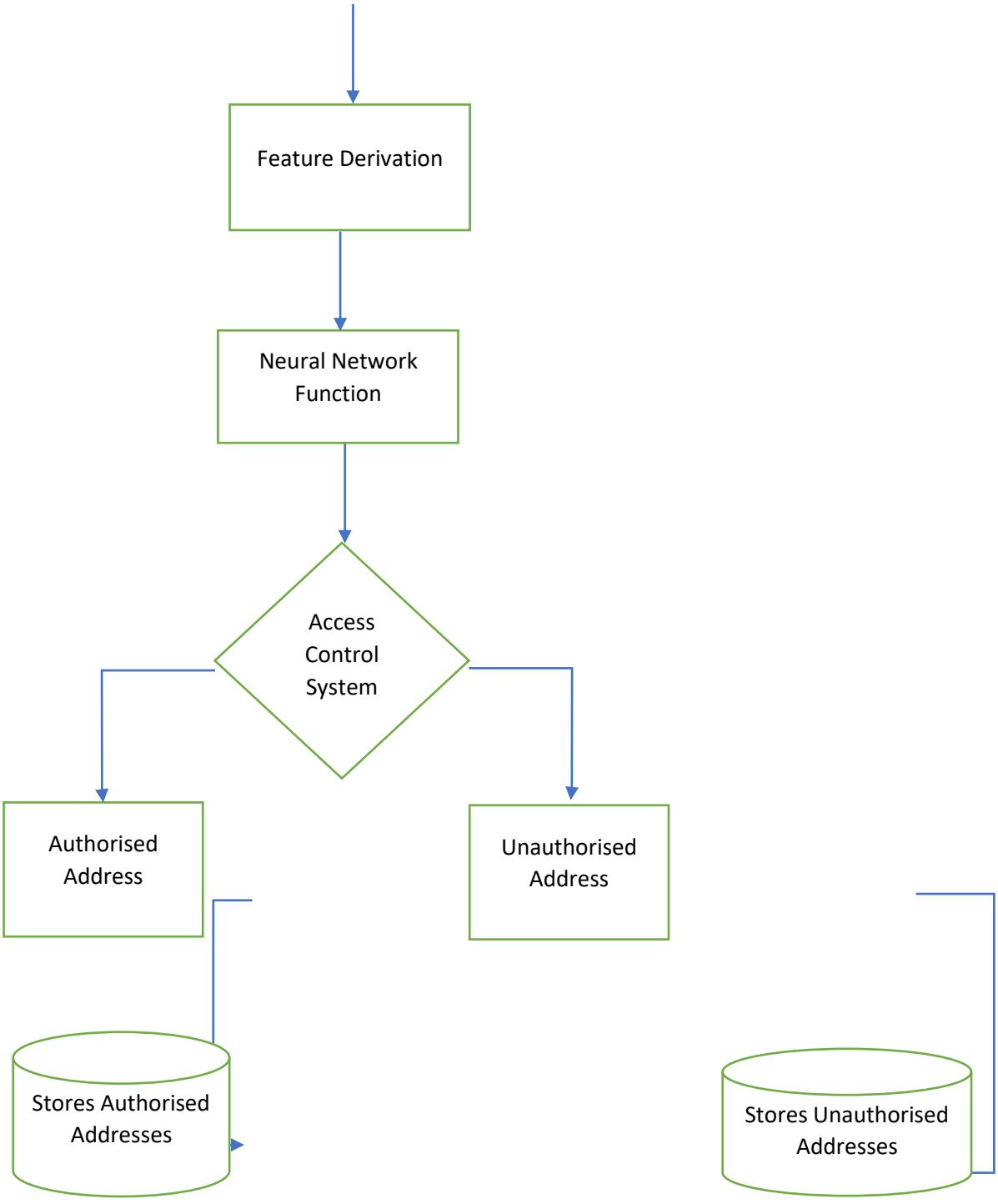


Figure 3.6: A Flowchart explaining the Operation Phase

Training Phase

1. Collecting of MAC addresses
2. Feature Derivation
3. Training of network using neural network:
4. Validation and Testing
5. Deployment

1. Collection of MAC Addresses:

MAC address of a device is a distinct hexadecimal number given to a Network Interface Controller (NIC). Due to the uniqueness of the MAC address, it makes easy to use to as a criterion for training. The MAC addresses used for this experiment were collected from device around so as to ensure the integrity of the MAC addresses.

2. Feature Derivation:

The MAC address is converted to binary and this in turn is used as the input for the input layer. The binary gives a 48-bit input which moves to the hidden layer for training. The neural network weight and bias is calculated in the hidden layer and is used as the feature for comparison. The weight of a neural network is the coefficient of the equation being resolved. The neural network bias is a constant value that is added to the product of inputs and weights.

3. Training of Network Using Neural Network:

There are different methods for training a neural network used for the training the neural network such as Batch training, Gradient descent training back propagation etc. The training used for this experiment are as follows:

1. **Quasi-Newton Backpropagation (trainbfg):**

Newton's method is a substitute for rapid optimization of the conjugate gradient me thods. The fundamental step of Newton's method is:

$$\mathbf{x}_{k+1}=\mathbf{x}_k-\mathbf{A}_{-1k}\mathbf{g}_k \quad (19)$$

where $\mathbf{A}_{-1k}$ is the Hessian matrix (2nd derivatives) of the performance index at the current values of the weights and biases. The method of Newton often converges faster than the methods of conjugating gradients. Sadly, computing the Hessian matrix for feed-forward neural networks is complicated and expensive. There is a class of algorithms based on Newton's model, but that does not allow second derivatives to be measured. These are called methods of quasi-Newton (or secant). At each iteration of the algorithm, they update an approximate Hessian matrix. The update is calculated as a gradient function. The Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update is the quasi-Newton method most successful in published studies. This algorithm is used in the routine of trainbfg. trainbfg can train any network as long as its weight, net input, and transfer functions have derivative functions.

Back propagation is used to calculate derivatives of performance (perf) with respect to the weight and bias variables X. Each variable is adjusted according to the following:

$$X = x + a * dX \quad (20)$$

where derivative of X (dX) is the search direction. The parameter *a* is selected to minimize the performance along the search direction. The line search function (searchFcn) is used to locate the minimum point. The first search direction

is the negative of the gradient of performance. In succeeding iterations, the search direction is computed according to the following formula:

dX = -H\gX;

where *gX* is the gradient and H is an approximate Hessian matrix.

Training terminates when any of these conditions are encountered:

- The maximum number of epochs (repetitions) is attained.
- The maximum amount of time is exceeded.
- Performance is minimized to the goal.
- The performance gradient falls below min_grad.
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

2. Levenberg-Marquaedt back propagation (trainlm): The Levenberg-Marquardt algorithm, like the quasi-Newton methods, was designed to solve second-order training speed without calculating the Hessian matrix. If the performance function has the form of a sum of squares (as is typical in feedforward networks training), then the Hessian matrix can be approximated as:

$$\mathbf{H} = \mathbf{J}^T\mathbf{J} \qquad 22$$

and the gradient can be computed as

$$\mathbf{g} = \mathbf{J}^T\mathbf{e} \qquad 23$$

where **J** is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases, and **e** is a vector of network errors. The Jacobian matrix can be computed through a standard back propagation

The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix in the following Newton-like update:

$$\mathbf{x}_{k+1}=\mathbf{x}_k-[\mathbf{J}^T\mathbf{J}+\mu\mathbf{I}]^{-1}\mathbf{J}^T\mathbf{e} \quad 24$$

Using the approximate Hessian matrix, this is only Newton's approach when the scalar μ is zero (0). Once μ is through, a small step size becomes a gradient descent. Newton's method is quicker and more reliable to a minimum of error, so the goal is to move as fast as possible to Newton's method. Therefore, after each successful step (reduction in output function) μ is increasing and only increased when a tentative step increases the performance function. Thus, at each iteration of the algorithm, the performance function is always reduced. trainlm can train any network as long as its weight, net input, and transfer functions have derivative functions.

Back propagation is used to calculate the Jacobian jX of performance perf with respect to the weight and bias variables X. Each variable is adjusted according to Levenberg-Marquardt,

$$jj = jX * jX \quad 25$$

$$je = jX * E \quad 26$$

$$dX = -(jj+I*mu) \setminus je \quad 27$$

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

Training terminates when any of these conditions are encountered:

- The maximum number of epochs (repetitions) is attained.
- The maximum amount of time is surpassed.
- Performance is minimized to the goal.

- The performance gradient falls below min_grad.
- mu exceeds mu_max.
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

3. Scaled Conjugate Gradient Back propagation (trainscg): Trainscg can train any network as long as it has derivative functions for weight, net input and transfer functions. Back propagation is used to measure output (perf) derivatives with respect to X variables of weight and bias. Training terminates when any of these conditions occurs:

- The maximum number of epochs (repetitions) is achieved.
- The maximum amount of time is surpassed.
- Performance is minimized to the goal.
- The performance gradient falls below min_grad.
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

4. Gradient Descent with Momentum Back propagation (traingdm): Gradient downward with momentum, implemented by traingdm, allows a network not only to adapt to the local gradient, but also to recent trends in the surface of errors. Acting as a low-pass filter, momentum helps the network to ignore small features in the surface of the error. A network can be stuck in a shallow local minimum without momentum**.** Gradient descent with momentum depends on two (2) training parameters. The parameter *lr* indicates the learning rate, similar to the simple

gradient descent. The parameter *mc* is the momentum constant that defines the amount of momentum. mc is set between 0 (no momentum) and values close to 1 (lots of momentum). A momentum constant of 1 result in a network that is completely insensitive to the local gradient and, therefore, does not learn properly.)

Training terminates when any of these conditions occurs:

- The maximum number of epochs (repetitions) is attained.
- The maximum amount of time is surpassed.
- Performance is minimized to the goal.
- The performance gradient falls below min_grad.
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

5. **Resilient Back propagation (trainrp)**: Multilayer networks typically use the hidden layers of sigmoid transfer functions. These functions are often called functions of "squashing," as they compress an infinite range of input into a finite range of output. Sigmoid functions are distinguished by the fact that their slopes must be close to zero (0) when the input is large. It causes a problem when you use the steepest descent to train a multilayer network with sigmoid functions, as the gradient can have a very low magnitude and thus cause small weight and bias shifts, although the weights and biases are far from their ideal values.

Training terminates when any of these conditions met:

- The maximum number of epochs (repetitions) is attained.
- The maximum amount of time is surpassed.

- Performance is minimized to the goal.
- The performance gradient falls below min_grad.
- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

6. Conjugate Gradient Backpropagation with Polak-Ribiere Updates (traincgf): Polak and Ribiére have suggested another variant of the conjugate gradient algorithm. As with the traincgf Fletcher-Reeves algorithm, the search route is decided by each iteration.

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1} \qquad 28$$

For the Polak-Ribiére update, the constant β_k is computed by

$$\beta_k = \frac{\Delta \mathbf{g}^{T_{k-1}} \mathbf{g}_k}{\mathbf{g}^{T_{k-1}} \mathbf{g}_{k-1}} \qquad 29$$

This is the inner product of the previous change in the gradient with the current gradient divided by the norm squared of the previous gradient.

Traincgp routine has traincgf-like performance. It's hard to predict which algorithm will best perform on a particular problem. Polak-Ribiére storage requirements (four vectors) are slightly higher than Fletcher-Reeves (three vectors).Training terminates when any of these conditions occurs:

- The maximum number of epochs (repetitions) is attained.
- The maximum amount of time is surpassed.
- Performance is minimized to the goal.
- The performance gradient falls below min_grad.

- Validation performance has increased more than max_fail times since the last time it decreased (when using validation).

4. Validation and Testing:

Twenty (20) MAC addressees were used during the course of the experiment, sixty percent (60 %) of the MAC addresses were used for training, twenty percent (20%) used for validation and the remaining MAC addresses were used for testing. The results were discussed in the next chapter.

5. Deployment:

After undergoing the training and validation stage and a successful testing stage, the process moves to the deployment stage where functions are written in a script to make the training useful. At the stage the process will undergo a feed-forward propagation.

Operation Phase

1. Monitoring traffic on Network:

This monitors all traffic trying to gain access to the network. This contains all both the IP address and MAC address of the device on the network

2. Feature derivation:

MAC addresses trying to gain access into the network are monitored and collected, then sent to the next stage where it undergoes another process of verification. After the MAC address has be extracted, it is sent to the next stage.

3. Neural Network Function:

This responsible for using the training technique chosen to analyse the MAC address and decides if it's within the limit of tolerance.

4. Access Control System:

This is responsible for granting permission to MAC addresses that passes the criteria for admission into the network and denying permission to the addresses that don't. It also saves the MAC addresses both the admitted and rejected.

CHAPTER 4

RESULTS AND DISCUSSION

In this chapter, results obtained using the procedures discussed in the previous chapter are analysed extensively. Some important performance criteria that were made reference to are: Mean Square Error which is the average squared difference between our estimated values and actual value. MSE can be described as a measure of quality an estimator gives, it is always a non-negative value and values closer to zero are more reliable. Another criterion is the number of epochs which describes the number of passes completed when training through the given dataset. Our feedforward neural network uses iterative algorithms that require as many epochs during training or learning process. Time of training represents the total amount of time a feedforward neural network training algorithm needs to complete one training process.

4.1 Selection of Optimal Training Algorithm

There is no specific formula for choosing a training algorithm. There are various factors to consider in choosing an optimal algorithm such as; complexity of the problem, the samples available, accuracy, computation time and resources, and whether the network is used for regression analysis or discriminant analysis that estimates dependent and independent variable relationships. These various properties associated with these training algorithms determines if the training function used is fast, whether it can be used for really complex problems and whether it requires abundant memory. This experiment was performed using six back propagation training algorithms. The six training functions considered are:

- Levenberg-Marquardt (LM)
- Scaled Conjugate Gradient (SCG)
- Gradient Descent with Momentum (GDM)
- Conjugate Gradient Back propagation with Polak-Riebre Updates (CGP)
- Resilient Back propagation (RP)
- Quasi-Newton (BFGS).

4.1.1 Figures of Result

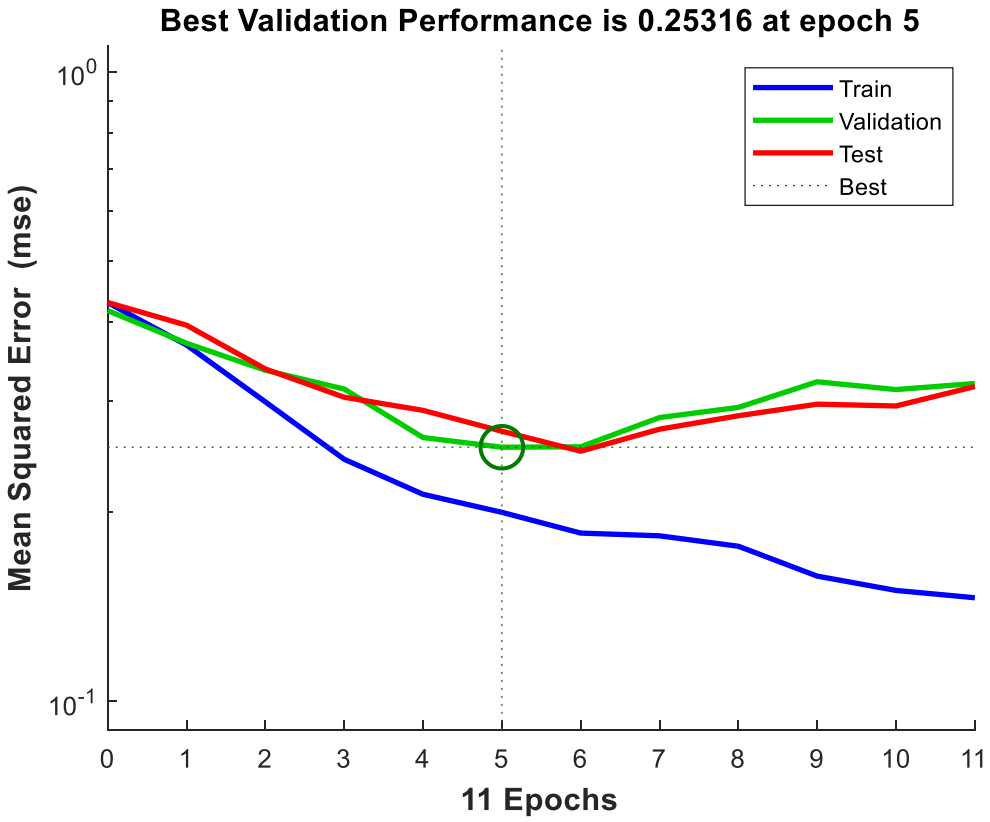


Figure 4.1: Performance Plot of Network with Training Function Trainlm

Figure 4.1 displays the performance plot of the network using the Levenberg-Marquardt (LM) training function. Training was completed after 11 Epochs. Mean Square Error (MSE) of 0.2240 was generated.

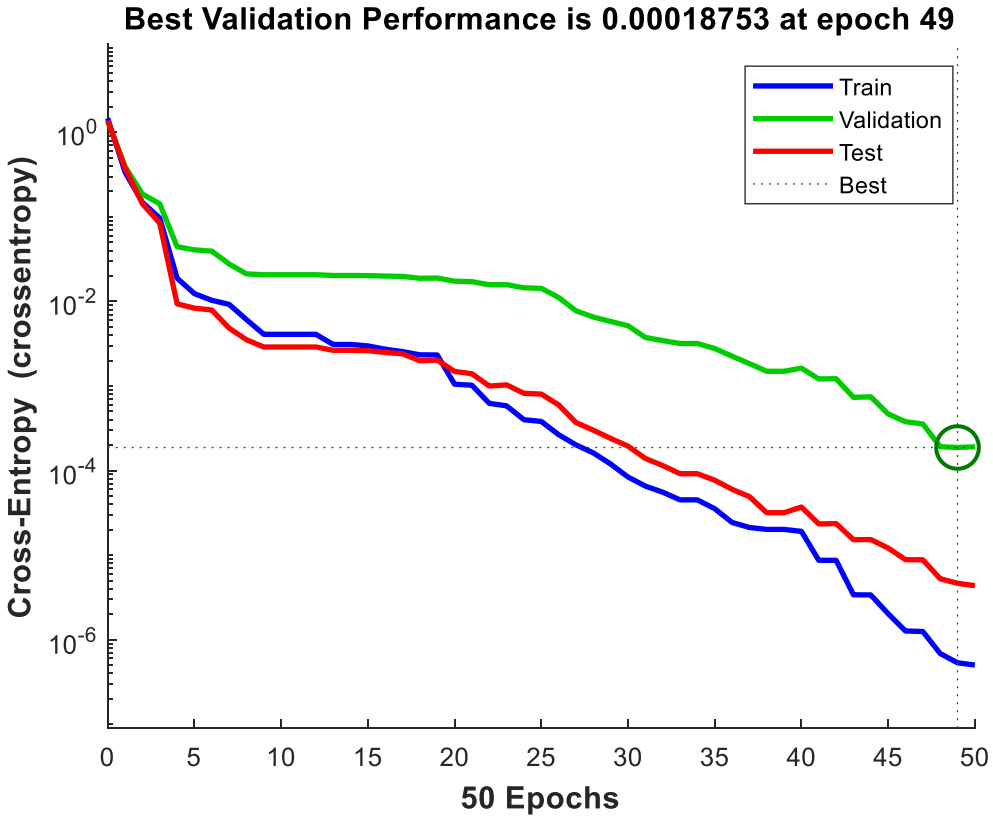


Figure 4.2: Performance Plot of Network with Training Function Trainscg

Figure 4.2 displays the performance plot of the network using the Scaled Conjugate Gradient (SCG) training function. The training was completed after 50 Epochs, a Mean Square Error (MSE) of 0.4988 was generated.

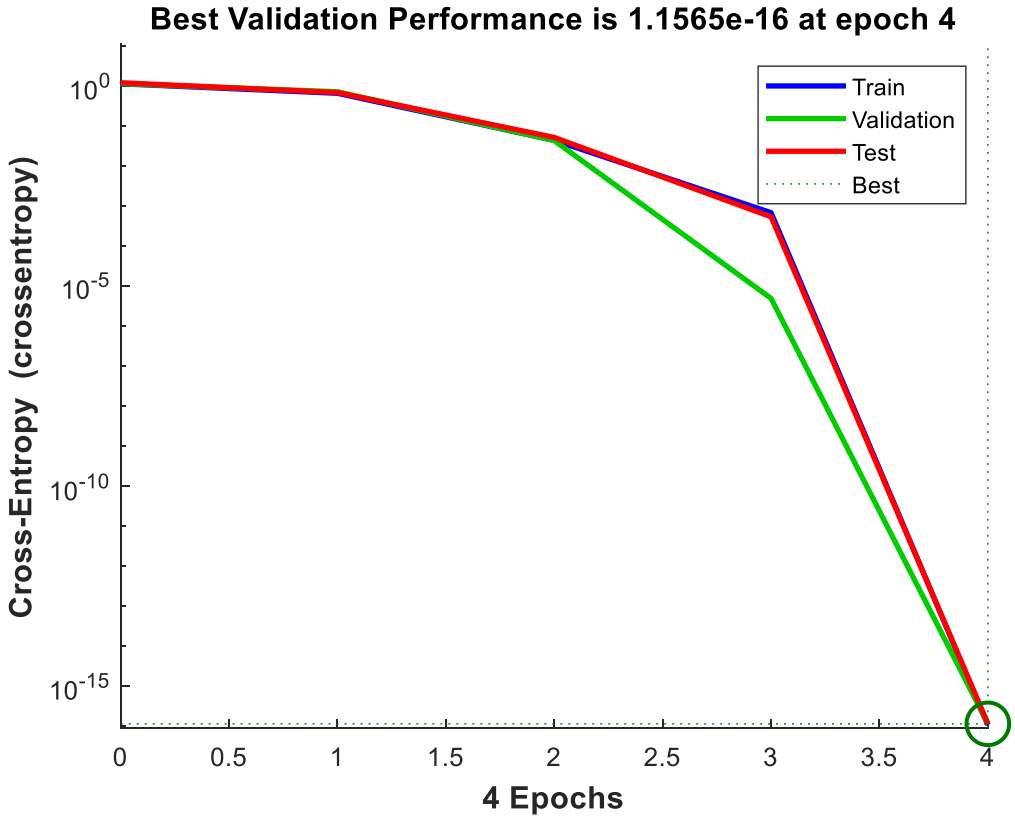


Figure 4.3: Performance Plot of Network with Training Function Trainbfg

Figure 4.3 displays the performance plot of the network using the Quasi-Newton (BFGS) training function. Training was completed after 4 epochs, a Mean Square Error (MSE) of 0.4996 was generated.

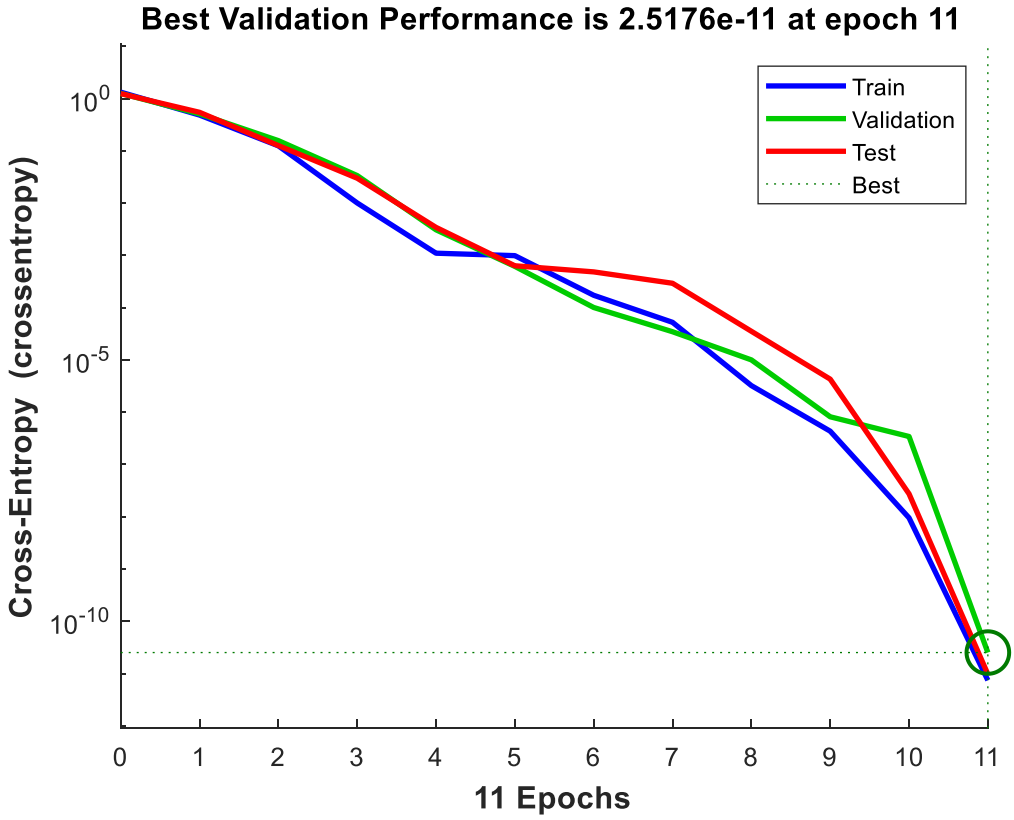


Figure 4.4: Performance Plot of Network with Training Function Traincgp

Figure 4.4 displays the performance plot of the network using the Conjugate Gradient Back propagation with Polak-Riebre Updates (CGP) training function. Training was completed after 11 epochs, a Mean Square Error (MSE) of 0.4996 was generated.

