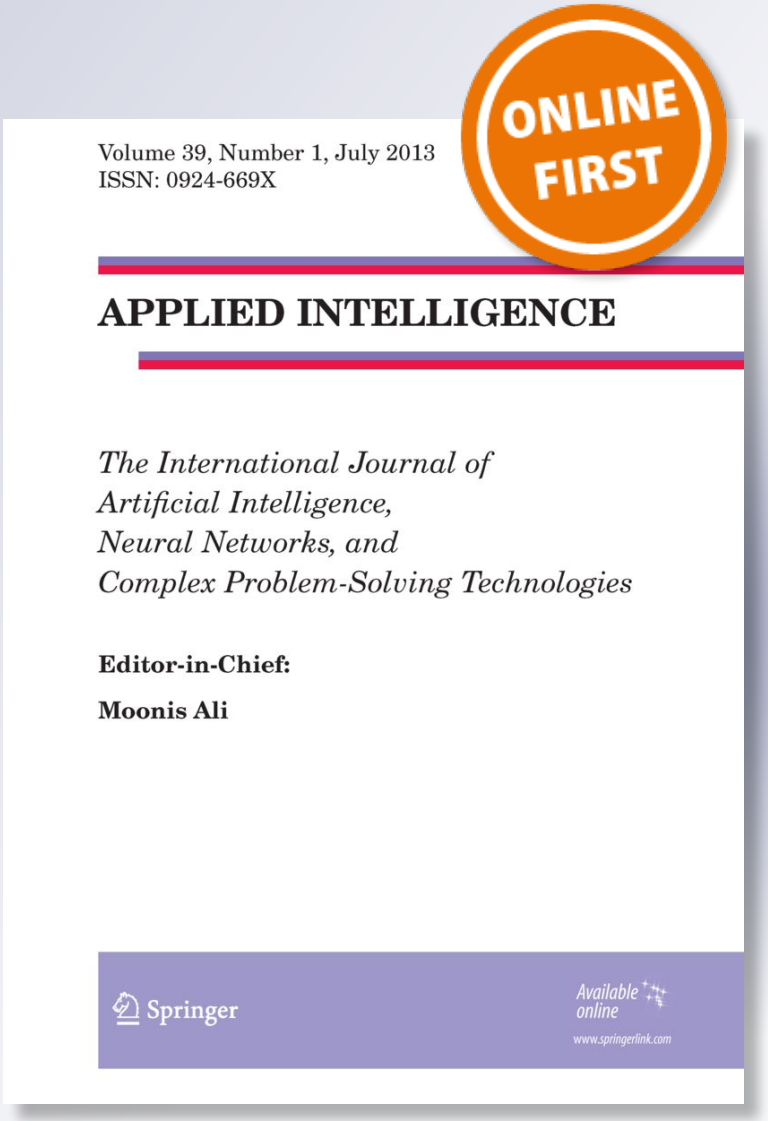
Vector Based Genetic Algorithm to optimize predictive analysis in network security

Sidra Ijaz, Faheel A. Hashmi, Sohail Asghar & Masoom Alam

Applied Intelligence
The International Journal of Research on Intelligent Systems for Real Life Complex Problems

ISSN 0924-669X

Appl Intell
DOI 10.1007/s10489-017-1026-9



Springer

Appl Intell
DOI 10.1007/s10489-017-1026-9

Vector Based Genetic Algorithm to optimize predictive analysis in network security

Sidra Ijaz[1] · **Faheel A. Hashmi**[2] · **Sohail Asghar**[1] · **Masoom Alam**[1]

Abstract A new Intrusion Detection System (IDS) for network security is proposed making use of a Vector-Based Genetic Algorithm (VBGA) inspired by evolutionary approaches. The novelty in the algorithm is to represent chromosomes as vectors and training data as matrices. This approach allows multiple pathways to calculate fitness function out of which one particular methodology is used and tested. The proposed method uses the overlap of the matrices with vector chromosomes for model building. The fitness of the chromosomes is calculated from the comparison of true and false positives in test data. The algorithm is flexible to train the chromosomes for one particular attack type or to detect the maximum number of attacks. The VBGA has been tested on two datasets (KDD Cup-99 and CTU-13). The proposed algorithm gives high detection rate and low false positives as compared to traditional Genetic Algorithm. A detailed comparative analysis is given of proposed VBGA with the traditional string-based genetic algorithm on the basis of accuracy and false positive rates. The results show that vector based genetic algorithm provides a significant improvement in detection rates keeping false positives at minimum.

Keywords Genetic algorithm · IDS · Misuse detection · Artificial intelligence

✉ Sidra Ijaz
sidrafaheel@yahoo.com

[1] Department of Computer Science, COMSATS Institute of Information Technology, Islamabad, Pakistan

[2] Department of Physics, COMSATS Institute of Information Technology, Islamabad, Pakistan

1 Introduction

Network Intrusion Detection is essentially a big data issue [1]. A gigantic amount of information is constantly being generated by various applications and network devices. In general, the two main types of intrusion detection are misuse detection and anomaly detection. The misuse detection or signature based systems detect offenses which have already been observed and defined through a set of rules. In misuse detection approach, each instance in data is either normal or a particular type of offense. This approach assists training an intrusion detection model on the input data with the attacks provided that they are labeled appropriately. The advantage of misuse detection is that it can help in detecting known attacks with a low false positive rate.

