IMPROVED GENETICALLY OPTIMIZED NEURAL NETWORK ALGORITHM FOR CLASSIFICATION OF DISTRIBUTED DENIAL OF SERVICE ATTACKS

BY

GADZAMA, Emmanuel Hamman
MTech/SICT/2017/6760

A THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, NIGERIA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF MASTER OF TECHNOLOGY IN CYBER SECURITY SCIENCE

SEPTEMBER, 2021

ABSTRACT

Distributed Denial of Service (DDoS) attack has continued to grow dynamically and has increased significantly to date. This form of attack is usually carried out by draining the available resources in the network as well as flooding the package with a significant intensity so that the system becomes overloaded and stops. This research proposes a classification of DDoS attack using neural network-based genetic algorithm (NNGA). The genetic algorithm was used to optimize neural network for the detection of DDoS attacks in order to improve the effectiveness and efficiency of classification accuracy and performance. To improve the NNGA, a fitness function was introduced in genetic algorithm that improved the performance of NNGA. The features of DDoS attacks from KDD 99 intrusion detection datasets were obtained to train the NNGA. The results obtained from the study indicate that the technique performed optimally in DDoS attack

recording the following; 98.58% and 0.351 respectively for accuracy and false positive rate. Therefore, revealed that the enhanced genetically optimized neural network algorithm has better accuracy and lower false positive rate in comparison with the conventional neural networks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPEDICES

CHAPTER ONE

1.0 INTRODUCTION

1.1 Background to the Study

The Internet was primarily designed for openness and scalability without any security concern, however, malicious users have exploited this weakness to achieve their purpose and recently, the number of network-based threats has been significantly increased (Booth and Andersson, 2017).DDoS attacks are one of the major types of these threats and the aim of these attacks is to make internet-based services unavailable to its legitimate users. Although widely known web sites, such as GitHub, Dyn (DNS Provider), BBC, Spamhaus and Bank of America (JP Morgan Chase/US Bancorp/Citigroup/PNS Bank) were well-equipped in security, reports by Makridis and Smeets (2019)showed that these sites suffered DDoS attacks in February 2018, October 2016, December 2015, March 2013 and December 2012 respectively. Hackers are incessantly generating new types of DDoS which work on the application layer as well as the network layer. The vulnerabilities in the aforementioned areas allow hackers to deny access to web services and slow down access to network resources. The Intrusion Detection System (IDS) is one of the solutions employed to solve the problem of DDoS and preserving the confidentiality, integrity and availability of web services and computer network resources (Adebayo and Abdul Aziz, 2019). Numerous types of DDoS attacks are already known, such as a Smurf attack, which sends large numbers of Internet controlled message protocol packets to the intended victims. A dissimilar category of DDoS is R-U-Dead-Yet (RUDY), which simply consumes all available sessions of a web application which means sessions will never end. In the same vein, the web service will be unavailable for any new request from new users. Lakshmi and Begum (2017) presented thatone of the most up-to-date DDoS categories is

HTTP POST/GET, where attackers send a totally legitimate posted messages at a very slow rate, such as (1 byte/240 second), into a web server hosting a web application. The HTTP POST/GET will have a harmful effect on a web service and cause it to slow down temporarily and interrupting the service. A dissimilar category of modern DDoS attack is an SQL Injection Dos (SID DoS) in which attackers insert a malicious SQL statement as a string that will pass to a website's database thereby illegally allowing access to the resources or to the stored data on servers(Adebayo *al.,* 2012).

Kubus, (2020) conveyed thatmost available open access data sets contained duplicated and redundant instances, which make the detection and classification of DDoS unrealistic and ineffectual.

Machine learning is usually used to detect and classify network traffic based on some features to measure and determine if the network traffic is normal while the amount of packets would increase in the attacked packet rather than the normal packet; also, the inter arrival time will be too small to allow attackers to consume resources rapidly (Adebayo and AbdulAziz 2014). DDoS packets usually have a high bit rate for network layer attack, thus, attackers focus on any attributes that help them to consume resources and make the service unavailable to end users.

The aim of this research is to develop an improved genetically optimized Neural Network Algorithm for classification of DDoS attacks in order to have high accuracy, high detection rate, and low false alarm rate (FAR) using relevant datasets.

1.2. Statement of the Research Problem

DDoS attacks have remained persistent and reoccurring decimal in the security issues of computing applications (Alenezi and Reed 2017). Machine learning classification algorithms were proven methods applied for improving DDoS detection and classification. Most frequently used techniques are Naive Bayes, neural network, support vector machine, decision trees, multilayer perception and random forest. However, these methods suffers from low accuracy in classification of DDoS attacks, completeness and confidence factors (high false alarm and high running time) such as experienced in researches conducted by Jawale and Bhusari (2014); Nourozian and Merati (2015); Kejie *et al.,* (2017) resulting in 90.78%, 90.78%, and 94.6% accuracy respectively with undisclosed high FPR respectively. This research presents an improved genetically optimized Neural Network Algorithm for classification of DDoS attacks using finely tuned objective functions and constraints, to augment the setback in DDoS attack detection.

1.3. Aim and Objectives

The aim of this research is to design an improved genetically optimized Neural Network Algorithm for classification of DDoS attacks. The objectives are to:

i. Formulate objective function in order to improve the genetically optimized neural network algorithm for DDoS attacks.
ii. Design an improved model using refined GA.
iii. Evaluate the performance of the developed model.

1.4. Significance of the Study

The formulated objective function shall bring about improvement in the GA which shall be used to optimize the DDoS data with neural network model. With the implementation of an

improved genetically optimized neural network algorithm, an optimal DDoS classification parameters can be obtained for developing the model, thereby providing adequate protection against network-based threats. This in turn will improve the level of trust and confidence in the cloud service providers by customers, as well as increasing its level of adoption for various cloud services. The proposed DDoS classification model shall be evaluated to ensure the effectiveness and efficiency in the operation compare to the existing system.

1.5. Scope of the Study

The research will be restricted to the use of genetically optimized Neural Network Algorithm for classification of DDoS attacks using the existing DDoS attacks dataset from the well-known KDD ‘99 dataset repository to evaluate the performance of proposed work.

.

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1 Related Work on Distributed Denial of Service

Yu and Lee (2010) proposed an incremental learning method which was called incremental tree inducer (ITI). The investigation established the performance of ITI, K-mean+ ITI, SMO+ ITI for DDoS detection on KDD'99 as 92.38%, 91.31% and 91.07% respectively.

Poojitha *et al.,* (2011) applied neural network to train samples from KDD'99. The method was able to simply feed forward neural networks trained by the back-propagation algorithm to classify the abnormal events. The authors reported the power of the algorithm to find 1500 DDoS attacks in the testing dataset.

Su (2012) collected attack data using one laptop that sent DDoS attacks against the victim machine in the LAN. The amount of traffic range was between 0-80 Mbps during the simulation. The author initially applied Modified Linde-Buzo-Grayclustering algorithm to reduce the amount of sample data. Afterwards, he employed KNN algorithm and reported the overall accuracy of 96.25% in the case of 2-flod validation.

Papalexakis *et al.* (2012) utilized the soft clustering to find different types of attacks in KDD'99. The researchers achieved an overall accuracy of 75% and 85% for normal and attack respectively.

Gavrilis and Dermatas (2013) conducted research and evaluated a Radial-basis function (RFB) Neural Network for DDoS attacks dependent on statistical vectors through short

time window analysis. The suggested method was tested and evaluated in a controlled environment with an accuracy rate of 98% of DDoS detection.

Ahmed and Mahmood (2014) applied the X-mean algorithm to detect anomalies in the DARPA dataset. The majority of the attack in the selected subset of the DARBA dataset was DDoS attacks and the study obtained 94% accuracy to detect anomalies in the dataset.

Jawale and Bhusari (2014) presented research on Artificial Neural Network (ANN) that achieved the highest accuracy rate. The research proposed a system that uses multilayer perceptions, back propagation and a support vector machine, consisting of multi modules such as packet collection and preprocessing data. This system achieved 90.78% detection rate.

Ahmed and Mahmood (2015) proposed a collective anomaly detection method using a partitioned clustering technique. The KDD'99 /DARPA datasets were used to train and test this method. The research confirmed the ability of the algorithm to find all available DDoS attacks in test data.

Hybrid Neural Network technique was used by Li *et al.*, (2019) consisting of a self-organizing map (SOM) and radial basis functions to detect and classify DDoS attacks. This experiment achieved a satisfactory accuracy rate result for detecting and classifying DDoS attacks.

Norouzian and Merati (2015) presented a most effective classification technique for detecting and classifying attacks into two groups normal or threat. The study offered a new approach to IDS based on a Multilayer Perceptron Neural Network to detect and

classify data into 6 groups. The research implemented MLP design with two hidden layers of neurons and achieved 90.78% accuracy rate.

A 2-layered NIDS that uses a feed-forward neural network was proposed by Haddadi *et al.* (2016) the proposed system classified normal connections and attacks. Diverse kinds of attacks were determined, and the research focused on using training function, data validation and a preprocess dataset that caused less memory usage, minimum resource consumption and faster training. After employing the projected system on a KDD cup 99 Dataset, the result was very satisfactory, both on accuracy rate and performance.

Zhang *et al.,* (2017) proposed an anomaly-based DDoS detection approach using an analysis of network traffic where a radial-based function (RBF) Neural Network was used in this approach, the method was tested UCLA dataset, achieving 93% accuracy rate for a DDoS attack.

Kejie *et al.,* (2017) proposed a framework to detect DDoS attacks and identify attack packets efficiently. The purpose of the framework was to exploit spatial and temporal correlation of DDoS attack traffic. The method employed accurately detected DDoS attacks and identified attack packets without modifying existing IP forwarding mechanisms at the routers. This work achieved 94.6% for detection probability using the proposed framework.

A synopsis and comprehensive classification of IDS was presented by Alenezi and Reed (2017) where difficulties and characteristics of DDoS attacks were discussed in the research. Three different classifications were chosen. The study focused on general

DDoS and flooding attacks. The cumulative sum approach had many advantages over statistical techniques which was effectively demonstrated in the research.

Recent study by Hoque *et al.* (2017) proposed a new DDoS detection framework which was implemented on software as well as hardware using the Field Programmable Gate Arrays (FPGA) device. This method merely considered the DDoS attack detection as a 2-class problem. This model formed the normal traffic profile during the analysis period. When a new input traffic instance was added, the attack detection module first computed the correlation value by analyzing the three distinct features of the added instance and normal profile. If the calculated correlation value surpasses the predefined threshold, the system generates an alarm.

2.1.1 Review of related concepts on genetic algorithm

A Genetic Algorithm (GA) is said to be a programming technique that mimics biological evolution as a problem-solving strategy (Bobor, 2006). This method is established on Darwinian's principle of evolution and survival of fittest to optimize a population of candidate solutions towards a predefined fitness (Li, 2004). It is important to state that GA uses an evolution and natural selection that uses a chromosome-like data structure as well as evolving the chromosomes using selection, recombination and mutation operators (Li, 2004). Typically, GA procedure starts with randomly generated population of chromosomes, which represent all possible solution of a problem. From each of the chromosomes, different positions are encoded as bits, characters or numbers. These positions could be referred to as genes. Thereafter, an evaluation function is used to calculate the goodness of each chromosome according to the desired solution; this function

is known as “Fitness Function”. Through this process of evaluation, “Crossover” is used to simulate natural reproduction and “Mutation” is used to mutation of species (Li, 2004). Likewise, for survival and combination, the selection of chromosomes is biased towards the fittest chromosomes.

When GA is used for solving various problems, three factors will have vital impact on the effectiveness of the algorithm and also of the applications (Goyal and Kamar 2008). These are:

i. The fitness function.
ii. The representation of individuals.
iii. The GA parameters.

The determination of the above-mentioned factors often depends on applications and/or implementation.

2.1.2 Related work on fitness function

Goyal and Kumar (2008) wrotea GA to detect the attack type of connection. The Algorithm used different features in network connections to generate a classification rule set; the researchers used the fitness function given by the formula:

$$F = \frac{a}{A} - \frac{b}{B} \tag{2.1}$$

Where:

A = Is the total of attacks,

a = The number of attack links which the individual correctly classified,

B = The normal links in the population,

b =The number of normal connections a network correctly classified.

A threshold value of 0.95 was set; the selected individual had a fitness value > 0.95.

Uppalaiah *et al.* (2012) used GA to detect Denial of Service (DOS) and Probe type of attacks, the researchers used a fitness function:

$$Fitness = \frac{f(x)}{f(Sum)} \tag{2.2}$$

Where f(x) depicts the fitness of entity x, while f(sum) is the total fitness of all entities.

Li (2004) used aGA for Intrusion Detection System, he calculated the fitness function by calculate the following four equations:

$$Outcome = \sum_{i=1}^{57} Matched * Weight(i) \tag{2.3}$$

$$\Delta = |Outcome - SuspiciousLevel| \tag{2.4}$$

$$Penality = \frac{\Delta * Ranking}{100} \tag{2.5}$$

$$Fitness = 1 - Penality \tag{2.6}$$

Using equation (2.3) the outcome is calculated based on whether the A field of connection matched the pre-classified data set and then multiply the weight of that field, the value of matched is 0 or 1. In the equation (2.4), the actual value of suspicious Level reflects observations from historical data. In the equation (2.5), ranking indicates whether or not the intrusion is easy to identify. Finally the value of fitness computed in equation (2.6) using the penalty.

2.1.3 Review of theories/empirical work

There are several researchers that used evolutionary algorithms and especially GAs in IDS to detect malicious intrusion from normal use. Similarly, there are several papers related to IDS which has certain level of impact in network security.

It is important to note that the process of using GAs for intrusion detection can be traced back to 1995, when Crosbie and Spafford (2008) applied the multiple agent technology and GP to detect network anomalies (1995). The GP was used to determine anomalous network behaviours and each agent can monitor one parameter of the network audit data. The proposed methodology has the benefit when many small autonomous agents are used but it has problem when communicating among the agents. Likewise, if the agents are not properly initialized the training process can be time consuming.

Li (2004) developed a process using GA to detect abnormal network intrusion. The approach used includes both quantitative and categorical features of network data for deriving classification rules. Nevertheless, it was observed that the inclusion of quantitative feature can increase detection rate but no experimental results were available.

Goyal and Kumar (2008) described a GA based algorithm to classify all types of smurf attack using the training dataset with very low false positive rate (at 0.2%) and detection rate at almost 100%.

Lu and Traore (2014) engaged the use of historical network dataset by using GP to derive a set of classification. The researchers used support-confidence framework as the fitness function and accurately classified several network intrusions. However, their use of genetic programming made the implementation procedure very difficult and also for training procedure more data and time was required.

Xia *et al*. (2015) used GA to detect anomalous network behaviours based on information theory. Few network features could be identified with network attacks based on mutual information between network features and type of intrusions and then using these features a linear structure rule and also a GA is derived. The approach of using mutual information and resulting linear rule appeared very effective because of the reduced complexity and higher detection rate. The only problem was it considered only the discrete features.

Gong *et al.* (2015) presented an implementation of GA based approach to Network Intrusion Detection using GA and presented software implementation. The approach derived a set of classification rules and utilized a support-confidence framework to judge fitness function.

Moorthy and Sathiyabama (2012) offered a novel approach to detect the malicious intrusions (hacks) by using a complex artificial intelligence method known as GA applied to IDS. This approach applies GA to learn how to detect malicious intrusions and separate then from normal use. Using GA result gave them the best fitness value which was very closely to the ideal fitness value of 1. The system was able to detect about 97% of attacks and 0.69% of normal connections were incorrectly classified as attacks.

Zhao *et al*. (2013) presented on IDS using GA that, Misuse detection system and anomaly detection system encode an expert's knowledge of known patterns of attack and system vulnerabilities as if-then rules. He also used two methods for cluster analysis, one was hierarchical and another one was K-means. It was concluded that only about 0.71% of normal connections were classified as attacks; also had a very low false positive rate.

Diaz-Gomez *et al.* (2006) used the evolution process set of probable solutions which were generated randomly. In that experiment, the researchers evaluated each chromosomes

using fitness function. The research also employed the use of single point crossover and single point mutation. Accordingly, the system was tested by implementing different formulas for fitness function. It was found that there were no false positives and the number of false negative decreases dramatically.