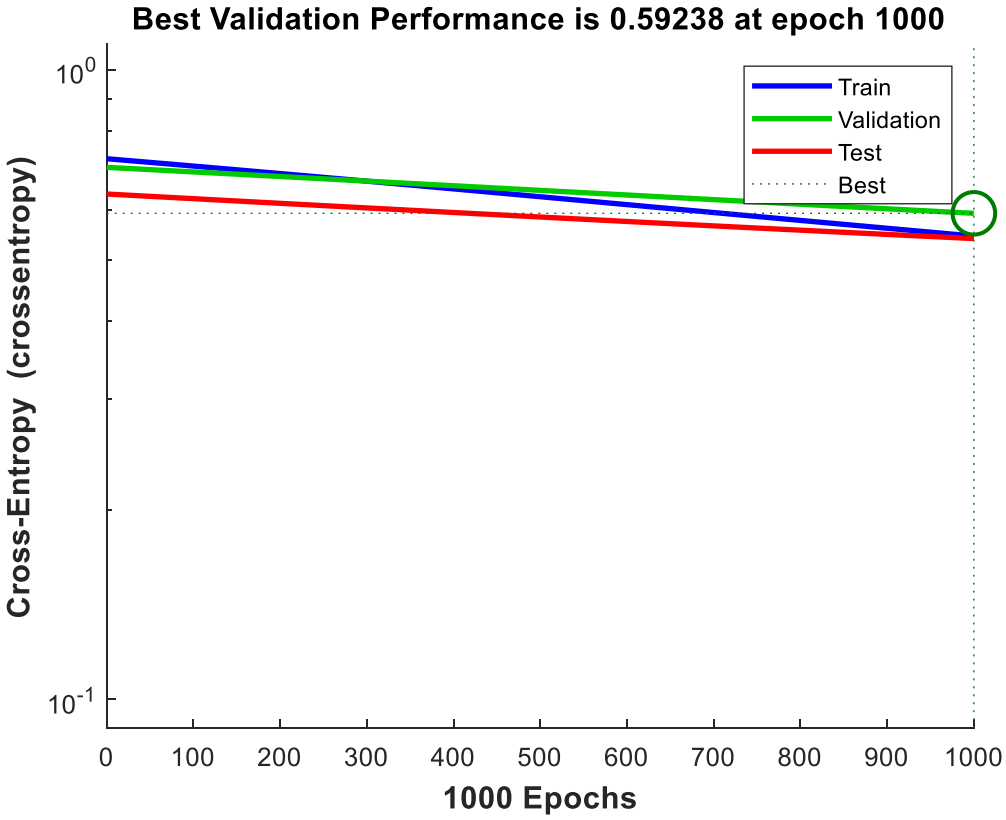


Figure 4.5: Performance Plot of Network with Training Function Traingdm

Figure 4.5 displays the performance plot of the network using the Gradient Descent with Momentum (GDM) training function. Training was completed after 1000 Epochs, a Mean Square Error (MSE) of 0.4032 was generated.

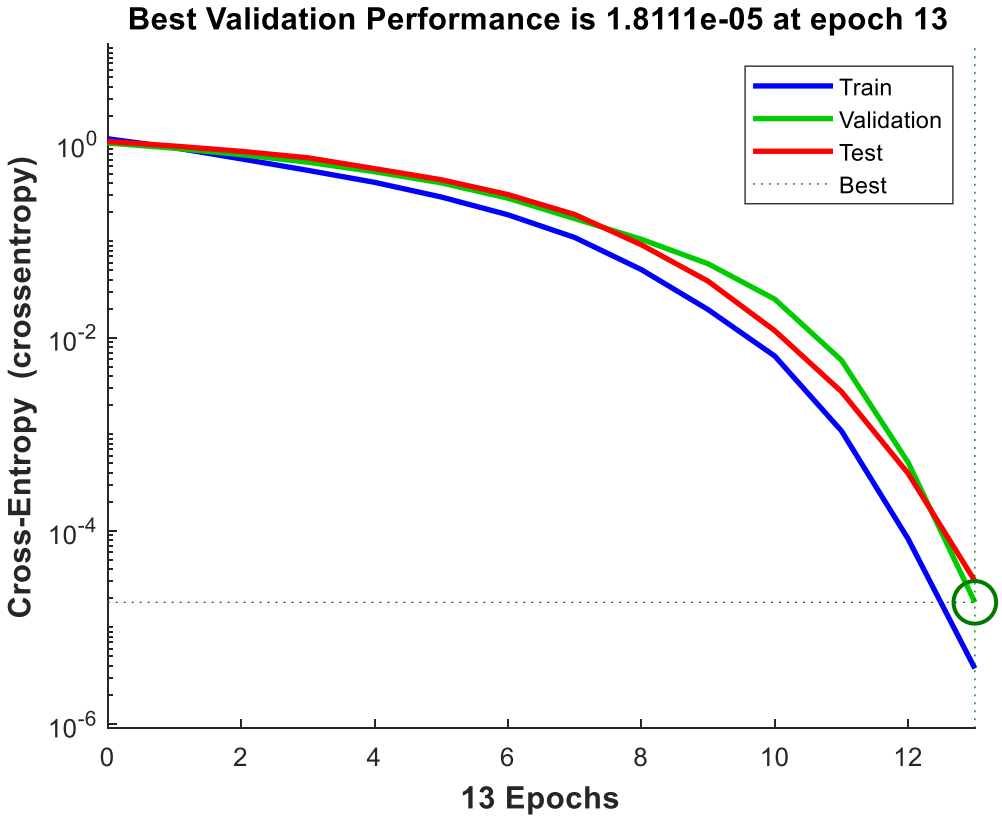


Figure 4.6: Performance Plot of Network with Training Function Trainrp

Figure 4.6 displays the performance plot of the network using the Resilient Back propagation (RP) training function. Training was completed in after 13 epochs, a Mean Square Error (MSE) of 0.4993 was generated.

Figure 4.1 to 4.6, shows the performance plot of using the training functions mentioned earlier, from figure 4.1 it can be seen that Levenberg-Marquardt (LM) gives the better optimization result. The Levenberg-Marquardt (LM) was used as our optimal training function because of its speed, and it does not require large computing and processing memory.

4.2 Determination of Optimum Network Architecture

After determining the optimal training function needed for our network, modifications are made to the number of neurons in the hidden layer of our network architecture, to

compare and analyse the optimal architecture needed to generate efficient results. In this section we will determine the optimal number of neurons required in the hidden layer to attain an optimal network for training. Different architecture was considered with varying number of neurons, also it is known that an increase in the number of hidden neurons is likely to cause overfitting (MathWorks Inc, 2018).

- {48-5-24} which is 48 neurons for inputs, 5 hidden neurons and 24 output neurons.
- {48-8-24} which is 48 neurons for inputs, 8 hidden neurons and 24 output neurons.
- {48- 10-24} which is 48 neurons for inputs, 10 hidden neurons and 24 output neurons.
- {48-15-24} which is 48 neurons for inputs, 15 hidden neurons and 24 output neurons.
- {48-20-24} which is 48 neurons for inputs, 20 hidden neurons and 24 output neurons.
- {48-25-24} which is 48 neurons for inputs, 25 hidden neurons and 24 output neurons.
- {48-30-24} which is 48 neurons for inputs, 30 hidden neurons and 24 output neurons.
- {48-35-24} which is 48 neurons for inputs, 35 hidden neurons and 24 output neurons.
- {48-40-24} which is 48 neurons for inputs, 40 hidden neurons and 24 output neurons.
- {48-45-24} which is 48 neurons for inputs, 45 hidden neurons and 24 output neurons.

- {48-50-24} which is 48 neurons for inputs, 50 hidden neurons and 24 output neurons.
- {48-55-24} which is 48 neurons for inputs, 55 hidden neurons and 24 output neurons.
- {48-60-24} which is 48 neurons for inputs, 60 hidden neurons and 24 output neurons.

Figure 4.7 to figure 4.19 shows the performance with respect to number of neurons as shown in table 4.1. Each one of these graphs has a train curve which decreases monotonically, and a test curve which has no effect on training and so provides an independent measure of network performance during and after training, validation curve measures network generalization, and stops the training when generalization stops improving. This is the case when considering figure 4.7 to figure 4.19.

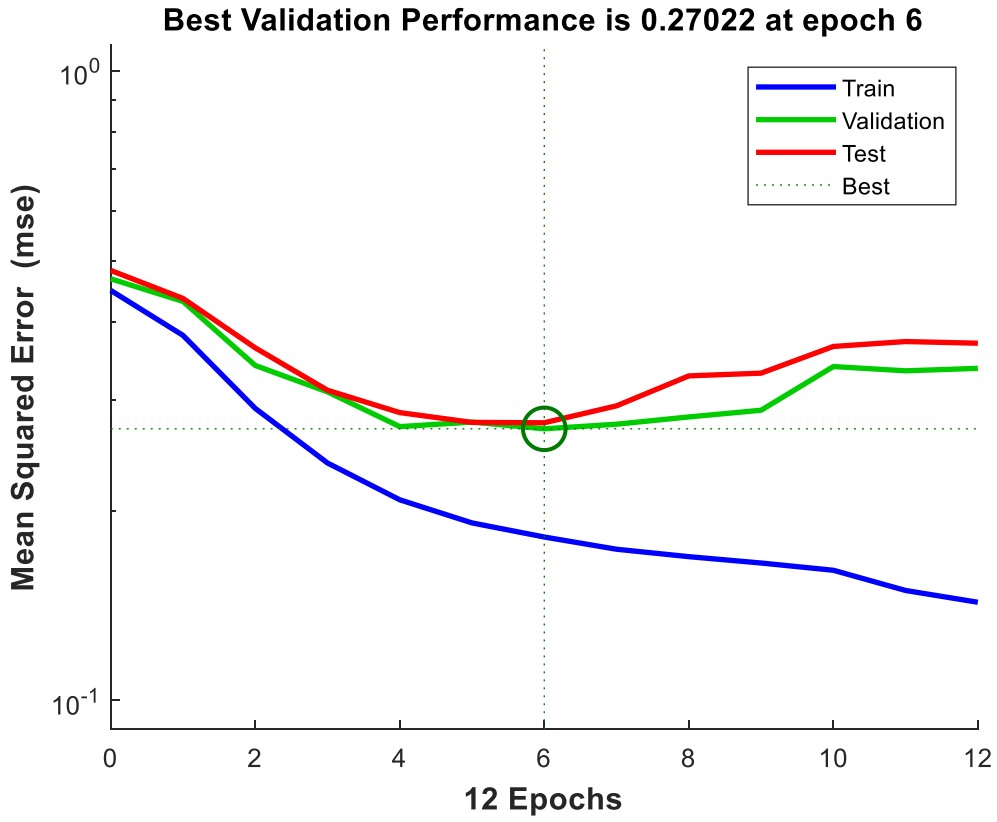


Figure 4.7: Performance Plot of {48-5-24} Network with Training Function Trainlm.

Figure 4.7 displays the performance plot of the network using the Levenberg-Marquardt (LM) training function and 5 hidden layer neurons. The training was completed in 1 second at 12 Epochs, validation curve shows a validation performance value of 0.2702 at 6 Epochs. Mean Square Error (MSE) of 0.2184 was generated.

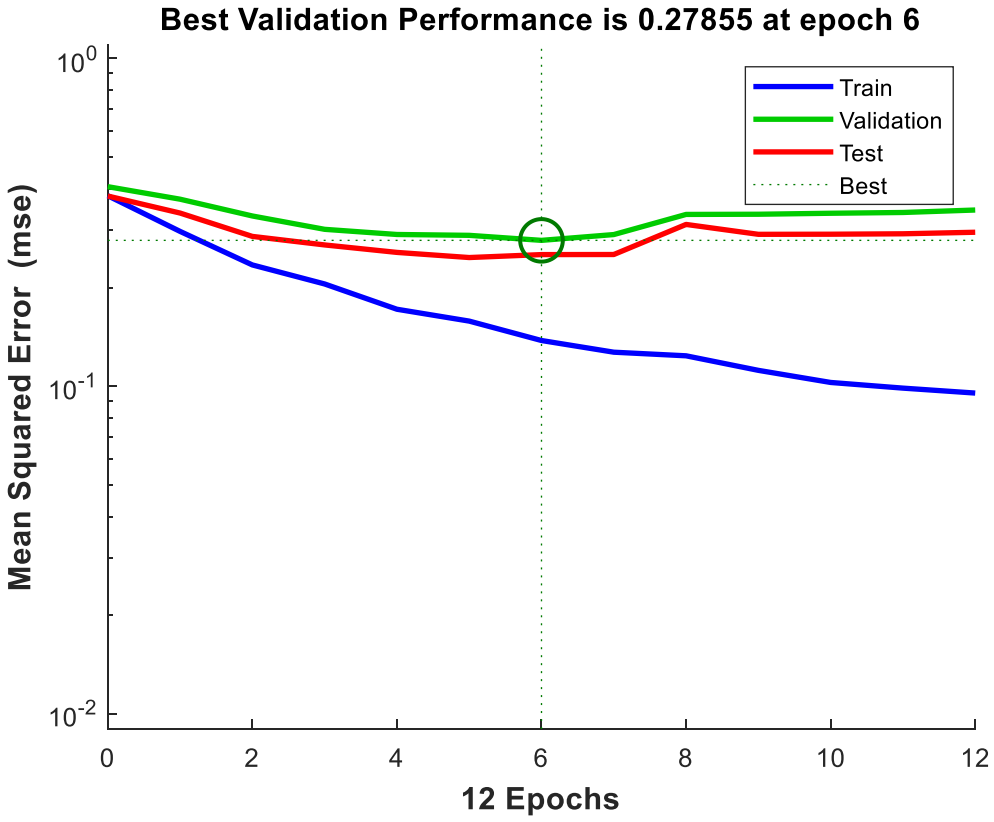


Figure 4.8: Performance Plot of {48-8-24} Network with Training Function Trainlm.

Figure 4.8 displays the performance plot of the network using the Levenberg-Marquardt (LM) training function and 8 hidden layer neurons. The training was completed in 2 seconds at 12 Epochs, validation curve shows a validation performance value of 0.27855 at 6 Epochs. Mean Square Error (MSE) of 0.1888 was generated.

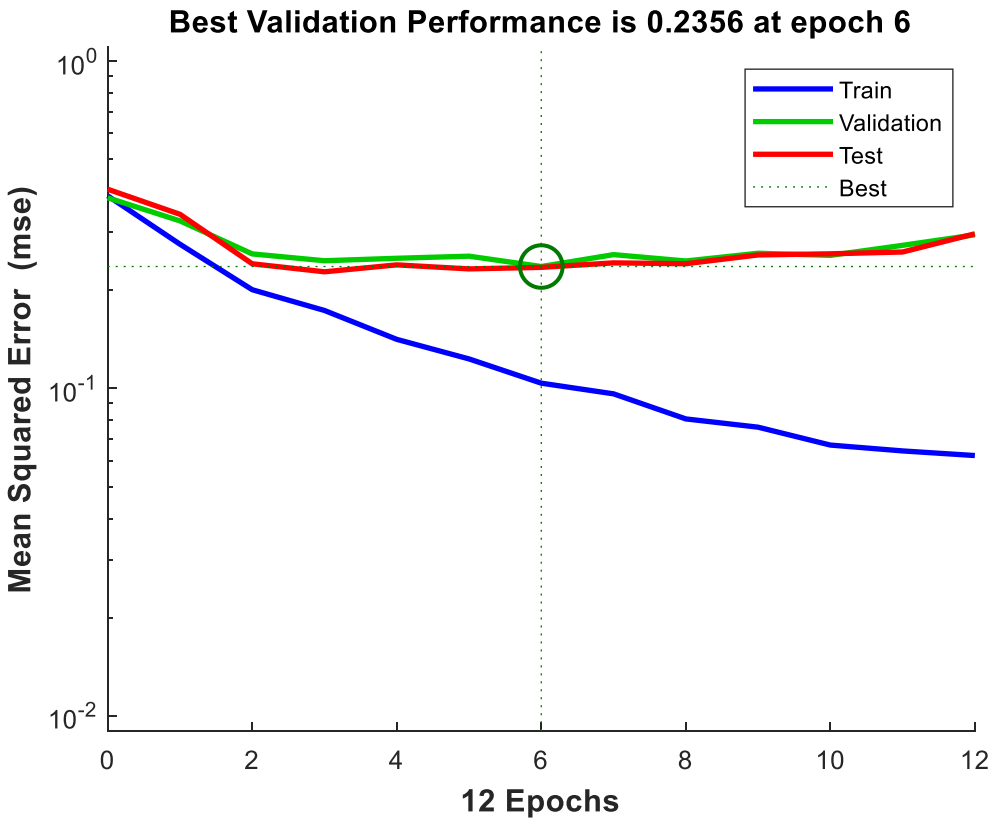


Figure 4.9: Performance Plot of {48-10-24} Network with Training Function Trainlm.

Figure 4.9 displays the performance plot of the network using optimal training function and 10 hidden layer neurons. Training was completed in 2 seconds at 12 Epochs, validation curve shows a validation performance value of 0.2356 at 6 Epochs. Mean Square Error (MSE) of 0.1562 was generated.

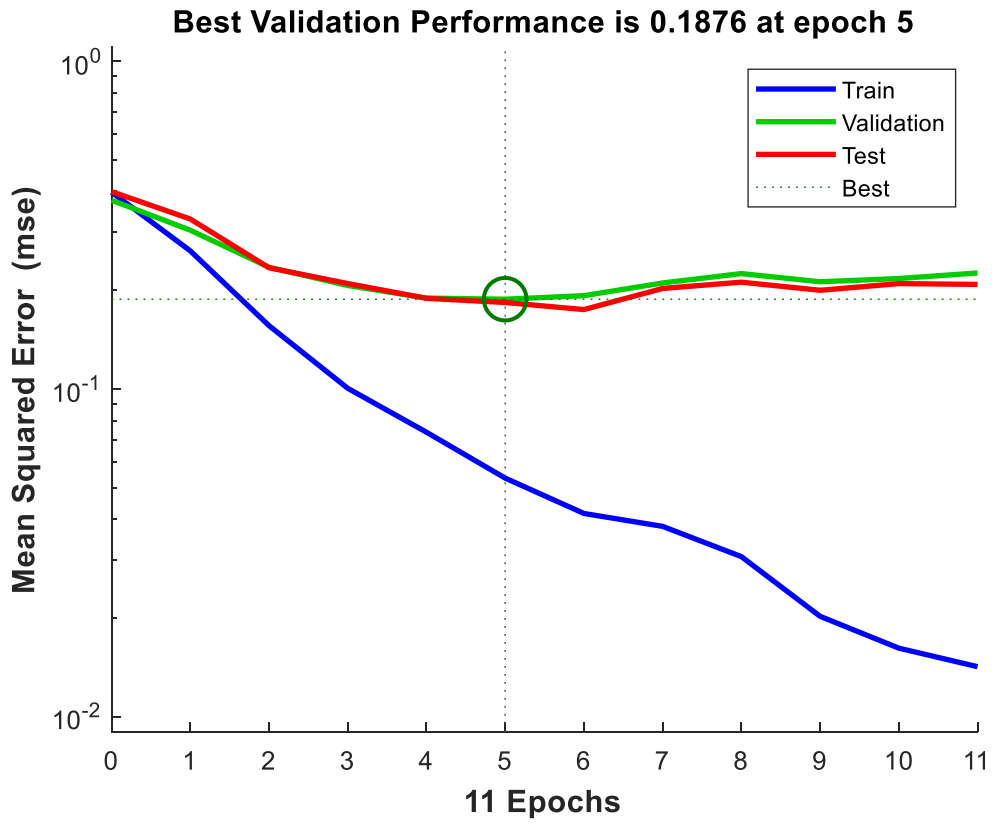


Figure 4.10: Performance Plot of {48-15-24} Network with Training Function Trainlm

Figure 4.10 displays the performance plot of the network using optimal training function and 15 hidden layer neurons. Training was completed in 8 seconds at 11 Epochs, validation curve shows a validation performance value of 0.1875 at 5 Epochs. Mean Square Error (MSE) of 0.1062 was generated.

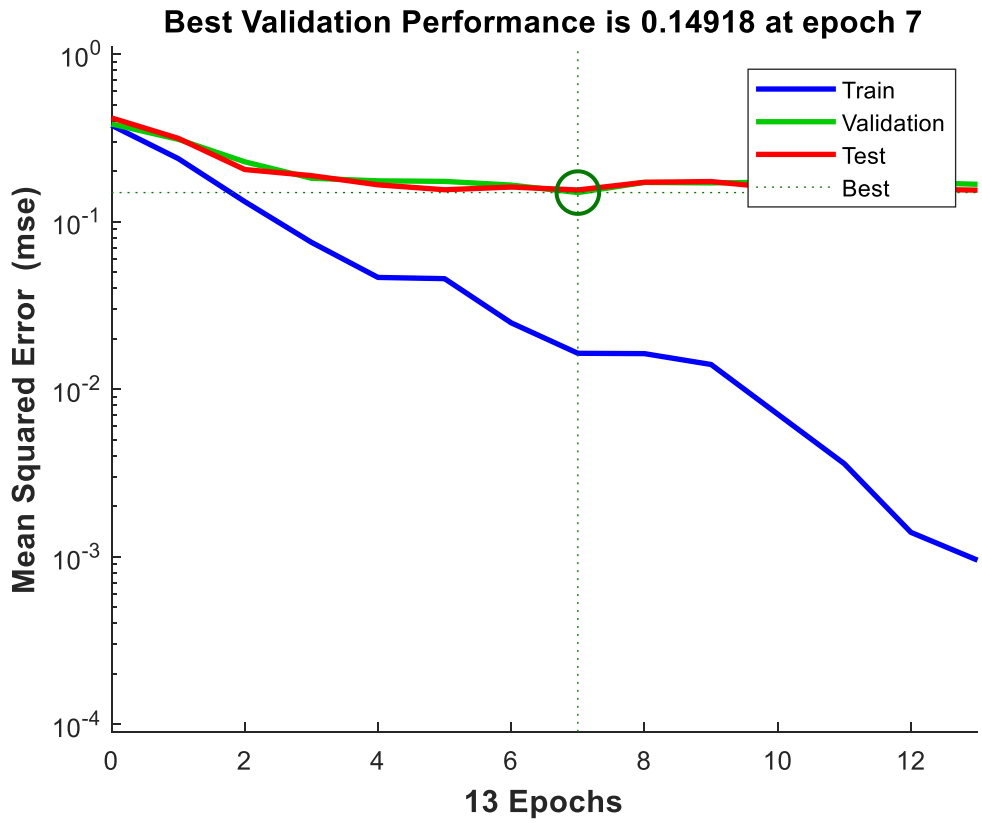


Figure 4.11: Performance Plot of {48-20-24} Network with Training Function Trainlm

Figure 4.11 displays the performance plot of the network using optimal training function and 20 hidden layer neurons. Training was completed in 10 seconds at 13 Epochs, validation curve shows a validation performance value of 0.14918 at 7 Epochs. Mean Square Error (MSE) of 0.0706 was generated.

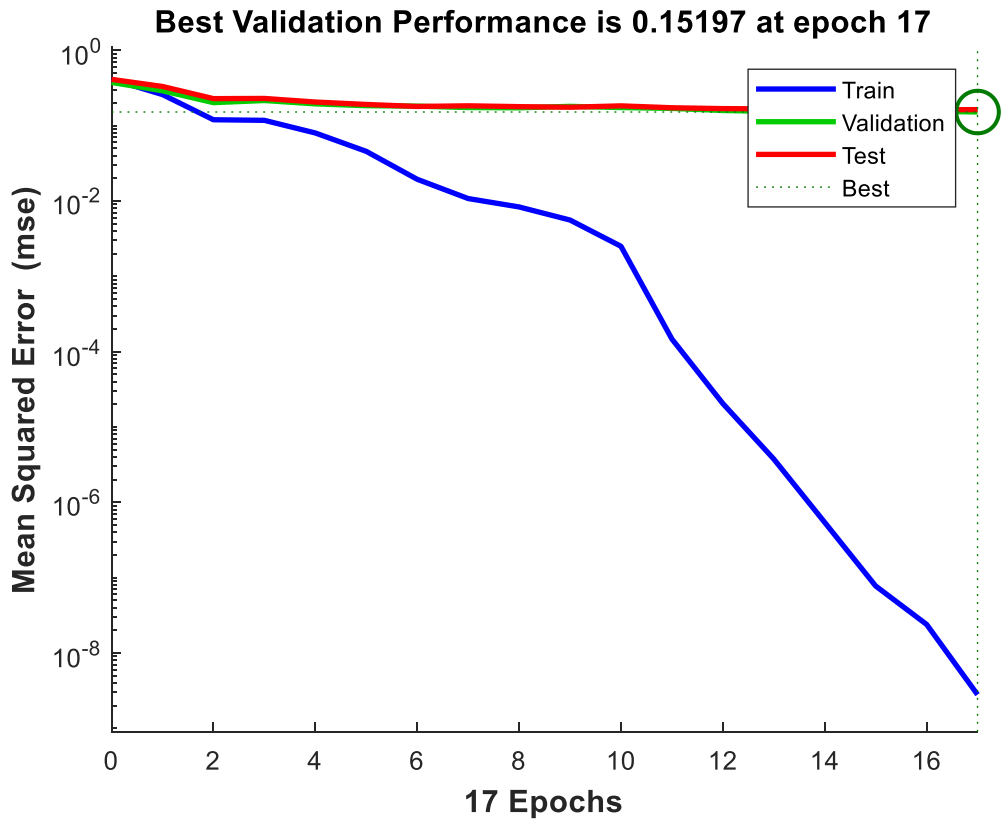


Figure 4.12: Performance Plot of {48-25-24} Network with Training Function Trainlm.

Figure 4.12 displays the performance plot of the network using optimal training function and 25 hidden layer neurons. Training was completed in 21 seconds at 17 Epochs, validation curve shows a validation performance value of 0.15197 at 17 Epochs. Mean Square Error (MSE) of 0.0631 was generated.