Evolutionary approaches have been an interest of AI researchers due to their similarity with natural phenomena. One such approach is Genetic Algorithm [2, 3]. It is a stochastic search algorithm based on the biological evolution theory. GA mimics the biological mechanisms of evolution such as selection, crossover, and mutation. Genetic Algorithm has been widely used in various applications in network security [4, 5]. Another recent evolutionary approach is Artificial Immune Systems that uses an implementation of the genetic algorithm [6]. Several different computer immune models have been proposed based on the study of immunology [7]. The goal of these models is to distinguish the normal from an anomaly.

1.1 Challenges and motivation

The intrinsic flaw in traditional misuse detection systems is the high false positive rate, that is a major concern of the research community [8, 9]. Due to this problem, the traditional intrusion detection systems highly depend

Published online: 31 August 2017

on human effort to differentiate intrusive from non-intrusive network traffic [10]. The need of augmentation of advanced analytic techniques with traditional IDS architecture is inevitable to reduce false alarm rate and increase the detection rate. This leads to an increasing trend towards complementing rule-based correlation with machine learning techniques in cyber security. Another problem that is growing with the attacks is the big data issue. The data is available in raw and unstructured form. The conventional tools are insufficient to process the information the data contains. There is noteworthy research work on predictive techniques used in network security [11, 12] in order to discover the behavioral patterns. The benchmark dataset commonly used in research for predictive analysis is the KDD cup 99 dataset. In this research, two datasets have been used, KDDcup99 dataset [13] and a recent CTU-13 botnet traffic dataset [14]. Evolutionary approaches in machine learning have not only been proven to be helpful in generating rules for IDS but are also used in dimension reduction, predictive learning and anomaly detection in network intrusion detection systems.

1.2 Contributions

A modified evolutionary approach is proposed in this research that provides the following contributions:

- A vector based genetic algorithm is proposed in which chromosomes are considered as a vector of $n + 1$ elements where n is the feature count in the data set.
- The proposed scheme provides competent accuracy rates and overcomes the problem of high false positive rates, making them negligible as compared to general misuse detection systems.
- A detailed comparative analysis illustrating the effectiveness of VBGA over traditional string-based genetic algorithm as an intrusion detection system is also given in this paper.
- VBGA system can evolve generation with respect to any selected class in order to improve the detection rate.
- The proposed solution has the flexibility to be used on any type of network logs.
- A comparative analysis of experimental results on the proposed solution for four feature selection techniques.

The rest of the paper is organized as follows: Section 2 gives a background on the role of evolutionary machine learning approaches in network security. In Section 3, the related work is discussed. The proposed VBGA is described in Section 4. The experimental results on KDDcup-99 and CTU-13 dataset are discussed in Section 5. Section 6 gives a detailed comparative analysis of proposed VBGA with string based GA IDS on the KDDcup99 dataset. The conclusions and future research ideas are identified in Section 7.

2 Background

In the modern era of Internet of Things (IoT) and smart cities, where everything is dependent on embedded technology, the issue of cyber security is becoming bigger than ever. The attacks and security breaches are more pervasive and evolved. The traditional security software and tools need a noticeable amount of human effort and time to identify attacks, extract new rules from the threats, and encode those into security tools to detect the infusion [15]. This labor-intensive process can be improved by applying machine learning algorithms. Machine learning can aid in various additional facilities that provide enhanced intrusion detection and protection. First of all, the literature review suggests that machine learning helps in enhancing predictive analysis and misuse detection by reducing false positives and false negatives. Moreover, it can enhance the existing rule-based systems by generating new rules deduced from threat intelligence feeds through machine learning. One of the major contributions of machine learning is anomaly detection. The idea is to identify the normal behavior and produce an alarm if any deviation or outlier is detected [16]. A variety of machine learning techniques are being used for a diverse nature of purposes in network security.

The idea of Genetic Algorithms was taken from the processes observed in natural evolution, hence giving an optimized evaluation of studied data. The natural evolutionary processes of selection, crossover, and mutation are mimicked into GA to solve optimization problems [17]. Hence, GA works very well with the computational problems that require a computer program to be adaptive in a changing environment. Hence there are four main steps in GA:

Initial population The initial step in GA is a population of chromosomes that is generally initialized randomly.

Selection The selection gives preference to evolved, fitter parents, to pass on their genes to the offspring. This can be checked upon calculating their fitness.

Crossover Generally, in the crossover, the sub-sequences of selected chromosomes are exchanged to create two offspring. It is primarily the factor that distinguishes GA from the other optimization techniques.

Mutation Mutation operator usually flips some bits in a chromosome randomly. The purpose of mutation is to keep the diversity inside the population and inhibit premature convergence. Figure 1 shows the process of a traditional genetic algorithm.

