LAUTECH Journal of Computing and Informatics (LAUJCI) – ISSN : 2714-4194
Volume 1– No.1, Sept. 2020 – www.laujci.lautech.edu.ng

Improved Genetically Optimized Neural Network Algorithm for Classification of Distributed Denial of Service Attack

Emmanuel Hamman Gadzama
Department of Cyber Security Science,
Science, Federal University of Technology,
Minna, Nigeria
ehgadzama@gmail.com

Olawale Surajudeen Adebayo, PhD
Department of Cyber Security
Federal University of Technology,
Minna, Nigeria
waleadebayo@futminna.edu.ng

ABSTRACT

This paper proposes a classification of distributed denial of service (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. In order 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 show the improved genetically optimized neural network algorithm has better accuracy and lower false positive rate in comparison with the conventional neural network.

Key words: DDoS, genetic algorithm, neural network, naïve bayes, machine learning.

1. INTRODUCTION

Distributed Denial of Service (DDoS) is a network security problem that continues to grow dynamically and has increased significantly to date. DDoS is a type of attack that is carried out by draining the available resources in the network by flooding the package with a significant intensity so that the system becomes overloaded and stops (Lu *et al.*, 2014). This attack results in enormous losses for institutions and companies engaged in online services. Prolonged deductions and substantial recovery costs are additional losses for the company due to loss of integrity. 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 (Pan and Li, 2015). However, these methods have suffered from low accuracy, completeness and confidence factors with undisclosed high false positive rate.

As the Internet becomes an essential part of human life, providing security of data passed over the internet is getting more crucial. The Internet was initially designed for openness and scalability without any security concern. Hence, malicious users exploit this weakness to achieve their purpose. In recent years, the number of network-based threats has been significantly increased. Distributed denial of service (DDoS) attacks are one of the major types of these threats. 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 *Global Cyber security Index* (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 attack and preserving the confidentiality, integrity and availability of web services and computer network resources.

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 different type 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. One 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 that is 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 different type of modern DDoS attack is an SQL Injection Dos (SIDDoS) 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.

It is pertinent to report that most of the common open access data sets have duplicated and redundant instances, which make the detection and classification of the DDoS unrealistic and ineffectual. Machine learning is usually used to detect and classify network traffic based on some features used to measure and determine if the network traffic is normal or is a type of DDoS. The number of packets will 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. DDoS packets all the time 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.

Artificial Neural Network is a biologically inspired computing model consisting of various processing elements (neurons). Neurons are connected to elements or weights that build the structure of neural networks. ANN has elements for processing information, namely transfer functions, weighted inputs, and output. (Rawat, Rana, Kumar and Bagwari 2018). A Genetic Algorithm (GA) is said to be a programming technique that mimics biological evolution as a problem-solving strategy (Bobor 2006). It is based on Darwinian's principle of evolution and survival of fittest to optimize a population of candidate solutions towards a predefined fitness (Li 2004). 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". During the process of evaluation "Crossover" is used to simulate natural reproduction and "Mutation" is used to mutation of species (Li 2004). Also, for survival and combination, the selection of chromosomes is biased towards the fittest chromosomes. When GA is used for Σsolving 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: 1) The fitness function, 2) The representation of individuals and 3) The GA parameters. The determination of the above-mentioned factors often depends on applications and/or implementation. The optimization of genetic algorithm with neural network for classification of distributed denial of service shown better performance is achievable.

The rest of the paper is organized as follows: related work is discussed in section two while section three contains experimental settings. Section four is results and discussion while conclusion and future work were contain in section five.

2. RELATED WORKS

Several researchers have 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. 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 Genetic Programming (GP) to detect network anomalies (1995). The two agents used GP to determine anomalous network behaviors 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) described a method 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) used historical network dataset by using GP to derive a set of classification Both Agents 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 behaviors 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.

Chitturs (1995) in 2001 offered a novel approach to detect the malicious intrusions (hacks) by using a complex artificial intelligence method known as GA applied to IDS. The researcher 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. (2004) in 2005 represented 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. (2004) in 2006 used the evolution process set of probable solutions which were generated randomly. In that experiment, they evaluated each chromosomes using fitness function. They also used single point crossover and single point mutation. In their research they performed GA for offline Intrusion Detection. As a result they tested the system by implementing different formulas for fitness function. They 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 showed 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.

YU and Lee (2010) proposed an incremental learning method which was called incremental tree inducer (ITI). They stated 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, Kumar and Reddy (2011) applied neural network to train samples from KDD'99. Their method was able to simply feed forward neural networks trained by the back-propagation algorithm to classify the abnormal events. They reported the power of their algorithm to find 1500 DDoS attacks in the testing dataset. Su (2012) collected its own 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. He initially applied MLBG clustering 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, Beutel and Steenkiste (2012) utilized the soft clustering to find different types of attacks in KDD'99. They reported an overall accuracy of 75% and 85% for normal and attack respectively. Dimitris and Dermatas (2013) 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 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 their selected subset of the DARPA dataset was DDoS attacks and they researched 94% accuracy to detect anomalies in the dataset. Jawale and Bhusari (2014) presented research on ANN that achieved the highest accuracy rate. They 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. They also used the KDD'99 /DARPA datasets to train and test their method. They reported the ability of their algorithm to find all available DDoS attacks in test data. In addition, a hybrid Neural Network technique was used by Wei and Li (2015), who proposed a hybrid Neural Network consisting of a self-organizing map (SOM) and radial basis functions to detect and classify DDoS attacks. The proposed technique achieved a satisfactory accuracy rate result for detecting and classifying DDoS attacks. Norouzian *et al.*, (2015) presented a most effective classification technique for detecting and classifying attacks into two groups normal or threat. They proposed a new approach to IDS based on a MultiLayer Perceptron Neural Network to detect and classify data into 6 groups. They implemented their MLP design with two hidden layers of neurons and achieved 90.78% accuracy rate.