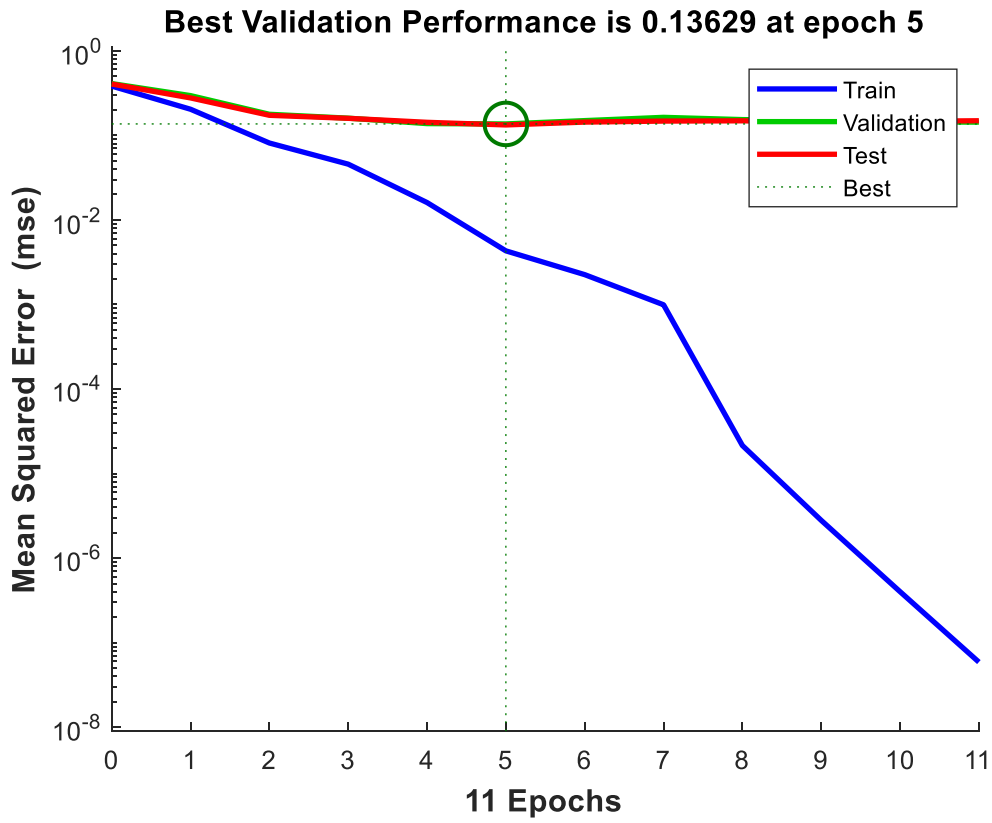


Figure 4.13: Performance Plot of {48-30-24} Network with Training Function Trainlm.

Figure 4.13 displays the performance plot of the network using optimal training function and 30 hidden layer neurons. Training was completed in 17 seconds at 11 Epochs, validation curve shows a validation performance value of 0.1363 at 5 Epochs. Mean Square Error (MSE) of 0.0563 was generated.

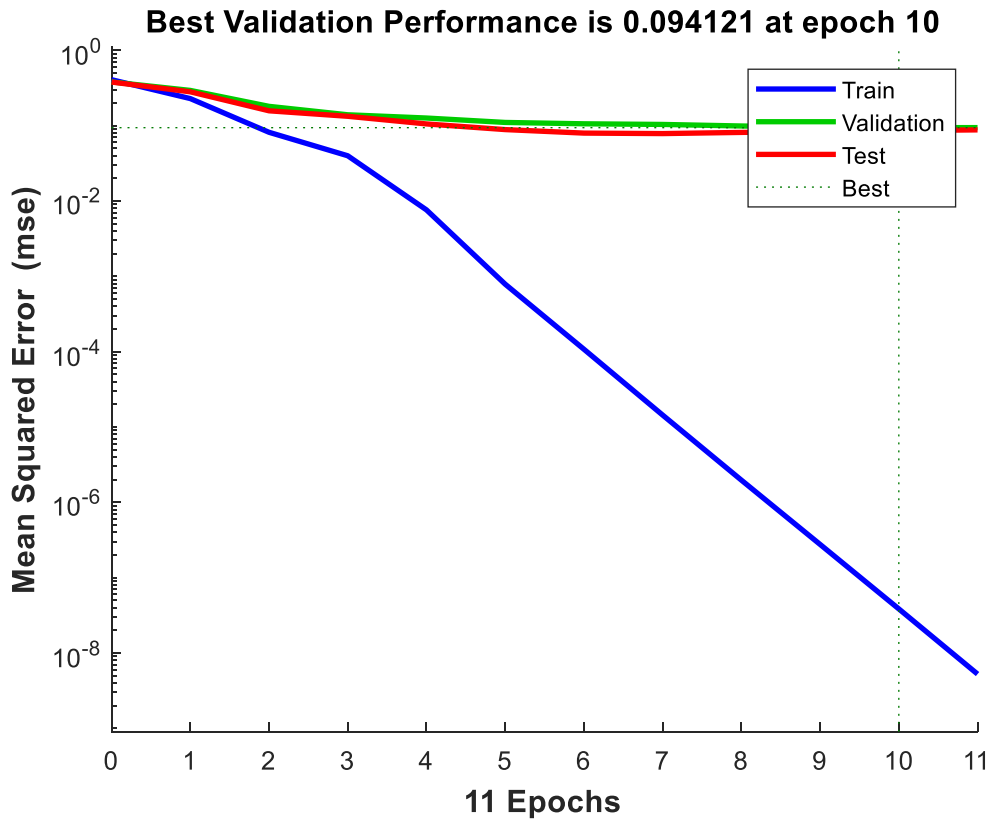


Figure 4.14: Performance Plot of {48-35-24} Network with Training Function Trainlm.

Figure 4.14 displays the performance plot of the network using optimal training function and 35 hidden layer neurons. Training was completed in 25 seconds at 11 Epochs, validation curve shows a validation performance value of 0.094121 at 10 Epochs. Mean Square Error (MSE) of 0.0360 was generated.

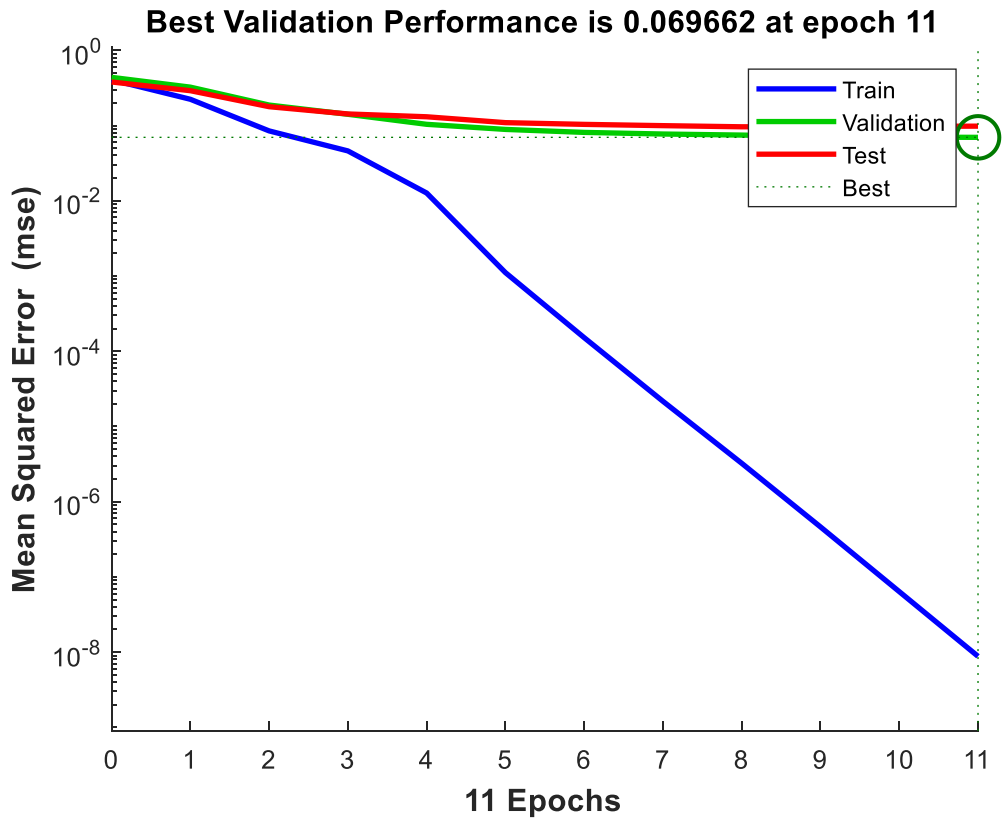


Figure 4.15: Performance Plot of {48-40-24} Network with Training Function Trainlm.

Figure 4.15 displays the performance plot of the network using optimal training function and 40 hidden layer neurons. Training was completed in 34 seconds at 11 Epochs, validation curve shows a validation performance value of 0.069662 at 11 Epochs. Mean Square Error (MSE) of 0.0336 was generated.

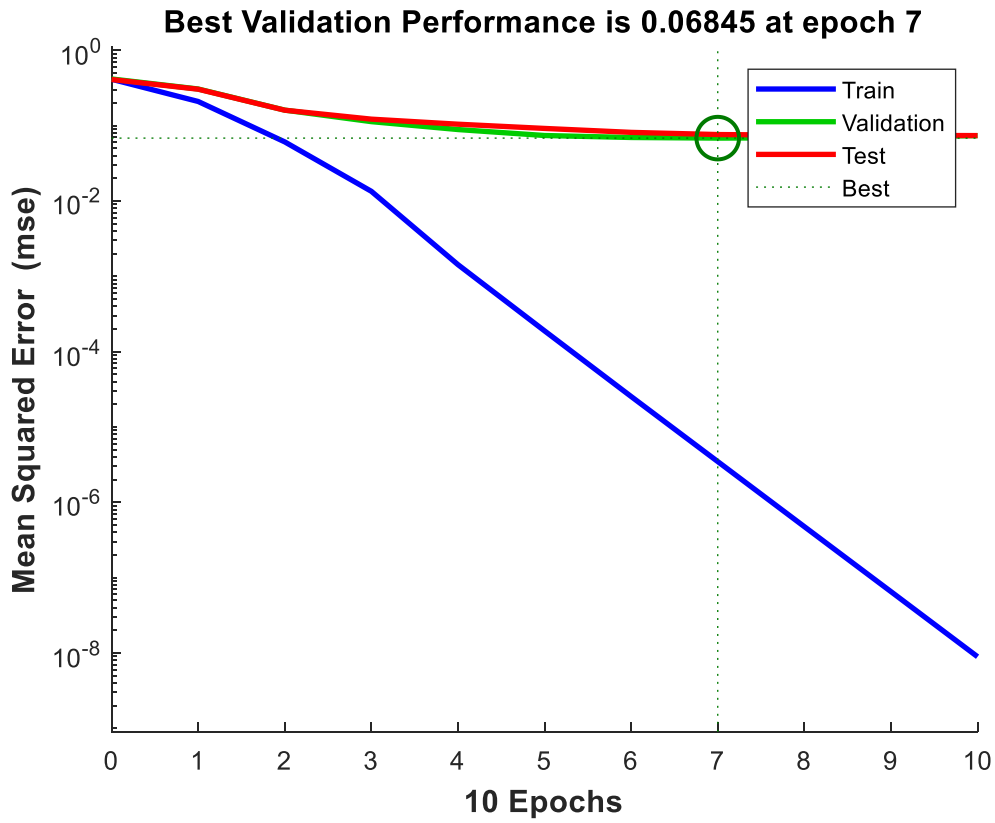


Figure 4.16: Performance Plot of {48-45-24} Network with Training Function Trainlm.

Figure 4.16 displays the performance plot of the network using optimal training function and 45 hidden layer neurons. Training was completed in 38 seconds at 10 Epochs, validation curve shows a validation performance value of 0.06845 at 7 Epochs. Mean Square Error (MSE) of 0.0290 was generated.

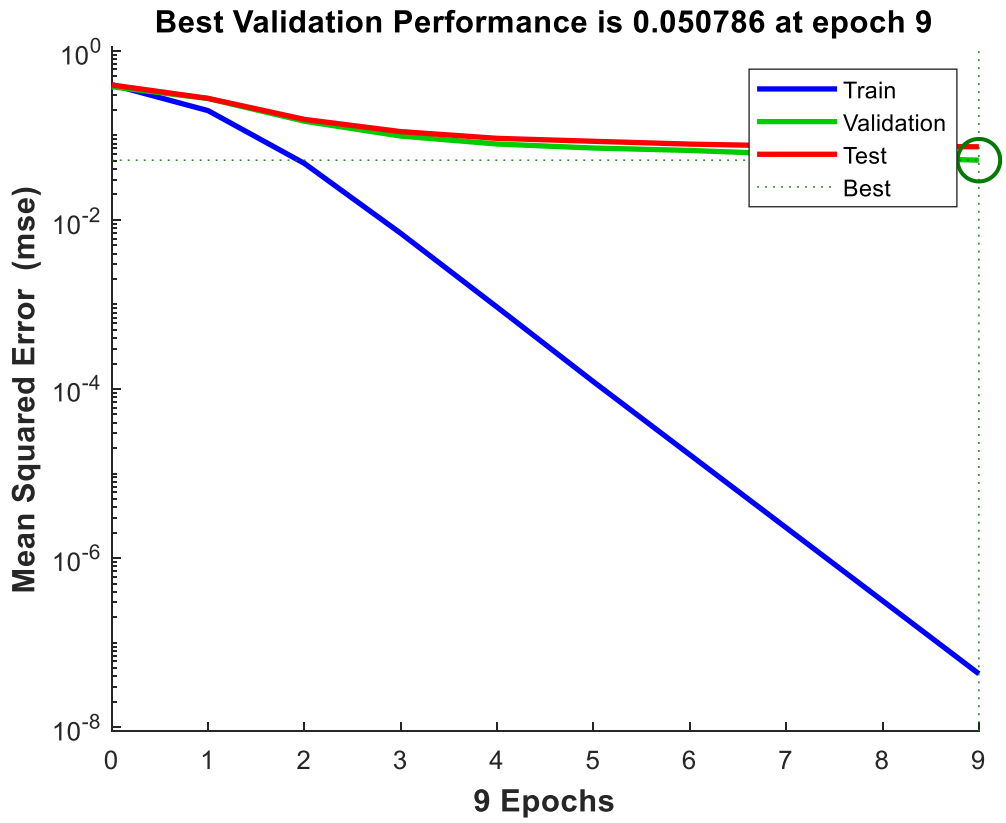


Figure 4.17: Performance Plot of {48-50-24} Network with Training Function Trainlm.

Figure 4.17 displays the performance plot of the network using optimal training function and 50 hidden layer neurons. Training was completed in 42 seconds at 9 Epochs, validation curve shows a validation performance value of 0.0507 at 9 Epochs. Mean Square Error (MSE) of 0.0289 was generated.

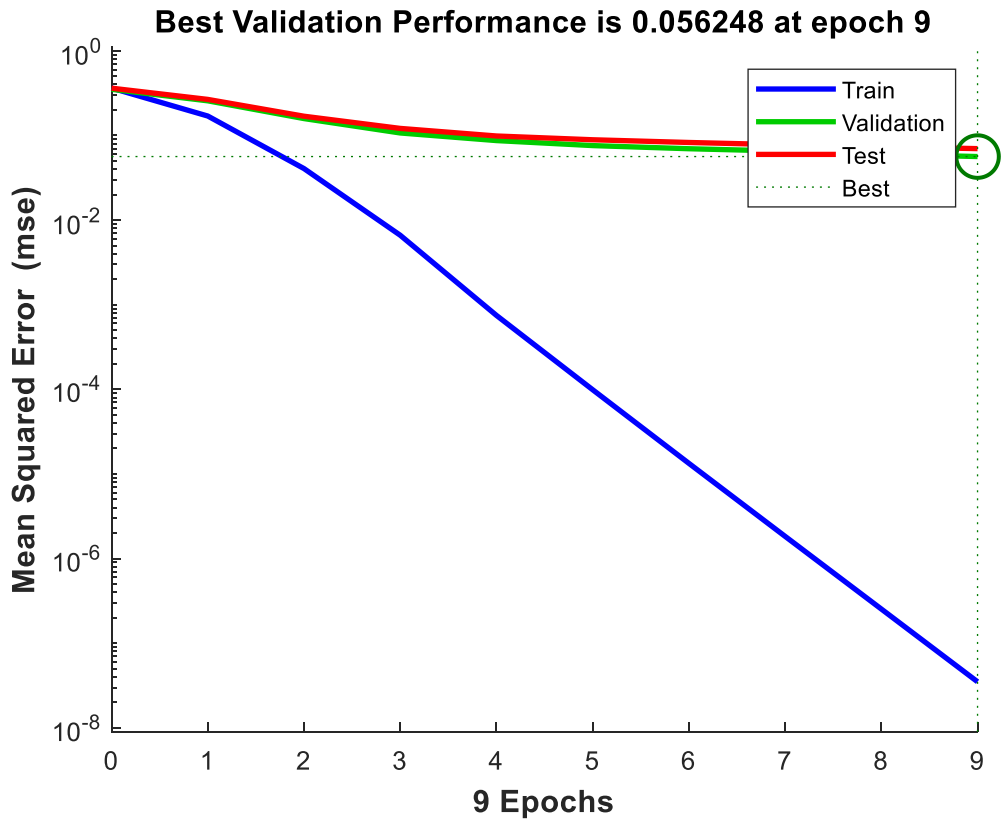


Figure 4.18: Performance Plot of {48-55-24} Network with Training Function Trainlm.

Figure 4.18 displays the performance plot of the network using optimal training function and 55 hidden layer neurons. Training was completed in 55 seconds at 9 Epochs, validation curve shows a validation performance value of 0.056248 at 9 Epochs. Mean Square Error (MSE) of 0.0252 was generated.

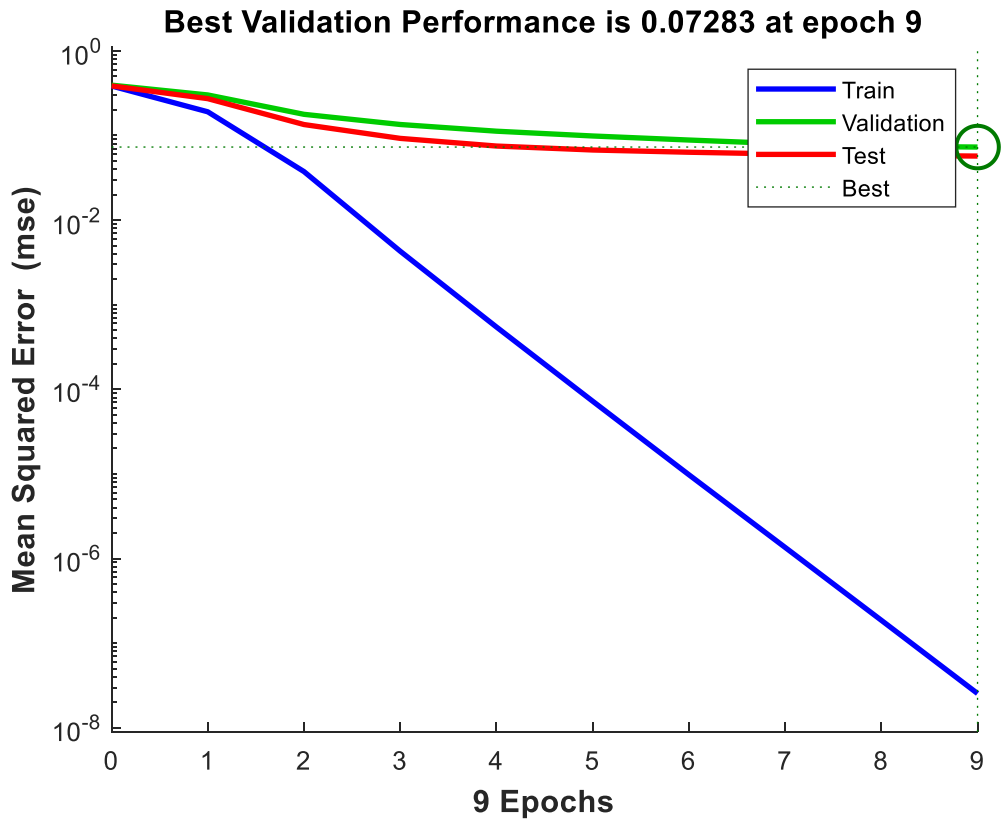


Figure 4.19: Performance Plot of {48-60-24} Network with Training Function Trainlm.

Figure 4.19 displays the performance plot of the network using optimal training function and 60 hidden layer neurons. Training was completed in 56 seconds at 9 Epochs, validation curve shows a validation performance value of 0.07283 at 9 Epochs. Mean Square Error (MSE) of 0.0260 was generated.

Table 4.1: Results of Different Network Architecture

Number of Hidden neurons	MSE	Epochs	Time(sec)
5	0.2184	12	1
8	0.1888	12	2
10	0.1562	12	2
15	0.1062	11	8
20	0.0706	13	10
25	0.0631	17	21
30	0.0563	11	17
35	0.0360	11	25
40	0.0336	11	34
45	0.0290	10	38
50	0.0289	9	42
55	0.0252	9	55
60	0.0260	9	56

From table 4.1 the following observations have been made from comparing the various values for performance criteria:

- The number of epochs becomes constant at a value of 9 epochs after 50 neurons.
- Mean Square Error (MSE) reduces as the number of hidden neurons increases.
- The time taken increases with increase in the number of neurons.

- It is obvious that the network topology with 50 hidden neurons in the hidden layer has the better optimization results among the four modifications even if it takes 42 seconds to train.

4.3 Testing Optimality Of {48-50-24} Network Architecture

The trained network displays a performance criterion known as the Mean Square Error which can be measured with respect to our dataset testing samples. It serves as a measure of how efficient the network will perform with real world datasets. The receiver operating characteristic plot (ROC) is another way to determine how well the neural network has matched the data samples. The relationship between false positive and true positive concentrations is established when the performance limit ranges from 0 to 1.

When markers appear farther left and up, they represent the fewer false positives that need to be accepted to generate a high true positive rate. The best classifiers are characterized by markers going from the bottom left corner, to top left corner, to the top right corner, or something close to that.

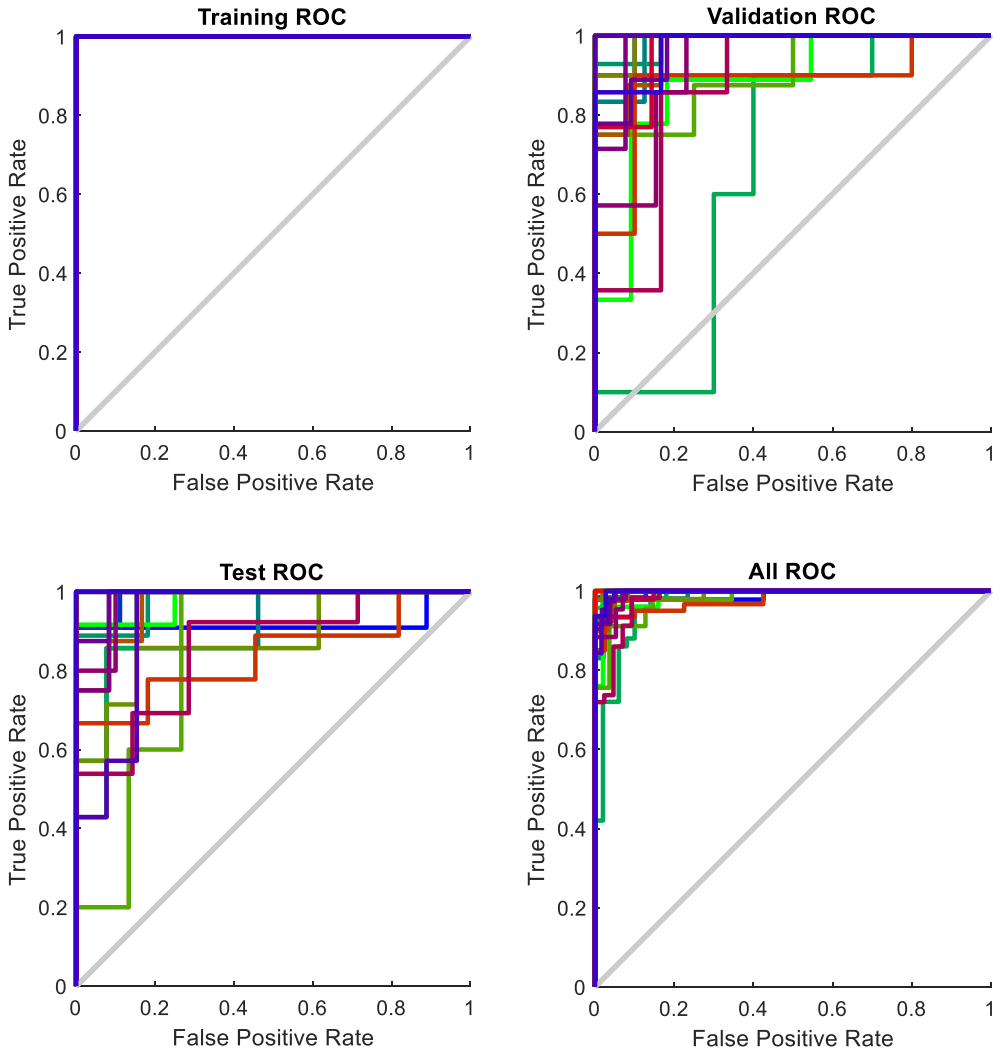


Figure 4.20: ROC Plot of {48-50-24} Network

Figure 4.20 shows that the neural network architecture is efficient.