Gong *et al*. (2004) in 2005 selected the approach to network misuse detection. The result of this research indicated that the GA approach was very effective and also had the flexibility to detect the intruder and also classify them. In this approach there was good detection rate and depending on the selection of fitness function weight values, the generated rules could be used to either generally detect network intrusions or precisely classify the types of intrusions.

2.2 Standard Genetic Algorithm Process

The standard GA process is shown in Figure 2.1 In the first instance, a population of chromosomes is created. Then, the chromosomes are evaluated by a defined fitness function. Thereafter, some of the chromosomes are selected for performing genetic operations. Finally, genetic operations of crossover and mutation are performed. The created offspring replace their parents in the initial population. In the reproduction method, only the selected parents in the third step will be replaced by their corresponding offspring. The GA procedure repeats until a user-defined criterion is reached. In this research, the standard GA is modified and new genetic operators are introduced to improve its performance.

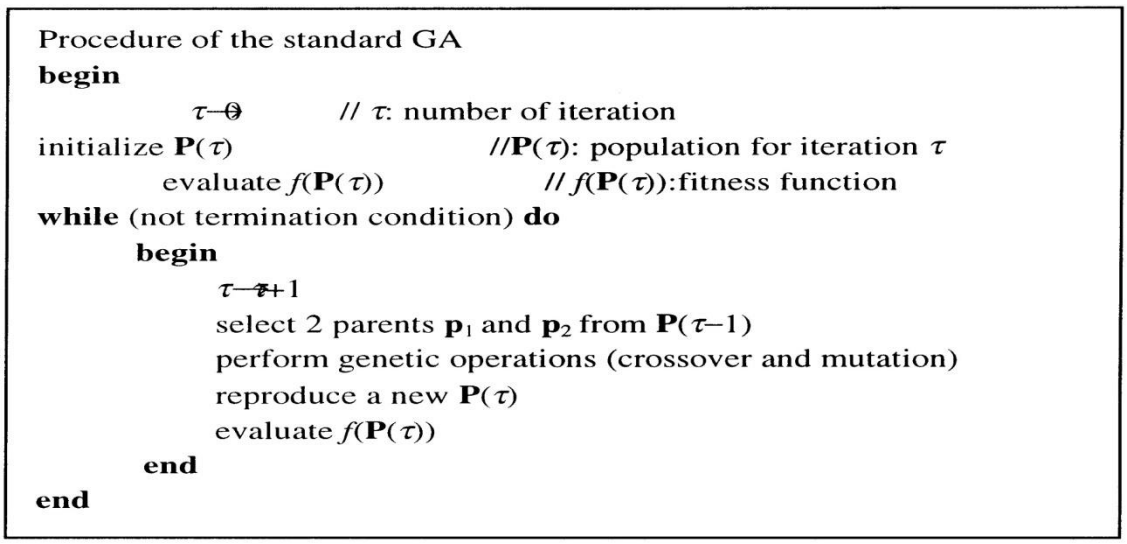

Figure 2.1. Procedure of the Standard Genetic Algorithm (Pham and Karaboga, 2000)

2.3 How Distributed Denial of Service Attack Work

As the name implies, DDoS attack enables an attacker to gain control of a network of online machines in order to carry out an attack. Computers and other technologies (such as IoT devices) are infected with malware, thereby turning each one into a bot (or zombie). The attacker at that moment takes remote control over the group of bots, which is called a botnet. Immediately a botnet has been recognized, the attacker is able to direct the machines by sending updated instructions to each bot via a method of remote control (Adebayo *et al.,* 2018). Each and every time the IP address of a victim is targeted by the botnet, each bot will react by sending requests to the target, potentially triggering the targeted server or network to overflow capacity thereby resulting in a denial-of-service to normal traffic. Taking into cognisance that every bot is a valid Internet device, separating the attack traffic from normal traffic can be hard.

2.4 Distributed Denial of Service Attack Strategy

Jaafar *et al.* (2019) reviewed DDoS attack components namely: attacker, control masters (or handlers), agents (or slaves or zombies), victim (or target machine). The Attacker initially scans millions of machines over the Internet for finding vulnerable machines whose security can be exploited easily. The machines are referred to as masters or handlers and are directly under the control of attacker. Additionally, the method of recruiting handlers is completely automated and is done through continuous scanning of remote machines looking for any security loopholes. Malicious codes are normally installed by the attacker into these compromised machines which then become capable of deploying further infected machines.

The machines which are normally used by handlers are directly under their control and are identified as slaves or zombies. Attacker indirectly controls these machines through handlers. Both the handlers and zombies, on the signal of attacker are deployed to start a coordinated attack on target machine thereby making the target machine incapable of communicating or utilizing any of its resources. The attacker frequently uses IP spoofing in handlers and zombies to hide the identity of these machines which enables future scope for attacker of using the same machines for creating DDoS attack.

2.5 Common Types of Distributed Denial of Service Attacks

To understand how different DDoS attacks work, it is vital to know how a network connection is established. A network connection on the Internet comprises several different components or “layers”. This is similar to building a house from the ground up where every

step in the model has a different purpose. The OSI model presented below, is a conceptual framework used to describe network connectivity in seven distinct layers.

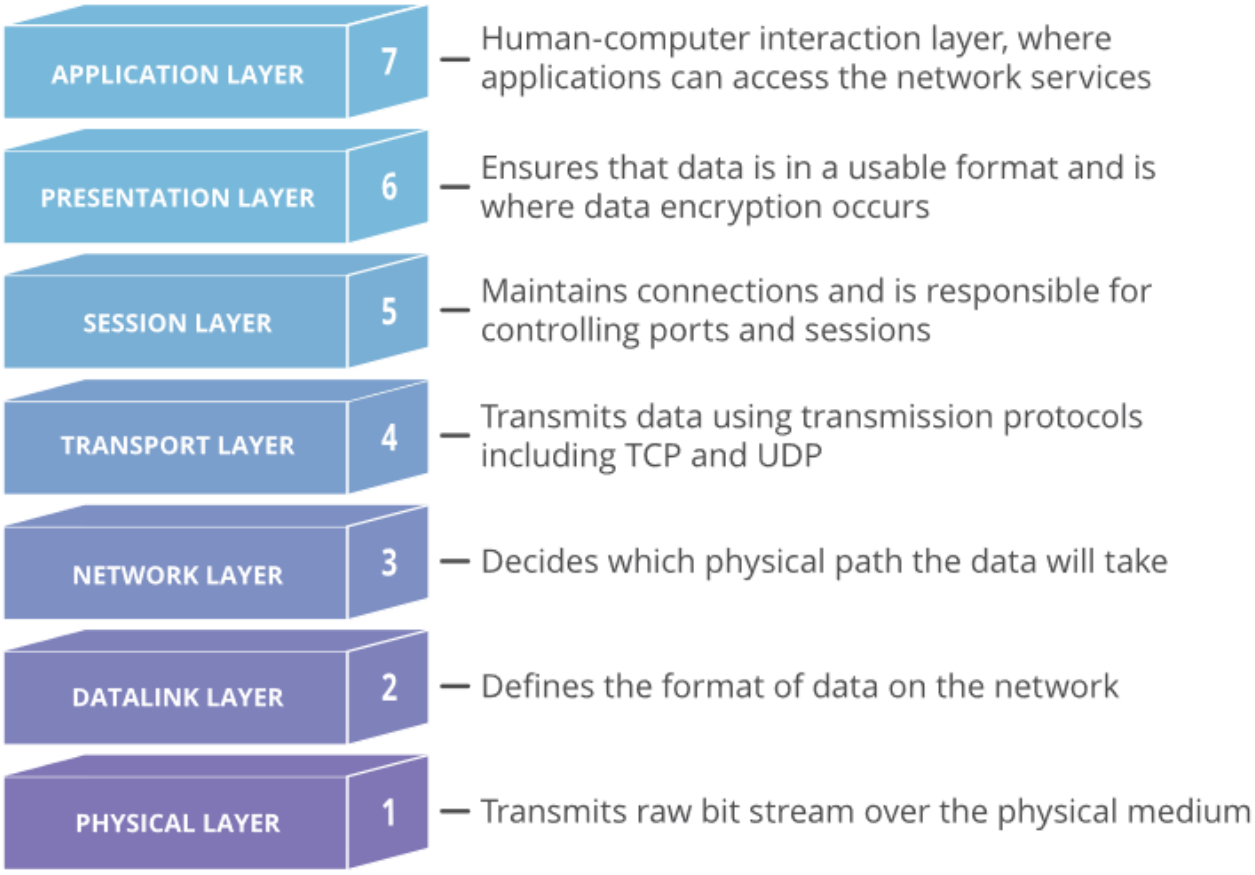

Figure 2.2 Conceptual Framework used to describe Network Connectivity in Seven Distinct Layers (Cardieri,2010)

While nearly all DDoS attacks involve overpowering a target device or network using traffic, attacks on the other hand can be divided into 3 categories. An attacker may make use one or multiple different attack vectors, or cycle attack vectors possibly based on counter measures taken by the target.

2.6 Application Layer of Distributed Denial of Service Attacks

In reference to the OSI model, the application layer occasionally referred to as a layer seven DDoS attack. The goal of this attack is to deplete the resources of the target. The attack is aimed at the layer in which web pages are generated on the server and delivered in response to HTTP requests. A single HTTP request is economical to execute on the client side, and can be expensive for the target server to respond to as the server often must load several files and run database queries in order to create a web page. It is pertinent to state that the Layer 7 attacks are hard to defend as the traffic can be difficult to flag as malicious.

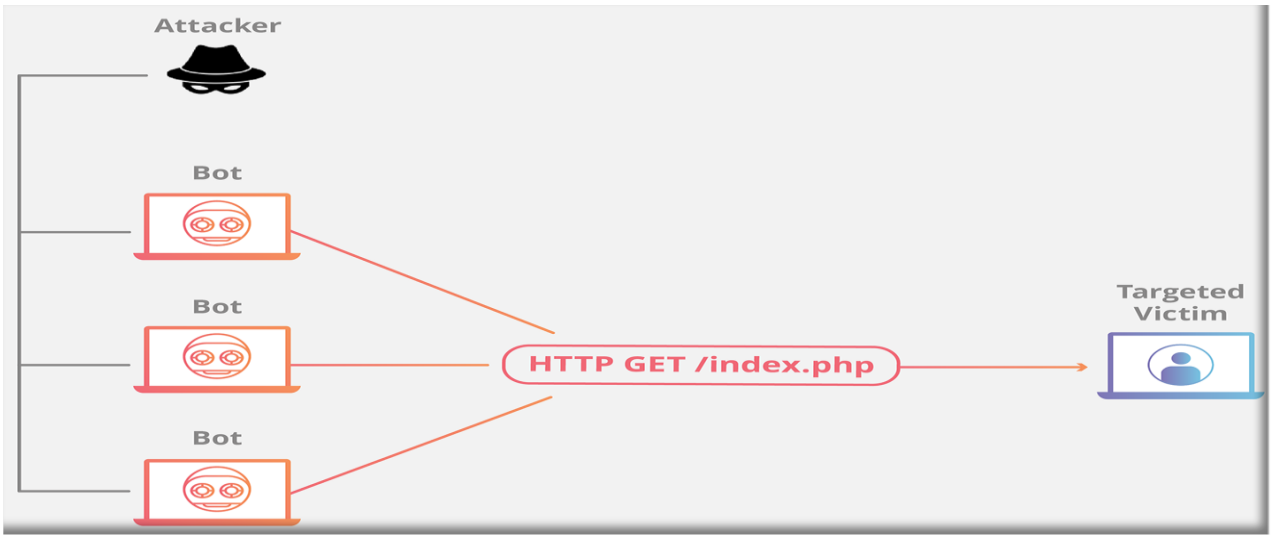

Figure 2.3 An Application Layer DDoS Attack(Cardieri,2010)

2.7 HTTP Flood Attack

The HTTP flood attack is likened to pressing refresh in a web browser over and over on many different computers at once where large numbers of HTTP requests flood the server resulting in denial-of-service. The HTTP attack ranges from simple to complex. Also, simpler applications may access one URL which has similar range of attacking IP

addresses, referrers and user agents. Complex versions may employ the use a large number of attacking IP addresses, and target random Universal Resource Locator using random referrers and user agents.

2.8 The Protocol Attack

The main aim of the protocol attack also referred to as state-exhaustion attack is to cause service disruption by consuming entire available state table capacity of web application servers or intermediate resources like firewalls and load balancers. Attacks in the protocol usually exploit the flaws seen in layer 3 and layer 4 of the protocol stack to render the target inaccessible.

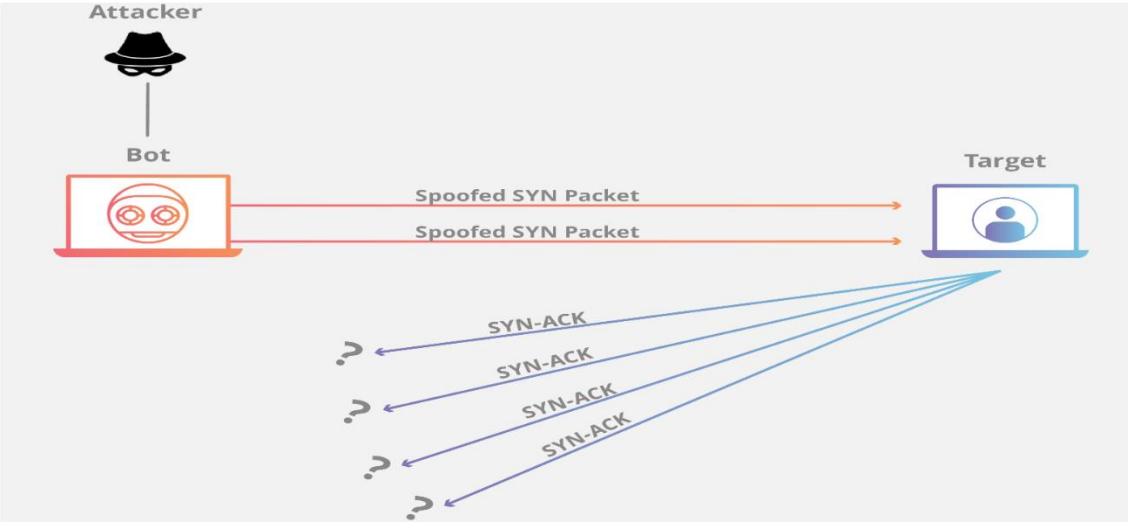

Figure 2.4 Protocol DDoS Attack (Cardieri, 2010)

2.9 SYN Flood Attack

A SYN Flood attack could be likened to a worker in a supply room receiving requests from the front of the store in which the worker receives a request, collects the package, and then waits for confirmation before bringing the package out front. The worker thereafter

acquires many more package requests without confirmation until the worker cannot carry any more packages, become stunned, and requests start going unanswered. The SYN Flood attack often exploits the TCP handshake by sending a target an enormous number of TCP "Initial Connection Request" SYN packets with spoofed source IP addresses. The target machine usually reacts to every connection request and then waits for the final step in the handshake, which never occurs, thereby exhausting the target's resources in the process.

2.10 Volumetric Attacks

This category of attacks attempts to make congestion by consuming all available bandwidth between the target and the larger Internet through which large amount of data are sent to a target by using a form of amplification or another means of creating massive traffic, such as requests from a botnet. Figure 2.4 shows an example of amplification.