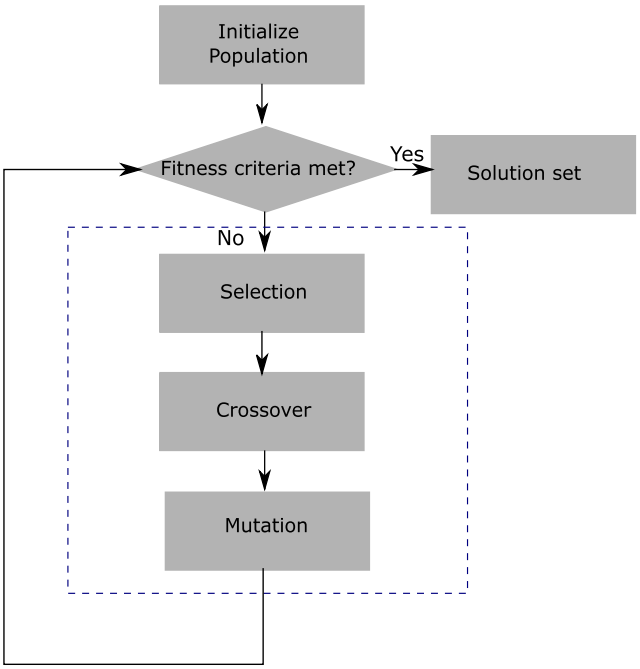


Fig. 1 Genetic algorithm process

3 Related work

Genetic Algorithm has been widely used in various applications in network security. It is proven to be helpful for a variety of purposes. For instance, GA is helpful in generating rules or signatures for IDS. It has also been used in misuse detection and anomaly detection. GA has been used for efficient feature selection in order to enhance the predictive learning in network intrusions. Literature suggests that genetic algorithms are used to tune the membership functions used by the intrusion detection systems. This survey gives detail on the implementation of GAs on IDS [18]. AIS is inspired from Human Immune System (HIS) that is a natural mechanism to identify and defend against harmful viruses without prior knowledge of them. Hence, AIS has been a topic of interest in network security [19, 20]. Artificial immune systems (AISs) have been built for a wide range of network security applications, particularly anomaly detection [21, 22].

A widespread methodology that considers both temporal and spatial information in encoding the network connection information into rules for signature-based IDS [5] uses GA to generate rules. Though the author provides the methodology to use GA in intrusion detection, it does not provide experimental results for proving its effectiveness. In [4], the authors used GA to select best classification rules for intrusion detection. This approach consisted of two main stages. The training stage, in which a set of rules for detecting intruders is generated by using network audit data offline. The rules with the highest fitness value were selected to feed the IDS in the real-time environment in the second stage. But, they have experimented on a very small sample subset of the kDDcup99 dataset that may not give a good representation of the entire population size. The authors of this paper [23] have discussed the drawbacks of SNORT IDS and various research areas which were taking place to enhance the performance of SNORT with the help of a genetic algorithm. The authors argue that in existing SNORT system, rules are not created at run time. The authors have discussed the role of GA in generating rules at run time. In another paper [24], the authors proposed a methodology to select and filter existing rules of SNORT through GA. Siahmarzkooh et al. [25] compared the results of Naive Bayes and SVM algorithm by using Naive Bayes method with GA technique to improve the prediction rates. The dataset again used is the KDD99. Their proposed system increased the efficiency of Naive Bayes classifier as compared to simple SVM or Naive Bayes. IDS have been implemented in past by GA application in order to efficiently detect various types of network intrusions by Houqe et al. [26]. The KDD99 benchmark dataset was used for experimentation. They used the standard deviation equation with distance in order to measure the fitness of a chromosome. The experimental results of their strategy do not provide optimized and effective predictive results. Moreover, a fuzzy genetic algorithm to classify network attack data [27] has recently been proposed. In this research, the authors considered both public KDD99 dataset and their own private network dataset. Fuzzy genetic algorithms in intrusion detection have also been effectively implemented as a predictive learning tool for intrusion detection [28]. The researchers have proved this point by comparing their results with results obtained using other traditional methods such as decision tree.

Kim et al. [29] proposed a fusion of SVM and GA for efficient optimization of both features and parameters for network detection models. This methodology provides optimal anomaly detection model capable of feature selection and maximization of the detection rates. The authors in [30] proposed a methodology of GA-based feature selection along with Decision Tree classifier for efficient network intrusion detection. Tsang et al. [31] proposed a genetic-fuzzy rule mining approach along with the evaluation of feature selection techniques for anomaly intrusion detection. This system can act as a genetic feature selection wrapper in order to search for an optimal feature subset. In another recent research work, Kanan et al. proposed a novel intrusion detection model in which they used GA-based feature selection algorithm and Fuzzy SVM for classification [32]. The researchers have also used a genetic algorithm based feature selection on KDD cup 99 dataset with ANN [33] and produced 99 percent predictive results. The authors in [34] have suggested a hybrid algorithm as well using SVM and GA that is capable of achieving a true positive

rate of 97 percent. The feature selection methodology is characterized into three priorities giving GA algorithm as the highest priority. Moreover, the authors in [35] used GA and self-organized feature map (SOFM) to augment the feature and information taking out from a gigantic dataset that has similarities with the famous KDD 99 cup dataset. Their strategy helped in reduction of the size and dimensions of the dataset to train SVM, hence increasing detection rate.

As discussed earlier, network intrusion detection techniques are either rule-based or anomaly based. The anomaly detection techniques are a significant technology to protect against evolving new attacks including zero-day attacks [36]. The main advantage of GA in anomaly detection is that it is a flexible search method that converges to a solution from multiple directions [16]. The authors in [37] proposed an anomaly based network intrusion detection system implementing GA. The experimental results boast that the proposed GANIDS (Genetic Algorithms based Network Intrusion Detection System) is efficient, having good detection rate with the lower false positive rate. In their paper, [38] the authors proposed an anomaly detection model using GA along with several features selection techniques. These include principle components analysis, sequential floating, and correlation-based feature selection. The comparative analysis showed that the sequential floating technique with the GA has the best results.