4.3.1 VALIDATION OF {48-50-24} NETWORK ARCHITECTURE

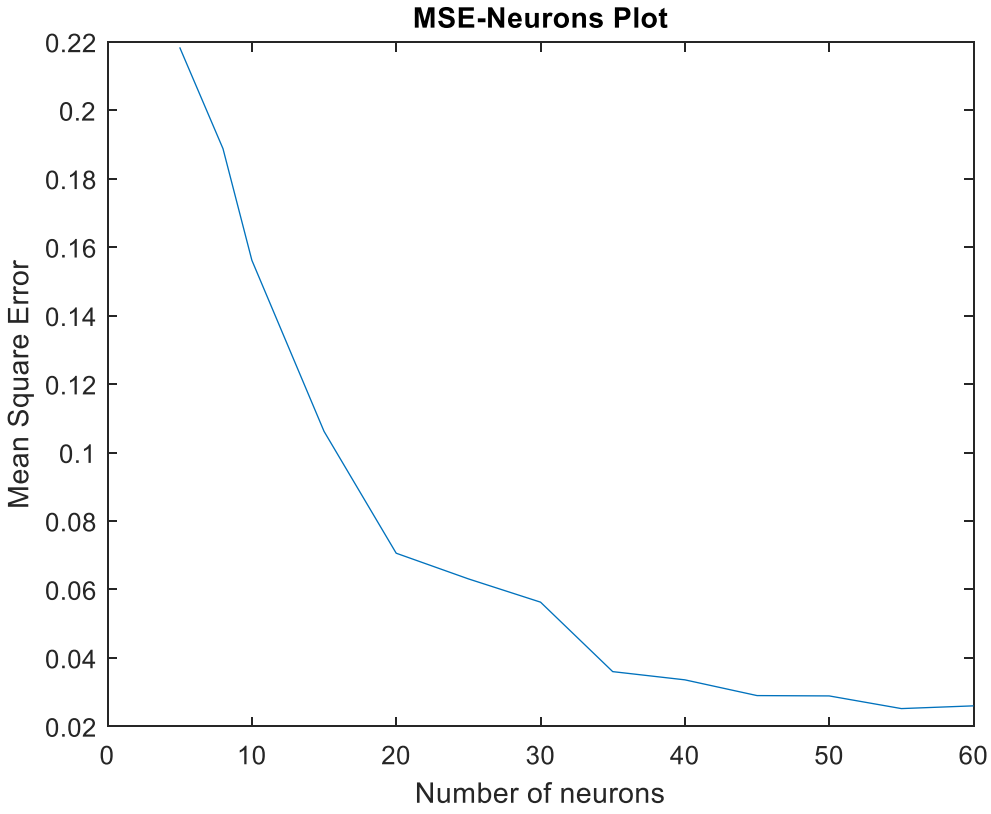


Figure 4.21: Graph showing relationship between MSE and Number of hidden layer neurons.

Figure 4.21 shows that the Mean Square Error monotonically reduces as the number of neurons in the hidden layer increases. This validates the network topology with 50 hidden neurons in the hidden layer as the better optimization results.

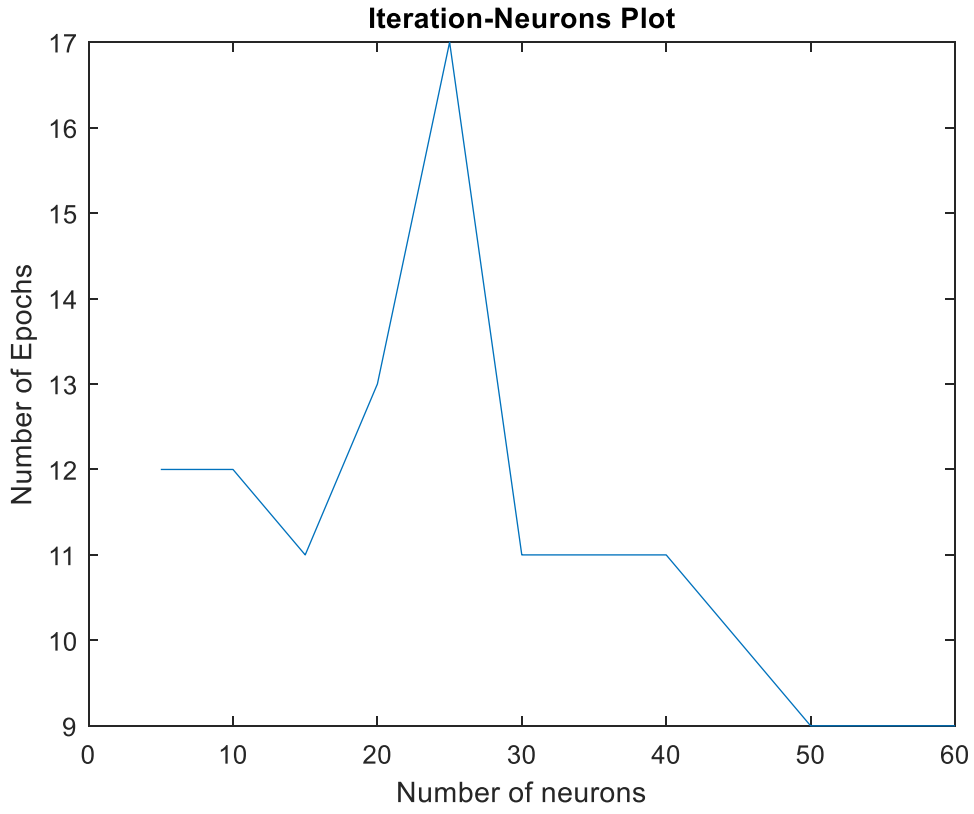


Figure 4.22: Graph showing relationship between Number of epochs and Number of hidden layer neurons.

Figure 4.22 displays the maximum iteration reached for each modification of number of neurons in the hidden layer. From the figure 25 neurons modification has the highest iteration at 17 epochs while the 50 neurons modification stopped after 9 epochs. This validates the network topology with 50 hidden neurons in the hidden layer as the optimal network architecture.

4.4 Performance Evaluation

The performance of the overall system was evaluated by determining overall accuracy, false-positive rate (FPR) and false-negative rate (FNR). The Positive and Negative labels are compared with predicted labels gotten from the neural network classification for evaluating performance. The classification produces four outcomes:

- True positive (TP): This outcome represents a registered MAC address and the network classifies it as positive prediction
- False positive (FP): This outcome represents an attacker's MAC address and the network classifies it as positive prediction
- True negative (TN): This outcome represents an attacker's MAC address and the network classifies it as negative prediction
- False negative (FN): This outcome represents a registered MAC address and the network classifies it as negative prediction

Accuracy is calculated using:

$$\frac{TP+TN}{TP+TN+FP+FN} \quad 30$$

False-positive rate (FPR)

$$1-\frac{TN}{TN+FP} \quad 31$$

False-negative rate (FNR)

$$1-\frac{TP}{TP+FN} \quad 32$$

Table 4.2 shows four scenarios and their corresponding TP, FP, TN, FN values with 20 sampled MAC addresses in each case.

Table 4.2: Scenarios and corresponding TP, FP, TN, FN values

Evaluation metric	Scenario 1	Scenario 2	Scenario 3	Scenario 4
TP	14	15	14	17
FP	2	2	3	1
FN	3	3	3	1
TN	1	0	0	1

The results from these evaluation metrics performed on four scenarios can be found in table 4.3.

Table 4.3: Result generated from evaluation metrics.

Dataset	Accuracy	FPR	FNR
1	75.00%	66.66%	17.65%
2	75.00%	100.00%	16.67%
3	70.00%	100.00%	17.70%
4	90.00%	50.00%	5.56%
Average	77.50%	79.20%	14.40%

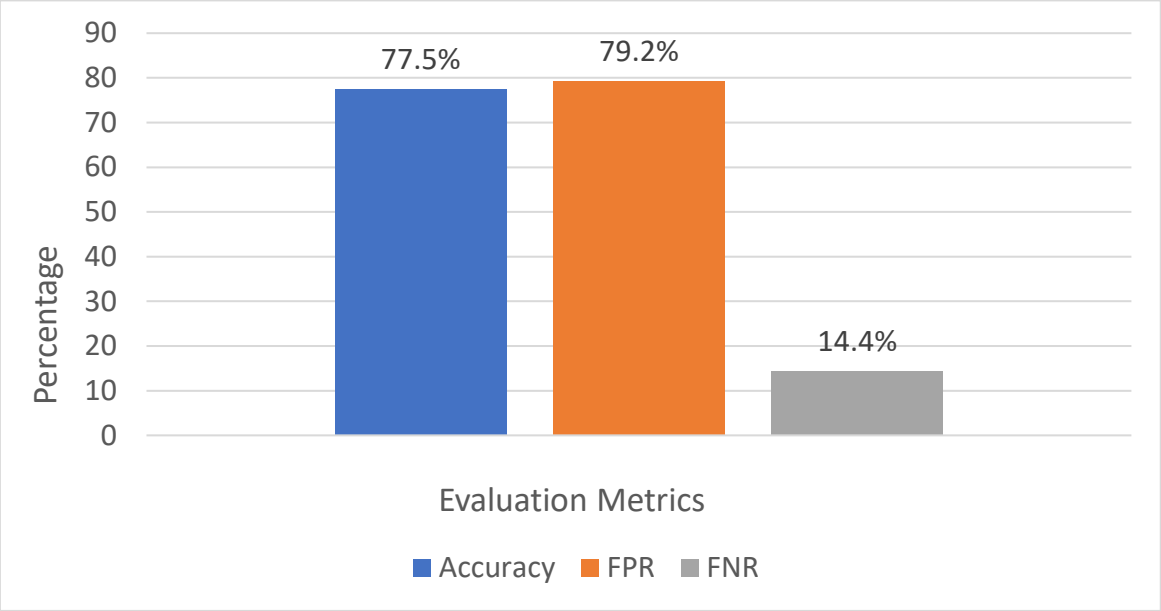


Figure 4.23: Bar graph representing various evaluation metrics

From table 4.3 and figure 4.23, the system achieves an overall accuracy of 77.50%. The false positive rate has a value of 79.20%, while the false negative rate value is seen to be 14.40%. The value of the FNR is quite low which is a sign of the effectiveness of our proposed system, this also implies that our proposed system misclassifies unauthorized MAC addresses by a small percentage of 14.40. This in no way exposes the network to attacks.

CHAPTER FIVE

CONCLUSIOIN

The project was aimed at designing a network authentication mechanism that can detect and prevent cyber-attacks from unauthorised users irrespective of the penetration tool used. In other to achieve our aim we collected MAC Address from trusted sources to ensure the authenticity of our input data and was able to train the neural network to detect the false MAC address using the neural network toolbox (nntool) on MATLAB undergoing different algorithms as explained above.

From the results gotten from the training, we can infer that MAC address plays an important role in the securing of our wireless communication.

5.1 Recommendation

There is more to be done in the area of securing our wireless communication, there is a need to further the study of wireless security. We therefore make the following recommendations for future improvement on wireless security.

5.2 Deployment on Multiple Platform

Protection of information and private data is a challenge faced across all spheres of communication. With this solution deployed across all operating systems and platforms, information and data become more secure and user can rest easy. Due to the unique physical address in the MAC address, it makes intrusion by hackers more difficult and the network more secure.

5.3 Widening of the input parameters

Further research needs to be done on the area of using multiple input parameters for screening and permission of users attempting to login to the network, either authorised

or unauthorised request. With multiple input parameters the network is more secure and impregnable.

REFERENCES

Aimasizadeh , J., & Azgomi, M. (2008). A New Method for Modelling and Evalution of the Probability of Attacker Success. Security Technology, Internation Conference.

Baek, K.-H., Smith, S., & Kotz, D. (2014). A Survey of WPA and 802.11i RSN Authentication Protocola. Dartmouth.

Bellardo, J., & Stefan, S. (2003). 802.11 Denial of Service Attacks: Real Vulnerabilities and Practical Solutions. Proceedings of 12 USENIX Security Symposium.

Choi, M.-K., Rosslin, J., Robles, J., Hong, C.-H., & Kim , T.-h. (2008). Wireless Network Security: Vulnerabilities, Threats and Countermeasures. International Journal of Multimedia and Ubiquitous Engineering.

Gast, M. (2009). TTLS and PEAP Comparison . Interop net Labs.

Gopalakrishman, S. (2014). A Survey of Wireless Network Security . Jornal of Computer Science and Mobile Computing, 53-68.

Halvorsen , M., & Haugen, O. (2009). Crptanalysis of IEEE 802.11i TKIP. Norwegian University of Science and Technology.

Harold, G., Kevin , B., Janne , L., Damon, M., & Douglas, S. (2010). Practical Defense for Evil Twin Attacks in 802.11. Proceedings of the Global Communications Conference. Miami.

Kaminsky, D. (2008). It's The End of The Cache As We Know It. Black Hat. Japan.

Khan, S., Jonathan, L., Tahir , N., & Mohammad, K. (2008). Denial of Service Attacks anf. International Journal of Computer Science and Network Security .

Mc Farland, D., Shue, C., & Kalafut, A. (2017). The Best for the Byte: Characterizing the Potential of DNS Amplification Attacks. Computer Networks. Internation Journal of Computer and Telecommunications Networking, 12-21.

Mockapetris, P. (1987). Domain Name- Implementation and specification. RFC Editor.

Narasimhan, H., & Padmanabhan, T. (2013). 2CAuth: A New Two Factor Authentication Scheme Using QR-Code. Internation Journal of Engineering and Technology.

Nevon Projects. (n.d.). AES Source Code in Matlab. Retrieved from Nevon Projects: http:// www.nevonprojects.com/aes-source-code-inmatlab

Nicholson, A., Chawathe, Y., Chen, M., Noble, B., & Wetherall, D. (2006). Improved access point sselection. Proceedings of the 4th International Confrence on Mobile System Applications and Services . Uppsala.

Refaat, T., Abdelhamid, T., & Mohamed, A.-F. (2016). Wireless Local Area Network Security Through Penetration Testing. International Journal of Computer Network and Communication Security.

The Government of the Hong Kong Special Administrative Region. (2010, December). Wireless Networking Security. Retrieved from http:// infosec.gov.hk

Vijaya, G., Scrikanth, V., & Chandra, M. (2013). Survey on Different Typrs of Attacks and Counter Measures ub Wireless Networks. Internation Journal of Computer Science and Information Technologies.

Vocal Technologies, Ltd. (2003). Counter CBC-MAC Protocol(CCMP) Encryption Algorithm. Retrieved from Vocal Technologies: www.vocal.com

Zargar, S. T., Joshi, J., & Tipper, D. (2012). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. IEEE Communications Survey & amp Tutorials.

A. Abduvaliyev, A. S. K. Pathan, J. Zhou, R. Roman, and W. C. Wong, "On the vital areas of intrusion detection systems in wireless sensor networks," IEEE Communications Surveys Tutorials, vol. 15, no. 3, pp. 1223–1237, Third 2013.

A. A. Gendreau and M. Moorman, "Survey of intrusion detection systems towards an end to end secure internet of things," in 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), Aug 2016, pp. 84–90.

A. A. Hatkar *et al*., "Media Access Control Spoofing Techniques and its Counter Measures," International Journal of Scientific & Engineering Research, Vol.2, Issue.6, 2012, pp. 1-5.

A. Hegde, "MAC Spoofing Detection and Prevention," International Journal of Advanced Research in Computer and Communication Engineering, Vol.5, Issue.1, 2016, pp- 229-232.

A. Le, J. Loo, Y. Luo, A. Lasebae, Specification-based IDS for securing RPL from topology attacks, in: Wireless Days (WD), 2011 IFIP, 2011, pp. 1–3.

A. Le, J. Loo, K. K. Chai, M. Aiash, A specification-based IDS for detecting attacks on RPL-based network topology, Information 7 (2) (2016) 25.

A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, “Evaluating computer intrusion detection systems: A survey of common practices,” ACM Comput. Surv., vol. 48, no. 1, pp. 12:1–12:41, Sep.2015.

Arisoa S Randrianasolo, Larry D Pyeatt, “An artificial immune system based on holland's classifier as network intrusion detection” in 2012 11th International Conference on Machine Learning and Applications 1, 504-507, 2012.

A. Shiravi, H. Shiravi, M. Tavallaee, A. a. Ghorbani,- Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Computers & Security 31 (3) (2012) 357–374.

Atefeh Torkaman, Ghazaleh Javadzadeh, Marjan Bahrololum, “A hybrid intelligent HIDS model using two-layer genetic algorithm and neural network” in The 5th Conference on Information and Knowledge Technology, 92-96, 2013.

Bhupinder Singh and Sumit Bansal, “A Review: Intrusion Detection System in Wireless Sensor Networks” 2017, pg 16.

C. Cervantes, D. Poplade, M. Nogueira, A. Santos, Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things, in: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), 2015, pp. 606–611.

Chung-Ming Ou, Chung-Ren Ou, “Immunity-inspired host-based intrusion detection systems” in 2011 Fifth International Conference on Genetic and Evolutionary Computing, 283-286, 2011.

D. B. Faria and D. R. Cheriton, “Detecting Identity-Based Attacks in Wireless Networks Using Singalprints,” WiSe’06: ACM Workshop on Wireless Security, 2006, pp. 43–52.

D. Chrun, Model-Based Support for Information Technology Security Decision Making, Ph.D. thesis, University of Maryland (2011).

D. Oh, D. Kim, W. W. Ro, A malicious pattern detection engine for embedded security systems in the Internet of Things, Sensors 14 (12) (2014) 24188–24211.

E. Cho, J. Kim, C. Hong, Attack model and detection scheme for botnet on 6LoWPAN, in: C. Hong, T. Tonouchi, Y. Ma, C.-S. Chao (Eds.), Management Enabling the Future Internet for Changing Business and New Computing Services, Vol. 5787 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 515–518.

Farnaaz, N., Jabbar, M.A.: Random forest modeling for network intrusion detection system. Procedia Comput. Sci. **89**, 213217 (2016).

Farhoud Hosseinpour, Kamalrulnizam Abu Bakar, Amir Hatami Hardoroudi, Nazaninsadat Kazazi, "Survey on artificial immune system as a bio-inspired technique for anomaly-based intrusion detection systems" in 2010 International Conference on Intelligent Networking and Collaborative Systems, 323-324, 2010.

G. J. Wright. (2003) Detecting wireless LAN MAC address spoofing [online]. Available: http://www.forum intrusion.com/archive/wlan_macspooof_detection.pdf.

Hai-Yan HOU, Xiao-Lan ZHANG, Jing-Yu GUO, Jun-Jie DUAN, "Off-line Chinese Signature Verification Based on GA-WNN [J]" in Journal of Henan University of Science & Technology (Natural Science) 3, 2009.

Hettich, S., Bay, S.D.: The UCI KDD Archive, University of California, Department of Information and Computer Science, Irvine, CA (1999). http://kdd.ics.uci.edu.

Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735 1780 (1997).

I. Butun, S. D. Morgera, and R. Sankar, “A survey of intrusion detection systems in wireless sensor networks,” IEEE Communications Surveys Tutorials, vol. 16, no. 1, pp. 266–282, First 2014.

Ingre, B., Yadav, A.: Performance analysis of NSL-KDD dataset using ANN. In: International Conference on Signal Processing and Communication Engineering Systems (SPACES), pp. 9296 (2015).

J. Amaral, L. Oliveira, J. Rodrigues, G. Han, L. Shu, Policy and network-based intrusion detection system for IPv6-enabled wireless sensor networks, in: Communications (ICC), 2014 IEEE International Conference on, 2014, pp. 1796–1801.

J. Hall *et al*., “Enhancing Intrusion Detection in Wireless Networks Using Radio Frequency Fingerprinting,” Conference on Communications, Internet, and Information Technology, Baltimore, USA, 2012, pp. 201-203.

J. Jabez; G.S. Anadha Mala, “A study on genetic-fuzzy based automatic intrusion detection on network datasets” in International Conference on Software Engineering and Mobile Application Modelling and Development (ICSEMA 2012), 01 July 2013.

J. O. Nehinbe, A critical evaluation of datasets for investigating IDSs and IPSs researches, in: Proceedings of 2011, 10th IEEE International Conference on Cybernetic Intelligent Systems, CIS 2011, 2011, pp. 92–97.

Junyuan Shen, Jidong Wang, Hao Ai, “An improved artificial immune system-based network intrusion detection by using rough set” in Communications and Network 4 (01), 41, 2012.

J. Vacca, Computer and information security handbook, Morgan Kaufmann, Amsterdam, 2013.

Jing Yu, Feng Wang, “Simulation Modeling of Network Intrusion Detection Based on Artificial Immune System” in 2010 Second World Congress on Software Engineering 2, 145-148, 2010.

KS Sujatha, Vydeki Dharmar, RS Bhuvaneswaran, “Design of genetic algorithm-based IDS for MANET” in 2012 International Conference on Recent Trends in Information Technology, 28-33, 2012.

Kuang, F., Xu, W., Zhang, S.: A novel hybrid KPCA and SVM with GA model for intrusion detection. Appl. Soft Comput. **18**, 178–184 (2014).

L. Arockiam and B. Vani, “Medium Access Control Spoof Detection And Prevention Algorithm (MAC SDP Dos) For Spoofing Attacks In WLAN,” International Journal of Computer Science and Information Technology & Security, Vol. 3, Issue.2, 2013, pp-165-171.

L. Sheeba *et al*. “A Brief survey on Intrusion Detection System for WSN”.

L. Wallgren, S. Raza, T. Voigt, Routing attacks and countermeasures in the RPLbased Internet of Things, International Journal of Distributed Sensor Networks 2013.

M. T. Hagan *et al*., “Neural Network Design,” PWS Publication, 1996.

M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in Proceedings of the Second IEEE International Conference on Computational Intelligence for Security and Defense Applications, Ottawa, Ontario, Canada, 2009, pp. 53–58.

M. V. Mahoney, P. K. Chan, An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection (2003) 220–237.

Manali Singh, Khushbu Babbar and Kusum Lata Jain, “A Survey on Intrusion Detection System in Wireless Sensor Networks” International Journal of Wireless Communications and Networking Technologies (IJWCNT) vol. 3, No.3, Apr-May 2014.

Md. Safiqul Islam, Razib Hayat Khan and Dewan Muhammad Bappy, “A Hierarchical Intrusion Detection System in Wireless Sensor Networks” International Journal of Computer Science and Network Security (IJCSNS), vol. 10, No. 8, August 2010.

Megha Gupta, “Hybrid Intrusion Detection System: Technology and Development” International Journal of Computer Applications (IJCA) vol. 115, No. 9, April 2015.

Meijuan Gao ; Jingwen Tian, “Wireless Sensor Network for Community Intrusion Detection System Based on Improved Genetic Algorithm Neural Network” 26 June 2009.

N. K. Thanigaivelan, E. Nigussie, R. K. Kanth, S. Virtanen, J. Isoaho, Distributed internal anomaly detection system for Internet-of-Things, in: 2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC), 2016, pp. 319–320.

Niyaz, Q., Sun, W., Javaid, A.Y., Alam, M.: A deep learning approach for network intrusion detection system. EAI Endorsed Trans. Secur. Saf. **16**, 21–26 (2015).

P. Kasinathan, C. Pastrone, M. Spirito, M. Vinkovits, Denial-of-service detection in 6LoWPAN based Internet of Things, in: Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on, 2013, pp. 600–607.

P. Pongle, G. Chavan, Real time intrusion and wormhole attack detection in Internet of Things, International Journal of Computer Applications 121 (9) (2015) 1–9.

P. R. Babu *et al*., "A Comprehensive Analysis of Spoofing," International Journal of Advanced Computer Science and Applications, Vol.1, Issue.6, 2010.

Reddy, R.R., Ramadevi, Y., Sunitha, K.V.N.: Effective discriminant function for intrusion detection using SVM. In: International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 11481153 (2016).

S. Raza, L. Wallgren, T. Voigt, SVELTE: Real-time intrusion detection in the Internet of Things, Ad Hoc Networks 11 (8) (2013) 2661 – 2674.

T.-H. Lee, C.-H. Wen, L.-H. Chang, H.-S. Chiang, M.-C. Hsieh, A lightweight intrusion detection scheme based on energy consumption analysis in 6LowPAN, in: Y.-M. Huang, H.-C. Chao, D.-J. Deng, J. J. J. H. Park (Eds.), Advanced Technologies, Embedded and Multimedia for Human-centric Computing, Vol. 260 of Lecture Notes in Electrical Engineering, Springer Netherlands, 2014, pp. 1205–1213.

V. Jyothsna and V. V. Rama Prasad, "A Review of Anomaly based Intrusion Detection Systems" International Journal of Computer Applications (IJCA) vol. 28, No. 7, September 2011.

V. Verendel, Quantified security is a weak hypothesis, in: Proceedings of the 2009 workshop on New security paradigms workshop - NSPW '09, 2009, pp. 37–49.

Wang Yunwu, "Using Fuzzy Expert System Based on Genetic Algorithms for Intrusion Detection System" 2009.

Yin, C., Zhu, Y., Fei, J., He, X.: A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access **5**, 21954–21961 (2017).

Yonghui Shi; Jun Bao; Zhongzhen Yan; Shengping Jiang, "Intrusion Detection for Transportation Information Security Systems Based on Genetic Algorithm-Chaos and

RBF Neural Network" in 2011 Third Pacific-Asia Conference on Circuits, Communications and System (PACCS). 18 August 2011.

Zhang, J., Zulkernine, M., Haque, A.: Random-forests-based network intrusion detection systems. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **38**, 649–659 (2008).

Zhou Shaojing, "Research of the artificial immune intrusion detection system model based on the E-learning" in 2010 International Forum on Information Technology and Applications 3, 126-130, 2010.

APPENDIX

APPENDIX A: MATLAB IMPLEMENTATION

Preview

This appendix contains a listing of the functions in the MATLAB Pattern recognition Toolbox used to obtain results for this study, and all the custom functions developed.