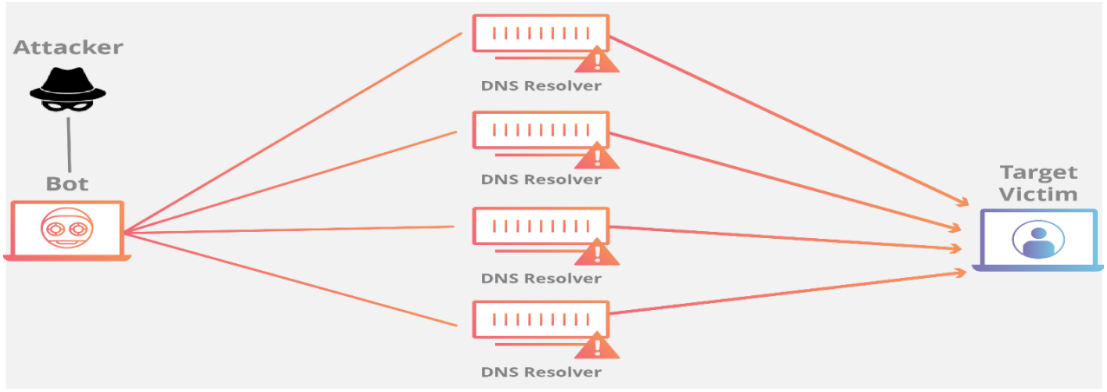

Figure 2.5 Diagram of Amplification (Cardieri,2010)

2.11Classification of DDOS Attacks: An Overview of Modern Approaches

Currently, there are a lot of options for organizing DDoS attacks. These options differ in the implementation techniques as well as the characteristics of parasitic traffic generated by botnets. There are three main classifications that specialists use nowadays.

2.11.1 Classification by protocol

DDoS attacks are divided into three large groups: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) and other. These are simplified classification based on the main protocols used to transfer data across the Internet, whose vulnerabilities are exploited by hackers to organize attacks. Both TCP and UDP are used more often than others, therefore attacks with their use are allocated in separate groups. Other forms include attacks via ICMP, GRE, IPIP, ESP, AH, SCTP, OSPF, SWIPE, TLSP, Compaq_PEE and other protocols. However, other school of thoughts divide DDoS attacks in 5 types: TCP, HTTP, UDP, ICMP and others. These divisions help us to ascertain tendencies regarding which protocols are more affected by parasitic traffic and the ones that are less affected, thereby paving way to adjust protection strategies and also helps in working on new algorithms for filtering "garbage."

2.11.2 OSI classification

The OSI is a network model of the OSI / ISO network protocol stack, according to how modern Internet operates. The OSI model comprises 7 layers: physical, data link, network, transport, session, presentation, and application (or application layer). Every protocol of the

network through which data (traffic) is transmitted denotes to a certain layer. Accordingly, the DDoS attacks classified by the layers of the OSI model.

Figure 2.6 Classification of DDoS Attacks by the Layers of the OSI Model(Cardieri,2010)

DDoS attacks are aimed at Layer 2 - data link, Layer 3 – network, Layer 4 – transport and Layer 7 - application.

2.11.3 Classification by mechanism of action

Going by the numerous types of denial of service attacks, 3 groups by the mechanism of action could be distinguished. All attacks aimed at overloading the communication channel, that is, various types of flooding falls under the first group. It is important to stress that the aim of flooding is to create a powerful flow of requests (data packets) which will take up the entire bandwidth which is dedicated to the victim's resource. The aforesaid groups comprise DNS amplification, Fragmented UDP flood, ICMP flood, NTP amplification, NTP flood, Fragmented ACK flood, Ping flood, UDP flood, Non-Spoofed UDP Flood, VoIP flood, Flood by media data, Smurf Attack, Fraggle Attack, Fragmented ICMP-flood, DNS flood and other attacks with amplification. Next is the group two which has fewer

types of denial of service attacks which also comprise of attacks that exploit the network protocol stack vulnerabilities comprising SYN flood, IP null attack, Fake TCP session attack, TCP null attack and Type of Service (TOS) Flood. Others include sack, RST/FIN flood, SYN-ACK flood, TCP null / IP null attack, Multiple SYN-ACK fake session attack, Synonymous IP Attack (Same Source/Dest Flood; LAND Attack), Misused Application Attack, Ping of Death and Multiple ACK Fake Session Attack (Multiple ACK Spoofed Session Flood). The third group includes the following application level DDoS attacks: HTTP flood, Faulty application attack, Single request HTTP flood, Fragmented HTTP packets attack, Single session HTTP flood and Session attack.

2.12 Artificial Neural Network

The Artificial Neural Network (ANN) is a biologically inspired computing model consist of various processing elements (neurons). These neurons are normally connected to elements or weights that build the structure of neural networks. The ANN has elements for processing information, namely transfer functions, weighted inputs, and output. (Rawat, Rana, Kumar and Bagwari 2018).

2.13 Intrusion Detection: An Overview

An intrusion detection system (IDS) is one of the solutions which can be used to prevent an intruder from launching a DDoS attack in a protected network. A very effective IDS can detect a new DDoS in a short time without human intervention. The IDS system can be categorized into two types as follows:

i. Host Intrusion Detection System (HIDS): this type of IDS can be implemented in network devices or workstations. HIDS techniques can be

used to prevent a DDoS attack on a selected device, but it does not support monitoring of a whole network.

ii. Network Intrusion Detection System (NIDS): The NIDS can be implemented as a security strategy within a protected network, and can be used to detect and classify all network traffic from all devices.

2.14 Networking Attacks

Every attack on a network can conveniently be placed into one of these groupings:

i. DDoS Attack: The DDoS attack as the name suggests attempt to make an online service inaccessible by overwhelming it with traffic from multiple sources. It is imperative to note that the DDoS attack usually targets a wide range of important resources, from banks to news websites, and present a major challenge to making sure people can publish and access important information. Some specific and particularly popular and dangerous types of DDoS attacks include: User Datagram Protocol (UDP) flood attack: An UDP flood attack is initiated by sending a large number of UDP packets to arbitrary ports on a remote host thereby on receiving the packets, the target system looks at the destination ports to identify the applications waiting on the port. However, once there is no application, it generates an ICMP packet with a message “destination unreachable”. This process can sap host resources thereby ultimately leading to inaccessibility.

ii. Internet Control Message Protocol (ICMP) flood: Similar in principle to the UDP flood attack, an ICMP flood overflows the target resource with ICMP Echo Request packets, generally sending packets as fast as possible without

waiting for the replies. The ICMP flood has the propensity to consume both outgoing and incoming bandwidth, as the target server will attempt to respond with ICMP Echo reply packets, resulting a significant overall system slowdown.

iii. SYN flood Attack: In the SYN flood attack, the attacker sends several packets but does not send the ACK back to the server. The connections are therefore half opened and consuming server resources. In this regards, a legitimate user will try to connect but the server will refuse to open a connection thereby resulting in denial of service.

iv. Remote to User Attacks (R2U): This is a type of attack in which a user sends packets to a machine over the internet, in which s/he does not have access to. This is in order to expose the machines vulnerabilities and exploit privileges which a local user would have on the computer. Examples of remote to user attack are: xlock, guest, xnsnoop, phf, sendmail dictionary.

v. User to Root Attacks (U2R): These type of attacks are exploitations in which the hacker starts off on the system with a normal user account and attempts to abuse vulnerabilities in the system so that to gain super user privileges. Examples include perl, xterm.

vi. Probing: The Probing is a type of attack where the hacker scans a machine or a networking device in order to determine weaknesses or vulnerabilities that may later be exploited so as to compromise the system. This technique is commonly used in data mining. For instance, saint, port sweep, mscan. nmap etc.

2.15 Classification of Intrusion Detection

Intrusions Detection can be categorized into two main groups namely:

i. Host Based Intrusion Detection: The Host Based Intrusion Detection (HIDSs) evaluate information found on a single or multiple host systems. This include the contents of operating systems, system and application files (Planquart, 2013).

ii. Network Based Intrusion Detection: The Network Based Intrusion Detections (NBID) evaluate information captured from network communications as well as analyzing the stream of packets which travel across the network (Planquart, 2013).

2.16 Components of Intrusion Detection System

An intrusion detection system usually consists of three functional components (Bace 2012). The first component of an intrusion detection system, which is also known as the event generator, is a data source. Data sources can be classified into four categories namely Host-based monitors, Network-based monitors, Application-based monitors and Target-based monitors.

The second component of an intrusion detection system is referred to as the analysis engine. This analysis engine takes information from the data source and examines the data for symptoms of attacks or other policy violations. The analysis engine can embrace one or both of the following analysis approaches:

i. Misuse/Signature-Based Detection: The Misuse/Signature-Based detection engine is intended to detect intrusions that follow well-known patterns of attacks (or signatures) that can exploit known software vulnerabilities. The

main limitation of this approach is that it only looks for the known weaknesses and may not care about detecting unknown future intrusions (Kumar and Spafford 1995),

ii. Anomaly/Statistical Detection: An anomaly based detection engine will search for something rare or unusual. The engine analyses system event streams by using statistical techniques to find patterns of activity that appear to be abnormal. The main drawbacks of this system are that engine are extremely expensive and can recognize an intrusive behaviour as normal behavior because of insufficient data. The third component of an intrusion detection system is referred to as response manager. In basic terms, the response manager will only act when inaccuracies (possible intrusion attacks) are found on the system, by informing someone or something in the form of a response. (Kumar and Spafford 1995).

CHAPTER THREE

3.0 RESEARCH METHODOLOGY

3.1 The Proposed Research Design

The design of the research work follows the processes elaborated in Figure 3.1. The design was divided into three phases namely: first, second and third phases. The first phase involves the critical, systematic and specific focus review stage. After the review, baseline papers were selected and thoroughly studied to formulate the problem. Thereafter, objective function was formulated for the GA optimization. The genetically optimized neural network classifier was developed. During the performance evaluation, comparative analysis of the conventional neural network classification and proposed classifier (NN-GA) was done. The work was technically concluded after this.

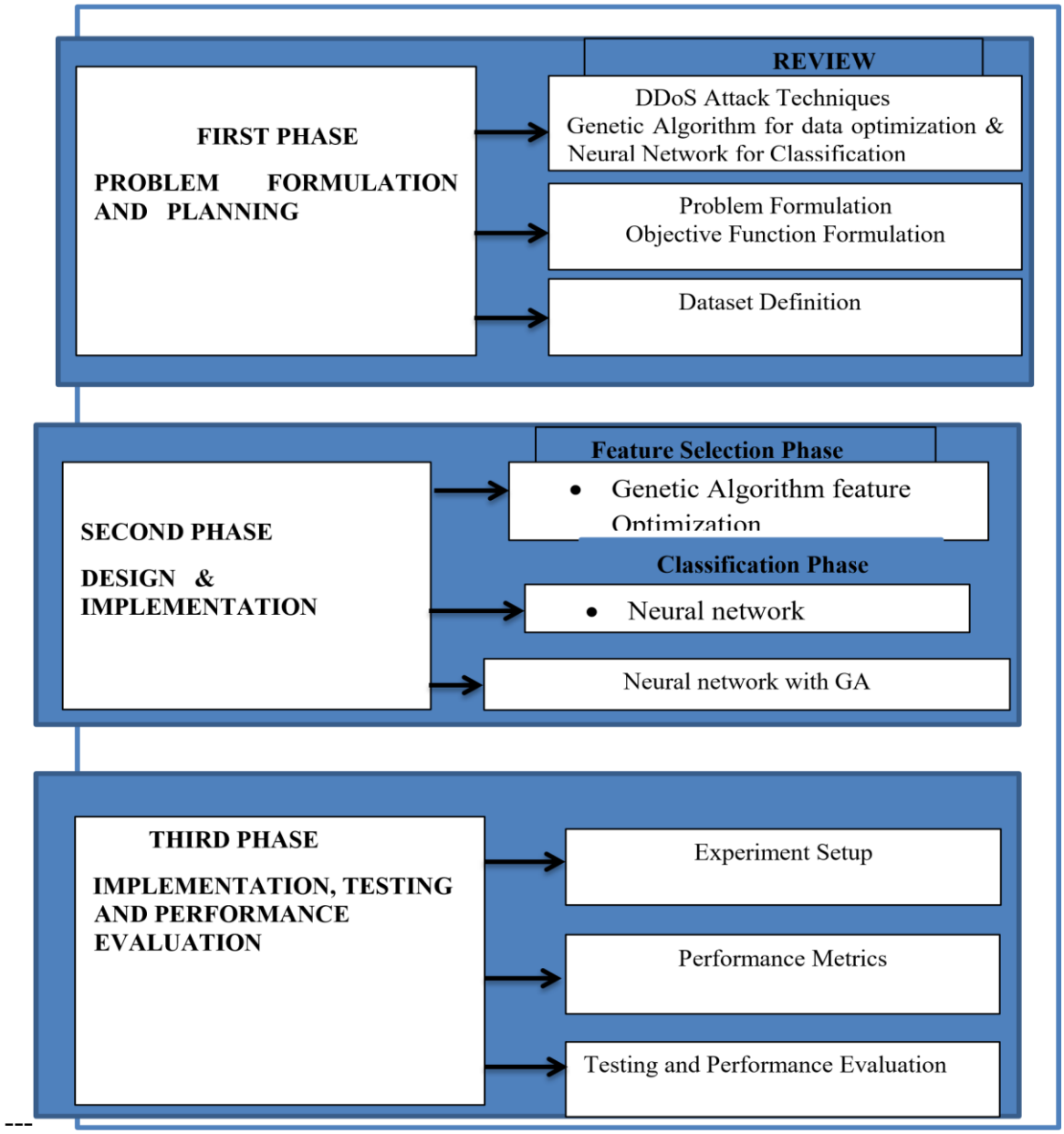

Figure 3.1: Block Diagram of the Research Design

3.2 Problem Formulation Process

The research problem was formulated by considering the key problems in the statement of problem. These issues are poor accuracy, confidence factor and completeness factor in the existing classifiers. The accuracy issue was addressed by formulating error in converse

relation to accuracy while the formulas for confidence and completeness factors would be used for genetic optimization.

3.3 Improved Genetic Algorithm

It would be recalled that in Figure 2.1, the standard GA process first created a population of chromosomes where the chromosomes were evaluated by a defined fitness function. Thereafter, some of the chromosomes were selected for performing genetic operations. Finally, genetic operations of crossover and mutation were performed. The produced offspring replaced their parents in the initial population. In this reproduction process, only the selected parents in the third step was replaced by their corresponding offspring. The GA process repeated until a user-defined criterion was reached. In this research, the standard GA was modified Figure in 3.2 and new genetic operators are introduced to improve its performance.

```
Procedure of the improved GA
begin
        τ←0        // τ: number of iteration
      initialize P(τ)                    //P(τ): population for iteration τ
        evaluate f(P(τ))        // f(P(τ)):fitness function
while (not termination condition) do
        begin
        τ←τ+1
        select 2 parents p1 and p2 from P(τ−1)
        perform crossover operation according to equations (7) to (13)
        perform mutation operation according to equation (14) to generate three
        offspring nos1, nos2 and nos3
        // reproduce a new P(τ)
                if random number < pa   // pa: probability of acceptance
                    The one among nos1, nos2 and nos3 with the largest fitness value
                    replaces the chromosome with the smallest fitness value in the
                    population
                else begin
                    if f(nos1) > smallest fitness value in the P(τ−1)
                        nos1 replaces the chromosome with the smallest fitness value
                    end
                    if f(nos2) > smallest fitness value in the updated P(τ−1)
                        nos2 replaces the chromosome with the smallest fitness value
                    end
                    if f(nos3) > smallest fitness value in the updated P(τ−1)
                        nos3 replaces the chromosome with the smallest fitness value
                    end
                end
                evaluate f(P(τ))
        end
end
```

Figure 3.2 Improved Algorithm: Predict data/intrusion type using GA

3.4 Objective Function Formulation Procedure

Multi-objective function was developed in this research in order to optimize the neural genetic classification of the DDoS attacks. The first objective is to minimize error. The error is computed from the difference of classier output and ground truth. The mathematically expression of the first objective function is given as:

$$\text{FitnessFcn1} = X_{co} - X_{gt} \tag{3.1}$$

Where

X_{co} = Classier Output

X_{gt} = Ground truth

Subject to

Constraints:

$0 <= X_{co}(1) <= 0.3$

$0 <= X_{gt}(2) <= 0.3$

The constraints for the two variables are chosen to be bounded between 0 and 0.3. This is because small range leads to pre-mature convergence and large range leads to poor performance.