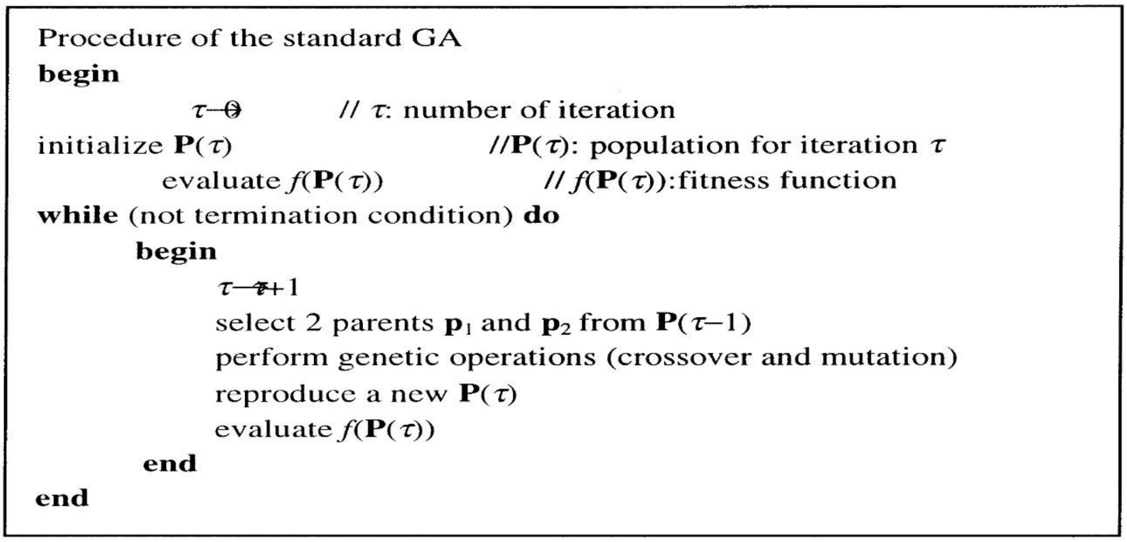
A NIDS using a 2-layered, feed-forward neural network was proposed by Sara Khanchi *et al.*, (2016). The proposed system classified normal connections and attacks. Different types of attacks were determined, and they focused on using training function, data validation and a preprocess dataset that caused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

Reyhaneh and Faraahi (2017) proposed an anomaly-based DDoS detection approach using an analysis of network traffic. A radial-based function (RBF) Neural Network was used in this approach, and they tested their method on a 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 technique 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. An overview and broad classification of IDS was presented by Mohammed and Reed (2017). The difficulties and characteristics of DDoS attacks were discussed in the research. Three different classifications were chosen. They focused on general DDoS and flooding attacks. The CUSUM approach had many advantages over statistical techniques which was effectively demonstrated in the research.

Recent study by Hoque, Kashyap and Bhattacharyya (2017) proposed a new DDoS detection framework which was implemented on software as well as hardware using the Field Programmable Gate Arrays (FPGA) device. The proposed method solely considered the DDoS attack detection as a 2-class problem. The proposed model created 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 Standard Genetic Algorithm Process

The standard GA process is shown in Figure 1. First, a population of chromosomes is created. Second, the chromosomes are evaluated by a defined fitness function. Third, some of the chromosomes are selected for performing genetic operations. Forth, genetic operations of crossover and mutation are performed. The produced offspring replace their parents in the initial population. In this reproduction process, only the selected parents in the third step will be replaced by their corresponding offspring. This GA process 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.



```
Procedure of the standard GA
begin
        $\tau \leftarrow 0$            // $\tau$: number of iteration
initialize $\mathbf{P}(\tau)$            //$\mathbf{P}(\tau)$: population for iteration $\tau$
        evaluate $f(\mathbf{P}(\tau))$        // $f(\mathbf{P}(\tau))$:fitness function
while (not termination condition) do
    begin
        $\tau \leftarrow \tau + 1$
        select 2 parents $\mathbf{p}_1$ and $\mathbf{p}_2$ from $\mathbf{P}(\tau-1)$
        perform genetic operations (crossover and mutation)
        reproduce a new $\mathbf{P}(\tau)$
        evaluate $f(\mathbf{P}(\tau))$
    end
end
```

Figure 1: Procedure of the Standard Genetic Algorithm
(D. T. Pham and D. Karaboga 2000)

2.2 Related Work on Fitness Function

Goyal and Kumar (2008) presented Genetic Algorithm to identify the attack type of connection, the algorithm used different features in network connections to generate a classification rule set; they used the fitness function given by the formula;

$$F = \frac{a}{A} - \frac{b}{B} \qquad (2.1)$$

Where:

A: Total of attacks.
a: Number of attack connections the individual correctly classified.
B: Normal connections in the population.
b: number of normal connections a network correctly classified.
They set a threshold value of 0.95; they select the individual which have a fitness value > 0.95.