A.1 Optimal Topology Function

This function generates a neural network for determining optimal network topology, it may also be referred to as the predictive algorithm.

```
% Solve a Pattern Recognition Problem with a Neural Network
% Script generated by Neural Pattern Recognition app
% Created 13-Nov-2019 03:42:46
%
% This script assumes these variables are defined:
%
%   ConvertedInput - input data.
%   TargetData - target data.


 [y1, y2] = xlsread('MAC adresses.xlsx');
 g = hex2dec(y2);
 f = dec2bin(g);
 display(f')
 xlswrite('MAC adresses.xlsx',f','convertedaddress');

y3 = xlsread('MAC adresses.xlsx','convertedaddress');

x=y3; % ConvertedInput;

t=y3(25:48,:); %t = TargetData;

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. Suitable in low memory situations.
trainFcn = 'trainlm';  % Scaled conjugate gradient backpropagation.
```

```
% Create a Pattern Recognition Network
hiddenLayerSize = 50;
myNet = patternnet(hiddenLayerSize, trainFcn);


% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivision
myNet.divideFcn = 'dividerand';  % Divide data randomly
myNet.divideMode = 'sample';  % Divide up every sample
myNet.divideParam.trainRatio = 60/100;
myNet.divideParam.valRatio = 20/100;
myNet.divideParam.testRatio = 20/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
myNet.performFcn = 'crossentropy';  % Cross-Entropy

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
myNet.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
    'plotconfusion', 'plotroc'};

% Setting maximum number of iterations:
net.trainParam.epochs = 1000;

% SPECIFY TRANSFER FUNCTION
 myNet.layers{1}.transferFcn = 'logsig';
 myNet.layers{2}.transferFcn = 'logsig';

% Train the Network
[myNet,tr] = train(myNet,x,t);

% Test the Network
y = myNet(x);
e = gsubtract(t,y);
performance = perform(myNet,t,y)
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t .* tr.valMask{1};
testTargets = t .* tr.testMask{1};
trainPerformance = perform(myNet,trainTargets,y)
valPerformance = perform(myNet,valTargets,y)
testPerformance = perform(myNet,testTargets,y)

% Measure performance based on Mean Square Error
MSE = mse(myNet,t,y)

% View the Network
 view(myNet)

% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, ploterrhist(e)
%figure, plotconfusion(t,y)
```

```
%figure, plotroc(t,y)

% Deployment
% Change the (false) values to (true) to enable the following code
blocks.
% See the help for each generation function for more information.
if (false)
    % Generate MATLAB function for neural network for application
    % deployment in MATLAB scripts or with MATLAB Compiler and Builder
    % tools, or simply to examine the calculations your trained neural
    % network performs.
    genFunction(myNet,'myNeuralNetworkFunction');
    y = myNeuralNetworkFunction(x);
end
if (false)
    % Generate a matrix-only MATLAB function for neural network code
    % generation with MATLAB Coder tools.
    genFunction(net,'myNeuralNetworkFunction','MatrixOnly','yes');
    y = myNeuralNetworkFunction(x);
end
if (false)
    % Generate a Simulink diagram for simulation or deployment with.
    % Simulink Coder tools.
    gensim(net);
end
```

A.2 Neural Network Function

This function is a MATLAB script that is deployed to enable the examination of various calculations by the trained neural network.

```
function [Y,Xf,Af] = myNeuralNetworkFunction(X,~,~)
%MYNEURALNETWORKFUNCTION neural network simulation function.
%
% Generated by Neural Network Toolbox function genFunction, 15-Nov-
2019 01:35:29.
%
% [Y] = myNeuralNetworkFunction(X,~,~) takes these arguments:
%
%   X = 1xTS cell, 1 inputs over TS timesteps
%   Each X{1,ts} = 48xQ matrix, input #1 at timestep ts.
%
% and returns:
%   Y = 1xTS cell of 1 outputs over TS timesteps.
%   Each Y{1,ts} = 24xQ matrix, output #1 at timestep ts.
%
% where Q is number of samples (or series) and TS is the number of
timesteps.

%#ok<*RPMT0>

% ===== NEURAL NETWORK CONSTANTS =====

% Input 1
x1_step1.xoffset =
[0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
;0;0;0;0;0;0;0;0;0;0;0;0;0];
```

```
x1_step1.gain =
[2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2;2
;2;2;2;2;2;2;2;2;2;2;2;2;2];
x1_step1.ymin = -1;

% Layer 1
b1 = [-
2.6627401436122974943;3.0262853588054112741;1.3486686878856126715;2.16
68584289635228757;-
2.4705541679772520247;1.8425920861676623463;1.3703685261825211406;1.75
84272004758292329;1.7162047170809469065;-
1.6799579887297972913;1.4071186217292737108;1.2998753992586231654;-
1.0805543752595569984;-0.91051751046639417542;-
1.1356010183763471932;0.77630651454188592808;0.36813087803451677305;0.
42939564298344035098;0.24752936222475765971;0.93725468484487839405;-
0.82780191267657721177;-
0.29819997516482521505;0.69535027996022602537;0.082079592944388976572;
0.4861201450071405139;0.4895370394132358971;-
0.34334962876122659958;0.36984397798637563826;0.43280139896021557755;0
.88245817462244613427;0.26179640537488685359;0.20536008376072056869;-
0.76548610658737559209;-0.39987023422059508881;-
1.0657631402811935661;0.75811968433065490824;0.61932508804576569261;1.
3124303278756024316;-0.89174107739053198873;-0.4028455406116306925;-
1.2354340677805688564;1.6288529402875038254;2.1754321236161002773;1.19
81065246645103972;-
2.3117778567025100145;1.7626494864741300006;2.21403515095319392;-
2.2035014133684778237;-2.3579240769194131921;-2.3735376143874260535];
IW1_1 = [0.62050385161698518033 -0.5107491906823535821 -
0.19060208049281651688 -0.33177038581265122508 -0.94119535573089763147
-0.63077527926544374459 0.50392751042147332363 -0.29275659556514810333
-0.06557630572377128042 -0.41718282353174734878 -
0.86867099133230962771 0.13705049908537750292 -0.29631909225342911718
-0.10129741574965166195 -0.84694462729063491047 -
0.042382570498114363611 0.077023198284037253103 -
0.71965926667200352806 -0.31332165606245282463 0.36209133517536401525
0.90703766153591003363 0.2735002596735486291 -0.26170192302449035715 -
0.36301260343788926965 1.24923544641054729 1.4332543327599078786
0.27503236836395072995 0.74668012613626977281 -1.0366794455225669669 -
0.27304053499127012294 -0.51006810084251608117 0.70614176623880708927
0.47693936814373055766 -0.16258353358333590966 0.29191359727694926596
1.9443818999135411563 0.876270557837068087 0.24062680933632785152 -
0.97083160607122731545 0.47301460707593845356 -0.67677481688790697145
-0.38233970453930082867 1.378346814942603471 0.82788938543589640773
0.99658944759192802376 -1.1574684906157166608 -0.65356717269131259229
-0.73178190121591701889;0.10367046358610443324 -0.76322658737983151145
-0.16341752901703129885 0.29778674457195530856 -1.0129324889553525946
0.80268019349520647321 0.22850540346064446284 -0.84629706798807102786
0.33407535578484182093 -0.037619425424326197183 -
0.038900248522527519635 0.46125021483388078947 -0.05052668082056400306
-0.13319420499121226165 -0.2325244566937187396 -0.29895288005579773172
0.31961609074883551296 0.30777791498793349856 -0.14147521109002703099
-0.029933334289178482185 -1.162928379752085073 -
0.089811538769799734161 0.10178830198129446005 -0.75688932872521019757
-0.61661850811142637774 0.10617727077244426548 0.89525614345899739988
0.24530672691762292481 -0.086135862901511275891 -1.2887196673543912517
-1.4464468287200706875 -0.47115595589972725232 0.33274490530080230721
-0.23989637087787496483 -1.2710976464689958121 0.44470465134938591545
0.12891720471062784181 -0.91994670946027168323 0.2648092096180000854
2.2007365147354942536 -0.17010739930734308789 0.69101600498216642077
0.63331958221874118387 -0.78114443071950323549 0.34240041952301025763
-0.38211482900066040536 0.38815058531719859314
0.19600596365022429635;-0.38907161111977578427 0.25663326796184326994
```

-0.17596418815107944811 -0.56066865189725045848 0.26128782548446921608
-0.55164108976644898874 -1.0399833083076395202 0.32479690743425909671
0.62516865571662483259 -0.87857559110821059623 0.29340937603189570781
0.81056459259624458191 -0.49868095283565683218 0.089783480205195595469
-1.2175321547225064212 -0.81392401127565638941 0.57392510936689944412
0.28084537942415860323 0.15676725090635168502 -0.041123848032636661776
-0.8556902480851439341 0.21759739122727700389 0.008543096772819135098 5
0.60547210400977802269 -0.47806832586158254372 -1.8114964869889804699
-0.093216650178497714729 -0.72870400364808263216 -
0.35772534021624313727 -1.1309877795376888088 0.19916410379822069299 -
0.29371522100650704346 -1.1061503003278876367 -2.0464774642675989114 -
1.7836458701442590158 0.18147396076242844343 -0.83100216130887882837
0.89623543013914064304 0.11558257716664135928 -0.46413566446907789098
-1.6953925835925944821 -1.2404017866178598251 -1.2396777601525539314
0.070172280273117129679 -0.52641931751540682605 -
0.95809665228867457731 -0.33398970587269294352 -
0.66567863816410877131;-0.28431228061283303488 -
0.078149566260879890778 -1.0597774525063110573 0.18608146556433025065
-0.30619995264691735049 0.038617477285568281575
0.069475904544407912633 -0.19438041657250360039 0.79736195214396055064
-0.24026214553944622021 0.18218533336366554698 -0.53758802879571543354
0.72033616563970415125 0.074979022586213917023 0.082745378098628710739
-0.4872989563540363056 -0.32833156924568296375 -0.49860117622385147218
0.62696577023446731225 -0.74767533403530905822 0.89449031368586484625
-0.44730952326246198325 -0.07769243873774611675 0.14143689769864675743
1.0898381626478053086 0.43748357345910260374 1.3908079002312465633 -
0.84960552542719003366 0.34827481405221405453 -
0.007499983143343962478 -0.34248578044486710503 1.6808168773990734834
0.18568311049319283224 -0.069821991051267073836 0.55062556034197662758
-0.10645555738780196342 0.073511836507077821246 -
0.56125965345920592053 -0.5607944288918439435 -0.32324082517389640667
0.89472373439730845668 -1.3519929953242086906 -0.7467735508954315149 -
0.99963371974677628362 -1.1272696469388565621 0.59909142848069918408 -
1.3929842550556874592 -0.54727716539960857833;0.13069996751829909831
0.30183369825366129779 -0.12414471333316337565 0.88968751708750470897
-0.065706676291235027731 -0.090843193916915532249
0.10910263961635568997 -0.82163814866732265418 0.2155508649738523963 -
0.28713852456829480619 0.1237469366290719075 -0.46605957494319827372
0.86625385431982171447 0.5278525078609743737 1.1382847147500791962
0.36982100416011870481 0.082707344354442088874 0.28571636427586127871
-0.069639544916889464798 -0.45867637861832527557
0.56853886116130691875 0.007455515587741950360 51 0.20107736583443003298
-0.38856695080130526643 0.59561739904055788219 -1.1470691520730451618
0.14506765378761049545 1.4976117825519810189 -0.83552792400530229155
0.30468394766088863213 -0.10444667281155467498 -0.63995578976409739358
1.279766215680901098 -0.16035199896486621 03 0.83483061912869438448
1.2182546582667168611 -0.31912161597040888 95 1.331910383352695515 -
0.37270641646437507877 0.54541286752634732249 0.55779000004506062993
0.36378165232283032893 -0.35660758939711872939 0.18930542709430742132
-0.52467437507303704702 1.2498915306716542606 -0.46037227086911575169
0.13210534121852951506;-1.1493494848079992554 1.055281560008508368 -
0.94682831835037584156 -0.48599021750706461464 -0.31772196727032014918
-1.1703717890298155346 0.83734381897874021 75 -0.12763335964972397085
0.33667993088828995774 -0.96263671524969152138 0.030981434087299010438
-0.069396174402245269786 0.32588598589207756762 -
0.22052205236637087427 -0.84191260858845440307 0.11657688500133601972
0.91131309282418704321 -0.19356163121053396314 0.46887889227131512859
0.52519461988671534147 -0.27632562425373224935 -
0.045979167305046343694 0.50336707634578659043 0.19315591519174260537
-0.24933171661222691684 -0.49730346889745108419 0.85980043091049218607
0.61237145660345448928 -0.1925022494331955214 0.40417803179053385376 -
0.93257626854531117111 0.77672172581328469132 0.87051068057194558314 -

0.21479993519934911794 -0.82619283018529898754 1.25489115738060141866 -
0.06700568846934880362 0.45491976884406548631 0.62527591100718793449
-1.6332743782592495929 -0.36055658598785023106 0.84721752680033823868
1.0147607957534563461 0.80880047304319291346 0.42002406795023716768 -
0.46189998259342052167 -0.46595577514044644873
0.64739107776745530298;-0.00020697775401612439339 -
0.14874403346506021029 0.21122894481012133294 -0.10113604517962455909
1.306536623417763332 -0.82648909253018432164 -0.18792289286185537289
0.37500699477526139303 0.51073393420873292037 -0.20455856982830894486
-0.47922460124762611411 0.70564944944361129853 -0.59181694718405419842
-0.55875766359997691701 -0.64982077535699611648 -
0.29118575945060248822 -0.81775011810556597069 0.31024348343576158005
-0.51497391246700996437 0.96080870432380194313 0.15540295626693373388
0.086947528629625805974 0.0493195356868702528 0.51966136588591627099 -
0.14488717987975777146 0.96622294060778490810 -0.17700408970108122420 -
1.0040875395082418109 0.86820761192112982929 1.1857231978407685169 -
1.5967607298169557151 0.88263982608693103415 -0.05650958119575012897 62
-0.38391694369095830108 -0.44557063824689091547 -
0.54327167407853016456 0.78358837803805359012 -0.33115494715372068013
-1.3456085887380242472 0.2802744258851034842 0.52829697928210839208
0.080812670184228163617 -0.013568396537891130871 -
1.2366928246056010376 -0.59812576023989716401 0.05062623932566499840 3
0.90686499818489152425 0.50963904482579169830 -0.99044325696390123071
0.69808480047505538924 0.57465797416819253129 -0.60203179291736508283
0.40023580778780759593 0.7027821265686075014 -0.39506610132276298586
0.74865244745590253661 0.63707785215020484859 -0.45229956113595742861
-0.76040674482407077939 0.75377786740960239786 -1.15339964533974526080
0.85606313029797065361 -1.2656765707066695015 -0.17196449541672481631
-0.56523215172428697972 0.71666169294173120985 0.24982799740431654856
-0.61569809858577284611 0.70403979752697776195 -0.21292418000547047585
0.99997880159424490376 -0.30223567241304249986 -0.21833664131559457622
0.17590232008950262843 -0.18419960106355498519 -0.13082167767250360568
-0.47236805995988906393 1.377811871438254343 1.1445909913067919828 -
0.65631341382043795196 0.40707958288709283901 0.27568350718260847820
0.66154869385781811530 0.011737186636563815695 0.36481235237013703854
0.68111553517909062538 1.65197210913040781750 -0.026159929186841714144
0.35991756968892363311 0.31776343662768707921 1.0387776280288820718 -
0.64480744917886112511 0.74049698568973132407 0.50523370945195855697 -
0.41884012235548812031 -0.56279442298705972814;-0.638974602208658759570
0.34504504773766270231 0.53839471883590050982 0.42115410767486932686
0.20608281396304026223 0.093424560619159768837 0.39871360794904047900 79
-0.2783255401873326873 -0.91283638470450800817 0.57884333175537405758
-0.00016842243946029594737 0.78919953323676805823
0.39880872356589897043 -0.52015222026783114462 0.36570594256840782688
-0.38838061777129229135 -0.57341640386219117076 -
0.32106699456931042658 -0.15381382842491919671 -0.16854820515241603074
-0.43652493948986559325 1.0913080312201957067 -0.34256903714450637199
0.42341203329523158239 0.43881575357275642846 0.07023829550492857731 7
1.6996952957649056337 -1.2792655867050930496 0.02887158494846682466 1
1.2065358446395266778 -0.15890454134252463003 -0.65408535049026561570
0.70878933735092353174 0.81098227592112170914 -0.63198752263323754175
1.473903314901453987 -0.16707081387053868138 0.78629928498602852383 -
0.87318659829163403074 -0.91427994559950209741 0.61208896176752980445
-0.25620330796145357466 -0.080003767191570540618
0.19385929988390354639 0.62226337804636000506 0.37018815975904578730 -
1.2108282571946302753 -1.16112929094727523880;0.37655940990565683890
0.041479640155253620637 0.20300534029444422535 -0.74903306279417203317
-0.67223695089884794029 -0.65773619319830489705 0.31180041672614383641
0.41302606548078590887 -0.54263904608850854761 0.26564494364645829805
0.16736818097078012824 -0.56745113180576733924 -0.44630054113253497805
0.64845378162869671268 0.57165215078074427257 -0.30485639640315692622
0.64556882591696218565 -1.1094160310568477179 0.61875214118126331542

0.4672602788837187382 0.37699959226457047112 -0.32982998839841570771
0.72440820556992513435 -0.20817471722392777433 -0.42902214367845725729
-0.52710443361516345906 -1.4298736411695194271 -0.43849816665059371257
-0.65949402522813271865 0.47695642945845401206 -
0.033216134198765109864 -0.60422736369939289425 -
0.050496995595585465744 -0.061819960569145168683 -
1.8328417490616215613 0.27279194890155239461 0.96695389881736348858 -
0.9172618920784532115 -0.56412476931923738466 0.24735032153672908972
2.26321413777047864 -0.34267219836320261805 0.75701312430906575468
1.1382749266000138366 0.033024381731315388677 -0.012982065154080199218
0.60787262101089667432 1.1032258803042265249;-0.92385930400144444796 -
0.061580510626671948471 -0.43882812243208774161 0.5986255004020819559
1.3011343791426546179 -1.2815780227705095839 -0.041924717259839004713
0.13407053923379527727 0.43679926966029847257 -0.097763549405739641474
0.32905658686099653609 0.33093083661886457669 -0.24264793164409009796
0.053336143965040709591 0.35743396398255650181 -0.29211516156611316841
0.22551757483360049261 0.55718902026534344873 0.15128118960911871027 -
0.29358352006348953367 -0.27618246834156373604 0.11447392387199577246
-1.5428733873589315362 -0.70125636787761791435 -0.623731292095101363 95
0.89085043755225545858 0.34007129157528870333 -0.25958244959606863178
-0.88388747844370740836 0.98274775484000054249 -1.6927297783817014665
0.34741948350385898348 -0.32120034065626074105 -0.16103067666744352859
-0.76750919715330878645 -1.465087163490150779 0.028945973412722626 75
0.40331034523029662964 1.3981893613651101838 0.92867345924913258237
0.86774157110078264932 -0.08650063224557924535 -0.331922554254576784 86
0.68389916758226676485 1.1079339137635799695 0.94890521694662921526
0.11995152585536242307 -0.29664078245396496847;-0.8760290389581374404
-0.20887135319123426025 0.24663366577599066254 -0.480802688916273535 11
0.11176365846689317873 -0.093526258369021730843 -1.0575048046579558214
-0.42436178827609066788 0.55086138186365041225 0.70030818796634286549
0.74796216477913590737 -0.1245881530062139253 0.41860539336481644712
0.90536838870556679559 -0.2989230324722216281 -0.244725275976622913 54
-0.074680692085430683269 0.29047351706719110043 -
0.22422870462919256007 0.20064726048579339945 -0.464356552269310440 82
0.60177983357048736579 -1.209372645096119081 0.15745618836358427073
1.1784194016706219799 -1.5344238263773504194 -0.064753248460545947962
0.62276408817525208406 -0.4682615061065600881 1.7186389974364890687 -
0.052664826872457916329 -0.11542884308633473422 0.322764372132168964 05
-0.65882864721710943101 1.6431634366872898134 -0.435013585478165265 56
-0.1010140864677978173 0.67420598999812764163 -0.883396521577127558 09
-0.826950613448547589 -0.52366304947496811195 -1.1788883817385529262
0.20409325775263603697 -0.03093264515680523366 -1.442354509307017584 9
1.2100806021836305604 0.2832843801178105414 -0.55068262068899320738;-
0.28389223979981281776 0.65391938015762252068 -0.105690347274228546 58
0.23208288733253593095 0.16035456915419554758 -0.018331321287948516841
0.035612486328703986393 0.64378785741992827774 0.110941364448686782 82
-0.3884828530847725947 0.27981754478215564719 -0.236875975261246707 33
0.31040804666946114621 -0.1926627276299748559 -0.133450682184755864 41
0.17636634311689231014 0.69437559621515376751 0.37856265290475132 89 -
0.26341866809020819673 -0.48848275506203020502 0.403066771359641584 65
-0.27577089416651368037 0.15028406745826733415 -0.024018352677222985 78
-0.5411171762859722234 -0.30443169067452032 88 -0.763225463461705722 07
-1.7470800799943020376 -0.21575358113046669284 -0.123565261112619242 94
1.4201875282830800362 -0.1921670027110585488 0.401489010955894287 67
0.75115443537691617415 -0.59409060907805444351 1.0068388704853674653
0.30036460839996975691 -0.11353694599260868991 0.041903987437559894746
-2.0297992125787081008 0.76058594102859555797 0.97210981777053984132 -
0.40136556932463168801 -0.55760816060650353343 -0.33894248039122659 71
-1.1114384249780251412 0.55995090686491100218
0.74404776552323936833;0.43645886436474701231 0.29682339074069996609
0.12647825959844288812 0.53018162501256982377 0.06763387239797809258 3
1.0275611816464460535 0.19815263614888778 41 -0.046026744804868194305

0.22040302959622176671 -0.14805087008083048228 0.098720403708547468935
-0.49254044159233389966 0.81078866203859667472 0.55866904809194184445
-0.049031133390406128614 -0.542552213533870841820
0.26494675224182240303 0.357087156850840026220 0.368515602864077584930
0.827672503389273339640 -0.021184445971553958477 -
0.303039986601591926350 0.125403727240453094490 0.411987999951198891060
1.0198051304283253238 0.067120776020647143767 0.332291189888612503720 -
0.75810340893363992176 -0.362964723681616807660 0.339738844461169198220
0.59567980495045036449 0.971992840494742216250 -0.674116262725059115017
0.08390838109257821309 -0.657319817035682207870 -0.150177057551360182420
-1.6006427598935788481 -0.1155645253539007572 0.023278087311217155603
0.32663497661119550441 -1.963735078385709043 -0.32099939672847893624
0.043410783191068932907 0.480535386333911462840 -1.3474841039969669421
1.3198613265554002272 0.734843930257301103380 0.032714675440624625680;-
0.59387799333767299359 0.211706844736283206210 -0.52123431937092590260 -
0.40025471374484083986 -0.191433097694857390360 0.377280912379526545130
0.60158663355751373913 -0.219935763322583549550 0.076531969079296269243
0.24655334687963931239 -0.059877358404171375506 -
0.54321929606450214401 1.233086193767681582 -0.402154895644225907070
1.1814033487337758377 -0.799171395959387131920 0.529463961783705583390 -
0.54474945880587122282 0.416455557615554904270 -0.626508714491953444540
1.232249388095087328 0.446064048907663379670 0.071955141697017055424 -
0.64658232304272633684 -1.8758387767466517815 0.488862411391992046640
0.51195163700369872295 -0.408922664002080216150 -
0.054564244351317431336 0.490865197083989679430 0.361397697347447310890
-0.54446631789235777621 -1.8957156741947591705 -1.5220685133999098948
-0.29267187577407038912 -0.036695547508950411564
0.69352480423532247844 0.480854332639398862610 -0.221455367562445437630
0.24755062551727907705 0.103462346290294063270 0.379326132962295003990
1.4240103622498312408 0.373698200557684323540 -0.038964132639783308210 -
0.29670438162059975529 -0.145709939774605445880 -
0.406090524199021973840;-0.58643606385701962935 -0.284346076505656408790
-0.32542858699769255937 0.387618662766587851460 0.185284903210227502870
0.093074408777974321216 0.097950277735419005731 -
0.23493757461252573471 -0.954506654283822086880 0.420660210914133247950
0.22869913222433754796 0.498426058140377992170 -0.068938909119945421256
-0.51559947078328338677 -0.039886324945989884427 -
0.94392180746267662084 0.394498442856041908740 0.082136158242715362476
-0.2580891028695139755 0.292342409945783088250 0.636428511562052268150
0.59987091383377111331 0.544511453254976074410 -0.026226858836124942315
1.4315174551558611515 0.467390475819104533880 -1.4668079091601728958
0.50636837501483444868 -1.3279331825197429051 0.578223667999381851780
0.30437130725947564303 0.097848636807690653194 -0.331579957008795334070
0.55162753932701202864 -1.3806486159286475157 -1.1384028803492216575 -
0.67397854319027428183 1.2007950032108558602 -0.749053233284795116910 -
0.20214709114820825220 0.045195543840823082515 -1.1017242585848328051 -
0.63070436174633348614 -0.69754933703248844790 -1.0635443657296415054
0.40251593090190213031 0.675544990717183657660 1.362693163912909399800;-
0.019854327362682354374 -0.17204665400712571 -0.485794514136113519950
0.19541548730237284781 0.524178229511738091870 -0.031466079348436461205
0.69103352410940577233 -0.82593266252271824168 -0.119187502153419097590
0.31835521670772071179 0.734738036289458440460 -0.094628196486314958280
-0.71789885095024830175 0.24600078174271877707 -0.155922716388551430010
-0.64990490673416279055 -0.3415503691785957141 0.117390014342936388480
0.11290906892411821383 0.117227251793720219090 0.039196757522833941545
0.08614338480366469375 -1.0711126739920371342 0.286385239409652436750 -
0.48679981544968015106 1.47684158002630328 -1.2650322125423136832
1.5884029494786877645 -0.060294537425415871412 -1.6508937394419738176
0.77501155836573543123 -0.9660691623815494955 -0.194879037631908114130
-0.30081664537460384157 0.38636749488008792497 0.906424768933531743720
0.99880771387433797681 -0.889028784008233818880 1.0861587227526723343 -
1.4819489747421950021 0.360123222255651465050 0.092825943781901190666 -