The second objective function is computed from the product of confidence factor and completeness measure. Confidence factor measures the predictive accuracy of a rule by taking into account true positive (TP) and false positive (FP). Mathematically, confidence factor is measured as,

$$Confidence\,factor = \frac{TP}{TP + FP} \tag{3.2}$$

Where TP is the number of samples that are correctly classified

FP is the number of samples that are incorrectly classified

Subject to

Constraints: TP >= 0
FP >= 0

Boundaries: TP = 1
FP = 1

Completeness factor is a measure of the ability of a rule to select instances of a certain class. The mathematical expression is given as:

$$Completeness = \frac{TP}{TP + FN} \tag{3.3}$$

Subject to

Constraints: TP >= 0
FN <= 0

Boundaries: TP = 1
FN = 0

Where FN is the number of false negative of the considered class

So, the second objective function is expressed as,

FitnessFcn2 = confidence × completeness (3.4)

Subject to

Constraints: TP >= 0
FP >= 0
FN <= 1

Boundaries: TP = 1
FP = 1
FN = 0

3.5 Flowchart of Genetic Algorithm

The flowchart in Figure 3.3 shows the operations of the general GA according to which GA is implemented into this research.

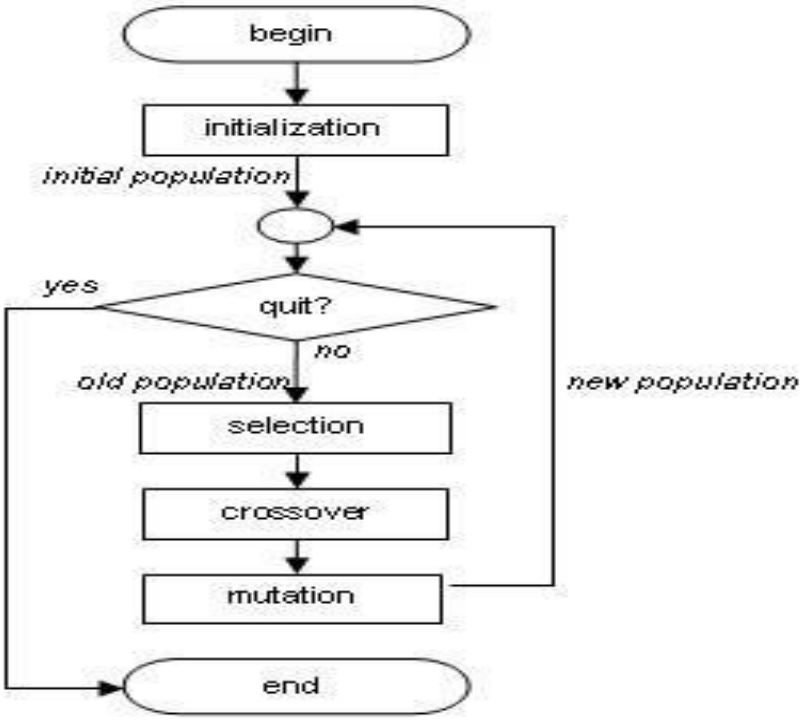

Figure 3.3Operations of a General Genetic Algorithm (Norouzian, 2011)

Figure 3.4 shows the Flowchart for the optimization/classification model. The DDoS data was optimized using GA. The optimization is in terms of optimizing the confidence and completeness factors and minimizing the error. The classification was done using neural network. Based on the set of rules generated during supervised learning, the classification was done as either non DDoS attack or Classified DDoS attack.

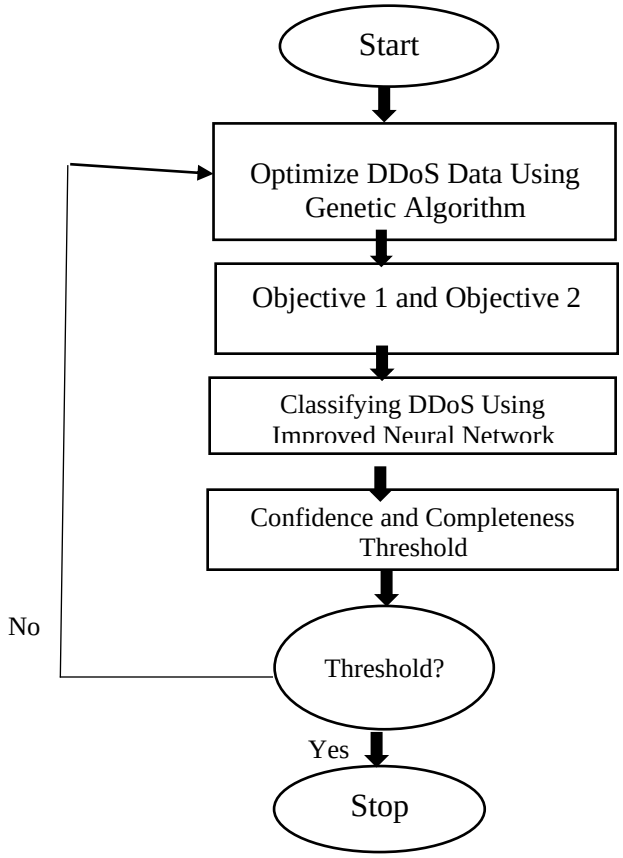

Figure 3.4: Flowchart of the Proposed Optimization/Classification Model

The proposed model starts by optimizing KDD DDOS data using GA. After feature optimization, neural network was applied for classification in order to build model for attack classification into DDOS attacks and non DDOS attacks. Once the model classify data into either benign or malicious, it stop the execution.

3.6 Implementation Procedure

The pre-calculation phase has 23 groups of chromosomes according to the training data. There are 23 (22+1) groups for each of attack and normal types presented in training data. The number of chromosomes in each group is variable and depends on the number of data

and relationship among data in that group. Hence, total number of chromosomes in all groups were tried to keep in reasonable level to optimize time consumption in testing phase. For each test data in the testing/detection phase, an initial population is made using the data and occurring mutation in different features. This population is compared with each of the chromosomes prepared in training phase. Portion of population, which are more loosely related with all training data than others, are removed. Crossover and mutation occurs in rest of the population which becomes the population of new generation. The process continue to run until the generation size comes down to one. Among the extracted features of the datasets and forthe sake of simplification of the implementation, only the numerical features (both continuous and discrete)were considered. The procedure for the implementation is captured in Figure 3.5.

Procedure for Implementation

Input: Pre-calculated set of chromosomes

Output: Type of data.

1 Initialize the population
2 CrossoverRate, MutationRate
3 While number of generation is not reached
4 For each chromosome in the population
5 For each pre-calculated chromosome
6 Find fitness
7 End for
8 Allot optimal fitness as the fitness of that chromosome,
9 End for
10 Remove some chromosomes with worse fitness
11 Set on crossover to the selected pair of chromosomes of the population
12 Set on mutation to each chromosome of the population.
13 End while

Figure 3.5 Implementation Procedure

3.7 Dataset and Data Processing

A literature review was carried out in order to find the appropriate datasets to be used for this research. Most frequently used datasets for intrusion/classification are the KDD cup 1999, DARPA 1998, CAIDA DDoS attack 2007, ISCX 2012, ADFA 2013, PREDICT 2014, DEFCON 2014, NSL-KDD 2014, KYOTO 2014, ICS Attack 2014, TUIDS. DARPA/KDD datasets are publicly available, but are old fashioned databases which are still used in modern studies. This research will leverage on the KDD 99 datasets for the implementation of our algorithm.

The KDD 99 benchmark consists of different components: kddcup.data kddcup.data_10_percent; kddcup.newtestdata_10_percent_unlabeled; kddcup.testdata; unlabeled; ddcup.testdata.unlabeled_10_percent; corrected, training data types and typo-correction. The "kddcup.data_10_percent" was used as training dataset and "corrected" as testing dataset. The training set consists of 494,021 records among which 97,280 are normal connection records, whereas the test set contains 311,029 records among which 60,593 are normal connection records. Table 3.5 shows the distribution of each intrusion type in the training and the test set.

Table 3.5. **Distribution of Intrusion types in Datasets**

Dataset	**Normal**	**Probe**	**ddos**	**u2r**	**r2l**	**Total**
Train ("kddcup.data_10_percent")	97280	4107	391458	52	1124	494021
Test ("corrected")	60593	4166	229853	228	16189	311029

In order to further improve the performance of the proposed classifier, the dataset would be preprocessed and the necessary features would be encoded into binary digits. This is because both neural network and GA operate optimally with binary digits. Encoding processing approach would be used to digitize the 23 classes of attacks in the dataset. In order to accommodate 23 attack classes, 5 bits would be used to encode the attacks as Code1 to Code5. With 5 bits, it can accommodate up to 32 attack classes using the formula of 2^n, where n is the number of bits.

3.8 Performance Evaluation Measures

In this Research, an accuracy-based measure was used to evaluate the classifier. These measures are the metrics that have to do with correction classification rate. The accuracy-based measures include:

i. **Confusion matrix**: Confusion Matrix as the name implies gives us a matrix as output and describes the complete performance of the model. This is an essential parameter for measuring machine learning based model. It consists of four (4) major components including True Positive, True Negative, False Positive, and False Negative. These components are described in table 3.1 thus:

Table 3.1: Confusion Matrix

		Predicted Class	
		Normal	Malicious
	Normal Web page	TN	FP
Actual Class	Malicious Web Page	FN	TP

Where:

TP (True positive) implies the total number of malicious network traffic instances "correctly" labeled by the classifier.

TN (True Negative) represents the total number of normal network traffic instances "correctly" labeled by the classifier.

FP (False positive) depicts the total number of normal network traffic instances "incorrectly" labeled by the classifier as malicious.

FN (False Negative) shows the total number of malicious network traffic instances "incorrectly" labeled by the classifier as normal.

ii. **Accuracy**: Accuracy measures how accurate a model can detect whether an instance of network traffic is normal or malicious (intrusion). It can be expressed in equation (14) as follows:

$$\text{Accuracy} = \text{TP+TN/ (TP+FP+FN+TN)} \tag{3.5}$$

iii. **True positive rate (sensitivity):** True Positive Rate is defined as *TP/ (FN*+TP). True Positive Rate corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points.

$$TruePositiveRate = \frac{TruePositive}{FalseNegative + TruePositive} \tag{3.6}$$

iv. **False positive rate (specificity):**False positive rate is defined as *FP/ (*FP+TN). False Positive Rate corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points**.**

$$FalsePositiveRate = \frac{FalsePositive}{FalsePositive + TrueNegative} \tag{3.7}$$

It is important to note that both False Positive Rate and True Positive Rate have values in the range [0, 1]. FPR and TPR both are computed at threshold values such as (0.00, 0.02, 0.04, …., 1.00) and a graph is drawn.

v. **Mean squared error**: Mean squared error(MSE) is quite similar to Mean Absolute Error, the only difference being that MSE takes the average of the square of the difference between the original values and the predicted values. The advantage of MSE being that it is easier to compute the gradient, whereas Mean Absolute Error requires complicated linear programming tools to compute the gradient. As, we take square of the error, the effect of larger errors become more pronounced then smaller error, hence the model can now focus more on the larger errors.

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2 \tag{3.8}$$

vi. **Regression**: Regression is a machine learning algorithm that can be trained to predict real numbered outputs. Regression is based on a hypothesis that can be linear, quadratic, polynomial, non-linear, etc. The hypothesis is a function that based on some hidden parameters and the input values. In the training phase, the hidden parameters are optimized w.r.t. the input values presented in the training. The process that does the optimization is the gradient decent algorithm. If you are using neural networks, then you also need back-propagation algorithm to compute gradient at each layer. Once the hypothesis parameters got trained, the result produced least error during the training, then the same hypothesis

with the trained parameters are used with new input values to predict outcomes that will be again real values.

CHAPTER FOUR

4.0 RESULTS AND DISCUSSION

4.1 Results Presentation

4.1.1 Preprocessed dataset classes of attacks

The result of 5-bit encoding of the 23 attack classes in the chosen dataset is presented in Table 4.1. These 5-bit encoded values are the output features of the dataset. The snippet of the input features of the dataset is in Appendix E.

Table 4.1: 5-bit Encoded DDoS Attacks

Attack type	Code1	Code2	Code3	Code4	Code5
1	0	0	0	0	0
2	0	0	0	0	1
3	0	0	0	1	0
4	0	0	0	1	1
5	0	0	1	0	0
6	0	0	1	0	1
7	0	0	1	1	0
8	0	0	1	1	1
9	0	1	0	0	0
10	0	1	0	0	1
11	0	1	0	1	0
12	0	1	0	1	1
13	0	1	1	0	0
14	0	1	1	0	1
15	0	1	1	1	0
16	0	1	1	1	1
17	1	0	0	0	0
18	1	0	0	0	1
19	1	0	0	1	0
20	1	0	0	1	1
21	1	0	1	0	0
22	1	0	1	0	1
23	1	0	1	1	0

4.1.2 Neural genetic algorithm classifier training

The neural-based classification results are presented in Figure 4.1. The architecture consists of 41 inputs, 10 neurons in the hidden layer and five output nodes. The training recorded 15 iterations.

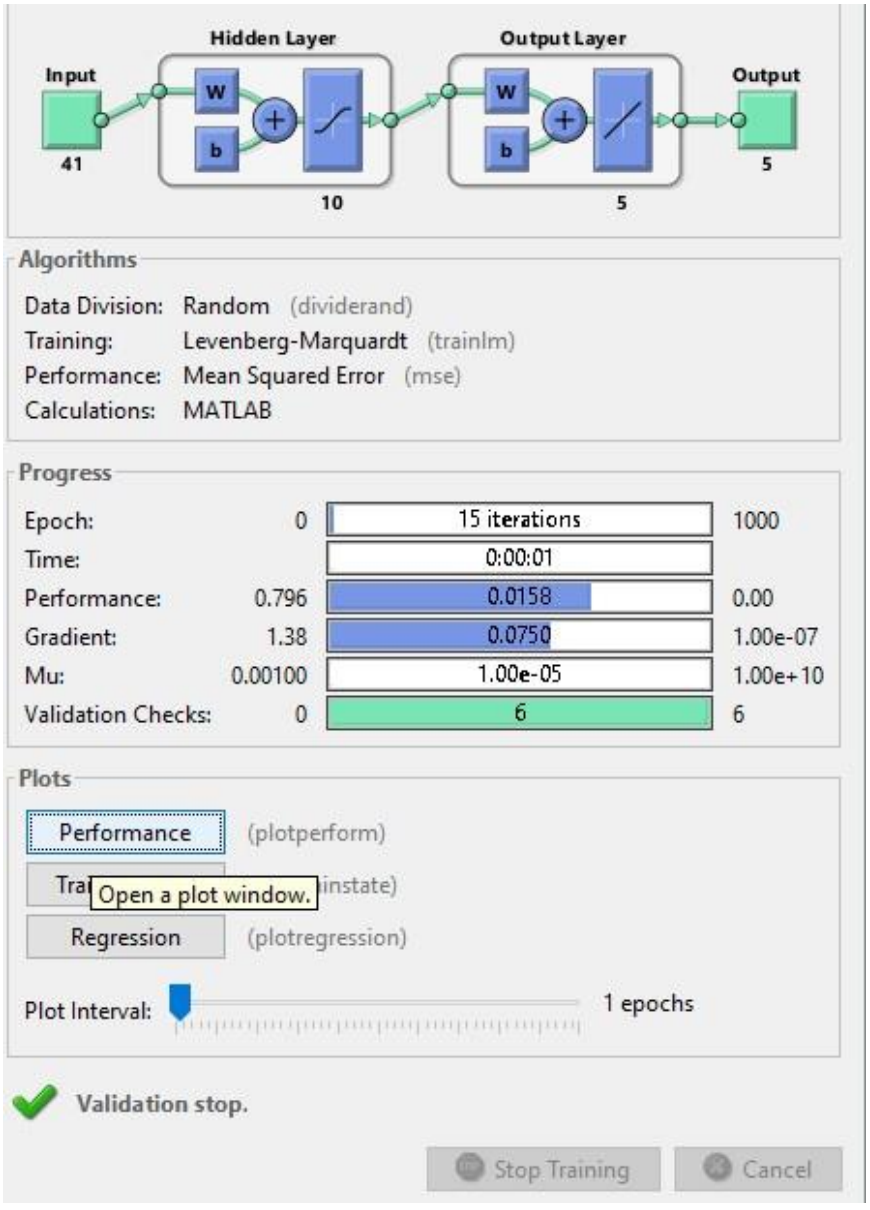

Figure 4.1: Neural Genetic Algorithm Classifier Training

The Mean Squared Error (MSE) of the neural-based genetic classifier records 0.07985 at the 9th epoch within 15 iterations as shown in Figure 4.2.