Uppalaiah, B., Anand, K., Narsimha, B., Swaraj, S., Bharat, T (2012) used GA to detect Denial of Service (DOS) and Probe type of attacks, they used a fitness function:

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

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

Li, W. (2004) Used Genetic Algorithm for Intrusion Detection System, he calculated the fitness function by calculate the following four equations:

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

$$\Delta = |Outcome - SuspiciousLevel| \quad (2.4)$$

$$Penality = \frac{\Delta * Ranking}{100} \quad (2.5)$$

$$Fitness = 1 - Penality \quad (2.6)$$

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.

3. EXPERIMENTAL SETTING

3.1 The Proposed Rresearch Design

The research 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 genetic algorithm optimization. The genetically optimized neural network classifier was developed. The use of improved GA with neural network optimized the performance of the new algorithm for effective detection. This process can also be termed as essemble of machine learning algorithm. The essemble of ANN and GA amongst several machine leaning cannot be overemphasized. ANN is a widely accepted machine learning method that uses past data to predict future trend, while GA is an algorithm that can find better subsets of input variables for importing into ANN, hence enabling more accurate prediction by its efficient feature selection. During the performance evaluation, comparative analysis of the conventional neural network classification and proposed classifier (NN-GA) was done. The diagram in figure 2 shows the block diagram of the research design.

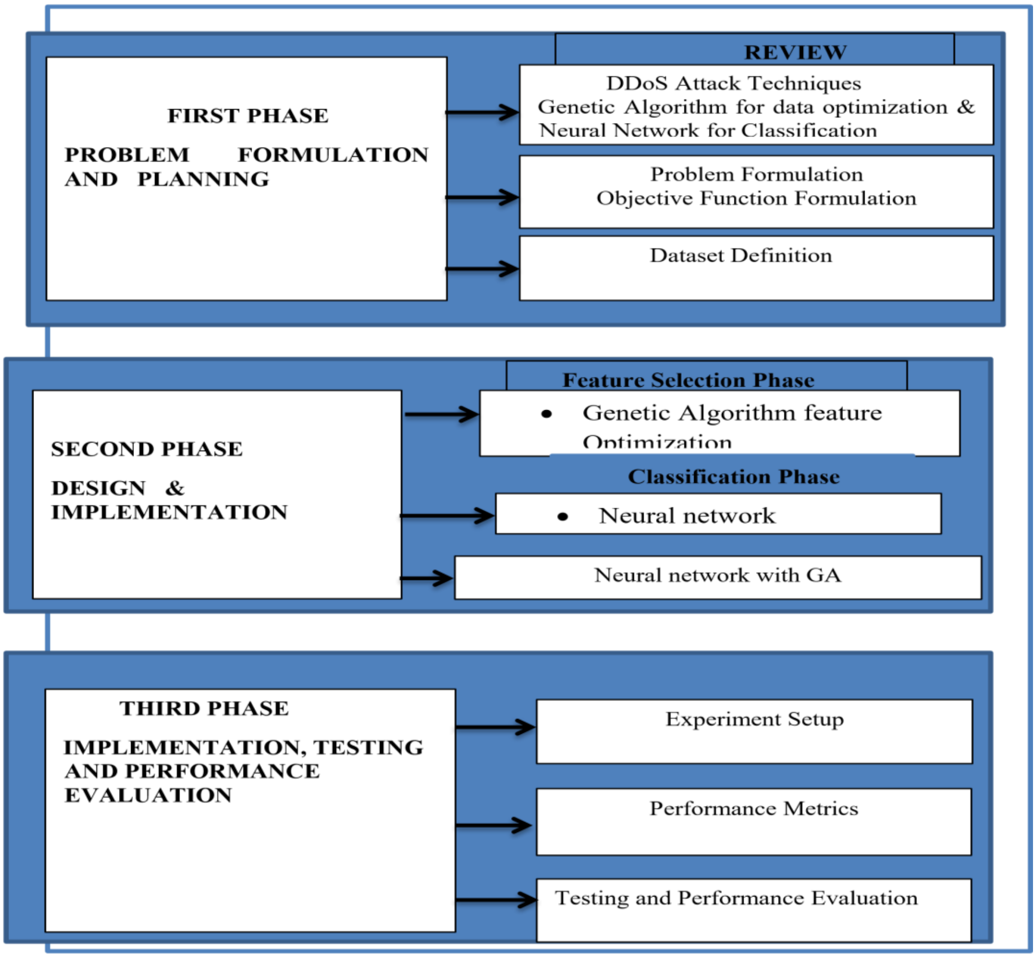
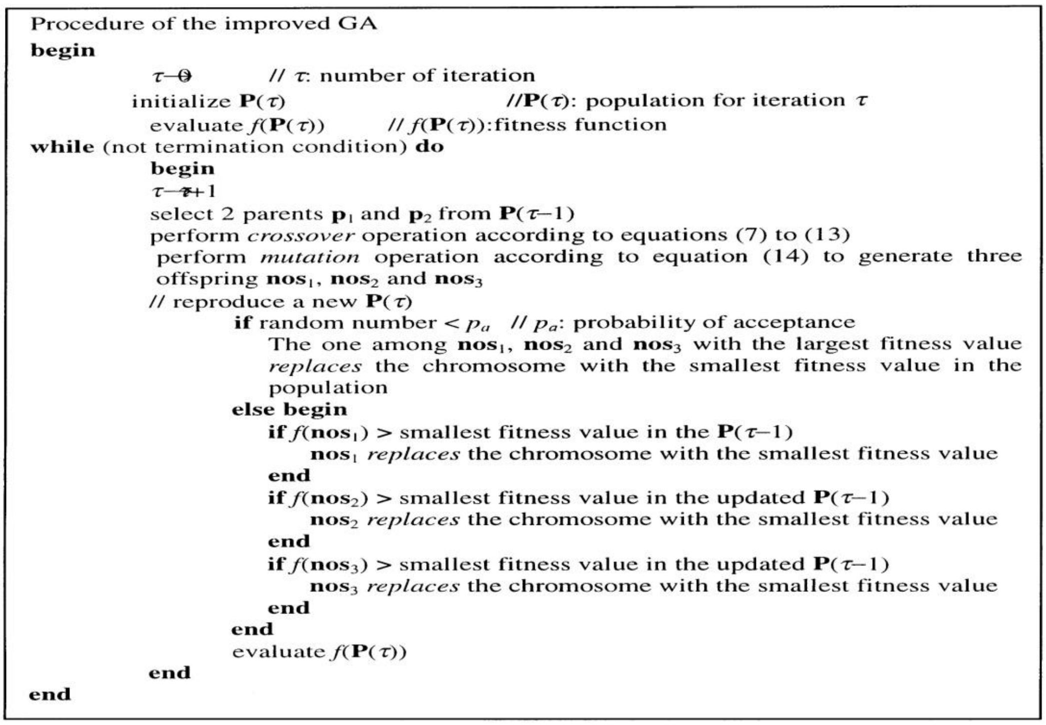