1.0658121827700999251 1.0472246442125474619 -0.38368610693875382367
0.106466821914932952 -0.22828857975186855889 -0.398359738504117322222;-
0.40165749690709778852 0.707224632721053292235 -0.708310878089254747360
-0.505167685496434648580 0.820664139787229851560 0.067616106420595137316
-0.356867592533335315610 0.715723535761407902990 0.518436052798052493530
0.84355935246070146150 0.542727819541055866990 0.097394550176802971442 -
0.308824146031942081690 -0.85082058538163063410 -0.101556708752137417330
0.180742948053106422530 0.219556239912242406880 -0.086046005344558446710
-0.255459939270964719870 -0.592991215421096007350 0.147807369325204218670
1.19055256207406356950 -0.195934390107740441510 0.46001567776134366650
0.42239764913404104218 0.10177181194610228354 -1.8173856735440121479 -
0.807532611477622563890 0.364847499919939310950 -0.269508745383451320250
-0.957999593631034596090 -0.398333502000210393220 -1.12284904213762515290
-0.290904841183662221660 0.654492858533397225960 0.739776949762150581870
0.13193985165729754661 0.596530116416129851940 0.435333411650002166660
0.901115085128475601370 0.0072000205947849466731 -0.1592217471499548509
0.90360796694025091735 -0.366779780702577329340 -1.147896726539654022 -
0.13374818397545351623 0.71282807154853111697 -
1.4192688376967479424;0.16620829234333908686 0.36115486679089159328
0.72074811062526655814 -0.043253449362830230507 -0.5356490772150768187
0.948398508507339199620 -0.047869038320387746754 0.364714353212795328040
-0.107552984116364866440 0.384309915321086603690 -0.130121863877046495260
-0.806863249206648869280 0.407180854557391280760 0.191779336395534349970
0.168302632728328682710 0.49819724118092112430 0.247451615362922150430 -
0.166914547254898626430 0.250156002107764841020 -0.178048598531100221720
0.640461178109335782520 0.109134582934938551 0.219413449437142860090 -
0.369272500290968308520 0.721149336849245758960 0.402195851212681643180 -
0.445901701952183038280 0.480164833105380495230 -0.701086847258386325340
0.198508532850508845340 -0.169223254985774340660 -0.549751947209679348030
1.1200267580799236899 -0.039487974465332836838 -1.8654487712228617902
-0.516097115644387316460 1.7075402431299722128 0.611103860733186232680
0.2126274136255789371 1.6592887767255468656 -0.812069118898781194820
1.1723236394841487495 -1.3006918656211836627 1.7194866495366065706 -
0.27877292276139231664 -0.391427791881722053620 0.98328320259292545380 -
1.0707952758243188818;-0.70259647220434928716 -0.1051062169299411958
0.15764166663674597513 -0.011462127571494529557
0.066175675106659767599 -0.14428518340945162279 0.27438570559253472325
0.42078426918872219309 -0.79714606970289636489 -
0.016383791642623685147 0.594062394653258274420 0.2135645745226860126
0.61148134435484369220 -0.617947185180385383240 -0.478830134667051487970
0.38261984794770648177 0.2633451678982419164 -0.834296028775538101030 -
0.069515641438115005624 0.739126192189822450730 -0.297007614053485247130
0.30503787569325446771 0.450420378413401156780 0.345986601747855004870
0.50387505691547651576 1.3738900671005640497 0.080434819892942960307 -
2.0770452234882981202 0.96712461298872620663 -1.1644748038363141873
0.24853838820305218515 0.95317676972113629663 0.53149370645605320540 -
1.2226788827964076845 0.77484407482630990316 -0.050546029843309192575
-1.2294086845822753684 0.57685064918847339666 1.0108842206174835088
0.87392743941015893494 0.22185084748870936844 0.55613663315926975539 -
0.47986381136975692074 0.98367491787515293211 -0.250280835193085859910
0.66199291396384774 1.134464567045164074 -0.380451537335583773770;-
1.1602129355135204491 -0.660035012147374144930 0.510607718437234070490
0.70725388151476942244 -0.109253176470615248620 -0.206224656224420294090
-0.56303370934881913357 -0.065749857073225601023 -
0.29037172803265309806 0.576087860927848360860 -0.4270650933493285728
0.3391071416921291326 -0.023522238038506317059 -0.203144444940601964780
0.03075800014113103803 -0.51584391984714983437 0.247568454649621849660
0.23319219294830739742 1.1699194900752598603 -0.290971321087996892540
0.82607371713745858521 0.28474456634701289515 -0.105330200261591122590
-1.0651781789539473966 -0.95146536002853321357 0.672116605468335959190
1.3706886891336289569 -0.098031348550332228498 1.0989583580398389895
0.42790421599560496402 -0.31810133726879530336 0.49133245339291597986

-0.27401640846594138567 -1.2184939457486589909 0.20204145075930246556
0.36618653276228313453 0.68025359246110461697 -1.4732732423846497394 -
0.30067456488188915031 0.42748646715083971070 -0.57581559987747987783
0.93726011985496560097 -1.2778547542566809359 -0.99679334652159512498
-1.3758034158530447755 -0.41764329488019591441 0.50312980477357416298
0.93330857381192255939;0.31660635285495547508 0.023305337009079447885
-0.21821192930709384927 0.032697340151256068519 -0.45457883070277321780
0.37872906566709735321 -0.30245967957790359515 -0.25966083411043006102
-0.47285126031374746791 -1.1730512982185545834 -0.39624554850160659258
0.42536388851083861695 -0.50643664849065272993 -
0.089303724926686853247 -0.65901624600789421304 -
0.18160597114852794487 0.10144563208821613665 0.33236468181673106681
0.550934006024078049 -0.173728187970121517040 -0.470113027761038970720 -
0.782484543768174667380 0.358483996162758222770 -0.248355435717700606670
-0.072067219147897168541 0.858418130772249599050 -1.752104888993567533
-0.994480918380019951460 -1.7475121297669038434 0.857548954385521589040
0.627966554992748893230 0.506390554293245420680 -1.2889242618007319408 -
0.021190419752218296601 0.887234000415250689820 0.074010288451931111053
0.380576951319594680710 -0.3189954309024796153 -0.324946575302653528630
0.903221936945226455680 0.252554056553434325140 0.537977340575852891470
0.341323502969384462440 -0.1148618081713721345 0.954714490056210673960
1.1487813444188041956 -0.54400962482869985593
0.655364216794702247880;0.12310755314897343486 0.050054252354839956496
-0.018932721990132093948 -0.54118315728195709458 -
0.077451097247291433856 0.201792048257296741110 0.921799584528402160540
-0.519494685062839689490 -0.307714916983186481670 0.376251995298455654830
-0.17008527064320888567 0.430876592649222012190 -0.202906486783531403440
-0.119276980753827976930 0.383496320199302187510 0.210061842973311208030
-0.444056419326436579990 0.171120682596216489020 -0.174208031570093679900
0.349052132841860596190 0.848537204473883477360 -0.296214418405203738780
0.240135801416005045810 0.125129716301214666260 0.554667670158049408440 -
0.070833185938669912529 1.0005674750877835422 -0.921103247675434166910
1.8513162969230270161 0.472706686787744745360 0.69611275707961994819
2.0891249837268031087 -0.100980858467358117210 -1.0740003094255132776 -
1.3120040676117832312 -1.1529749567658704734 0.954227081732254567290
1.0194085091679159216 -0.161009416953655809520 -1.0859107487055261743
0.12956034530334287203 0.898135249310672567620 -0.2076861042257543144 -
0.40932087571105207902 0.283682199981718508040 0.491280417565294724720
0.93686581106988808809 -1.3069447875205368881;-0.233770218353998027180
0.62720598000398719574 -0.40699835371215514757 -
0.014183932391128178246 0.19585246472703035647 0.553530259136742697220 -
0.19843120354649898540 0.015442574090960427269 -0.550302462156247895740
0.38140017303986989416 -0.61821581604301278912 -0.318956442758170588550
-0.13693779330245198222 -0.52824191378640295014 -
0.151294381975266900580 -0.062025296772333871065 -
0.11860094310653206129 0.140580828672671298560 -0.104785539549535441920
0.12183976367419077635 -0.583296006740467332110 -0.698187199986589512870
0.21718195357641284504 -0.610822800120380482450 0.473774879480828470250
-0.62144754958419157198 0.2585456899328634206 1.2633993788054027618
1.134476348324638062 -0.344066578342147821880 0.883929599121818959951 -
1.0191663587463255514 -0.382711101727381375340 -1.0111443787586866794
0.37352086384525778895 -0.3146337909342741268 1.1037428559541571094
1.507001689422855728 0.869226897934584408210 0.216327703356206629830
0.78427610048936879927 -1.200137035697488086 -0.021530384673361303849
-1.3423657011924645666 0.973638833276513238070 -0.450511157391535466130
1.4356512675764587783 -0.476993562284932781650;-1.0649503282816079608
0.053670458920199573538 0.025490197681959855802 0.496901836040828992490
-0.845745730694049346890 0.992809337899398958170 0.29444325970066342890 -
0.812143552285942704750 -0.353010418017541738320 0.401857522282363766040
-0.46519175786580635457 0.051921504461649246875 -
0.29454541981284754204 0.051236214693006487675 -0.758207824920976714010
0.54170136023796833236 -0.2836508590399505203 0.063654648996037260789

0.065062534332281346527 0.33652002589341400984 -0.17896845286090945493
-0.30926825096069332499 0.30986826866941558078 -
0.088835778936052225974 0.80696562494642243291 -0.3599963162499020819
-0.8940674606113619971 -0.31887804435014005522 -0.52593898315612708227
-0.91885604835311440741 0.10068676765780391891 0.4424188867766079003
0.93342393335632156681 -1.238635333016992135 -0.0087841661475498485773
0.77348524433847365422 0.30022776594117189797 -0.097375342903631589841
-0.4782803386016875602 -0.078649401297902030161 1.8544630854074293236
0.39607548005570814009 0.55022518421765531649 -0.46200446816496232305
-1.7418514003326330819 0.20585208128963994323 -0.32206693185695661752
-0.19652948026078329025;0.61834349274521460682 -0.97316594116944032944
0.11967974233814254736 -0.030042987024354029374 0.41047066684685029614
0.19348974392512721665 -0.43134774331913228185 0.33158504476125100213
0.68010804167926108832 -0.05290324366535006928 -0.26073849503307333642
0.069585333115076586363 -0.018849631857745813984
0.31340719073293044072 -0.11449983779836078734 -0.5788039011478612883
0.18981146324926750912 -0.056757218731515818311 -
0.27527182748994794004 0.53224289747993780431 0.036086837720674197483
0.66552805501965683899 0.49180833437380927498 -0.13455824160253840316
1.3821807401821666339 1.3189313960807504955 -0.61978322053719048235
0.073623787114954902799 0.10377778505615516125 0.85361833385488328219
-1.169259012452433355 0.66741449591657031615 -0.94892153265929690686 -
0.51026207217268149119 1.2421910600295307248 0.064384067223687738157
1.1636060578657239084 1.0268507359557845771 0.44565830040864301553
0.49564357702878042433 0.32802375177959675057 1.3223789995267665187 -
1.0620455596417444077 1.7748007143070865244 -0.24047913291042721795 -
0.29201643022298973129 -0.4424847535477478
0.43349571930733354908;0.29252012954448608495 0.34752728652823061672
0.33069676647177614903 0.1066359657063097599 -0.51639024428240232201
0.25012855018393614914 -0.82628531556319595452 -0.22799336282606680326
0.27380380526339404001 0.15366850574113202765 0.95764539277241755855 -
0.46222022339427032422 -0.50627904448651395164 -
0.035220525755929280032 0.057293173866750736478 0.12586512095816171364
-0.45184579494877435968 0.79569577402916480402 -0.38809825600503738974
-0.60159338338173973249 0.27646599847690123131 0.1772412466842776535
0.47166591270837798788 -0.13145919669033320765 -0.1521759048099060041
1.1266363882182923728 0.5928305468437222725 -0.27224124335140331699
0.54422773060971318948 0.072249481496358847199 0.68779647525352871629
-0.52132790843673293413 1.4407468071860780601 -2.1419290520486242713
0.28110390765785470135 -0.12740176500708924601 0.081122692508355576102
-0.15593184584263394821 0.23942491606086707678 0.13146502816199526209
0.27875650259572648526 0.48682617659134869159 -0.30699898681572090897
0.01247150053392475158 1.0270561779157567628 -0.14393495094330016615 -
1.9271951097528929875 1.0783508909360468575;-0.60177031431855232402
0.20777805567496696049 0.47179591251004282526 0.13040428371406098673 -
0.61848063928919683185 -0.34336015365797944998 0.61927193611774367099
-0.58134853228831961935 0.18545546659588363347 0.08931518339030597442
-0.15906019067170742987 0.16845773601819574727 0.23853228151694197412
0.33940164000831785085 -0.52443415090854283989 0.45970291650234740288
-0.31407980255632356181 0.13527838328386582178 -0.28461879452371552368
0.49848775497512365362 0.13945689604341748868 -0.60895325615813222786
0.1118904459068164603 -0.06489671416366918355 -0.57965037094435034604
-0.72505849463847149572 -0.69321948080817308302 -1.7637201409841438693
0.051259925736105091398 0.1341578933611862201 -1.6295828520515582838 -
1.6829110356668122161 -0.561975560853269606 73 0.26438161389235093068
0.26239480692436023546 -0.11882381667888912424 0.4088939677761507796
1.0069493625083798527 0.92134173818277553814 1.026378903033427159 -
1.0025097361216812608 0.48924226414402632868 -0.24190972054795986668
0.27683220357175436499 -1.8812881516277846217 0.50065843772133222789 -
0.8177368053408494708 -0.89116494103871213284;0.66698330879008349559
0.045081961190442054743 -0.22889521040556651266 0.53419119681613813633
-0.074150488294314392812 0.41643748839055089173 -

0.23675302460986344388 0.32660480277940945326 -0.082887491375144034533
0.26897049426824642948 -0.16979447699104166514 0.19454140059527078033
0.65852843026690721651 -0.172027164268945848 0.58542437255396184703
0.079813340188458065105 0.35839818441599807786 0.2384040806613990926 -
0.22281926727146389444 0.26293413010523408735 -0.30801998204398406411
0.60016176493468553765 -0.57883210771041626597 0.44553117560827082411
-0.33285525747636873772 0.79856069363915271886 1.0287700765606972997
0.76594997145572385477 -0.81539669664389469173 -1.4490898470433044576
1.2697200656612188663 -1.0622451051754906448 0.50108689641612447208
0.49902551007223716617 0.51595733057843273794 -0.27300312506137430768
0.27735074438830342736 1.228183257119319638 -0.2425260323127765294
0.99749306796946246578 -1.0956872124505960819 0.21371260105788864037 -
0.10208402351559126009 -1.0187054207694319352 0.16616314171067975147
2.0872059647461500553 0.027884439067290253678
0.50085224459843713518;0.07524467873328112566 0.029016216185828419549
-0.61418783298760915024 0.60178605984978095211 -0.56427751510476553687
-0.33565588071322161046 -0.57277851745936247951 -
0.47967088478236347138 0.34401291863093880208 0.18247578724885676893 -
0.66310558840060718211 -0.24881519611657068602 -
0.059365384616402568085 0.0562774865100515026 0.44092006584919579693
0.096459217084117143681 -0.36250807689522213106 -
0.10946925846884578148 -0.13939190345842408791 0.48740906607025219488
0.62785007647655188823 -0.48094173186961630684 1.2295090419298631002 -
0.42387845240904709065 -0.56845183442078417446 -1.2115747599371839982
0.4729277580575879103 0.86102090595250557747 0.52141791571387663407
1.2793418790196477985 0.037603190264255834108 -0.16175244512656300477
-0.1564412891831417074 0.25947898913686145983 0.52559459247468320875
1.0806100244885730177 0.98327916914674939886 -0.729864450493587835 21
1.267767117904498031 0.05385238889185078176 -1.6838825998387760841 -
1.4498079769906724312 0.022850590536053358437 0.020604274208977584049
0.060580615545928445664 0.59864148892309787531 0.80524426497537859859
1.0217543252364851281;0.33969696604575339016 -0.29197579644568238955 -
1.0755948929385164003 1.3525890138912335381 -0.072145169196188768668 -
0.34410118913835224808 -0.81231885637876455331 0.47917586313426491218
-0.67956858945236764402 0.76256072408726749412 0.37968605788740217299
0.85861881282779983771 0.28547419818354718979 0.41154986922492109436 -
0.53857016046819794486 0.28679072531871513396 -0.11643221479397330986
0.59167461372084528826 0.59812750899344024891 -0.43988389994491094992
-0.44787649799309620091 0.075037360047279733077 -
0.13141286772165192143 -0.071578475811090752634 -
0.59777598901353268257 -0.63047305288406330082 -0.44741177927739739228
-0.30180060010844939056 0.53976253862916245829 1.2138603476907792178
0.5106368282139961412 -0.33936055882902044223 0.81387597557585578123 -
1.2391872052748620359 0.89111966208598580863 -0.92141142941881604145
1.117567112348964331 0.64755541991178400263 -1.1774917652641148091 -
0.49640402695361757202 0.51572847781071917694 -0.22216018637841739025
-0.96702516600869103947 1.3548985350045779086 1.2459244173445245263
0.024804551690740542225 0.53766197841320018114
0.34097684291201746376;0.94813862451680752663 -0.66373453829912565372
-0.46474307183966023782 0.022276898073857791249 0.25822299372933271711
-0.035959214920672041838 0.31363311679123323517
0.051241281103717790935 -0.084195006056851831588 -
0.032692199459089493763 -0.34586470785131828709 0.47018784112292072441
0.42567677062169834157 -0.59395953217062824603 -0.63030349203923752288
1.037719298060344908 0.098664572488028345898 0.019356376895508412006
0.33452869531304718631 -0.099912401310329801429 -
0.54470021540076207778 0.49259097923285111165 0.52217608804823378854 -
0.4288282776984859157 -0.13766230742222010375 -0.2212034892735367797 -
0.99485615875077892944 0.59226597588527285509 0.71660244349527335572 -
0.3129105763916880667 -0.45900380268247625848 1.3345751358047484025 -
1.296604216473136395 0.99397986082346134751 -0.74058147684971487035 -
0.93317911964270994218 0.57021804538915199512 0.2499910390204650168 -

0.17634621845299819354 -0.63483567589521683328 -1.0369765506652868226
0.52853278866892405841 -0.89823228254744202914 0.65299785013422828595
-1.7300195018500121868 -0.30309624101957471920 -1.6705793375107194088
0.032903737950159356318;-1.2986022500273213254 0.50708969356088173797
0.61359827813104983463 -0.84014089916264289570 -0.27847005078249903454
-0.28326015998739212387 -0.36687366232513524134 -
0.015433567173784239202 -0.20568808229915497288 0.32847094270115001491
0.79710384637651987561 0.15073149991341674991 -0.15727302443984339519
-0.52728846136623241936 0.20619051069526952991 0.40607843990514091992
0.64080321026463549217 0.48099855065901298001 0.07370473630133028697 4
-0.3940685325586140153 0.31572462479786417289 -0.31990212707722509577
-0.33941690364483517728 -0.12422187393767185337 0.74371470347235679732
0.13866923839921202477 -0.69355855005219846365 0.5977791758565781377
1.5075790211155390974 1.2330595927835483749 0.41266171344649271724
1.1550521762070367071 -0.74120929835663007523 0.36104704065042625416 -
0.25682971868800180681 0.76646933481828005696 -0.69769306548147358438
-1.3152873794513793015 0.16758820206300362177 1.2133551511143585433 -
0.78302486432673701344 0.65328187538989723748 -1.8505299502041940851 -
0.57897109352154474404 1.0293287266826307569 1.4347555379628877947 -
0.11529107341013873633 -0.28157331399964152796;0.30394434211626619735
0.6269219228332193472 1.242576330438202703 -0.085258900454022720772 -
0.18196258598689088459 1.1424371498350458509 0.20916600802008175619
0.88450162767832107846 0.18117155971724729913 0.51614078990336587083 -
0.088442058525711353401 0.25739706617669388944 1.18741169530321522 -
1.0589901223395215002 -0.067069504958975303954 0.063945872321586119669
0.26149606642339029428 0.47871312860754106078 -0.39968242845401374952
-0.97096901409854186049 1.1414274036512621624 -0.45299471496271975779
-0.81093780024327732914 0.22960368535178451999 1.5496036805863973029
0.73879694734901879549 0.86222914971031650033 -1.0191631144748303228 -
1.1463930171411444991 0.556221478713230022 -1.4953489530701975507 -
0.68775717429215821319 -0.49842943385704685522 -0.24661281897446685196
0.67712483182118110570 0.22289968084938621695 -0.33314422939730098383 -
0.66580882989452427889 0.46946584404785896005 -1.2611379758951355434
0.18978887002828245811 -0.25691823482304582127
0.001340850116925411394 9 0.001128660471282072475
0.36390573023736250136 0.41993090679419403655 1.3826253218523687494
1.196128126533145597;-0.18980042760650986389 -0.27627507934505168841 -
0.41984876839202117660 0.96652049903164216627 -0.22894908883202444438
0.75471973807538228485 -0.05971282828229187295 -0.11411108645496501057
0.54417263585310804430 -0.63351248563265682989 0.10496526645097868646 -
0.36392472399529018912 -0.16755025185695834877 -0.23338517932792668397
-0.66937358275500014049 -0.4071413728561244505 -
0.008764431747117513299 4 0.089763436052357037376 -
0.27842582585279179730 0.15971821839425132117 0.41385721183185048 -
0.60700601428358502698 -0.24694055530664932308 -
0.016143173912903180378 0.12068082966907098252 0.47277761392589601197
0.90911962681959579413 0.20714211157853626433 0.34287759370837905815
1.2073732356407329558 0.72201984779808636716 -0.64633319793464516501 -
0.84332979920796447004 -0.50713650215989258818 -0.90023002382306971381
-0.99029963848847746277 -1.2769837560382331088 0.72128569290116206769
0.62138085339601023538 0.82967254587243954855 1.187934851581207818
0.88558943714968352268 0.24578027448991457415 -0.239395306268639807 -
0.97894440879975630487 -1.1069601166613267651 -1.9267188963065109419 -
1.1210730953759782125;0.99087529848603994598 -0.038842834836158218748
-0.19105752238909889940 -0.23424661282621270231 -
0.098109589784530271128 -0.93840206467232445409 0.91085674683593809142
0.48458918360428859806 -0.58622873729868174575 0.21097583565250066506
-0.16813591227237195680 -0.084358809203193885207 -
0.32314563742036511718 0.004471778162979514470 4 -
0.012341503804932662386 0.73628577672408079824 0.30409905791923941276
0.30033733717461685853 -0.36238095726105934435 -0.39342375816307167957
-0.34510819783649770987 -0.927485031133837734 -0.54773576165892645662

0.97545118286777121597 0.49384398441264126811 1.2889033652029824317
0.82605314157473996861 0.66952043180078568696 -0.25537335628770529983
-0.047203225018450242767 0.79650925933759808562 0.49942483554971517368
-0.63570020008496186392 0.10664135101087626745 2.2917772200670767369 -
1.1084713806583628148 -0.32179575402239640747 0.56046735991615537209
0.16640169807475693742 0.73235313022104420178 0.21929958206806626775 -
0.78837226876889265892 0.8174112811509105514 0.41154827911963626219 -
0.2294214387328977478 -1.0327382421508213284 -0.17846321215858007569
0.21014244853282162961;0.34562225756922193565 -0.24755591718405839385
-0.24565488566509793844 0.15196445275321010526 -
0.048635638284349781701 -0.30615400638458800664 0.28940511672197827275
-0.27100731109390241436 0.69325483355787165252 0.60736004340439919691
1.0015057807259499789 -0.20256402908206375124 -0.91718963047964163771
0.13094539102402980024 -0.21220970230401600687 -0.4974407884756441578
-0.11616156012696117472 0.39137368608066169795 -0.63201499756816281739
-0.75331525444077640508 0.060633781604070680071 0.3402445586662053989
0.44237417196192607705 -0.4216383991500848305 -0.66281534261367103245
-0.6054633086358478522 0.93732404011238862313 2.0667707791611036328 -
0.13675661448107123697 0.80354848103474474375 -1.3639876356204148067
0.62949610601904082419 -0.850083406091367344 -1.519227361023663514 -
0.23356932563102689171 0.86858715737681335689 -1.2990297823546648193
0.13788282229703152648 -0.20149937893852742921 -0.51391811580501500778
-0.00112302779685904911990.57493460599067480743 0.90289146251907248519
-1.4603445126501692908 0.11467679915974962612 0.27794844776509058937
1.2779593892521474707 -0.62025801895574161904;0.29246241919233284534
0.28519263216981183273 0.13054216772180760042 -0.21612709704654647913
-0.23107921082868981832 0.078307870731661469166 0.90606508383753459057
-0.21697360576957441802 0.25502465895284848196 -0.73937270299772250848
-0.51476888469976289642 0.24198557373724544073 -0.56760641281051171791
0.08123611949724038328 0.29458948479447577551 -0.25230838492844542831
0.30191114180841499559 -0.027324622524902718312 -
0.48887339164208443165 -0.7747713864436706066 0.46728356504098966795
0.01360460008403276827 -0.071424905425090084998 0.59243276710623371528
-0.85302640673055751819 -0.64469827996793449554 0.40949431196929347854
0.52924134118397636595 -0.81581945177667025693 -0.1045934887673213437
-0.14813217613198892408 1.2813976438567464822 -0.3825420136053473974 -
0.98911014019172915912 0.7849402663744756703 0.31820857436877281188
0.40098377664282086608 0.70208209486318795545 0.9515173792394004959
0.37164117605762514444 0.87472367466121259127 0.37940573424400869662 -
1.7624204269327312655 -1.3003694197029480595 -0.98458321785328883902
0.99014908791602318505 -1.2366682844833394526
0.761004253782797346340;0.012371952345642766441 -0.37618255091611380037
-0.17793559031815109339 0.12339929464167824491 0.47378178507220364279
-0.15420636322414771335 -0.31560846526342672957 0.54586370377216253402
0.28078401928925822428 -0.086045022018419445309 0.48051416144910308326
1.4110474157730321654 -0.73228922757653003295 0.37515661340756617514
0.10553650007850051695 0.30400810967003855101 0.35844383203322660592 -
0.016927166484212413655 0.089465765503967883965 -
0.24451161061590739876 0.21362225575422916224 0.10289392304111757981
0.76174971036557659332 0.012603038244537545162 -0.779320294126054413650
0.61504558914280704762 1.4639908419846159315 1.0820396297297556476 -
0.72733928464502972666 0.61825431172716716066 1.151349468327840686
1.1140957094382835102 0.85356627927093275954 0.55216586207857265567 -
1.0147203497944943251 -0.35138803574592453183 -0.69738527648407300231
1.4051150449192757463 1.0177706211948707171 0.14565369260802216167
1.2789619926489232959 -1.7268529509632912777 0.21762982177124079453
0.99870058613902346067 -1.0430869458357698853 1.2481577007106863242
0.80611806611419767332 0.31665631000248101179;-0.61954875577795787134
-0.47749021725711909836 -0.030631163495359544496 -
0.28057762317867163127 0.26787962864986269995 0.34061735506663770590
0.72705844393635765677 -0.21722100002798272467 -0.58703803257809272775
-0.24880924020293970766 -0.26690372118525063083 -