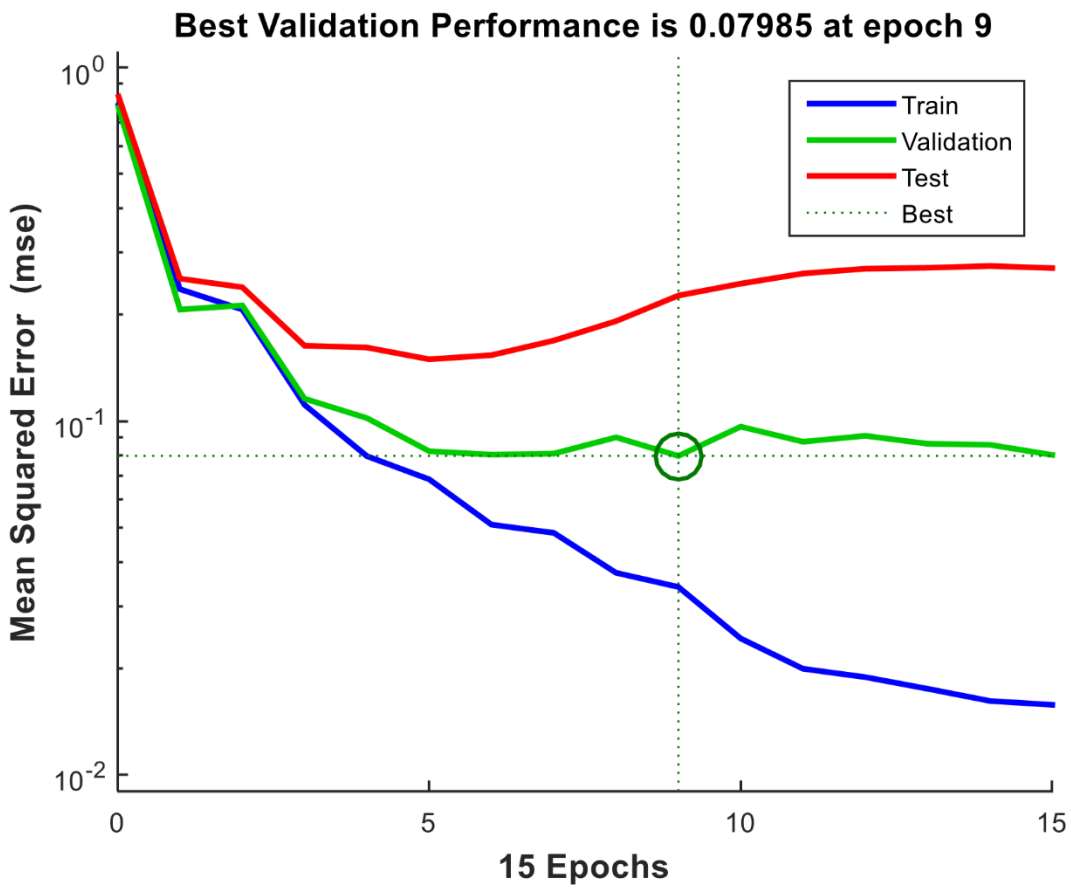

Figure 4.2: Classifier Mean Squared Error

The classifier also regresses between 0 and 1. The better result is obtained when the regression is closer to 1. The regression for training, validation and testing are respectively achieved as 0.92879 (92.879%), 0.83382 (83.382%) and 0.58577 (58.577%). The overall regression achieved by the classifier is 0.85423 (85.423%) as presented in Figure 4.3.

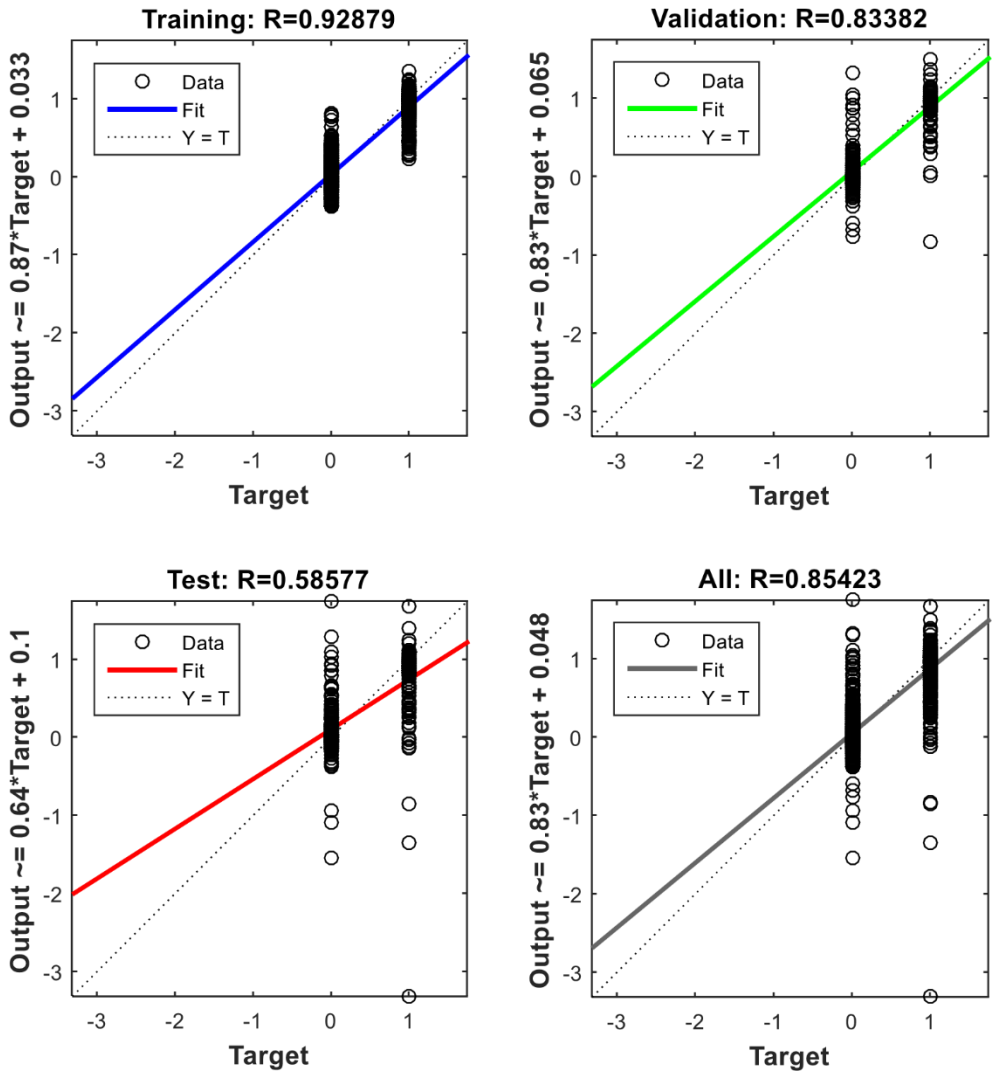

Figure 4.3: Classifier Regression

4.1.3 Genetic algorithm implementation toolbox

The essence of GA is for optimization. It optimizes the classification of the DDoS attacks. The implementation toolbox is presented in Figure 4.4. It consists of a section for problem setup and results. It also makes provision for option setting and customization of

genes formations, chromosomes mating, individuals, population, crossover operation, mutation operation, generation, stop criteria and so on.

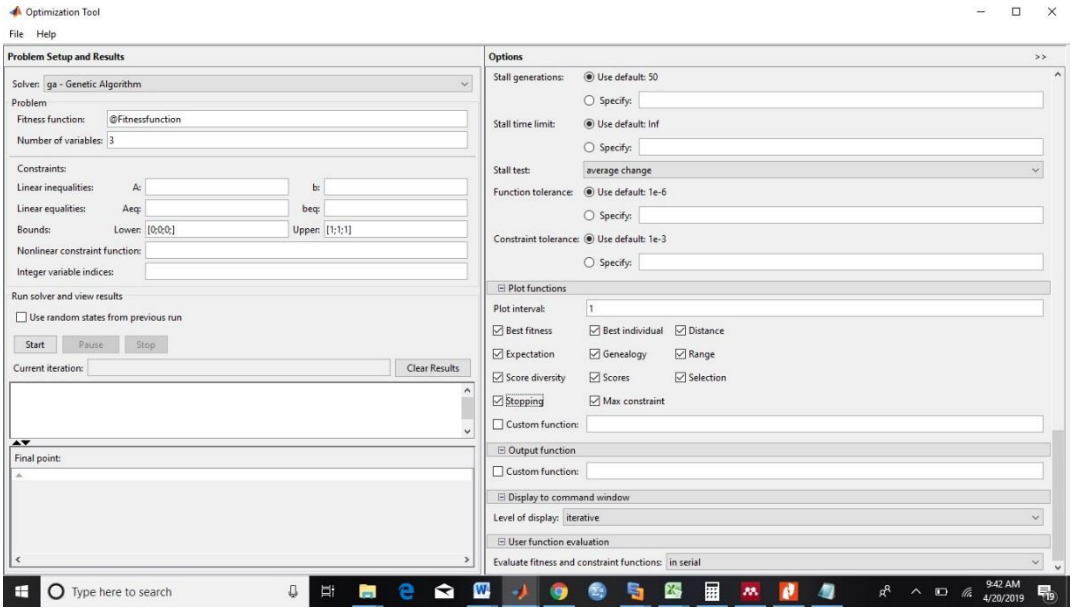

Figure 4.4: Genetic Algorithm Implementation Toolbox

The fitness value is asymptotically distributed along the generation. This clearly shows that the genetic classifier converges at a definite point as shown in Figure 4.5. The best fitness has a value of 1.2656×10^{-6} with mean value of 7.13018×10^{-6}.

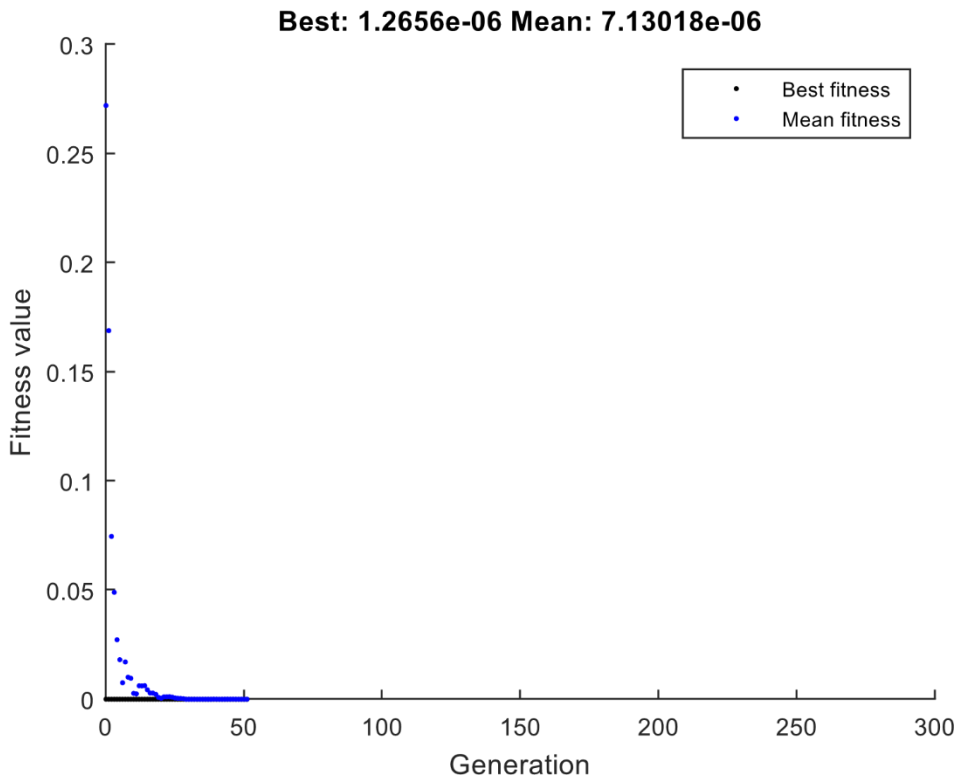

Figure 4.5: Plot of Fitness Value against Generation

At convergence point, current best individuals were recorded as second and third variables which are false positive and false negative as shown in Figure 4.6.

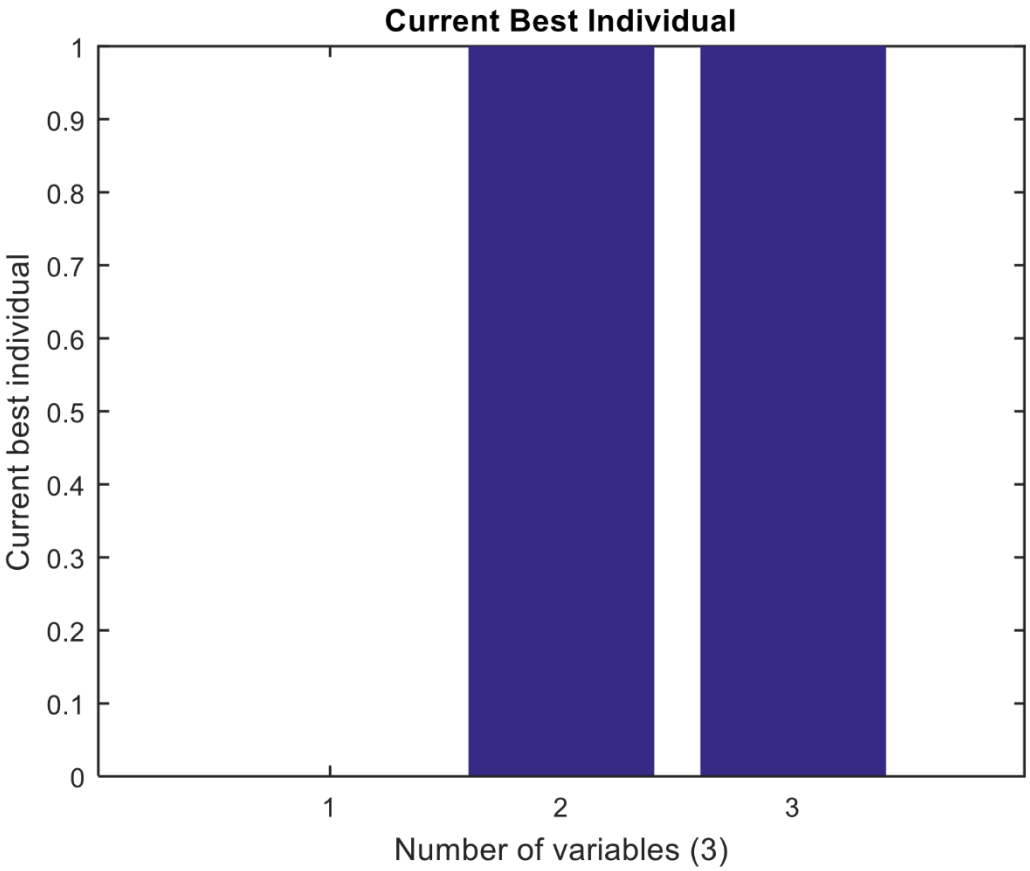

Figure 4.6: Current best individuals against Number of variables

The average distance between individual also shows a convergence trends as it asymptotically approaches a finishing line across the generations as shown in Figure 4.7.