4 Proposed methodology

In this contribution, a novel vector based genetic algorithm is proposed for network intrusion detection systems (IDS). The method has been implemented in Python and has been tested on the KDD Cup 99 and CTU-13 dataset. The method consists of three phases, each of which is handled by a different python class. The system level diagram is shown in Fig. 2. In the following, we discuss the three phases of proposed method.

4.1 Phase one: data pre-processing

The dataset first needs to be preprocessed in order to make it suitable for computations. In this phase the data is read from the file, the alphanumeric entries in the data are converted into numeric types making use of the conversion tables that are dynamically updated (as the data is read from the file) and are later stored as legend files for future use.

Feature selection Feature selection is an essential step in building machine learning based intrusion detection systems. There may be quite a few features that not only increase the computational cost but also the error rate of training models. Four feature selection techniques for the purpose of dimension reduction have been used. In the first experiment, a basic feature selection is used for all types of attacks. We exclude the features 'num_outbound_cmds' and 'is_host_login' as these two have only a single value in the entire dataset.

The KDD cup 99 dataset has total 41 features. These features are generally divided into three main groups [25]: Basic features, that are the features extracted from a TCP/IP connection (from 1 to 9). The second type is content features that help signify the suspicious behavior in the data (such as a large number of failed login attempts). These are labeled from 10 to 22. Moreover, the third type is the traffic features, that encapsulate the features computed in a window interval. These can further be divided into same host features(23 to 31) and same service features (32 to 41). It is to be noted that different feature set is important in finding out different attack categories. We have chosen three feature selection techniques mentioned in [39] for our experimentation for denial of service attacks (for the rest of attacks, basic feature selection is used). The techniques include linear correlation-based feature selection (LCFS), modified mutual information-based feature selection (MMIFS) and Forward Feature Selection Algorithm (FFSA) [39]. The LCFS methodology proposed by Amiri et al. [39] is based on the linear correlation coefficient. In FFSA, the input features are selected on basis of maximizing mutual information (that is evaluated from the arbitrary dependency between random variables) between selected inputs and output. On the other hand, MMIFS is a feature selection method proposed by Amiri et al. [39] that selects features on basis of maximum relevance and minimum redundancy. For CTU-13 dataset, all of the 14 features are selected. The rest of the phases for all three experiments remained same.

The next step during this phase is data normalization. The entire data set is normalized by dividing each column (excluding the last column that contains the labels for attack types) by the maximum value in that column. This ensures that all columns except the last one have values going from 0.0 to 1.0, and each record in the dataset (excluding the last column) can be treated as an n dimensional vector having the norm bounded between 0.0 and $\sqrt{n}$ where n is the feature count.

The dataset is further divided in the ratio 60 : 20 : 20. The last two parts constitute the matrices Training_Data and Test_Data. The first 60% are sorted by the attack labels into different Attack_Matrices for each attack type. The output of this phase are the matrices Training_Data, Test_Data, and different Attack_Matrices.

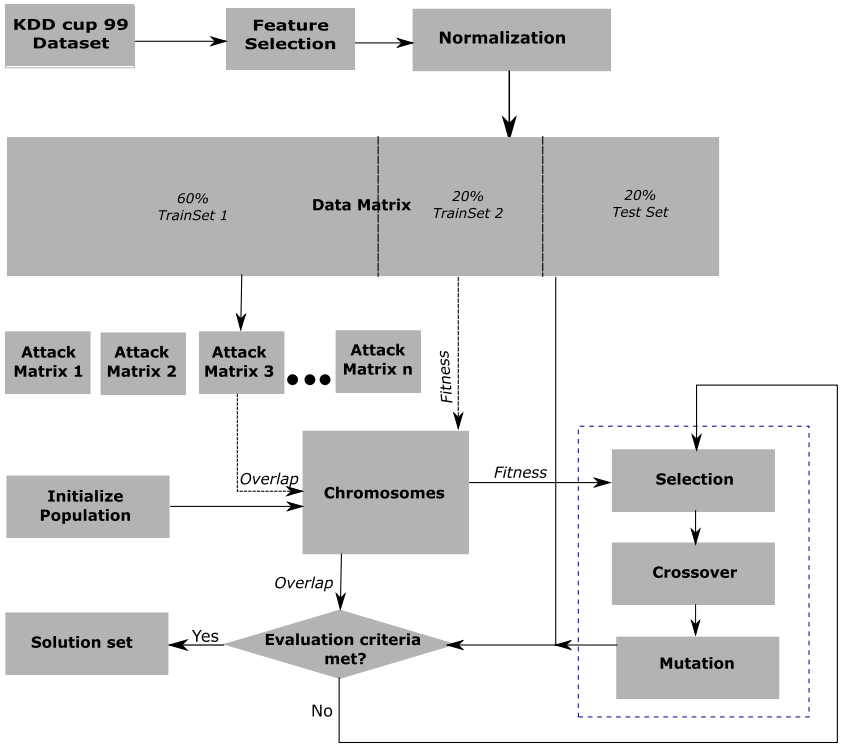


Fig. 2 VBGA intrusion detection system

4.2 Phase two: creation of chromosomes

The central element in our method deals with the functionality to handle the chromosomes for the Genetic Algorithm (GA). We are using a vector based approach in which each chromosome is a vector of $n + 1$ elements where n is the feature count in the dataset. The last element in the chromosome corresponds to the last column of the dataset and is set to 0.0 during the entire process. Each element (gene) in the chromosome is a random float value between 0.0 and 1.0. A typical chromosome is shown in (1). The superscripts correspond to a respective feature in the dataset.