0.34421472147593451529 -0.5338477610614690283 -0.11121068380934778785
-0.45686060020251711977 0.60417349733395986355 0.63106878895860374357
0.2909995406637146953 -0.92638045329285434448 0.086451687695445436632
0.34603413695132956285 -0.68363375995859976264 0.74270437925045651451
0.45058234973596078543 0.35634601735998977512 -1.7780637306193809444
1.0254116095497305405 0.36138685529091069126 -0.097946519741266252 49 -
0.60104617737915899678 0.58003174299176341044 0.30041752277618716249 -
0.60808600696433767752 1.3357187926573355075 0.57500061326983908927 -
0.12840617285151387228 1.2212918150165390507 -1.536221675823409516 -
0.061623989561173452001 0.87627173989276918586 0.48571598948662408901
-0.28009575463668517914 -1.1408317463085393406 0.92923901842274203577
-0.32308341089704362137 -0.67926482528706566555 -
0.41911007835638830477 -0.76850656654210680596;0.21315467687421130183
-0.58990048972694242302 0.75761832590911903917 -1.219329481574149554
0.40854791351883401607 0.67171460667798088551 0.060691381431576214844
0.005122283846935631 9138 -0.32452872905480473209 -
0.9108821273804249552 0.03079779875196752767 -0.046495454280980327844
0.22170502846553405418 -0.63305078108990420205 -0.23129758049462495473
-0.33657585981672533126 0.35296552793641050583 0.078908186903550128766
0.66522035291105274801 0.4319033973277986993 0.56452938210496250715
0.21798220552890046342 -0.19870812005367388675 -
0.005137943481557898 4425 0.11231044735033436643
0.062498281481972020601 0.6088882203963669193 -0.56451257907824980986
1.398443027677055861 0.23905821075831509837 0.93800590731182542648 -
0.8451066285620539853 -0.40884895911527774359 2.1500453651957238677
0.029139620527578485865 0.80432666113791806417 0.42266916429402096034
0.33103354185776145968 -0.47966634976084920883 -1.5941044454138224573
0.27776465362024371775 1.2261142829205760041 1.9794051084912105409 -
0.70477344269037178748 -0.2477802242250894682 0.93672547246549064948
0.1007191920931699014 0.032374808563860289767;0.10644826124631046049 -
0.43444643273766142544 0.093938451980376105888 -0.56497994010132435161
-0.85583580055482055826 -0.43386671659439079995 0.154051232927178927 13
0.64888083544186914153 -0.29187022637999449071 0.72324508663023379107
0.32210193264947595271 -0.4484645706704463719 0.53253217023642784866
0.78465166553796450444 0.84818118949960286113 -0.35293599631957395246
-0.21758565689637421325 -0.040288714703763328639 -
0.14682303709255267732 -0.011250476892280999661 -
0.43398807607397954511 0.28868344992139399752 -0.10768634579989655453
0.50826869428044052501 -1.9584729941776730655 0.79454515713089102213 -
0.21523513012910780451 -0.67207661096955739755 -1.6883068555354312501
-0.34565987301997197489 0.30573205298231181937 -0.15802576920469219024
-0.88560835424453299147 0.20434419729869460203 -1.2817233730589678853
0.28107659520204703041 -0.52517333053882830907 -0.11346836723753053744
-1.2441660266203200536 -0.23517380864097769955 -0.4651851017132552224
1.7184871793320648514 -1.0324135588054859092 -0.45723873762154632061
0.79069711846671875133 0.84491756802519613068 1.0545923202062650326 -
0.66209408145293568992;-0.66084836997199392084 -0.42802469656727276659
0.41666457004374590767 -0.67847761295629138711 0.51899203214554689989
-0.16021811073870967368 0.12290266918933522022 -0.40628380852904921738
-0.49657139306449560312 -0.13887467032689096436 -
0.26421912414045300999 -0.19111177968423401219 0.21039681714578950533
-0.69252442808750536152 0.24196532981901155979 -
0.040586628706407834599 -0.91047605197605352423
0.027341355521682292329 -0.054549192832007695197 -
0.67822659643915084171 -0.2982780202233025757 0.25077678202396014306
0.41613917008620437965 -0.04641092980233028 2146 0.17344284152960523504
0.37629230137155827984 -0.04558330624389146656 9 0.12753485596460378826
0.12167914719523696687 1.0216109968745081815 0.51813916600272724722
0.07749960475520527 3032 -0.19470509573425778815 0.21315820180449474019
0.42354521408996370635 -0.58928465422977094335 -2.1855713351034626868
-1.0464169546548489631 2.160908898338366857 -0.44757537573870381786
0.3149729907950517549 0.14432299934087194915 0.52290645598214735568

0.63015383728541307651 -0.43689367997300659896 -1.0681918406968806678
0.27360545677209097137 0.72765477860948102329;0.70531333810818530594
1.0426380875480840693 0.12798425673772226285 -0.45013638650610193759 -
0.16633029937928242581 -0.33194088473795640981 -0.14790886551559342954
0.49621302463784106029 -0.83578112750370459061 0.023716598440216035693
-0.11344937232431001606 0.017819286951853283846 -1.3783181147758254426
0.79650365188503591263 0.29952259101560813059 1.0514802434476004933
0.71655304761842997241 -0.28331446785035796498 0.40375787426603143571
0.63964861549599627555 0.059606858327070411097 -0.47491182145033222772
-0.52500229986807833704 -0.3169771372412163446 1.3428550763618853914 -
0.26265315787541937409 1.1338736848586734585 -0.5000991992905209349
0.010552442434588887865 -0.46804226888626132563 0.10995633219339280562
-1.5282661449787966479 -0.074510550850409085211 -
0.36292482804015174791 0.57748139231748030209 -1.3303700490591152672
0.029506478694102028504 -1.3960181279761467721 -0.20496473555320007809
0.097106885363362330232 -0.38186325871440590651 0.35232951544562052248
0.12775155522769054972 -1.4529728441524742966 -0.53883938845486190061
-1.1491111028192977361 -0.68469369206665720018
1.2368820728373375317;0.31030947795358998498 -0.27564308335477483158
0.26281270251288690876 0.34586319986538349713 0.28116842009805814717
0.18581908826652271394 0.058084497394567384532 0.25497871707947716846
0.35847767576801747857 0.58884283375565782404 0.33499920122476439799
0.53349465030182552372 -0.65443391747050327911 -0.24994261224832067803
-1.040711203607857626 7 0.45980213652909185384 0.012611241604373482281
0.20787295904547853365 -0.45923780342535736043 0.15105285761334180661
0.41188117365758780686 -0.50655177885391777792 -0.41059062822889735855
0.30161995274322889182 -1.3412730354300075319 0.72324860865449747127 -
0.15842908572816349899 -0.11833961319314090233 -0.61815858614493346401
1.4772330405409856979 -0.21891444140310661881 -0.045800058269656669652
-0.10417145363733523733 0.63610605437226108716 0.32551081793347508286
2.2466999296861582813 0.1386710665232158135 0.1348220128666689499
0.21091066343477671285 1.2300884088925405635 0.8825905374976634743 -
1.4713539801683555019 -0.5512021298295022298 -0.38906072525032986409
0.14236628589485295082 0.21640972211898790167 -0.74447331647536085786
-0.99700615551680848991;0.91041331621253107631 -
0.061855145995757553068 -0.57939628601956039144 -0.2218077069627583664
0.23430416051244237763 0.52339785879813538472 0.26480510180589111879
0.31101329423913742778 -0.063649375261838864182 0.41054861545779208765
0.65957967451500687517 -0.51538455449499109839 0.5116607614817394456 -
0.16759606600057466719 -0.54540803958951578068 -
0.024642497976170074647 0.22598942712376096642 -0.32273213085090035612
-0.82462620382808016206 0.29910110575283715972 0.78072858188806193436
0.22812978598558930621 0.087305829749372163007 -0.41931081985839863124
-0.34720724419750526035 0.13952199648048946301 -0.6887947093083859329
-0.3096104677315043574 0.25731934909096620956 0.37383796601819063721
1.7894050786379509521 -0.860934969272540429 05 -1.8231993630960103836 -
0.30602354877842841141 -1.448560342814005919 0.63187445845218936924
0.34007470683060153638 -0.21705988811586834908 1.2352902832204528671
0.31659165197268118730 -0.81398338017763460783 -0.94823225132102784141
-0.096607779128592724538 0.72487129826030871627 1.206509451585047632 -
1.2847897720430685453 -0.88168892614515126382
0.13434780128648582398;0.47513111682967668070 -0.36623594043483254090 -
0.10254884920532571224 -0.11189882431440789645 0.27599573212221473195
-0.62893372831451344052 -0.73051858207122466471 0.35399395339852846032
-0.35400815761791226688 0.48622234502981936943 0.079821600881212595446
0.38094248838708616445 0.86409489152482832353 -0.50933652060808931594
-0.015590902790666702482 0.083389597053007552541
0.033780298411323822494 -0.19935419882966243232
0.027201631190954991824 -0.10091393818047457442 -
0.88101455210504386883 -0.34727776932195802972 -0.17586051916751382906
0.12661873649550497989 0.49549613936897057842 0.26873782606663632055 -
0.44576268350573128307 1.7897776413287853625 1.4603638598802417548

```
0.52058419589556415019 0.67719688026980284778 -0.020120986987290513659
-0.19672706152936003665 1.5221589043620216142 -0.21975700448056542657
2.0125288001721171049 -0.85162385520708983044 1.8281002193655717658 -
0.56502626226944308O3 -0.26970933384399337074 -0.32462038988769925663
-0.24220712944830224966 -1.1673392553424721818 -
0.063147166665049436762 -0.54683999523061987968 0.43818465674214213124
0.004833514544768130472 -
0.049475009016355354918;0.83265646761350620153 0.36670087240146037599
-0.14556787691509440186 0.18975370823988277347 -0.26227366078909175595
-0.55919902998585846454 0.83119017381947324363 -
0.005881301470435537984l 0.048074825913906535368 0.5609599004214573803
-0.47152087632022132713 0.98069778541614349709 -0.56014296165113286463
-0.41745836074737513455 1.1783440807359093494 -0.30927655023891159614
-0.28907033848253671504 -0.4035134546660829602 -0.84243904730326468755
-0.68345238690817822214 -0.15510868570285921142 -
0.008658027500462779288A -0.11102697847270978637
0.34551869812792390668 1.9867241084533502349 -0.29769223616980461378 -
0.16502154220677903163 0.35926957983085799908 -0.69233116020678409086
-1.2613425708017966809 0.026619666888981281111 0.32840995610717288455
0.14378871881993518511 1.0192776492406898647 -0.37494398100126069551
0.052891452827463714126 0.42330027716660806325 -0.42480229505223948694
-0.21941827285409737369 0.15981802992416474507 -
0.089879044884538086824 -1.0871014415920814411 0.90971631837506705676
-0.47494930059459988803 1.0214101817397411853 -0.020609256710689038278
-1.4622073911335498053 -0.4200507050884053073;-0.65849511619738820833
0.23042942404675678669 0.078044729058332276694 -0.76050831829138720863
-0.79801238274102326287 0.153286709920649688591 0.58585778134512156434
-0.35485510484316556967 -1.0733919899713546542 -0.14139760618714419804
-0.50171547923798442969 -0.50042955210912021258
0.049139959587882114933 1.0138716444574289444 -0.75327902646441080581
-0.627005257486768941 -0.81403545473066352134 -0.34207092630649699183
-0.17168771816107358807 0.67724079055132924143 -0.49514467472695899986
0.662328270223105586291 1.441965215102334108 0.00400758089648078214 -
0.26344241080709690506 -0.069991061665052922702 1.9686003471030739664
0.54462211152382200563 -0.97766204318199800838 0.86214322270618803934
0.83124358695766153993 -0.80835975107310376053 -0.91132816459190357783
-0.26737810494718472487 1.4176832561365975138 -0.12738714079794719036
0.23580146688711783876 0.49532929490878231279 0.28170005533102426831 -
0.21019069161683029412 0.64676602559376061841 -0.10530448440629058826
-0.69466305462319621888 0.66374612535253196821 -0.24095729018144385059
-0.71980699428737160428 0.85568015065383029594
0.21252663098705124045;-0.28922118774312693956 0.15175612091875040344
-0.79611343670697476416 -0.25081975521337235291 0.38535158087967896767
0.29627608235775504397 0.74498321342240159115 0.082221059653209041529
0.096809963498459078446 -0.071523754870304045994 -
0.45381962567646877416 0.59256304402455450083 0.38522460072059122593 -
0.36350848138052699321 -0.068480355371113799112 0.46019345652774135136
-0.8458809906014047586 -0.011241808185405009757 0.45293611484290419167
-0.44709973971483985 0.34598934796266012093 -0.24767556599799500638
0.65624383382171613821 0.54595273572796687578 0.57241311761351765242
0.93006613280014782674 -0.37790031948142177587 1.3306093034464288039 -
0.37321450555576363151 0.22733297321815501557 -0.75019584173140496652
-2.0954774633032884346 0.077676876646361664447 0.76089839266069725365
-0.20498393937987027802 0.90405501807560673111 -0.93304707210742787726
1.2202089326792067148 -0.85153522642116952213 -0.053971591755666167844
0.71562824996814311529 0.44170630296959012551 -0.78586765076995201973
-0.2688857715431853812 0.48573177741051037914 0.21914256970392403945 -
0.52518760427481825381 2.7875365761639545603];

% Layer 2
b2 = [-1.5216632980917248474;-4.1929980881676094384;-
6.5286988193700157268;-1.4689731616075560883;4.6409873439001092521;-
```

```
7.48070491414751259 64;-0.864809064799270998 95;0.541412812976076018 86;-
0.367980693749826559 55;3.920239728173226634 3;2.396184517316897633 1;-
5.668879294148239544 9;-4.235992402034925774 5;-1.885754527063801422 3;-
4.375025952887742875;-4.560550071619789314 1;-4.830035682961364429 4;-
3.686719539954933289;9.473761850515936444 6;1.131292481760298063 2;-
1.009623561570459404 6;3.174979173975652546 6;1.143147568629018451 2;-
3.552152082600638305 4];
LW2_1 = [3.3607077038435146932 0.027017660931308279393 -
0.84955516359954230676 3.5740311953296521885 2.2801077263765088254
0.79456694674414352075 -2.7666947556895596037 -0.679592964230125717 86
0.81001113425263282508 0.19407935574741969065 -3.1600332192822957467
3.4744516203024380374 -2.4518715649248803423 2.7743954971234581208 -
2.4884512801903704293 3.1419201904687645843 -0.604143222312145740668
2.1515665502223249028 2.3180261512849429373 1.9724534974611374061 -
4.6789765381840897618 -0.31563993271963924903 1.6308688977233025685
0.71741636690960941536 -0.43170437953881501958 4.1843445336198712425 -
0.91597417676818249088 -0.90975465523383602928 -1.9764697551988839042
-2.7581441779880497833 -0.35877859905451342515 -1.5228227267967280323
2.4551314352204638425 3.7241891996647940033 -0.184965728375853344 37 -
0.46882771824667351934 -1.6497059613874913708 -1.334143989413616227 -
2.0499563133325784392 -0.45684830878053184211 -1.5834406613957217047 -
4.6199198523960838259 -0.043563452819331272003 2.5370736458054685691 -
3.1310257111058001911 -0.86444346825450035787 0.7885400394551951297 5
5.105969615925809002 -1.1825439932457255665
0.95254571547135391544 4.2344694116700933861 -0.055164890373306976756 -
2.7836190542297369888 0.34273191594281610106 -3.9287441197570704965
0.15961512247955256254 1.2348430879794847215 1.1787412720134355926 -
0.53853219557766163206 -0.5245754383458595127 1.7186650180308578051 -
2.4010033207720797854 -1.4286601858445397983 -1.3936231142288868501 -
0.6289976307888186513 0.32469327696627836577 4.7572799231763260863
1.3284425484627222502 0.30211960101128260137 5.0279336367709683842
5.1538168346282562382 0.88731200117778019631 -1.5888696172752869717 -
1.7768868061503981792 -1.5680095712686883402 4.7273851789011400726
2.3071239002605956614 -1.7552391977017838265 1.5715237788703702027 -
1.9309673540191223395 -3.1577020025559483507 -0.5766333429056910153 7
1.5731306227075521065 1.4732186066079198117 1.6437972815259818127
1.7958648474433769326 -1.8257346329604491331 -2.7001966869167701724
2.2494011180262178939 -3.9273417072756733326 0.7914219220326312864 7
1.4242382546219092276 0.87111454150630529547 -0.687920717888977240057
1.5237808930797971385 -1.4631360632766323704 0.055793402191383317801 -
0.030022806124911503856 -0.99240978148907421641
2.6551078125967464416;0.872889708468067127 2.6796027888775904024 -
0.91834340858980545441 3.4406136691033704089 -0.485150999058823140247
1.819421265139037347 -1.9838682838743939563 -1.8881749313377027555
5.1347646659755161735 -3.1305047624171771226 0.109268531304959712347 -
1.0025178120739552945 -2.487317381759010626 0.163392757095417118237
1.6794265620203077116 -3.5833357698658550028 -1.8247842781649648014 -
4.7250716847038676249 0.024374345105575984105 -0.734254557699856702517
3.3485932940563722937 -5.0848859580517782319 2.7609273976688846197 -
0.48966196442673443068 -3.7934823196603799644 -2.0565094195602031668
1.0051326418709030097 -1.3277696828843250643 2.3977156593325612555 -
0.42898911917306870034 -1.2567643076831440663 -1.1959996895903453051 -
2.5292265318589879364 2.5124658885130783936 1.7809748563561069723
1.8946335558343290195 2.5587619354848403219 1.1148725422058480827
4.0682126318465314796 3.4554038123068000488 1.3489954105792549832
0.33129060792590431239 0.63782109007117726573 1.7013719643529467973
0.27807844527127650869 1.5684949985825309415 -0.27513723378623472504 -
0.65453474322879423575 4.4850760032225709395 -
0.12276385886252638713;1.9883351062279801535 0.63813037554674556517 -
0.90215821270565088597 -3.1975366156680662399 2.8595032356523044648
2.5487350149729266491 -3.2898978083478214174 -0.50961043755484336515 -
4.5820831957175842675 0.92559966351187761288 2.5704916764496510417
```

3.3974051504493729681 -3.1374263284725798506 -1.6807543451406035118 -
1.15776625021522106 0.76519683879795197257 4.511775343088428869 -
1.4958395475416532694 3.1669070570779709328 -4.7022927798248472797
2.6062647218380097236 -2.2766816894960748563 -3.3329593423964460897
2.3244935394371255555 0.62111719568595258334 -0.10840779115565746249 -
0.76046632886072174085 -4.5298561701255284362 1.6878423539528208952
2.4302458717626040929 -2.0945400234740101375 1.0854706526728914451
1.9590939968054068565 -2.6433680888300914447 0.44435278440030911984
1.8915772196392568461 5.2434458129412488958 0.67250057673177354722
0.86644004563224708004 0.18972779743296439015 -0.56261907867953686591
-1.0039476079334352931 2.1445597890110414951 -0.037055473718408282746
0.07961169356509975592 -3.3860562267272955594 1.9829183662969598334 -
0.65156526576575957144 1.7861584529708176561 2.330772925007270846;-
3.2354283537735146759 0.73508086125989091641 -1.8064143984650824493
1.9331579478390761295 -1.7818469624448183275 1.1296344447553572365
1.4014888941489795915 -1.3824781801579852925 -0.31464319187471156969 -
1.1866175726247052413 0.71659240596786377697 -3.5193145197728088469 -
0.91696794117464575091 -2.1829459570220772768 0.21446029956920642512 -
3.6485656182074470699 -0.64880890958316872474 0.54624962668233356666 -
0.92560932300154719066 2.5798724923635214523 2.6109487469217977917 -
3.7226831451552575558 4.6301960958964762938 4.5518248582399216673 -
1.3528162043088269861 0.26809303706888309993 -0.33956697388420464812 -
2.48101705897720759 -1.7418209582811765568 1.3730426341881984964
2.0970097134083123258 1.8534896883840141779 2.8734825741218683604 -
2.8484561417482607837 0.26961706164078169667 -1.0115562963910817995 -
1.4330714237361008578 -2.3155308871319157937 -1.8441537039455089264 -
2.1719896445107780281 3.2441255221173581624 -3.448226407555444073 -
0.82400379670198664606 1.4456405405209773463 -1.47907933733683250003
0.002548054017083328792 7 5.0062092068284504975 -2.9455119634000737072
-3.3587598586580620363 -1.3433113107304666123;-0.44947535701940177777
-4.3543645999279396719 -3.1101807801430116918 -0.52647454665603354851
0.62281918835079408403 1.7255861372489362804 2.3921380214251199 98
3.6631005625896624345 2.5459821217141458405 1.2926135078923937094
2.5894292491233383835 5.4646588325607892145 -1.5588572072650968003
0.19086171106948585097 2.2533181014139538334 2.6591916571790576995 -
6.5380206847582709173 -1.6092200755533869749 1.0886066375970073761 -
3.6190796341033268924 3.4975985182125430839 1.9553739705917048131
1.3557999459772243078 -2.7309452541604031239 -2.1702275124672549822
1.2928744080425249585 -0.79302853879653145697 1.3803331406968815109 -
3.5069702305267629328 2.8466008448334267911 4.355680773989270449 -
1.5164211038617205762 2.5604165847119486621 1.3520231258120281037
2.524052502266475706 -0.65377590778087479872 0.671870456008759942 -
0.37942070293845325812 -0.4941924420182531863 -1.6512531158910714879 -
0.87266740155637811949 0.53992374724017877519 0.91709372788364196261 -
1.3456657468384034537 2.6356560551085950195 0.001965607557630158 4862
1.0278416293561596362 -2.128299941650972027 1.8090005014836583186
0.13889632860912393131;-1.071782267539120 4257 -3.900373697457509 6141 -
0.31208963420204283157 -0.42741048886387872363 0.70979006474796191295
-2.095134710187792404 -4.1419285408171573337 1.7845465780971989656 -
1.1381169922052394305 0.71820449542930486775 -4.5810517366450493881 -
0.61922495168603730598 4.4461135845849053538 1.4533111958704594269
1.8523561592491084582 2.0822216171742220503 2.4074163962232839609 -
3.2564268464535266467 -0.78224166999182465077 1.5944056796684660249
0.28825555918823220214 2.4074153030461440395 1.5213427512927704122
2.6822311365809432537 0.14594285544736246285 -3.2863742606569199545
1.1648149179744036186 -4.0911046721821673344 4.1683798623932251104 -
0.63208043539650782261 1.0648030975120119646 -3.1613143320433989558
2.9876116324148349612 -3.2262822768294268094 2.3573486250081208127
1.1516220819151838928 -4.3123267813132990511 0.65080969761031004506
2.8867456345283000196 1.8020354747665487949 2.5947518012955628208
1.5966748510487311918 -0.80413985105719176794 0.011956755459867907 11
0.078083468506440753543 3.8824249474472694033 0.99247996019928319544 -

0.24836614513149762873 1.0173347768954117409 -
2.9166645767391710109;2.5409746899200289505 -0.8216962608476888752 -
1.6584977300215295593 3.8866425727568159054 -0.53947266255610881736
1.7686133713843448234 1.451513821246719127 -0.21521589885526029584 -
1.6381896665311845851 -1.2828045176493618751 -1.7907994758741265784 -
1.4818839256469162891 -1.40843824427751784479 1.6594916893043956119 -
1.8669841532786042304 0.97219777198881873748 -1.71638179985833216 89 -
1.9022434652195752136 -1.9770669623274039761 1.33603770320167192 23
0.56087437382375193717 2.8859470311734400916 4.6837611878484803896 -
3.0078530704308437116 0.81468871217445526511 1.3920901044470448404 -
0.38603724294270536088 -4.4044991458154818531 -1.33172945289275102
0.54268587571571047512 -1.0174241433306563387 2.6262841093886972743
4.1337706389403985341 -2.6134035597102616144 -0.96374239453181886894
0.14419199000775700936 0.873739625575745026 96 5.27005214185874759 64
3.3329432917960639671 1.1609448390025134401 -2.4267659142096378311 -
1.5057171323065172697 0.48789713184500610454 -4.1696144386265583393
0.68321099052842393373 -2.3087558179751299647 -0.3078609688222398999
1.9148224575706831452 -1.7345790130453062616 -
4.4517856503422184034;0.92428358699379009611 2.2166488951637823845 -
2.7022816035726511252 1.069083776382804496 3.201529981936744651
3.2239833422234807081 0.52467417872681521018 1.6780798357767801043
0.91225807644738732272 -0.86249227614602164138 0.30981098999352418844
2.2652722978467947357 0.62880042170875849461 0.53231522471797076967 -
5.9391529374391565099 0.20765726669137377081 -0.57175420275032706741 -
3.2699198962684303815 3.3478237282536822406 1.97134980377174451 33
2.3555038108609771541 -2.5955253945681295846 1.3812259436701284088 -
0.14725606220161990834 4.7368500585354267329 -2.0323934351921946195
4.2461847794798446287 -0.13362892684505631169 0.90992029005058450153 -
1.2730869477957602598 4.0693281681370327973 -2.4012802273236282247 -
1.987984957013530618 -0.96361365585348635054 -1.9962513138036304117 -
0.22125726760309991081 -1.8793704149262357994 -3.8639147175270918133
1.6563119395266767686 -1.949911961952877526 -0.55742616648681164726 -
2.1767041427006939003 0.27618030339912591486 0.67502678424358308895
0.33934204566215847576 -4.9918133844624694717 0.65482454652519805904
1.1237226521174676908 -3.1629102168494220848 0.85671659890691620465;-
2.2426651948586875385 0.84102655449492669248 -4.8405230481928560593
0.11098395850832629139 0.069895489827022413976 0.50958965350736540323
1.8803251729741090603 0.858808993991496683 05 2.5324251988689443493 -
0.87589836270742615909 -0.42876519469805590701 -1.6377024513604998557
2.0987609208461468491 2.678998847416335316 -3.7386497402523493072
0.29615784062209793426 -0.48332138954266895547 -0.59263542230793775367
0.011367382918659888272 -4.1137861724911752859 -3.0194518546004056603
-0.44625686023195149099 -1.8406076165350246931 -2.4272684098205954406
-3.1811054941315597411 0.55044209597943738466 -3.3129540890145752208
0.007186048967544322874 1 0.78570514592722073832 -
0.45199901204868764282 -2.3471575228842400662 1.9366185678105858337
1.0190168417082234864 -1.1652399541319342635 -0.71436162866563646112
0.68489147322593479927 -2.1742859984229623294 -2.4302510509827386365
1.129826365528636023 4.3752818043434684725 4.8311339734693534709
0.48061890622304409559 1.668019893059005776 -2.4659455220220598548
1.4260521110856678728 -1.2382603884278173823 3.6675962052172041972
3.9631677371036690971 -0.33652026639894783155 0.95626332322086958 35;-
0.20697510775692476437 -2.9907398554851258865 -3.2900189633589449478
1.1745600516038789962 0.65676817539692899928 -1.0530363247426388629
0.66150814761891907345 1.392582183887642211 -2.1645072288205979838 -
5.0334286843698929914 -3.5882663599564343926 4.4292142331554709145 -
1.7714282044649947157 -0.54167575476833307313 0.024541187893441257506
-3.4601536759042894609 0.48931673681037146 65 1.2196457200260055753 -
3.8452906201016991972 2.3597118489869344238 1.5170972008466754311
2.8616468949904647623 -3.435104267131621647 0.58225202494342553905
0.13864487863287056069 3.2728235397736162859 1.2685838418556336649
1.6078413196813967634 0.94051387442940903583 1.5610973693306267052