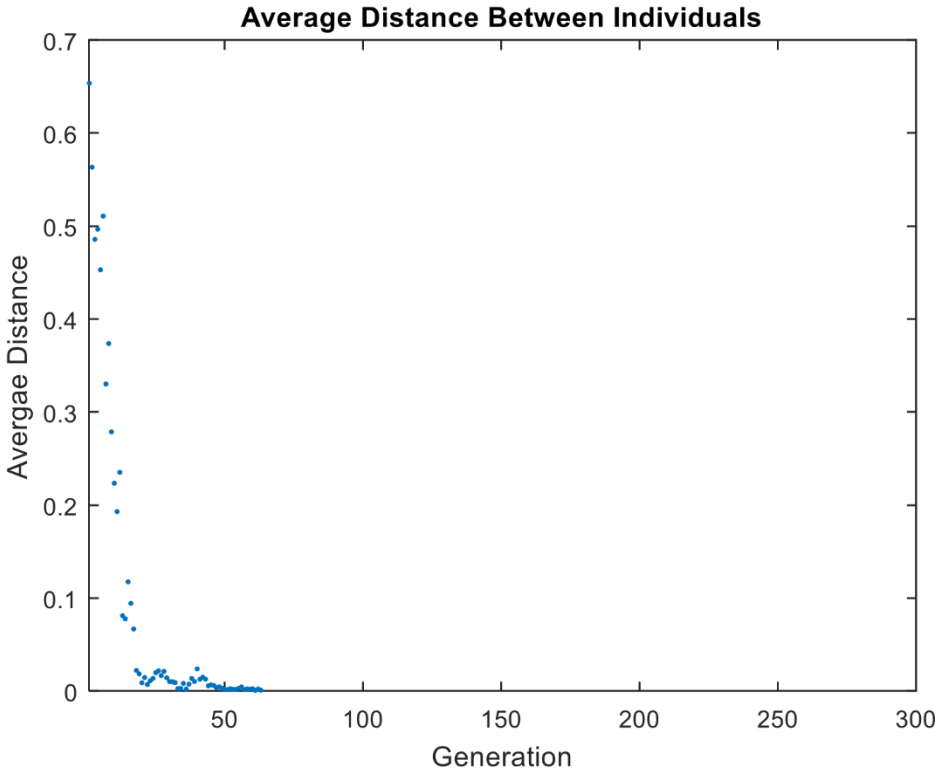

Figure 4.7: Average distance between individuals against Generation

In Figure 4.8, the individual against generation is fairly good, showing individual participation across generations.

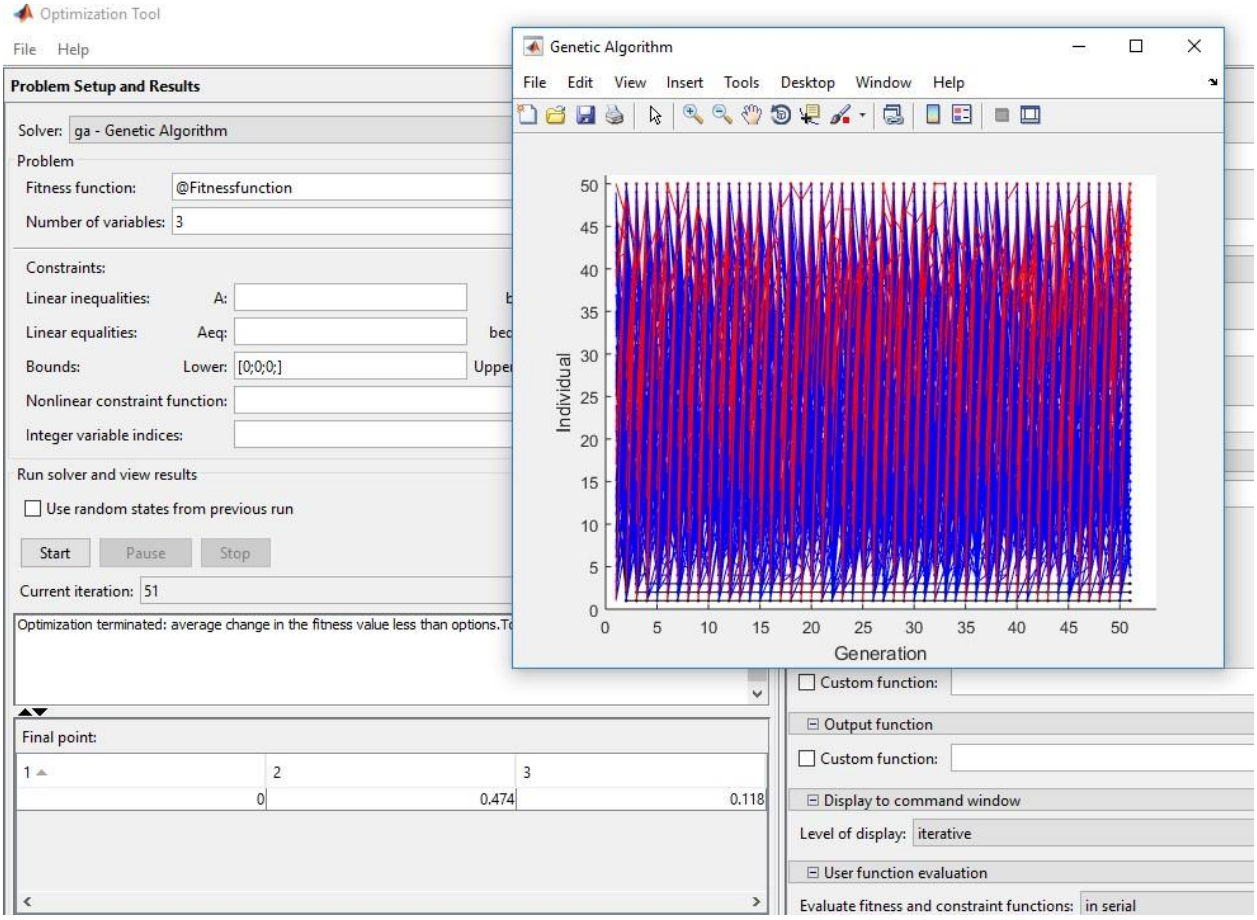

Figure 4.8: Individual against Generation

The best versus worst analysis across generations shows a good performance as presented in Figure 4.9. The best performance is shown by the histogram lines while the mean is indicated by the underlined shading. The worst case does not surface at all. Generally, this indicates a good performance for the genetic classifier.

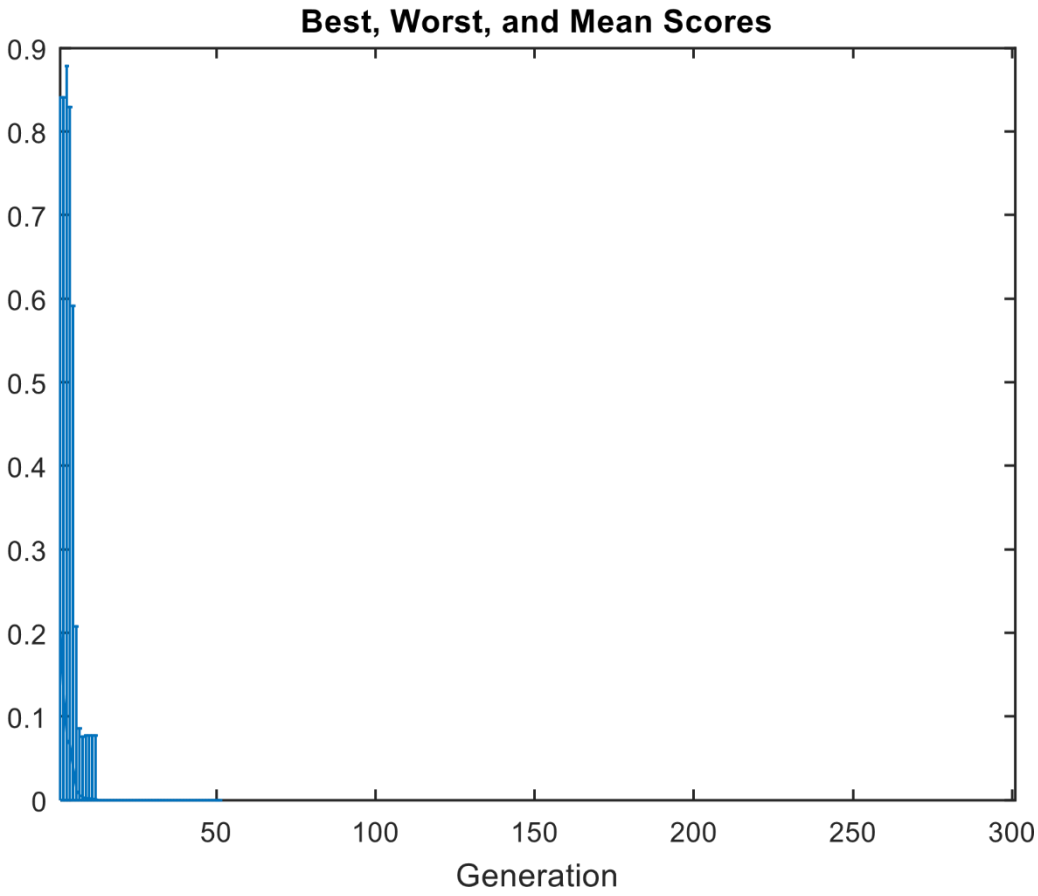

Figure 4.9: Best, Worst and Mean Scores against Generation

In Figure 4.10, the number of individuals with good score range is very high. Over 45 individuals was recorded at the point of high score range.

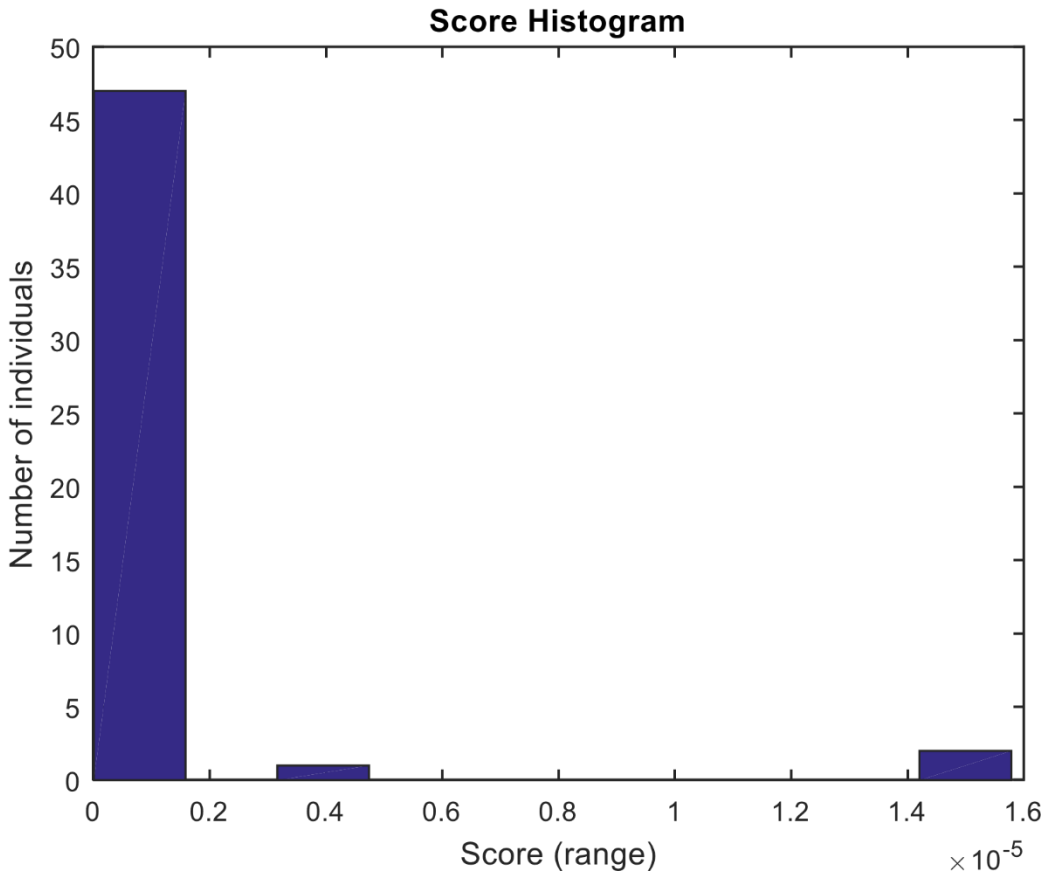

Figure 4.10: Number of Individuals against Score

The genetic classifier attains high fitted individuals at the 15th generation as shown in Figure

4.11.

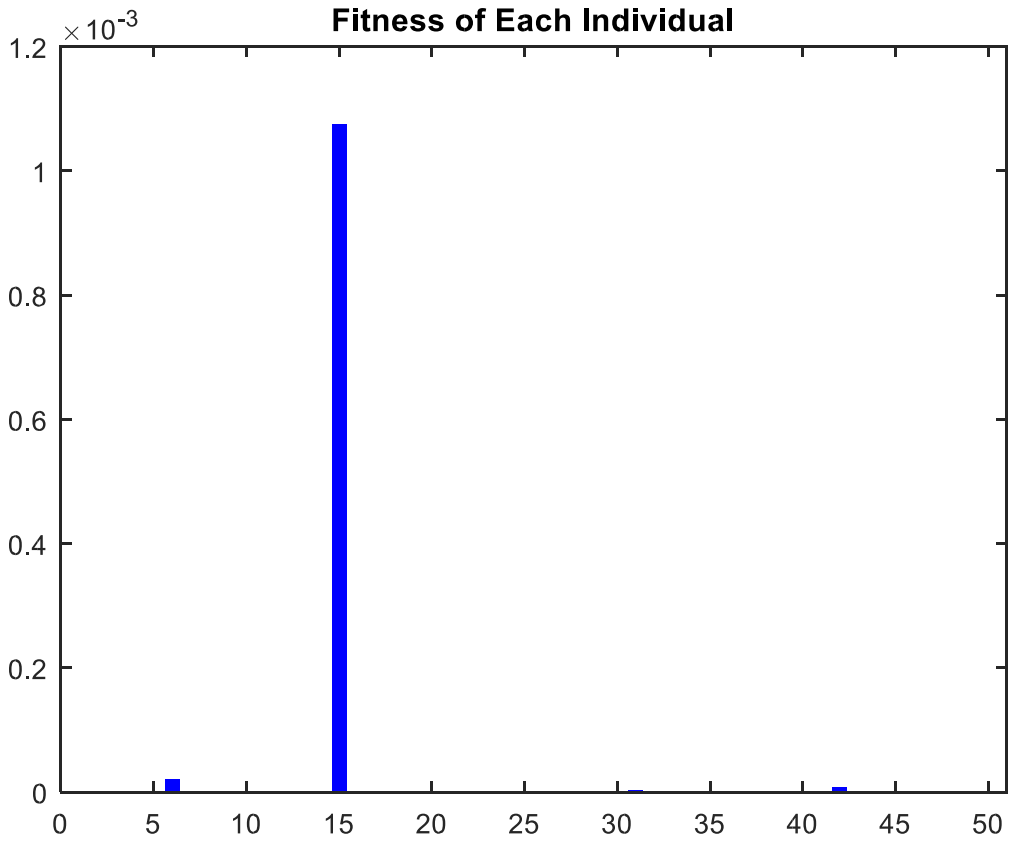

Figure 4.11: Fitness of Individuals against Generations

The classifier also shows good fitness of individual parents in reproducing fitted children. Average of five children was reproduced by the individual parent during chromosomes mating as shown in Figure 4.12.