REVIEW
FIRST PHASE
PROBLEM FORMULATION AND PLANNING
DDoS Attack Techniques
Genetic Algorithm for data optimization & Neural Network for Classification
Problem Formulation
Objective Function Formulation
Dataset Definition
SECOND PHASE
DESIGN & IMPLEMENTATION
Feature Selection Phase
Genetic Algorithm feature Optimization
Classification Phase
Neural network
Neural network with GA
THIRD PHASE
IMPLEMENTATION, TESTING AND PERFORMANCE EVALUATION
Experiment Setup
Performance Metrics
Testing and Performance Evaluation

Figure 2: Block Diagram of the Research Design

3.2 Genetic Algorithm of the Proposed System

In Figure 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 the 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 is modified and new genetic operators are introduced to improve its performance as shown in Figure 3.



```
Procedure of the improved GA
begin
            $\tau \leftarrow 0$         // $\tau$: number of iteration
          initialize $\mathbf{P}(\tau)$                //$\mathbf{P}(\tau)$: population for iteration $\tau$
            evaluate $f(\mathbf{P}(\tau))$        // $f(\mathbf{P}(\tau))$:fitness function
while (not termination condition) do
            begin
            $\tau \leftarrow \tau+1$
            select 2 parents $\mathbf{p}_1$ and $\mathbf{p}_2$ from $\mathbf{P}(\tau-1)$
            perform crossover operation according to equations (7) to (13)
            perform mutation operation according to equation (14) to generate three
            offspring $\mathbf{nos}_1$, $\mathbf{nos}_2$ and $\mathbf{nos}_3$
            // reproduce a new $\mathbf{P}(\tau)$
                  if random number < $p_a$   // $p_a$: probability of acceptance
                     The one among $\mathbf{nos}_1$, $\mathbf{nos}_2$ and $\mathbf{nos}_3$ with the largest fitness value
                     replaces the chromosome with the smallest fitness value in the
                     population
                  else begin
                     if $f(\mathbf{nos}_1)$ > smallest fitness value in the $\mathbf{P}(\tau-1)$
                        $\mathbf{nos}_1$ replaces the chromosome with the smallest fitness value
                     end
                     if $f(\mathbf{nos}_2)$ > smallest fitness value in the updated $\mathbf{P}(\tau-1)$
                        $\mathbf{nos}_2$ replaces the chromosome with the smallest fitness value
                     end
                     if $f(\mathbf{nos}_3)$ > smallest fitness value in the updated $\mathbf{P}(\tau-1)$
                        $\mathbf{nos}_3$ replaces the chromosome with the smallest fitness value
                     end
                  end
                  evaluate $f(\mathbf{P}(\tau))$
            end
end
```

Figure 3. Improved Algorithm: Predict data/intrusion type (using GA)

3.3 Objective Function Formulation

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:

$$FitnessFcn1 = X_{co} - X_{gt} \qquad (3.1)$$

Where
X_{co} = Classier Output
X_{gt} = Groundtruth

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,

$$Confidence\ factor = \frac{TP}{TP+FP} \qquad (3.2)$$

Where TP is the number of samples that are correctly classified
FP is the number of samples that are incorrectly classified

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

$$Completeness = \frac{TP}{TP+FN} \qquad (3.3)$$

Where FN is the number of false negative of the considered class. So, the second objective function is expressed as:

$$FitnessFcn2 = confidence \times completeness \qquad (3.4)$$

3.4 The Proposed Model

The proposed model was designed by optimizing KDD DDOS features using genetic algorithm. 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. Figure 4 shows the flowchart for the proposed optimization/classification model. The distributed denial of service data was optimized using genetic algorithm. 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.