$$\text{chromosome} = \begin{bmatrix} 0.25^{1} \\ 0.68^{2} \\ 0.96^{3} \\ 0.71^{38} \\ . \\ . \\ . \\ . \\ 0.81^{39} \\ 0.0^{40} \end{bmatrix} \tag{1}$$

For each chromosome we calculate on overlap array as shown in (2):

$$O[i] = \text{Mean}(\text{InnerProduct}(\text{Attack\_Matrix}[i], \text{chromosome})) \tag{2}$$

Here $O[i]$ is the overlap index of the chromosome for the i^{th} attack type. The overlap array for the chromosome in the last paragraph is given in (3). The subscript corresponds to different attack types present in the dataset. Using the overlap array the chromosomes can predict the attack type of the unknown data. This can be done by taking the overlap (inner product) of the unknown data with the chromosome and comparing the result with the overlap indices of the chromosome for different attacks.

$$\text{O} = \begin{bmatrix} 3.26_{1} \\ 2.71_{2} \\ 3.01_{3} \\ . \\ . \\ . \\ . \\ 2.83_{22} \\ 2.41_{23} \end{bmatrix} \tag{3}$$

Lastly, in this phase of the algorithm, a criteria is needed to evaluate the fitness of the chromosomes. This is done be evaluating the 20% Training_Data by the chromosome for the attack i and using the (4).

$$\text{fit}[i] = \frac{\text{True\_Positives}[i]}{\text{Attack\_Count}[i]} - \frac{\text{False\_Postive}[i]}{\text{Total\_Attack\_Count}} \tag{4}$$

For the example chromosome discussed in this section, the fitness array is illustrated in (5). This fitness array shows

that the chromosome is best suited for detecting attack 6 which corresponds to the class. The pseudo code for calculating overlap and fitness of chromosomes is given as Algorithm 1.

$$\text{fit} = \begin{bmatrix} 0.05_1 \\ 0.39_2 \\ . \\ . \\ 0.98_6 \\ . \\ . \\ 0.59_{12} \\ . \\ . \\ -0.01_{23} \end{bmatrix} \quad (5)$$

Algorithm 1 Overlap and Fitness Calculation

```
    Calculating Overlap:
for x in range(1,n) do
    ATM ← Attack_Matrix[x]
    overlapArr ←innerProduct(ATM,chromosome)
    overlap[x]← Mean(overlapArr)
    Calculating Fitness:
for record in Training_Data do
    guess ← guessAttack(record)
    if guess==true then
        fitnessU[record[−1]]+ =1
    elsefalseAlarm[record[−1]]+ =1
    attackCount ← getAttackCount(Training_Data)
    attackCountTotal ← sum(attackCount)
for x in range(1,n) do
    if attackCount[x]>0 then
        fitness[x] ← fitnessU[x]/attackCount[x]
    elsefitness[x] ← fitnessU[x]
    fitness[x]− =(falseAlarm[x]/attackCountTotal)
```

4.3 Phase 3: population and prediction

The last phase of the method deals with the functionality required to handle the population of chromosomes. The population is created, sorted according to the fitness of chromosomes, used for making predictions of the Test_Data, and evolved making use of selection, crossover, and mutation of the chromosomes.

The first step in this phase is to create a random initial population. The initial population size should be large to explore a wide space of candidate chromosomes. Once the population has been created, the chromosomes in the population can be selected based on their fitness, either for a particular attack or overall fitness for all the attacks. The selected population is then used to test on remaining 20% of the dataset. This is done by evaluating the records in test data against each chromosome in the selected population, each chromosome makes a guess for the record attack type based on the comparison of its overlap with the record with its overlap array. We choose the attack type where a maximum number of selected chromosomes agree on the attack type. If the results are satisfactory, the process can be stopped at this level, else the selected population can be evolved through selection, crossover, and mutation to obtain more fit chromosomes.

5 Experimental results

Two datasets have been selected for experimentation. The first dataset is KDD cup 99, that is the most commonly used dataset over the past decade in the field of machine learning and network security. In order to further ensure the significance of proposed IDS, a labeled botnet traffic dataset named CTU-13 is selected. The detail on the datasets along with the experimental results are given as follows:

5.1 KDD cup 99 dataset

It was developed for the KDD competition in 1999 where the competition task was to build a network intrusion detector. It was created by Stolfo et al. [13]. This fairly large dataset contains 41 features and is labeled as either normal or a particular type of attack, that falls in one of following four categories:

Denial of Service Attack (DoS) DoS is the most common type of attack where the attacker makes the computational resources such as memory resource overly busy to handle legitimate requests, therefore leading to a denial of legitimate user's access and services.

User to Root Attack (U2R) U2R is a type of attack in which the attacker exploits the system by starting with apparently legitimate access to a normal user account on the system (through hacking passwords). After that, the attacker makes use of the vulnerability in the system to gain the root access.

Remote to Local Attack (R2L) The R2L attack usually occurs when the illegitimate attacker can send packets to a system over a network and get hold of some vulnerability to gain the local access as a legitimate user of that system.