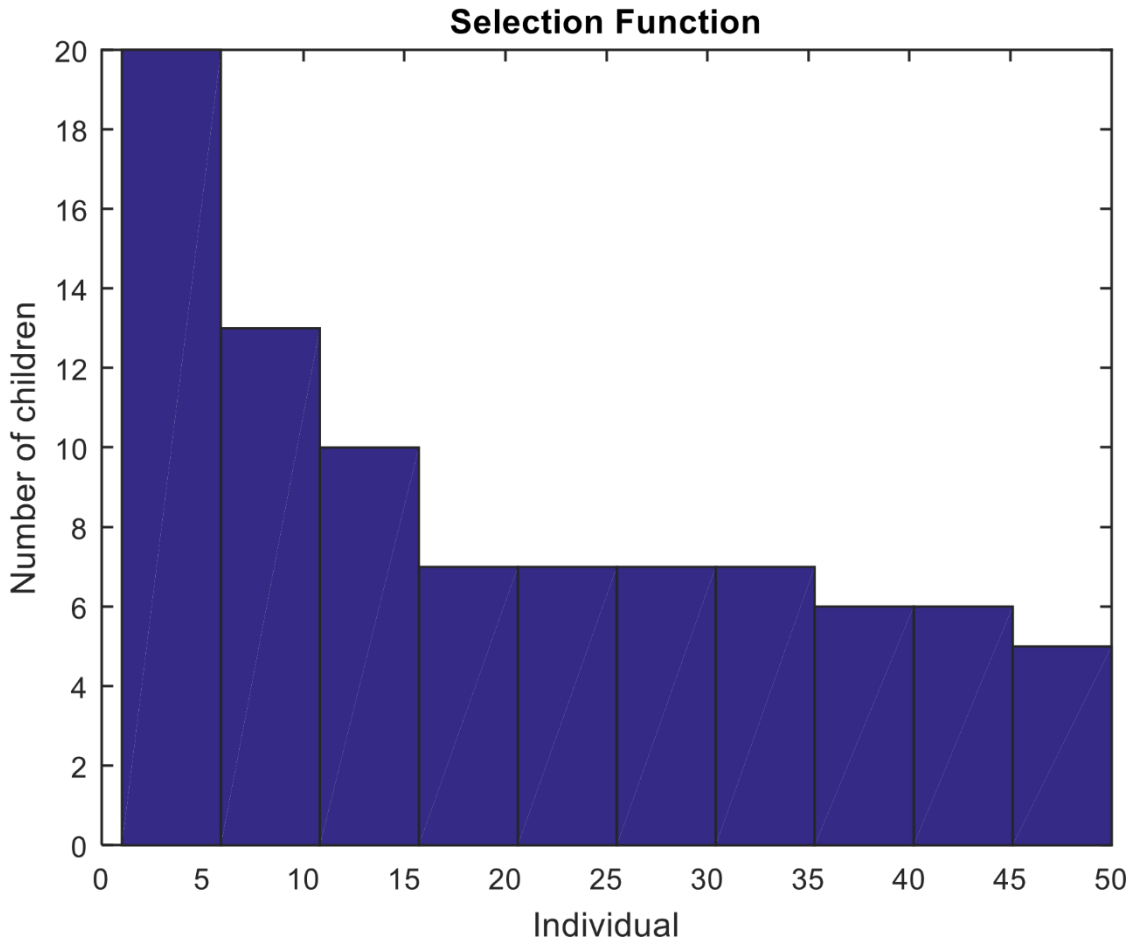

Figure 4.12: Number of Children against Individual

The stopping criteria were averagely okay. Over 20% stopping criteria was as a result of generational constraint while few were stalled as shown in Figure 4.13.

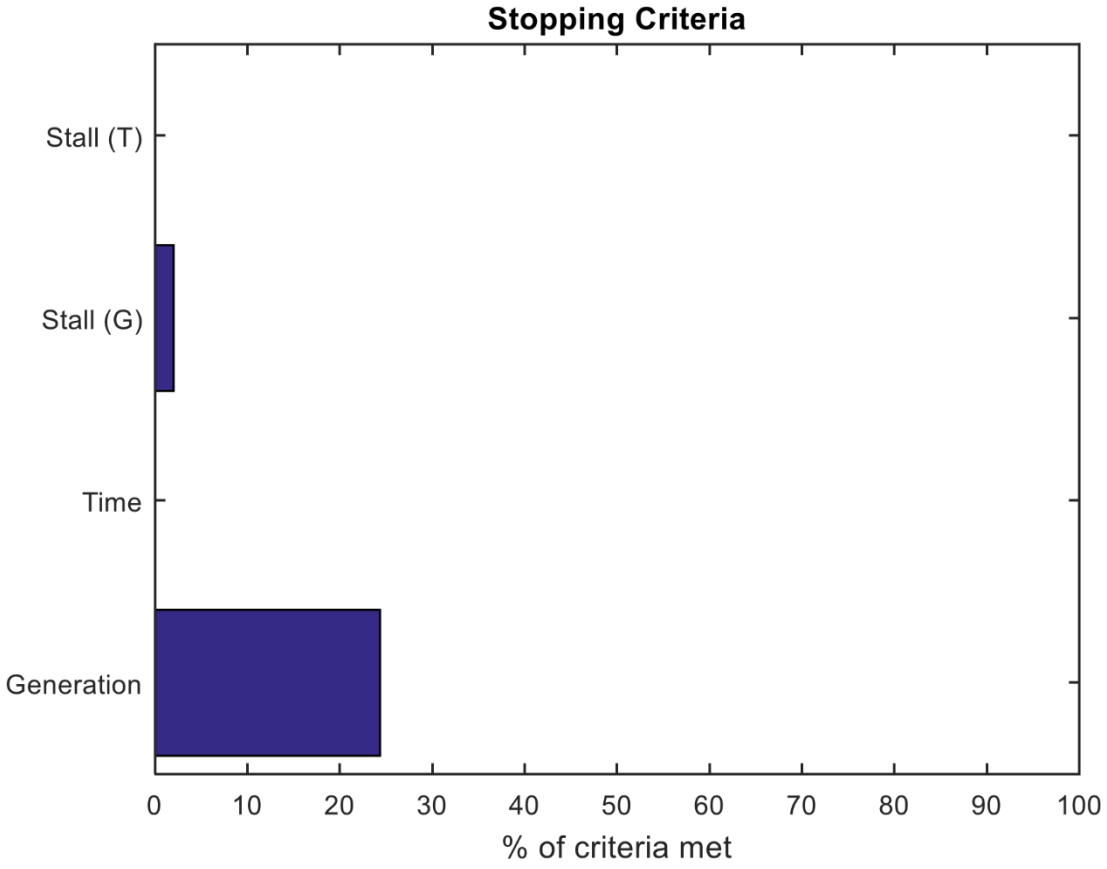

Figure 4.13: Percentage Analysis of Stopping Criteria

4.2 Discussion

The introduction of the GA for optimization-classification of the DDoS has shown that better performance is achievable. The performance evaluation of the neural GA classifier was done based on accuracy-based measures. In this research, we analyzed five classification algorithms and selected NN-GA as it provides better accuracy than the other four. The other conventional neural networks recorded lower classification accuracy compared to the genetically optimized classifier. The evaluation results show an absolute successful classification of all the attacks. Accuracy of 98.58% was achieved with detection rate of 96.49% and specificity of 95.97% as against the other conventional

NN results. The performance evaluations of the improved NN-GA with existing algorithms are presented in Table 4.2.

Table 4.2: Performance Evaluation of the Improved NN-GA with Existing Algorithms

	Accuracy %	**True Positive Rate(TPR) %**	**True Negative Rate (TPR) %**	**False Positive Rate (FPR) %**	**False Negative Rate (FNR) %**
MLP-ANN (UDP) (Sofi *et al*., 2017)	94.32	92.10	91.64	5.12	0.482
Decision Trees-UDP (Sofi *et al*., 2017)	92.23	90.40	89.2	5.87	0.56
Naïve Bayes-UDP (Sofi *et al*., 2017)	96.91	94.34	93.21	5.56	0.52
SVM using DARPA dataset (UDP attack) (Vijayasarathy, 2012)	88.50	91.42	94.60	5.71	0.558
NN-GA (Researcher 2021)	98.58	96.49	95.97	4.03	0.351

4.3 Implication of Findings

The results in the table 4.2 shows the new model NN-GA has better accuracy of 98.58% with lower false positive rate of 0.351 as against the other existing neural networks. Again, the confusion metrics depicted in Table 3.1 shows that, for most of the classes, this improved model performs well enough except normal data type which is because of ignoring non-numerical features. Comparing with the confusion metrics of the winning entry of KDD‟99, better detection rate for DDoS/user-to-root and close detection rate for probe & remote-to-local was achieved. The implication of this achievement is that DDoS attacks with low rate, low levels and mutating variant nature will be better detected than existing approaches.

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In this research, the author designed an improved model using refined genetic neural network algorithm to efficiently detect and provide adequate protection against network-based threats. To implement and measure the performance of the built model, the standard KDD99 benchmark dataset was used to obtain reasonable detection rate. Furthermore, to measure the fitness of the chromosomes, the standard deviation equations with distance were employed. It is envisaged that if better equations or heuristics could be employed in the detection process, the detection rate would improve to a great extent, especially the false positive rate would surely be much lower.

5.2 Recommendation

This study recommends that further experiment be performed using other DDoS datasets from different sources as well as other classification methods such as Bayesian, Support Vector Machine and K-Means to get an improved accurate result.

5.3 Contributions to Knowledge

The contributions of this research are:

i. An improved multi-step objective function based confidence and completeness factors.

ii. The improved genetically optimized neural network algorithm would improve the accuracy of the detector with little overhead.

iii. The improved technique will lower false detection rate.

REFERENCES

Aamir, M., & Arif, M. (2013). Study and performance evaluation on recent distributed denial of service trends of attack & defense. *International Journal of Information Technology and Computer Science, 5*(8), 54-65.

Adebayo, O. S., & AbdulAziz, N. (2014). An intelligence based model for the prevention of advanced cyber-attacks. In *the 5th International Conference on Information and Communication Technology for the Muslim World* (pp. 1-5). Minna, Nigeria : IEEE

Adebayo, O. S., & Abdul Aziz, N. (2019). Improved malware detection model with apriori association rule and particle swarm optimization. *Security and Communication Networks, 2019*. doi: 10.1155/2019/2850932

Adebayo, O. S., Mabayoje, M. A., Mishra, A., & Osho, O. (2012). Malware detection, supportive software agents and its classification schemes.*International Journal of Network Security & Its Applications (IJNSA),* 4(6), 58-65. Retrieved from http://repository.futminna.edu.ng:8080/jspui/handle/123456789/1960

Adebayo, O. S., Noel, M. D., Abdulmutalab, M., Baba, M., Abdulhamid, S. Í. M., & Suleiman, A. (2018). Performance analysis of classification algorithms for distributed denial of service attack detection in a distributed network environment. *International Conference on Information Technology on Education and Development (ITED),* 3(12), 67-75. Abuja, Nigeria: Base

Ahmed, M., & Mahmood, A. N. (2014). Network traffic analysis based on collective anomaly detection. *In 9th Conference on Industrial Electronics and Applications* (pp. 1141-1146). Hangzhou, China: IEEE.

Ahmed, M., & Mahmood, A. N. (2015). Novel approach for network traffic pattern analysis using clustering-based collective anomaly detection. *Annals of Data Science, 2*(1), 111-130.

Alenezi, M., & Reed, M. J. (2017). Methodologies for detecting denial of service distributed denial of service attacks against network servers. *In Seventh International Conference on System and Networks Communications (ICSNC).* (pp. 92-98). Lisbo, Portugal

Bace, R. G. (2012). *Intrusion detection and intrusion prevention devices*. Computer Security Handbook, 27 (1-27). doi: https://doi.org/10.1002/9781118851678.ch27

Bobor, V. (2006). Efficient intrusion detection system architecture based on neural networks and genetic algorithms. *Department of Computer and Systems Sciences, Stockholm University/Royal Institute of Technology*, 1(3), 88-94. doi:10.5120/89-188

Booth, T., & Andersson, K. (2017). Critical infrastructure network distributed denial of service defense, via cognitive learning. *In 4th IEEE Annual Consumer Communications & Networking Conference (CCNC)* (pp. 1-6). Las Vegas, NV, USA: IEEE.

Cardieri, P. (2010). Modeling interference in wireless ad hoc networks. *Communications Surveys & Tutorials*, 12(4), 551-572.

Crosbie, M., & Spafford, G. (2008). Applying genetic programming to intrusion detection. *In Working Notes for the Association for the Advancement of Artificial Intelligence (AAAI) Symposium on Genetic Programming* (pp. 1-8). Cambridge, MA: MIT Press.

Diaz-Gomez, P. A., & Hougen, D. F. (2006). A genetic algorithm approach for doing misuse detection in audit trail files. *In 15th International Conference on Computing* (pp. 329-338). Mexico City, Mexico: IEEE.

Elavarasi, M. (2016). Network forensics and its investigation methodology. *An International Journal Emergency. Trends Science. Technology*, *3*(05), 852-859.

Gavrilis, D., & Dermatas, E. (2013). Detection of web denial-of-service attacks using decoy hyperlinks. *In 5th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Patras*. doi:10.1.1.127.3245&rep=rep1&type=pdf

Gong, R. H., Zulkernine, M., & Abolmaesumi, P. (2015). A software implementation of a genetic algorithm based approach to network intrusion detection. *In Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network* (pp. 246-253). Towson, USA: IEEE.

Goyal, A., & Kumar, C. (2008). Genetic algorithm-network intrusion detection system: a genetic algorithm based network intrusion detection system. *Northwestern university*, *178*(15), 3024-3042.

Haddadi, F., Khanchi, S., Shetabi, M., & Derhami, V. (2016). Intrusion detection and attack classification using feed-forward neural network. *In 2010 Second International Conference on Computer and Network Technology* (pp. 262-266). Bangkok, Thailand: IEEE.

Hoque, N., Kashyap, H., & Bhattacharyya, D. K. (2017). Real-time distributed denial of service attack detection using field programmable gate array. *Computer Communications*, *110*, 48-58. Amsterdam: Elsevier

Ilgun, K., Kemmerer, R. A., & Porras, P. A. (1995). State transition analysis: A rule-based intrusion detection approach. *IEEE Transactions on Software Engineering*, *21*(3), 181-199, doi: 10.1109/32.372146

Jaafar, G. A., Abdullah, S. M., & Ismail, S. (2019). Review of recent detection methods for hyper text transfer protocol distributed denial of service attack. *Journal of Computer Networks and Communications*. doi:10.1155/2019/1283472

Jawale, M. D. R., & Bhusari, V. (2014). Technique to detect and classify attacks in network intrusion detection system using artificial neural network. *International Journal Emerging Research in Management Technology*, *3*(10), 75-81.

Karimazad, R., & Faraahi, A. (2017). An anomaly-based method for distributed denial of service attacks detection using radial basis function neural networks. *In Proceedings of the International Conference on Network and Electronics Engineering,* 1(15), 208-214. doi:10.1.1.735.1761&rep=rep1&type=pdf

Kayacik, H. G., Zincir-Heywood, A. N., & Heywood, M. I. (2005). Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets. *In Proceedings of the Third Annual Conference on Privacy, Security and Trust,* (94), 1723-1722. doi: 10.1.1.66.7574&rep=rep1&type=pdf

Kejie, Lu., Wu, D., Fan, J., Todorovic, S., & Nucci, A. (2017). Robust and efficient detection of distributed denial of service attacks for large-scale internet. *Computer Networks, 51*(18), 5036-5056. doi: 10.1016/j.comnet.2007.08.008

Kubus, M. (2020). Evaluation of resampling methods in the class unbalance problem. *Ekonometria*, *24*(1), 39-50. Retrieved from http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.desklight-002cb3e1-70f4-4321-a489-f4ced00e9d3b

Kumar, S., & Spafford, E. H. (1995). A software architecture to support misuse intrusion detection. *In Proceedings of the 18th National Information Security Conference.* 194-204. doi:10.1.1.159.2516

Lakshmi, V. N., & Begum, S. (2017). Distributed denial of service defense: Enhanced flooding detection and confidence-based filtering method. *Advances in Computational Sciences and Technology*, *10*(8), 2257-2272. Retrieved from http://www.ripublication.com/acst17/acstv10n8_07.pdf

Li, W. (2004). Using genetic algorithm for network intrusion detection. *Proceedings of the United States Department of Energy Cyber Security Group, 1,* 1-8. Retrieved from http://bit.csc.lsu.edu/~jianhua/krish-1.pdf

Li, Z., Wei, L., Li, W., Wei, L., Chen, M., Lv, M.,& Gao, N. (2019). Research on distributed denial of service attack detection based on extreme learning machine in internet of things environment. *In the 10th International Conference on Software Engineering and Service Science,* 144-148. Beijing, China: IEEE.