Flowchart of the Proposed Model

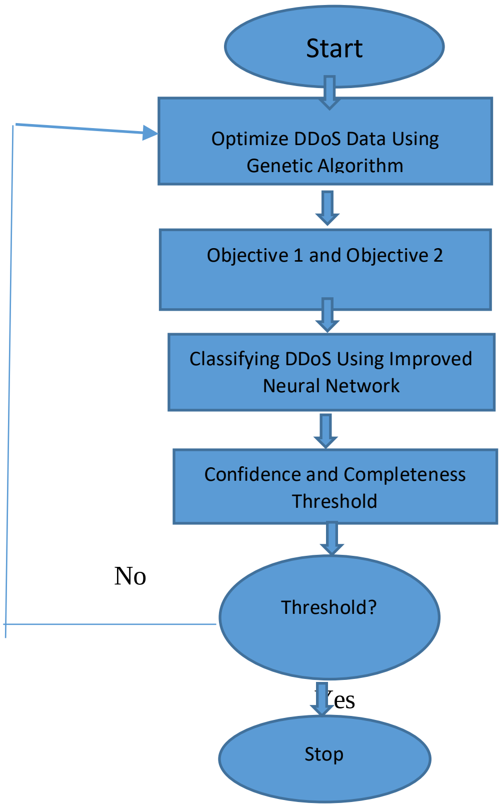

Start
Optimize DDoS Data Using Genetic Algorithm
Objective 1 and Objective 2
Classifying DDoS Using Improved Neural Network
Confidence and Completeness Threshold
No
Threshold?
Yes
Stop

Figure 4. Flowchart of the Proposed Optimization/Classification Model

3.5 Implementation Procedure

The pre-calculation phase have made 23 groups of chromosomes according to training data. We have 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. Thus, 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 1 (one). Amongst the extracted features of the datasets, we have taken only the numerical features, both continuous and discrete, under consideration for the sake of the simplification of the implementation.

3.6 Dataset and Data Processing

KDD 99 intrusion detection dataset was collected and used for the implementation of this improved algorithm, The KDD 99 intrusion detection datasets are based on the 1998 DARPA initiative, which offers designers of intrusion detection systems (IDS) with a benchmark on which to evaluate different methodologies. A simulation is therefore being made of a factitious military network with three „target‟ machines running various operating systems and services. Three additional machines to spoof different IP addresses for generate network traffic was also used. It is pertinent to highlight that, a connection is a sequence of TCP packets starting and ending at some well-defined times, between which data flows from a source IP address to a target IP address under some well-defined protocol. It results in 41 features for each connection. Finally, there is a sniffer that records all network traffic using the TCP dump format. The total simulated period is seven weeks.

Typical connections are created to profile that expected in a military network and attacks fall into one of four categories: User to Root; Remote to Local; Denial of Service; and Probe. Note that the KDD 99 intrusion detection benchmark consists different components:kddcup data; kddcup.data_10_percent; kddcup.newtestdata_10_percent_unlabeled; kddcup.testdata.unlabeled; kddcup.testdata.unlabeled_10_percent; corrected. For this research, “kddcup.data_10_percent” was used as training dataset and “corrected” as testing dataset. In this case the training set consists of 494,021 records among which 97,280 are normal connection records, while the test set contains 311,029 records among which 60,593 are normal connection records. In Table 1, we see the distribution of each intrusion type in the training and the test set. Table 1 is showing the distribution of intrusion categories in the datasets.

Table 1. 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 train the features with the classifier, the necessary features were encoded into binary digits and transformed into n by m dimension matrix for effective and effeicient training. This is because both neural network and genetic algorithm operates optimally with binary digits. Encoding processing approach was used to digitize the 23 classes of attacks in the dataset. In order to accommodate 23 attack classes, 5 bits were 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. 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 genetic algorithm 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.

3.7 Preprocessed Dataset Classes of Attacks

The result of 5-bit encoding of the 23 attack classes in the chosen dataset is presented in Table 2. These 5-bit encoded values are the output features of the dataset.

Table 2. 5-bit Encoded DDoS Attacks

S/N	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

3.7 Performance Evaluation Metric

Accuracy-based measures would be used to evaluate the classifier. The accuracy-based measures are the metrics that have to do with correction classification rate. The measures will comprise of:

3.7.1 Confusion Matrix

Confusion Matrix as the name suggests 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, thus:

Table 3. Confusion Matrix

		Predicted Class	
		Normal	Malicious
Actual Class	Normal Web page	TN	FP
	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.