Probing attack In Probing attack, the attacker attempts to gain the information about a network, apparently circumventing its security controls.

There are following variants of this dataset available at UCI KDD archive:

1. kddcup.data 10 percent
2. kddcup.newtestdata 10 percent unlabeled
3. kddcup.testdata.unlabeled
4. kddcup.testdata.unlabeled 10 percent
5. corrected

The 10 percent training set is selected that has 494,020 connections containing all the minority classes and part of the majority classes, as it is the smaller version of the original dataset. We have performed three sets of experiments with three separate feature selection schemes on KDD Cup 99 Dataset. The details of these experiments are as follows:

The features 'num_outbound_cmds' and 'is_host_login' have been excluded in the first set of experiment. A random population consisting of 20,000 chromosomes is generated initially. The large population size helps in exploring a large parameter space for the fit chromosomes. Then the best fit chromosome for each attack type is used to predict the same attack in Test_Data. The results are given in Table 1 and Fig. 3. The false positive rate in the Fig. 3 is obtained by dividing the False Positives in Table 1 by the total no of records in Test_Data (which is $\approx$ 100000).

Table 1 Detection rate and false positive rate of selected classes

Attack Label	Class	Total Attacks	Detected	False Positives
1	normal	19348	16093	379
2	buffer_overflow	9	6	835
3	loadmodule	3	0	129
4	perl	1	0	32
5	neptune	21436	21394	259
6	smurf	56282	56218	6
7	guess_passwd	13	12	30
8	pod	51	51	1754
9	teardrop	209	176	1764
10	portsweep	207	183	53
11	ipsweep	249	227	643
12	land	2	2	0
13	ftp_write	1	0	44
14	back	440	396	13043
15	imap	3	0	12
16	satan	294	253	78
17	phf	0	0	5
18	nmap	48	37	3696
19	multihop	1	0	1
20	warezmaster	3	3	401
21	warezclient	201	168	826
22	spy	0	0	640
23	rootkit	4	0	30

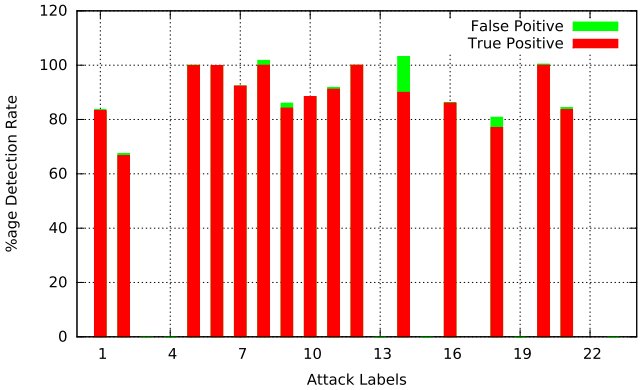


Fig. 3 Detection rate and false positive rate in first generation

For specific attack types (having a significant number of records in the dataset) populations containing the top 20 fit chromosomes are selected. Among one such population (for the chosen attack type) we cross over the genome of first 5 chromosomes with the randomly selected chromosome in the population. This crossover process is implemented by swapping a random segment of the first chromosome by that of the second parent chromosome. A mutation index going from 0 to 1 causes random alterations in the genes of the child chromosome. The mutation index is 0.2 which means that 20% of the genome in the child chromosome is randomly altered. The population is further evolved to reach a size of 200 chromosomes, after which this process of selection, crossover, and mutation are repeated. We continue this process for a total of 6 generations. The results of the top fit chromosome for chosen attack type for each generation are shown in Fig. 4.

Finally, instead of using a single chromosome for making the predictions, a population consisting of multiple chromosomes to make predictions is used. In this case, predictions are made only if multiple chromosomes agree on the prediction. This improves true positive prediction rate and significantly reduces the false positive rate. We use the populations consisting of top 1,2,5,10,20 and 50 chromosomes for each attack type to make predictions for Test_Data. The best results obtained in this manner are shown in Fig. 5.

As discussed earlier, three different feature selection techniques are explored in order to enhance the performance of intrusion detection system. The tests are performed on denial of service attacks. It is observed that the basic feature selection technique performs the best for our experimentation. The LCFS feature selection technique proposed by Amiri et al. [39] performs better for majority attacks, but the results for land and Neptune are quite low. The MMFIS and FFSA perform poorly for most of DoS attacks. The results are illustrated in detail in Fig. 6.