Lu, K., Wu, D., Fan, J., Todorovic, S., & Nucci, A. (2017). Robust and efficient detection of distributed denial of service attacks for large-scale internet. *Computer Networks, 51*(18), 5036-5056. doi: 10.1016/j.comnet.2007.08.008

Lu, W., & Traore, I. (2014). Detecting new forms of network intrusion using genetic programming. *Computational intelligence, 20*(3), 475-494. doi: 10.1111/j.0824-7935.2004.00247.x

Makridis, C. A., & Smeets, M. (2019). Determinants of cyber readiness. *Journal of Cyber Policy, 4*(1), 72-89. doi:10.1080/23738871.1604781

Moorthy, M., & Sathiyabama, S. (2012). A study of intrusion detection using data mining. *In International Conference on Advances in Engineering, Science and Management,* 8-15. Nagapattinam, India: IEEE.

Mualfah, D., & Riadi, I. (2017). Network forensics for detecting flooding attack on web server. *International Journal of Computer Science and Information Security, 15*(2), 326-331. Retrieved from https://sites.google.com/site/ijcsis/

Norouzian, M. R., & Merati, S. (2015). Classifying attacks in a network intrusion detection system based on artificial neural networks. *In 13th International Conference on Advanced Communication Technology,* 868-873. Gangwon, Korea (South): IEEE.

Pan, W., & Li, W. (2015). A hybrid neural network approach to the classification of novel attacks for intrusion detection. *In International Symposium on Parallel and Distributed Processing and Applications,* 564-575. Berlin, Heidelberg: Springer

Papalexakis, E. E., Beutel, A., & Steenkiste, P. (2012). Network anomaly detection using co-clustering. *In 2012 International Conference on Advances in Social Networks Analysis and Mining,* 403-410. Istanbul, Turkey: IEEE.

Pham, D. T., & Karaboga, D. (2000). Tabu Search. *In Intelligent Optimization Techniques,* 149-186. London: Springer

Poojitha, G., Kumar, K. N., & Reddy, P. J. (2011). Intrusion detection using artificial neural network. *In 2010 Second International Conference on Computing, Communication and Networking Technologies,* pp. 1-7. Karur, India: IEEE.

Rawat, A. S., Rana, A., Kumar, A., & Bagwari, A. (2018). Application of multilayer artificial neural network in the diagnosis system: a systematic review. *International Journal of Artificial Intelligence, 7*(3), 138-142. doi: 10.11591/ijai.v7.i3.pp138-142

Rodrigues, B., Bocek, T., Lareida, A., Hausheer, D., Rafati, S., & Stiller, B. (2017). A blockchain-based architecture for collaborative distributed denial of service mitigation with smart contracts. *In International Conference on Autonomous Infrastructure, Management and Security*,16-29. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-60774-0_2

Sofi, I., Mahajan, A., & Mansotra, V. (2017). Machine learning techniques used for the detection and analysis of modern types of distributed denial of service attacks. *Learning*, *4*(06). Retrieved from https://www.irjet.net/archives/V4/i6/IRJET-V4I6200.pdf

Su, M. Y. (2012). Using clustering to improve the k-nearest neighbor based classifiers for online anomaly network traffic identification. *Journal of Network and Computer Applications*, *34*(2), 722-730. doi: 10.1016/j.jnca.2010.10.009

Uppalaiah, B., Anand, K., Narsimha, B., Swaraj, S., & Bharat, T. (2012). Genetic algorithm approach to intrusion detection system. *International Journal of Computer Science and Technology*, *3*(1), 156-160. Retrieved from http://www.ijcst.com/vol31/1/uppaliah.pdf

Xia, T., Qu, G., Hariri, S., & Yousif, M. (2015). An efficient network intrusion detection method based on information theory and genetic algorithm. *In 24th IEEE International Performance, Computing, and Communications Conference, 2015*, 11-17. Phoenix, USA: IEEE.

Xiao-ming L., Gong Cheng., Qi, L. I., & Miano Z. (2012). A comparative study on flood denial of service and low-rate dos attacks. *The Journal of China Universities of Posts and Telecommunications*, *19*, 116-121. doi: 10.1016/S1005-8885(11)60458-5

Yu, W. Y., & Lee, H. M. (2009). An incremental-learning method for supervised anomaly detection by cascading service classifier and incremental tree inducer decision tree methods. *In Pacific-Asia Workshop on Intelligence and Security Informatics*, 155-160. Berlin, Heidelberg: Springer.

Zhang Chao-yang (2011). Denial of service attack analysis and study of new measures to prevent. *In 2011 International Conference on Intelligence Science and Information Engineering*, 426-429. Wuhan, China: IEEE.

Zhang, L. Y., Ming, Q. I. A. N., & CHI, Y. B. (2017). Distributed denial of service attack detection using sliding window method. *Destech Transactions on Computer Science and Engineering*, 531-535. doi: 10.12783/dtcse/wcne2017/19818

Zhao, Y., Zhou, F., Fan, X., Liang, X., & Liu, Y. (2013). Intrusion detection systems radar: a real-time visualization framework for intrusion detection system alerts. *Science China Information Sciences*, *56*(8), 1-12. doi: 10.1007/s11432-0-13-4891-9

APPENDIX A

Overview of Related Studies

S/No	Reference	Technique/ Method Used	Strength	Weakness
1.	Lu and Traore (2014)	It used historical network dataset by using GP to derive a set of classification	It used support-confidence framework as the fitness function and accurately classified several network intrusions	It made the implementation procedure very difficult and also for training procedure more data and time was required.
2.	Xia *et al*. (2015)	It used GA to detect anomalous network behaviours based on information theory	It was very effective because of the reduced complexity and higher detection rate	It considered only the discrete features.
3.	Gong *et al*. (2015)	It presented an implementation of GA based approach to Network Intrusion Detection using GA and presented software implementation.	It worked effectively for the selected datasets and has the flexibility to be used in different ways to meet users' special requirements.	It requires the whole training data to be loaded into memory before any computation. For large training datasets, it is neither efficient nor feasible.
4.	Kejie *et al*. (2017)	It proposed a framework to detect DDoS attacks and identi--fy attack packets efficiently.	It accurately detected DDoS attacks and identify attack packets without modifying existing IP forwarding mechanisms at the routers. It achieved 94.6% for detection probability using the proposed framework.	The simulation of the presented technique was provided in 'Weka'.
5.	Wei Pan and Weihua Li (2015)	It proposed a hybrid Neural Network consisting of a self-organizing map (SOM) and radial basis functions to detect and classify DDoS attacks	It achieved a satisfactory accuracy rate result for detecting and classifying DDoS attacks	It did not include modern attacks in different OSI layers such as transport layer in the work.
6.	Norouzian *et al*. (2015)	It proposed a new approach to IDS based on a MultiLayer Perceptron Neural Network to detect and classify data into 6 groups.	The research implemented MLP design with two hidden layers of neurons and achieved 90.78% accuracy rate.	The experimental result suggested that there is more to do in the IDS based on ANN.
7.	Haddadi, F, Sara Khanchi and others (2016)	It Proposed a NIDS using a 2-layered, feed-forward neural network.	It implemented the proposed system on a KDD cup 99 dataset, the result was very satisfactory, both on accuracy rate and performance.	It achieved equivalent performance and reduced computational overhead and memory usage.

8.	Reyhaneh Karimazad and Ahmad Faraahi (2017)	It proposed an anomaly-based DDoS detection approach using an analysis of network traffic.	It used a radial-based function (RBF) Neural Network on a UCLA dataset, achieving 93% accuracy rate for a DDoS attack.	large distinct frequencies in large space and large update/query time
9.	Jawale and Bhusari (2014)	It proposed a system that uses multilayer perceptions, back propagation and a support vector machine, consisting of multi modules such as packet collection and preprocessing data	It incorporates three well-known classification techniques: Multilayer Perceptron (MLP), Naïve Bayes and Random Forest.	During the transient period valid packets can be dropped
10.	Dimitris Gorillas and Evangelos Dermatas (2013)	It presented and evaluated a Radial-basis-function (RFB) Neural Network for DDoS attacks dependent on statistical vectors through short-time window analysis.	The proposed method was tested and evaluated in a controlled environment with an accuracy rate of 94% of DDoS detection.	During the transient period valid packets can be dropped

*****

APPENDIX B

Implementation Algorithms

```
% Setup the Genetic Algorithm
fitnessfunction= @ga_test;
N = 1310;  % number of optimization (decision) variables
popsize = 8 ; % set population size = number of chromosomes
max_iteration = 50;  % max number of iterations
minimum_cost = 120;  % minimum cost
mutation_rate = 0.01; % mutation rate
selection_rate = 0.5; % selection rate: fraction of population
nbits = 1;
Nt = nbits*N; % total number of bits in a chormosome
number_mutations = mutation_rate*N*(popsize-1); % number of mutations
% #population members that survive (Nkeep = Xrate*Npop); Nkeep survive for mating,
and (Npop - Nkeep) are discarded to make room for the new offspring
keep = floor(selection_rate*popsize);
iga=0; % generation counter initialized
pop=round(rand(popsize,Nt)); % random population of 1s and 0s
cost=feval(fitnessfunction,pop); % calculates population cost using fitnessfunction
[cost,ind]=sort(cost); % min cost in element 1
pop=pop(ind,:); % sorts population with lowest cost first
minc(1)=min(cost); % minc contains min of population
while iga < max_iteration  %Iterate through generations
iga=iga+1; % increments generation counter
% Pair and mate
M=ceil((M-keep)/2); % number of matings weights chromosomes based upon position in
list probability distribution function
prob=flipud([1:keep]'/sum([1:keep]));
odds=[0 cumsum(prob(1:keep))];
pick1=rand(1,popsize); % mate #1
pick2=rand(1,popsize); % mate #2
% parents contain the indicies of the chromosomes that will mate
ic=1;
while ic<=M
for id=2:keep+1
if pick1(ic)<=odds(id) & pick1(ic)>odds(id-1)
ma(ic)=id-1;
end % if
if pick2(ic)<=odds(id) & pick2(ic)>odds(id-1)
pa(ic)=id-1;
end % if
end % id
ic=ic+1;
end % while
%_____
```

```
% Performs mating using single point crossover
ix=1:2:keep; % index of mate #1
xp=ceil(rand(1,M)*(Nt-1)); % crossover point
pop(keep+ix,:)=[pop(ma,1:xp) pop(pa,xp+1:Nt)];
% first offspring
pop(keep+ix+1,:)=[pop(pa,1:xp) pop(ma,xp+1:Nt)];
% second offspring
%_____________________________________________________
% Mutate the population
number_mutations=ceil((popsize-1)*Nt*mutation_rate); % total number of mutations
mrow=ceil(rand(1,number_mutations)*(popsize-1))+1; % row to mutate
mcol=ceil(rand(1,number_mutations)*Nt); % column to mutate
for ii=1:number_mutations
pop(mrow(ii),mcol(ii))=abs(pop(mrow(ii),mcol(ii))-1);
end
%_____________________________________________________
% The population is re-evaluated for cost decode
cost(2:popsize)=feval(fitnessfunction,pop(2:popsize,:));
%_____________________________________________________
% Sort the costs and associated parameters
[cost,ind]=sort(cost);
pop=pop(ind,:);
%_____________________________________________________
% Stopping criteria
if iga>maxit | cost(1)<mincost
break
end
[iga cost(1)]
end
```

APPENDIX C

Objective Function on Confidence and Completeness Factor

```
function [y]=Fitnessfunction(x)

ConfidenceFitness = x(3)/(x(3)+x(4))

CompletenessFitness = x(3)/(x(3)+x(5))

%y=1/(1+(1/100)*(x(1)-x(2))^2)

y=ConfidenceFitness * CompletenessFitness
```

APPENDIX D

Objective Function on Error

```
function [ y ] = MinError( x )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
y = x(1)-x(2);
end
```

APPENDIX E

Neural Genetic Algorithm Codes

```
clear all; clc; close all;

%% Data Entry
% This section is to feed in data from the dataset in Excel Sheet
MyData=xlsread('NormalizedKDDDataset10PercentEncodedFull3.xlsx');

% This section is to enter both Input features and Output feature
Input=MyData(:,1:end-5);
Output=MyData(:,end-4:end);

%% Data Transpose Section
% This section is to transpose both the input and output data
InputTranspose=Input';
OutputTranspose=Output';

%% ANN Section,
% This section is to carry out computations using Artificial Neural Network
network=newff(InputTranspose,OutputTranspose,10)

%% ANN Training
% This section is .....
network=train(network,InputTranspose,OutputTranspose);
Output=network(InputTranspose);

%% Performance Evaluation
% This section is to evaluate the performance of your model

Error = Output-OutputTranspose;
[GAx,GAfval,GAexitflag,GAoutput,GApopulation,GAscores]=ga(@MinError,2,[],[],[],[]
,[0;0],[0.3;0.3])

Performance=perform(network,Output,OutputTranspose);

%% Simulation Section
% This section is to use the model for simulation

SimResult=sim(network,InputTranspose);

%% Thresholding Section...
% This section is to ...

ThreshSimResult=SimResult>0.5

%% Performance Evaluation

PerformanceParameters=classperf(OutputTranspose, ThreshSimResult)

[GA2x,GA2fval,GA2exitflag,GA2output,GA2population,GA2scores]=ga(@Fitnessfunction,
3,[],[],[],[],[0.5;0.5;0.5],[1;1;1])
```

APPENDIX F

Snippet of the Dataset Input features

L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	…	L41
21	tcp	telnet	SF	135	1290	0	0	0	0	0	1	0
85	tcp	telnet	SF	277	693	0	0	0	0	0	0	0
192	tcp	ftp	SF	119	426	0	0	0	2	0	1	0
179	tcp	ftp	SF	87	319	0	0	0	1	0	1	0
0	tcp	ftp_data	SF	866	0	0	0	0	0	0	0	0
0	tcp	ftp_data	SF	0	467968	0	0	0	0	0	0	0
1	tcp	ftp_data	SF	0	988002	0	0	0	0	0	0	0
198	tcp	telnet	SF	562	9139	0	0	0	3	0	1	22
718	tcp	telnet	SF	1412	25260	0	0	0	15	0	1	38
0	tcp	telnet	S0	0	0	0	0	0	0	0	0	0
0	tcp	telnet	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0
0	tcp	private	S0	0	0	0	0	0	0	0	0	0

APPENDIX G

Dataset Description and labels

The 41 Input Features captioned as L1 to L41 are labelled as follows:

1. duration: continuous.
2. protocol_type: symbolic.
3. service: symbolic.
4. flag: symbolic.
5. src_bytes: continuous.
6. dst_bytes: continuous.
7. land: symbolic.
8. wrong_fragment: continuous.
9. urgent: continuous.
10. hot: continuous.
11. num_failed_logins: continuous.
12. logged_in: symbolic.
13. num_compromised: continuous.
14. root_shell: continuous.
15. su_attempted: continuous.
16. num_root: continuous.
17. num_file_creations: continuous.
18. num_shells: continuous.
19. num_access_files: continuous.
20. num_outbound_cmds: continuous.
21. is_host_login: symbolic.
22. is_guest_login: symbolic.
23. count: continuous.
24. srv_count: continuous.
25. serror_rate: continuous.
26. srv_serror_rate: continuous.
27. rerror_rate: continuous.
28. srv_rerror_rate: continuous.
29. same_srv_rate: continuous.
30. diff_srv_rate: continuous.
31. srv_diff_host_rate: continuous.
32. dst_host_count: continuous.
33. dst_host_srv_count: continuous.

34. dst_host_same_srv_rate: continuous.
35. dst_host_diff_srv_rate: continuous.
36. dst_host_same_src_port_rate: continuous.
37. dst_host_srv_diff_host_rate: continuous.
38. dst_host_serror_rate: continuous.
39. dst_host_srv_serror_rate: continuous.
40. dst_host_rerror_rate: continuous.
41. dst_host_srv_rerror_rate: continuous.

The output feature consists of 23 classes of attacks. These are as follows: back,buffer_overflow,ftp_write,guess_passwd,imap,ipsweep,land,loadmodule,multihop,neptune,nmap,normal,perl,phf,pod,portsweep,rootkit,satan,smurf,spy,teardrop,warezclient,warezmaster.