3.7.2 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:

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

3.7.3 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.

$$\text{True Positive Rate} = \frac{\text{True Positive}}{\text{False Negative + True Positive}} \qquad (3.6)$$

3.7.4 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.

$$\text{False Positive Rate} = \frac{\text{False Positive}}{\text{False Positive + True Negative}} \qquad (3.7)$$

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.

3.7.5 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.

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

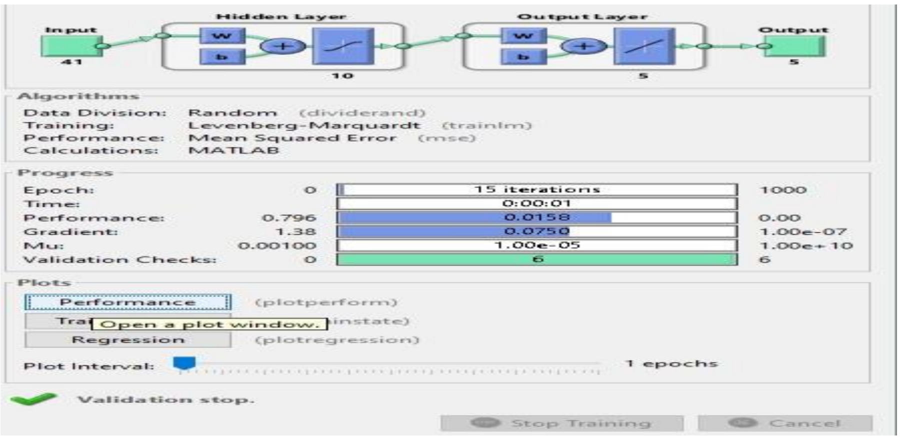
3.7.6 *Regression*

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 (when they gave 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.

4. RESULTS AND DISCUSSION

4.1 Results

The neural network-based classification results are presented in Figure 5. The architecture consists of 41 inputs, 10 neurons in the hidden layer and five output nodes. The training recorded 15 iterations.



```
Algorithms
Data Division:   Random                  (dividerand)
Training:        Levenberg-Marquardt     (trainlm)
Performance:     Mean Squared Error      (mse)
Calculations:    MATLAB

Progress
Epoch:                     0   15 iterations   1000
Time:                          0:00:01
Performance:           0.796   0.0158          0.00
Gradient:               1.38   0.0750          1.00e-07
Mu:                  0.00100   1.00e-05        1.00e+10
Validation Checks:         0   6               6

Plots
Performance   (plotperform)
Tra           instate)
Regression    (plotregression)

Plot Interval:   1 epochs

✓ Validation stop.
```

Figure 5. 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 6.

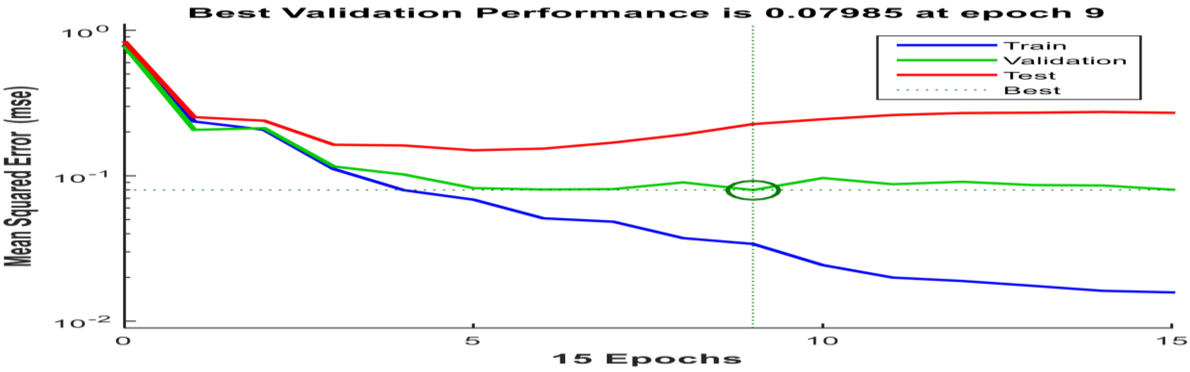


Figure 6. Classifier mean squared error

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 7.