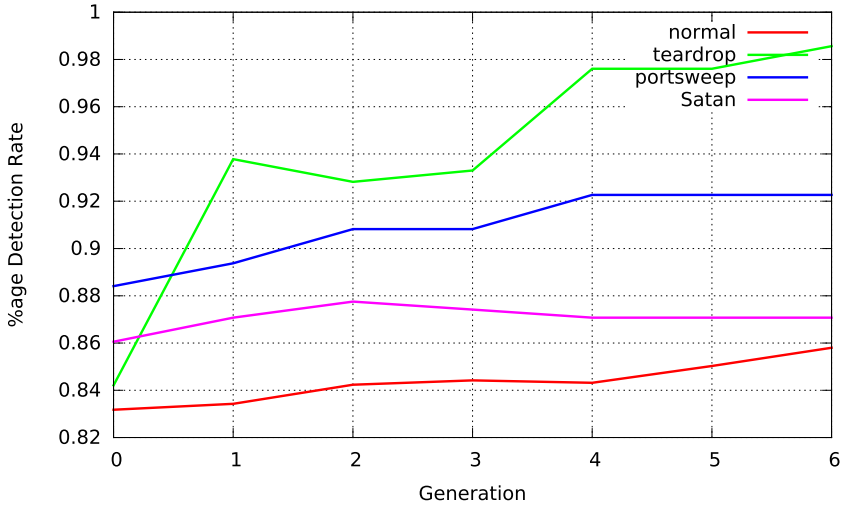


Fig. 4 Detection rate of selected classes through six generations

5.2 CTU-13 dataset

Garcia et al. [14] built a labeled dataset of network traffic that contains a massive capture of a botnet, normal and background traffic. There are total thirteen scenarios of various botnet samples. Each sample contains labeled bidirectional traffic flow provided publicly for experimentation. There are total 15 features including the label in the dataset.

For experimentation, scenario 2 titled 'CTU-Malware-Capture-Botnet-43' is selected that has approximately 2 million bidirectional flow connections containing background, normal and botnet flows. The data is partitioned on ratio 60:20:20. A random population consisting of 2000 chromosomes (as it is bigger dataset) is generated initially. Then the 100 fit chromosomes for two labels is selected (flow=from-botnet-V43-TCP-attepmt and flow=from-botnet-V43-TCP-attept-spam) having approximately 2000 and 1000 records in test data respectively. The total no of records in Test_Data is $\approx$ 360000.

The population is further evolved to 500 with the mutation index of 0.3. It is evolved to reach a size of 100 chromosomes, after which this process of selection, crossover, and mutation are repeated. This process is continued for a total of 6 generations. The results of the top fit chromosome for chosen attack type for each generation are shown in Figs. 7 and 8 for each label respectively. The results for the label 'flow=from-botnet-V43-TCP-attept-spam' have been improved remarkably through the 6 generations.

6 Comparative analysis

For comparative analysis, KDDtup 99 dataset has been selected, as the published research on this dataset is widely available. It is discussed in the background section that the attacks can be generalized into four main type of categories: DoS, Probe, U2R, and R2L. The accuracy and false positive rate in accordance with the grouped categories are given in Fig. 9. The accuracy of DoS attack is 99.8 percent.

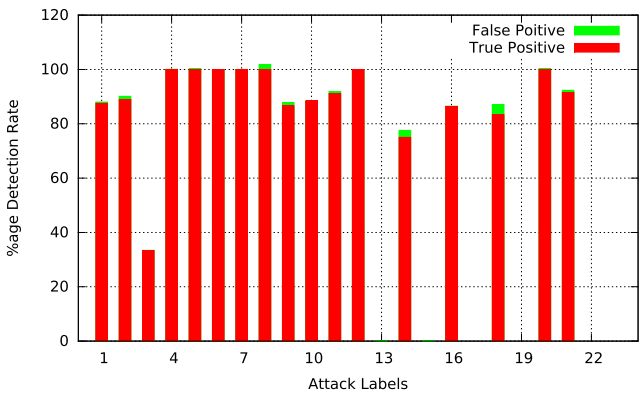


Fig. 5 Best detection rate for each class

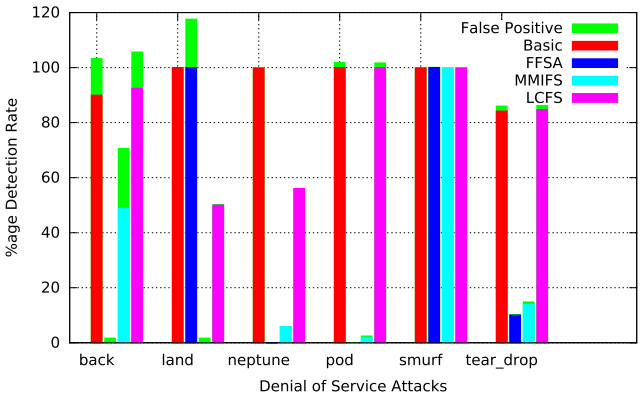


Fig. 6 Performance of VBGA with four feature selection techniques on DoS attacks

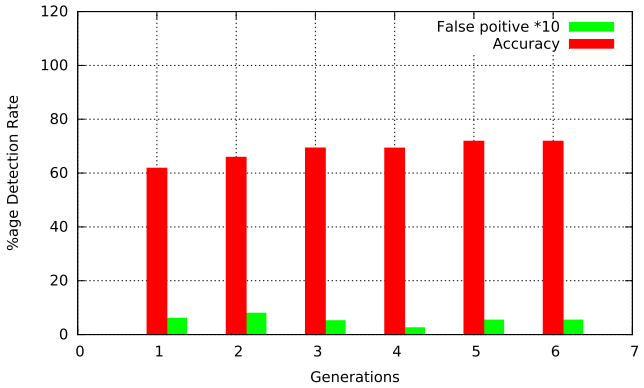


Fig. 7 Detection rate and false positive rate for flow=from-botnet-V43-attempt for six generations