2.2881012916723548756 -2.2942268910167280538 -0.58425025433766852867
0.13617518465254846438 -2.0170479382518160527 4.3315794502880873296 -
0.26483429422873305814 1.55437242160425825 87 -2.5849829008777946981
0.67898556481343297353 0.556314722595444536554 -1.6513362605164139119
1.5131466679668867936 1.031112530954481965 0.3854925849995434195 -
3.2833711512787195907 -0.991265535727117730 32 -1.4397587115640608779
3.3826822546393713331 -0.90811030640004786196;6.0177737386024823252
0.62657488030032770787 -0.59728385076960988087 2.1762204416549404407
2.6929311637396535772 2.5161567553446415602 -1.9868898350289578847 -
1.7827554941425201029 2.6602205984377431669 1.4319259310832148646 -
4.232218768479641291 -1.3701874478536815172 2.6412405615316578711
0.80592818645924424459 -0.20072216162234524095 -1.5173613218955301907
3.207374256346094743 1.6603164373872296711 -0.85312845134676085213
2.3312559061995261622 2.0068869894994922909 -2.2626735686531493918 -
2.0528158583820905747 -1.2073641323274699744 1.6737921047650656803 -
0.14890933942959286296 0.73874536571411197894 -0.50520845862903762669
-0.12062519211708824796 3.1030156314015879815 -3.0235767835613653887 -
3.0441833207605020206 1.52633667357399605 -0.77075389719983855574 -
2.5217722819348478147 -2.3609124924237279153 1.428358193274198662
0.002696859079086 4832574 0.24792296149214759771 -
0.06161557736454304629 1.8077754730666584226 -0.65680335681795554148 -
2.4665410796119466852 -3.477966234231954612 5.1617475463361870425
1.6271947544871432356 4.4084519264135808569 -1.2521668579069753502
1.0938239732195846887 1.8153321559251329731;1.3945114456667675462
1.1683219744515513927 -1.9257548046289543198 0.14256961670997431169 -
0.34063230338760525084 0.15243622524435573862 1.8675431937494622048
0.79367518163722461377 -0.48592182851380666397 2.6047043209022460175 -
1.3598220838500880703 0.3645319945917451232 1.0480914292012855071 -
3.3396606972002129687 2.1381364755678426981 -1.4244386711819179148
4.2505683528077069511 0.49539684694564833922 2.8154751459464075758 -
3.7783962846624357113 1.4659350709691798542 0.64253577914377102 42
3.9496080347292092938 2.9695397800542671085 -0.61812446863306336198
3.557089636149243006 0.18384600953969831849 1.5474354041056157527
0.21245234362591530752 2.4303274679284876392 4.4649363526572614091
1.53013652372265474 -1.179427833620556143 -0.58693501408550352583 -
4.0054159863127800634 -1.8033234656761205628 -2.1476790850084297446
2.4092686916371892636 1.0594320022114933622 2.0920667101173249236
1.1120521211806302375 -1.8173490152625741967 -7.0592642505090976712 -
1.8343822832618144059 1.1728278388466621962 -0.14018307285789471939 -
2.2630191306685061292 2.1926482868266257142 -1.4406016102601539863 -
2.0387121115928437121;-0.4795770972194073889 -2.659154188717569145
2.9358875052857595911 -0.72240860428627629 02 3.0498667355102457854
1.0247755418725918997 0.35130538047954917857 2.8946143230627123799
0.015145778531785406007 -2.2351560241048042954 1.4681605329328775156 -
1.4712679474245977662 -0.49844293296201513943 -1.2631924609860478714 -
0.68848237079278917783 2.6640615892811334575 -2.9754567485249259207
2.5749583318314068769 1.7657342766923129762 2.7423284910818361837 -
3.6715335719842014939 -0.23329673045994911429 3.4720674553023531672
4.2405379379696235276 -1.5573273614151910138 2.0176500734062905451
1.3226703585604975277 0.50825036263747669363 1.0107507360525904705 -
2.7166399377965051443 -0.66125945192182833399 -0.40711382199226803236
-3.3400117889198197574 -3.6465231804066160493 0.97558485054824806149
1.3737435489712210845 0.98845194811691783343 1.7521321406835947965
4.4765517263559644334 -4.3699389616315933438 0.97974790956938506703 -
2.719869537268824633 -3.8869608664840691326 -3.2199350753761333266 -
1.3694584693398808017 -1.3083053048998314338 3.5881009060169941627 -
0.99461390479056788205 1.419176116378992214 3.1449744260114305128;-
1.5132122930523903559 0.16461338333564917713 -0.66180481283392422753 -
2.833064547535946609 -2.0335790394967832029 2.5078664385339459741 -
4.7288966137517389754 5.2598584642539591982 -0.92824014691690426293 -
1.9555136982134886381 1.6109194299850391197 -3.9868191488225606101 -
0.20614642592353665806 1.4233847943979631712 0.05494864575339677981 2 -

2.4363304433326867162 4.4880288908692769212 2.3551047113169838454
1.1209088675669609536 1.7024316917326860477 -2.8588548234645232426 -
1.3746679189183748093 -1.0419345844497143183 2.0180867149144900985 -
1.7355433930216850325 -0.037738882823181857384 1.4633034041466861375
4.0421905540963596337 -1.2939268400514425394 2.58809938610228496 -
2.465402577958220931 -0.34517888622127079756 1.5189956800020580907
1.5957504279448022633 2.2324937476762047872 0.076423137033136953611 -
0.14741697692022961985 2.0509740847947779407 3.8408479821043450819 -
1.1120919604255012914 -2.5951029062483454979 -4.6446769660616196163
3.7713646725492870893 -0.49441823070000118223 -0.69399826369887573918
2.43892177606420546 -2.7387844728856545196 0.46193475543851902909
0.17446156958998979003 -0.34808118353241068732;1.6507930080875867773
4.666503766049093116 -0.56051834617967111196 0.24416405802093937849
0.71407936104514446729 -3.9507158728369056178 2.1107934716091953931 -
0.13369008914469324156 -1.5941648517733471024 0.099310824457217436456
2.8511440194530539571 -2.033528309387808708 -5.4954273888618754285
1.7604351966771532023 3.0572452340395237513 0.21842442637480222219 -
4.1840853003012847822 1.3821184800970800488 3.6130887668344442432
3.8262231880521824223 -0.67580299253470998977 2.8180104283118065922 -
2.2131999155450383654 2.4748116172700265203 -0.257333995801943937774
0.90524836620771964579 0.75784392205292927525 2.0472614079898763251
2.7677320459610994519 -0.26900319558756574656 -0.53539556178310465651
-0.62645409745416458236 1.8160179821618178675 -2.8480301900285551753
0.85160036434010299988 2.9461233823852501779 -1.2460707780746715923
1.6696003489694899447 -2.8145258520984022255 0.732127623554048256887 -
3.7557862428124866305 -0.629488066463789297520 -2.2420481698466905307 -
0.59210766384035418142 2.2426494942633476981 -0.34135374895547304641
0.24154781946643696933 -1.1395735151683534792 -0.45564786706460658205
-0.48990658694745420343;-2.5596938075458282036 -0.824259095273181197077
-4.6875405767852429406 2.4570238701832654016 0.48599017300931141028 -
1.201405522585789365 1.3228572127308297901 -0.360991100883163662520
1.1024731115903232403 5.9669396943206054473 1.98383955275515355690 -
1.1077263754209363267 1.6991409439196094588 -3.9991316062551924837 -
0.85362508109682555535 0.4053236761981992009 -0.001645266670481648763 1
-0.6757073408871153708 -2.55697309331824707 0.43480409669197067624 -
2.4894888757188886608 -0.88390174128074638915 1.4059546801319413589
2.3877047912864028945 3.3454978665680821237 1.5871960160180167421
2.4714449323488021371 -0.49733069947758812290 -3.5033175226727713003 -
4.4228071275088129966 1.4032106411854239347 -3.1441208453121727295 -
1.8582038196729813162 0.14742541464244598992 2.6853151891922526495
1.0243214590340685710 1.3365838673255108215 2.6719555750574639497
3.2046355191772537552 1.2325760182574962975 0.492566665852698039432
0.93489097011365640544 1.4974947766244630643 -1.8921331007662482993
3.4719682132083242720 -0.36993931670249635690 1.0533662430530215115 -
0.24445295436109365506 1.4659225707597982602 -0.65769655373428881351;-
0.96508565144665614532 1.4973182243718903273 -3.0734445908379535162 -
3.8698195207606040924 1.2509337941819178575 2.2747646232732421367 -
1.8103220168341185303 1.0814307400608662579 0.61649534901892888428 -
3.4818042495682353099 -1.3622534648948811942 -1.2079986134069264914
1.2744367575688897443 -1.3499319789397161706 0.16175788519365147389 -
2.6031661212059304944 -0.33824313299086111728 -1.6178183571421986819
1.9413720090732278223 1.7074267601679136153 1.594969388014103151
1.4329645046710510758 2.4692728529728031894 -3.0218424857777526071
1.8016842200204701285 3.0274616338181177255 2.5384124005075512365
1.8214724153574921583 1.4850011972651238068 -3.8141373138137262266 -
0.32616742899747291906 1.8333269596459711437 1.8743382829101393749
1.3298585398160198690 3.9694196200864308999 -2.4714174693987471798
2.2658554514289459547 -0.94057671651757446263 -3.8774564785660650124
0.60142811824398401122 2.5170114941064936076 4.4924738849350829284 -
0.98551014577150253260 0.074394316568321683381 -4.4645946797970870179 -
3.321391709544011750 0.972767502272960715710 -1.9162854990371265274
2.5462887241370188995 2.9589485608805179417;3.5850294009981418242

1.4359614512438279021 -3.3100267816416102384 -1.2253899720880743196
1.8428100115032912676 0.71534975078441875862 -0.82686906791896475077
3.3212605589033366016 -2.5497192772177994513 2.037227725212637619 -
0.51119039132294774586 1.2859112573402884383 0.46471681955954802223
0.38542606932908862749 4.8623531778776296619 -1.4543432710229977634 -
3.5300264694387593067 2.1673845635608071092 -1.5985130396908775463 -
0.45138200043392540106 -3.0364071435556412304 1.716967070579026533 -
1.9837497440947171601 1.5135304864540881642 1.2229779535871454499 -
3.3075130673655954716 -0.64893100714276985563 0.366122953056413180 66 -
0.94075747883724769594 -0.66183525097529638526 -3.1273393325986207358
-1.8049447419951647476 -5.5171419054692432127 1.1675109813648036905 -
1.1514836159842594121 1.2165311740491193682 3.1023033557214052536 -
3.4157441049279428924 -0.51975275050037261781 -4.0037086759368074595
3.3285946882460213381 -1.5302735005559240378 2.0301827124228952748 -
0.95188810562091963341 -2.3440277133447184887 -1.0627335841048679566 -
2.1725306106951207674 2.0877671644307707055 -1.6741529275074151784 -
1.6118387019895343837;1.5551525944116417755 -3.3224299845686005028 -
0.39610556268962321402 -1.9461942881339115985 0.593922751215483213 37
1.7099492222906316208 -2.1473901011443854436 -2.2667959587110013686
0.94218682137264364584 1.7973621819006451172 0.671971348959301773 01 -
0.73139824144255238814 -1.2831380752322976146 1.7388495941735675476
3.8451996864748556604 -2.5354398559205457708 2.450840183750224277 -
1.5290829763267510533 4.6770240259043127651 2.998254645697329579 -
3.3242427892887604912 -1.0722829858083311461 0.125290281844495199 34 -
1.6043976267535440261 -0.91276851125949587384 4.3768944042305291 12 -
0.75061022410944233663 0.11158035312848485487 -2.625765154473210572 -
0.16133241131337108798 1.7438936465003986864 2.1142686622885547 01 -
0.70235544005221861585 0.77846289323170025 92 -0.703764876484831436 44
1.1633705402631739911 -4.6633595457594276823 -2.7365768870813367286
2.4841892720980736087 2.8231630718144771208 -2.5422546772817202765
0.075951552215297535309 2.2330461021800780408 -3.0817503118309623744 -
1.7459444754282855694 0.51039352342868793411 -0.15384121708179238075 -
0.66948199487699178167 2.532033834885985879
0.73784559584084030703;2.0914070930616310839 0.6128885243314998732
0.58926320270787790889 -3.3538527540542437677 -1.6486856065241934921
0.69877666932496262 93 -0.82578096446588145074 1.2939221431240777171
3.3879480684231224608 0.13987674577125874809 4.4665821295873211483 -
3.8485581140220070928 0.14341159723657642688 -3.9477908857801020659 -
0.069194439907353702535 -1.8546959654725678757 -2.4138154950012209454
-1.7126198360662570064 -2.1250901301480542571 -0.072637607969661219243
-4.2330988486124399017 2.8218234653208398299 1.7555824748398964896
2.6548410571578977546 -5.4214852075080264271 0.862101423584842985 19
2.7180794762012769894 -3.4296008334402370288 0.862285609526348695 3
1.97283331779005211 4.2534536409405463431 -3.5181506732177179053
3.4194651231496226984 1.3785622152042171784 -2.564544003894507096
0.10600860022856177511 0.61810499347404301851 -1.4475543346091279684 -
3.2574726760858037089 1.3406117334863634216 -0.368372504953875190 56
2.8773718525533191759 -0.098601957653571936646 0.4725879819440402807 -
1.1140914501113714508 2.5388889981054023615 0.156714969288377081 74
2.4772896555415395703 -1.1561118671368066924 0.191895264433388640 37;-
2.7971392270191595131 -1.776558791719409669 -2.4142146012391916265
1.6885648440259284353 3.8294394200256016703 -2.0512130149485510167 -
3.9568340686023315556 0.084844101130369914 41 0.28872266176215660716
2.5028364918811432105 2.7141438281920176046 2.1953945563349170911 -
1.7755335265480947093 2.0608583169022063331 0.154538698266610263 77 -
0.58870339962437046033 -0.47100604900452835988 0.508243376393224810 53
0.83861359122735390503 0.36067578718933945847 -1.8795167493488951216
2.8153366600071474402 1.9817497540341317 83 -2.3343628108316973879
0.28395714826839524303 -1.712751799309306433 -0.806191813049709327 11
0.094514977998031657336 5.6145910851658094032 -0.341141145177458637 86
-3.2647032675062170881 -2.1562408336493463779 4.6634853110386087494
1.738811462458795587 -1.5724554157478423111 -3.6205563392586235238

```
0.29636070224160299125 3.3610167178376637587 1.9855731172406272389 -
1.5574759574444234644 3.0999685398503520339 1.724716103074459328 -
3.3933502273061355581 -3.0482037008235520759 0.27818070037413539231 -
2.587880135516252178 -1.7520704787475502862 0.33859206052950840959 -
2.7965378882074340972 -0.58424663482479799459;-0.86681722698911911618
1.1130237328094487737 -0.80484541292739253393 -3.290391228855346295 -
2.0364014808032582238 -2.8949963516366943317 2.3954752647194830217 -
1.295781422454284737 -2.863936672308351028 0.52501130078973523574 -
0.85826373846484205199 0.11873215577079007654 0.32860153894990445922
1.6995069587084337126 -0.92905362373752531369 1.4690820528640196052 -
0.59712391003732989692 2.1550581830586295062 2.6524734672075784836
3.4627793209671673402 0.16571594716845847173 -2.1125952191340053865
3.2065361022447689265 4.8110408156660326995 0.24429235480634300948 -
2.0758378869438196546 -4.8146396801979189917 -0.66065067804119592676
0.95840046398505884806 1.8931560222183643205 0.66641134254091838329 -
3.6064912106328392838 0.84837548473095880652 2.3407213751406730218 -
4.2681381558738120674 -1.3102512196602862371 3.437185827491583634 -
3.1448831166126729464 2.5389293226956715088 0.40173955055024185 29
0.83010492242492084358 2.7615815346177323697 -0.43137162859330058229 -
2.9732761059020940131 -0.93943430653774195882 -1.3540076356341075314
0.38220920530782348523 -3.6953357650215457042 3.5470626020325504157
0.8297806976051340655;-2.705031030232648348 0.91574357454806609713 -
1.2672020437641267065 -0.33505694890493809002 0.10250429791800055868
1.6608979437846651539 1.8186770676921655276 -0.34859582474340011782 -
3.601031573203826941 0.71340003091294135196 -1.7266212085404193388 -
2.758989731793603184 1.8911475288346093482 -0.02780761598144325733 -
1.0010856319659340308 1.885550616306974625 0.31581736986379377319 -
4.6836501175071756364 -1.0755550045142032722 -0.556391753142189471 33
3.583579411200856768 1.4100038094526798904 -2.291569461702809285 -
1.5263548919400828918 0.19835180161138105115 1.588050581543998252
2.1750980272220448164 -3.1650947444968267064 1.6465211668561348102
2.6661255859185493122 0.70396909901286119915 1.1088545273795440416 -
1.1425408841927060521 4.4160267844412501859 -0.58842522820620291046
0.63641357228490580411 -0.36300848289226766985 2.2353600951737711
1.1308893107315884929 -1.2953627739479243175 -0.2353398307701052572 -
0.16959101447667251916 -1.1511789881099205868 2.326446451752641309 -
2.5015056162758577685 1.9498779468395126546 0.96669536207434225705 -
0.76800159614573326738 -0.78542858245951918672 6.8561318645048041631];

% ===== SIMULATION ========

% Format Input Arguments
isCellX = iscell(X);
if ~isCellX
  X = {X};
end

% Dimensions
TS = size(X,2); % timesteps
if ~isempty(X)
  Q = size(X{1},2); % samples/series
else
  Q = 0;
end

% Allocate Outputs
Y = cell(1,TS);

% Time loop
for ts=1:TS
```

```
    % Input 1
    Xp1 = mapminmax_apply(X{1,ts},x1_step1);

    % Layer 1
    a1 = logsig_apply(repmat(b1,1,Q) + IW1_1*Xp1);

    % Layer 2
    a2 = logsig_apply(repmat(b2,1,Q) + LW2_1*a1);

    % Output 1
    Y{1,ts} = a2;
end

% Final Delay States
Xf = cell(1,0);
Af = cell(2,0);

% Format Output Arguments
if ~isCellX
  Y = cell2mat(Y);
end
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y = mapminmax_apply(x,settings)
  y = bsxfun(@minus,x,settings.xoffset);
  y = bsxfun(@times,y,settings.gain);
  y = bsxfun(@plus,y,settings.ymin);
end

% Sigmoid Positive Transfer Function
function a = logsig_apply(n,~)
  a = 1 ./ (1 + exp(-n));
end
```

A.3 Implementation script

This function uses the optimal topology function and the neural network function to generate the access control feature for authorized and non-authorized MAC addresses.

```
%Access MAC Addresses from a file
[y1, y2] = xlsread('DATA3.xlsx');
 g = hex2dec(y2);
 f = dec2bin(g);
 for j=1:length(y2)
   for  i=1:48
d1(j,i)=str2double(f(j,i));
   end
end
 d=d1';

%convert input to binary
%d = hex2bin(str);
```

```
%display(d)
for k = 1:length(y2)
    d2 = d(:,k);

%pass to Neural network function
 p = myNeuralNetworkFunction(d2);

%produce result
display(p)
% Access control section

%extract the target section from the mAC address

tg=d2(25:48);

%find the coefficient of correllation
r = corr(tg,p);

if r>= 0.75

    % grant access
    disp(' Access granted')
    disp(['correlation = ', num2str(r)])

    g4(k)=1;
else
    % deny access
    disp(' Access Denied')
     disp(['correlation = ', num2str(r)])
         g4(k)=0;

end
end
```

A.4 Plot validation Function

This function generates various validation plots used in this study.

```
% Array for number of neurons
x = [5, 8, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60];
% Array for Mean Square Error
y = [0.2184, 0.1888, 0.1562, 0.1062, 0.0706, 0.0631, 0.0563, 0.0360,
0.0336, 0.0290, 0.0289, 0.0252, 0.0260];
% Array for Number of iterations
y1 = [12, 12, 12, 11, 13, 17, 11, 11, 11, 10, 9, 9, 9];
% Plot graph 1
%figure, plot(x, y)
%title('MSE-Neurons Plot')
%xlabel('Number of neurons')
%ylabel('Mean Square Error')
% Plot graph 2
figure, plot(x, y1)
title('Iteration-Neurons Plot')

xlabel('Number of neurons')
ylabel('Number of Epochs')
```

APPENDIX B: MAC ADDRESSES

Preview

This appendix contains a listing of all the MAC addresses that are used in this study.

B.1 MAC Address used in training phase

FEA233C3E7A4

118EBDC40130

8B5B2BD4292D

84BFA83F294D

BC8AA5411424

C1A1BF4315A1

DE8A5B814A11

5323F50401A8

44FB6FCCBA51

F29061BF4E28

0D2DC794405B

0E42794F5996

A501154AB141

A600CE4FDC79

34C07454F067

CD56DCE353E5

8E127A78772F

A5E1F61F2503

0845B72001FD

CCEFEE3C94E0

B77830ED7E65

CBB1F50E4C88

4638AE6216E7

58EF62D9AA3B

AF993F467D85

15A0AE5B1EA1

0D1E90FDE113

72C4809ADDAB

1648047431A2

1A8A8D355104

2FFABFC3FC53

50F1725C874B

A2A5ECA5FE84

E0A3B8DDBDE4

4751B47A0752

7273661BFB59

B043FF3F298B

1220B472E820

B8F3637D99A7

3B714C4423C9

2BBB31CBA5D8

C41F53F2F779

D7FFE3D6979F

5A6981BAC3F9

0200DB957DA0

A88E3FB57430

97F78E2798AD

A07B84D29DBE

1E8969342321

C624A3CA004D

495C048C84AA

3A110479D0C1

E943434EC019

846940A8E0E0

7182794FA8CD

2807711A3F73

E6132E800B38

8F982829A6B0

D048B9EFB109

604E4734AFFE

4031A9F4961C

E3E6B3F7729D

F7D34DF5C5C5

352DE6168061

86BA236D05D9

17CBEBD7F477

DCBFA438EFB9

0722ACD2437D

0022A3068CAC

1625A6858DD4

A6345C375D0E

5D8976DE5360

5A3C932A525A

238D6355F9D1

0BF302744461

EAF32AECCEB9

CB0B9E398337

0D253CF2B790

A3F5FD719881

CFE1B5F95AB7

7A15B414C4E5

B7A530F70B8B

8ADB18E941DF

F7AC5A86ADE7

4394EB163CA3

42AD7099B726

3986E7394FF7

ACFC524AA8EB

25CBF86A473B

BDC1BF7C5FBA

99D3A3482D09

FFFA1A905907

E3D3CE7EFF2C

DA6CD8FA9DD9

2DC2936E50C9

039ECDE5696C

3312B7A98176

23A3EB7BD447

19BD884F2F90

61CBF8EA612A

B.2 Authorized MAC Addresses used in operation phase

CAF598F2E46B

91D48F4795EB

45BDF1BD77B1

4A9F0F3B53FC

7CE70B2894EE

4B19699511B4

D72BB351E702

39CCA86CB9C4

56A4E7A8D972

DEA7C47C2E8F

85B2EBEEFE3C

581EEA3229D8

E4803478856D

84C041986E64

A9F4F217A34C

A4094E3849F0

ACFD3831CE1B

C1C3BEC92C95

7781DDD324D2

1AA1120EB567

EC1099DF906F

133AD43FAC31

72885DA808B2

B7E63589E68C

4EC03CAA2CF7

A4B1A8FC63E1

53805C80BF7B

091A610C241A

1AA2800CAA41

5C61F6EFC7B3

442E09673F4D

7D6541D740CC

2A47A7BFED0B

214E2B04F2DB

B956D37C5908

4F597087C1F0

3251A29A3A01

47792F6E65E6

5051BCB3FDDC

5AEB335A20A3

9C4BEE0FAB80

C4077580358B

9CFC5DFABF1E

2E680282E42A

4261BB183D51

E9D45844AEC8

FE30ADC3E44A

E37A8EB1D802

61058EB1C028

0750E1FB5C04

E56545D5FB26

895E4279147F

FF61C6279B1C

3F9ADD46A76B

F995CF7BEF4E

26FA24F81E8F

B05AE2ACC7F4

5F839F692F3F

E6EBD19CB887

A51FEE067582

F4EC6019DAE3

482C229D41BF

5214BF3F4767

69F4327C447A

91AB2CA53047

69F105C90278

EE3DF77D7EE8

C2131644A92F

E0B81AAF18EE

C949551EC59B

21F97FD8BAED

18D6119E1CC2

E95043DB0047

0C6FAC98D561

6BBCE046BA7C

66F04A2D19A8

E1EF8638C4FE

5F90A0D8C72F

3CE1072F180B

45F458643335

B.3 Unauthorized MAC Addresses in operation phase

45CDF1BD78B2

DEA6C48C2E5F

ACFD5833CE2B

8DDB235C10A6

7EB02CEA1CE7

2D4BDC0FEF80

C35A6DB2C843

32055EF1C127

341B2CD82045

5433CB5EA340