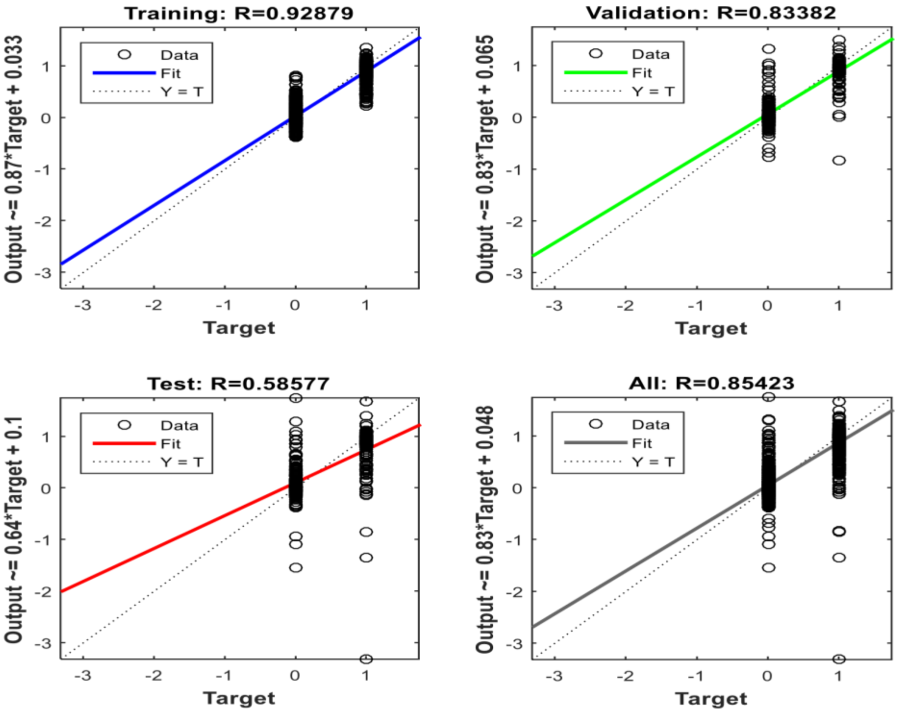


Figure 7. Classifier regression

The performance evaluation of the Neural Genetic Algorithm classifier was done on accuracy-based measures. The evaluation results show an absolute successful classification of all the attacks. Accuracy of 98.58% was obtained with detection rate of 96.49% and specificity of 95.97%. In addition to accuracy-based measurements, high predictive values both positive and negative were achieved (94.26% positive prediction and 95.04% negative predictive ability). The optimization-classification model was evaluated against the conventional neural network classification model (CNNCM). The results are presented in Table 4.

Table 4. Performance evaluation

	Accuracy	True Positive Rate (TPR)	True Negative Rate (TPR)	False Positive Rate (FPR)	False Negative Rate
NN	88.50	91.42	94.29	5.71	0.558 - (CNNCM)
NN-GA	98.58	96.49	95.97	4.03	0.351 - (Authors Result)

4.2 Discussion

The introduction of the genetic algorithm for optimization-classification of the distributed denial of service has shown that better performance is achievable. The conventional neural network recorded lower classification accuracy compared to the genetically optimized classifier. The results in the table 4 shows the new model NN-GA has better accuracy of 98.58 with lower false positive rate of 0.351 in comparison to conventional neural network classification model alone which has accuracy of 88.50 with false alarm rate of 0.558. Again, from the system the confusion metrics depicted in Table 4.3 shows that, for most of the classes, this 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 distributed denial of service & user-to-root and close detection rate for probe & remote-to-local was achieved.

5. CONCLUSION AND FUTURE WORK

This research, based on the experiments carried out show the optimization of improved genetic algorithm with neural network for classification of the DDoS has better performance in terms of accuracy and false alarm rate. In order to implement and measure the performance of this research, standard KDD99 benchmark dataset was obtained and trained with new model. The model gives reasonable accuracy and false alarm rate preferable to the existing ones. The results of this research show new model NN-GA has better accuracy of 98.58 with lower false positive rate of 0.351 as against the conventional neural network which yielded accuracy of 88.50 with false alarm rate of 0.558. In the future, using an improved fitness function or heuristic esemble has potential to give better detection and false alarm rate.

6. REFERENCES

[1] Aamir, Muhammad, and Muhammad Arif (2013). "*Study and Performance Evaluation on Recent DDoS Trends of Attack & Defense." International* Journal of Information Technology and Computer Science (IJITCS) 5, no. 8.

[2] Ahmed M. and A. Mahmood (2014) "*Network Traffic Analysis Based On Collective Anomaly Detection,*"in 2014 IEEE 9th conference on industrial electronics and applications (ICIEA), pp.1141-1146.

[3] Ahmed M. and A. Mahmood (2015) "*Novel Approach for Network Traffic Pattern Analysis Using Clustering-Based Collective Anomaly Detection*," Annals of Data Science, vol.2(1), pp.111-130.

[4] Alenezi M. & Reed M.J. (2017). *Methodologies for Detecting DOS/DDOS attacks against Network Servers*. In The Seventh International Conference on Systems and Networks Communications, ICSNC SemiMarkov Models.

[5]Anup Goyal & Chetan Kumar (2008). GA-NIDS: *A Genetic Algorithm based Network Intrusion Detection System*.

[6]Bace R.G (2010). *Intrusion Detection*. Macmillan Technical Publishing.

[7] 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, KTH/DSV.

[8] Crosbie, M & Spafford, E. (1995). *Applying Genetic Programming to Intrusion Detection*. Proceedings of the AAAI Fall Symposium

[9] Elavarasi M. (2016). *Network Forensics and Its Investigation Methodology.* An International Journal Emergency. Trends Sci. Technol., vol. 03, no. 05, pp. 852–859.

[10]D. T. Pham and D. Karaboga, (2000). *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*. New York: Springer-Verlag.

[11]Gong, R.H., Zulkernine, M & Abolmaesumi, P. (2005). *A Software Implementation of a Genetic Algorithm Based Approach to Network Intrusion Detection*.

[12] Goyal, A., Kumar, C., A (2008) *Genetic Algorithm based Network Intrusion Detection System*, not published Electrical Engineering and Computer Science Northwestern University Evanston. From: http://www.cs.northwestern.edu/~ago210/ganids/GANIDS.pdf

[13] Haddadi, F., Khanchi, S., Shetabi, M., & and Derhami V. (2016). *Intrusion Detection and Attack Classification Using Feed-forward Neural Network*. In Computer and Network Technology (ICCNT), 14th International Conference on, pp. 262–266, IEEE.

[14] Hoque N, H. Kashyap, and D.K. Bhattacharyya, "*Real-time DDoS attack detection using FPGA*," Computer Communications, vol.110, pp.
vol.110, pp. 48-58, 2017.

[15] Ilgun, K., Kemmerer, R., & Porras P.A., (1995). *State Transition Analysis: A Rule-Based Intrusion Detection Approach.* An IEEE Transaction on Software Engineering, 21(3):pp. 181-199.

[16] Jawale M.D.R., & Bhusari V. (2014). *Technique to Detect and Classify Attacks in NIDS Using ANN*.

[17] Karimazad R., & Faraahi A. (2017). *An Anomaly-based Method for DDOS Attacks Detection Using RBF Neural Networks*. An International Conference on Network and Electronics Engineering, IPCSIT, vol. 15.

[18] Kayacık, 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".*

[19] KDD Cup (1999). Data; *http://www.kdd.org/kddcup/index.php?section=1999&method=data*

[20] Kumar, S., & Spafford E. (1995). *A Software architecture to Support Misuse Intrusion Detection*. In The 18th National Information Security Conference, pp. 194-204.

[21] Li, W. (2004). Using Genetic Algorithm for Network Intrusion Detection. *A Genetic Algorithm Approach to Network Intrusion Detection*, SANS Institute, USA.

[22] Lu, K., Wu, D., Fan, J., Todorovic, S., & A. Nucci (2017). *Robust and Efficient Detection of DDOS Attacks for Large-scale Internet*. Computer Networks, vol. 91, no. 18, pp. 5036–5056.

[23] Lu, W. & Traore, I. (2014). Detecting New Forms of Network Intrusion Using Genetic Programming. Computational Intelligence, vol. 20, pp. 3, Blackwell Publishing, Malden, pp. 475-494.

[24] Mualfah D., & Riadi I. (2017). *Network Forensics for Detecting Flooding Attack on Web Server*. An IJCSIS) Int. J. Computer. Science Information Security. vol. 15, no. 2, pp. 326–331.

[25] Nagalakshmi V. “DDos Defense: *Enhanced Flooding Detection and Confidence-based Filtering Method*.” Advances in Computational Sciences and Technology (ISSN) volume 10.

[26] Norouzian M.R., & S. Merati S. (2015). *Classifying Attacks in a Network Intrusion Detection System based on Artificial Neural Networks*. In Advanced Communication Technology (ICACT), 13th International Conference on, pp. 868–873, IEEE.

[27] Pan, W., & W. Li (2015). *A Hybrid Neural Network Approach to the Classification of Novel Attacks for Intrusion Detection in Parallel and Distributed Processing and Applications*, pp. 564–575, Springer.

[28] Papalexakis E., A. Beutel, and P Steenkiste, “*Network Anomaly Detection Using Co- Clustering*,” In: Proceedings of the 2012 international conference on advances in social networks analysis and mining (ASONAM2012),ASONAM'12, IEEE Computer Society, Washington, DC,USA, pp.403-10. 2012.

[29] Poojitha G, K. Kumar, and P. Reddy (2010). “*Intrusion Detection Using Artificial Neural Network*,” 2010 International conference on computing communication and networking technologies (ICCCNT), pp.1-7.

[30] Rawat, A.S., Rana, A., Kumar A., & A. Bagwari (2018). *Application of Multi-Layer Artificial Neural Network in the Diagnosis System : A Systematic Review*,” vol. 7, no. 3, pp. 138–142.

[31] Su M-Y., “*Using Clustering To Improve The KNN-Based Classifiers For Online Anomaly Network Traffic Identification,*” Journal of Network and Computer Applications, vol.34 (2), pp.722–30, 2011.

[32] Todd Booth, Karl Andersson (2017). *Critical Infrastructure Network DDos defense via cognitive learning*.

[33] 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 IJCST. VOL 3, Issue 1, March 2012. From: http://www.ijcst.com/vol31/1/uppaliah.pdf

[34] Xia, T. G., Qu, S., Hariri & M. Yousif (2015). *An Efficient Network Intrusion Detection Method Based on Information Theory and Genetic Algorithm*. Proceedings of the 29th IEEE International Performance Computing and Communications Conference, Phoenix, AZ, USA.

[35] Yu W. and H. Lee, "*An Incremental-Learning Method for Supervised Anomaly Detection By Cascading Service Classifier And ITI Decision Tree Methods*," in Proceedings of the Pacific Asia Workshop on Intelligence and Security Informatics, pp. 155-160, 2009.

[36] Xiao-ming L., C. Gong, L. Qi, and Z. Miano, "*A Comparative Study on Flood DoS and Low-Rate DoS Attack*," The Journal of China Universities of Posts and telecommunications. Vol. 19, pp. 116-121, June 2012. .

[37] Zhang Chao-yang (2011) *DDoS Attack Analysis and Study of New Measures to prevent*.

[36] *https://www.vxchnge.com/blog/recent-ddos-attacks-on-companies*

[37] *https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/*