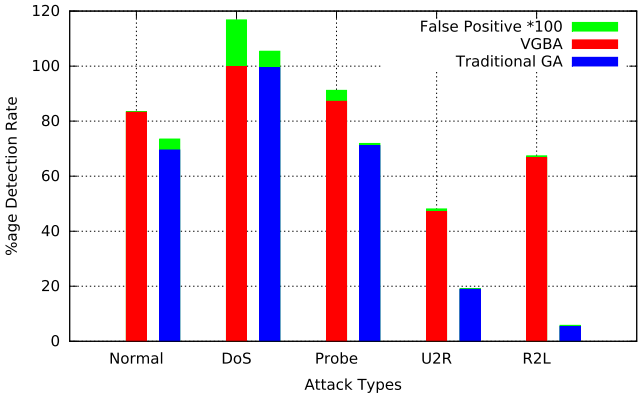


Fig. 9 Comparative analysis of VBGA IDS with tradtional string based IDS

As discussed earlier, that traditional GA methodology considers chromosomes as strings. The proposed modified genetic algorithm considers the chromosomes as vectors and introduces overlap as the evaluation criteria of result sets. In this section, a detailed comparative analysis of traditional GA-based algorithm is provided (as the results produced by Houque et al. [26]). It is clear from the results that the proposed VBGA IDS has provided higher accuracy rates and competitive false positive rates as compared to traditional methodology. In Fig. 9, the results of traditional GA-based IDS and VBGA IDS have been compared for all class types. The false positive rate is negligible, so we have multiplied it by one hundred in order to show the visibility in the graphical representation. For Normal instances, the detection rate is comparatively very high, with 14 percent increase in the detection rate. Furthermore, the false positive rate is also considerably reduced.

It is to be noted that the DoS attacks are the majority number of attacks in KDD cup 99 dataset. In Fig. 9, it is evident that the VBGA IDS increases the detection rate of DoS attacks to 99.8 percent with a negligible false positive rate of 0.17 percent. This beats the old benchmark set by string based GA intrusion detection system [26], that is 99.4 percent. The proposed VBGA intrusion detection system also increases the accuracy of Probe attack detection significantly. The algorithm detects 87.2 instances correctly increasing accuracy rate to significant 16.1 percent. Furthermore, the detection rate for R2L and U2R by the traditional string-based GA is extremely poor. It is significantly improved by our proposed VBGA IDS from 18 percent to 47 percent for U2R, and from 5 percent to 67 percent for R2L.

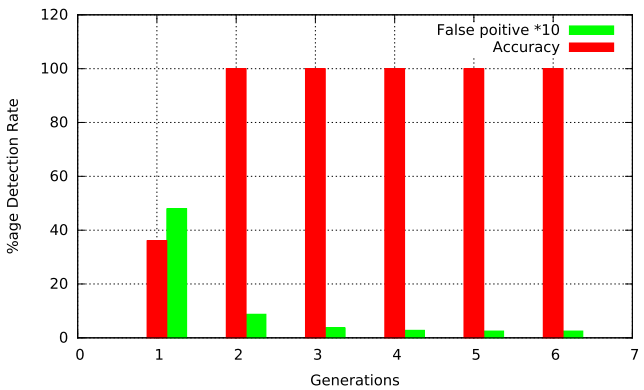


Fig. 8 Detection rate and false positive rate for flow=from-botnet-V43-attempt-spam for six generations

7 Conclusions and future work

Evolutionary algorithms have been used in various applications in network security. For instance, GA is helpful in generating rules for signature-based intrusion detection systems. It has also been used in anomaly detection. Moreover, GA has been applied for efficient feature selection in order to enhance the predictive learning in network intrusions. The evolutionary approach of GA is modified to optimize the predictive analysis hence producing prediction accuracy rates that set the benchmark to test the performance of the GA for intrusion detection. VBGA considers overlap of the attack matrices with chromosomes as evaluation criteria for predictions. The fitness function is used in the selection of chromosomes to evolve population. The proposed system is flexible to evolve generation with respect to any selected class in order to improve the detection rate. The experimental results show high accuracy rates with negligible false positive rates on two benchmark datasets. There are a number of things that can be accommodated in the future work. We can try to make more efficient use of Attack_Matrices. In the present version, these matrices are used only to calculate the overlap array of the chromosome in which only

the mean value of the overlap of the chromosomes with the entire matrix is used. In the future work, we can make use of the variance, minimum and maximum information to come up with a better strategy of defining the overlap arrays. Finally, it should be possible to come up with a novel fitness function in which the fitness is calculated directly from the overlap arrays of the chromosomes.

References

1. Gantz J, Reinsel D (2012) The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. IDC iView: IDC Anal Fut 2007:1–16
2. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4(2):65–85
3. Srinivas M, Patnaik LM (1994) Genetic algorithms: A survey. Computer 27(6):17–26
4. Banković Z, Stepanović D, Bojanić S, Nieto-Taladriz O (2007) Improving network security using genetic algorithm approach. Comput Electr Eng 33(5):438–451
5. Li W (2004) Using genetic algorithm for network intrusion detection. In: Proceedings of the United States department of energy cyber security group, pp 1–8
6. De Castro LN, Timmis J (2002) Artificial immune systems: a new computational intelligence approach. Springer Science & Business Media
7. Dasgupta D, Attoh-Okine N (1997) Immunity-based systems: A survey. In: 1997 IEEE international conference on systems, man, and cybernetics, 1997. Computational cybernetics and simulation, vol 1. IEEE, pp 369–374
8. Om H, Kundu A (2012) A hybrid system for reducing the false alarm rate of anomaly intrusion detection system. In: 2012 1st International conference on recent advances in information technology (RAIT). IEEE, pp 131–136
9. Hean L, Shuguang W (2013) Research on false alarm rate of intrusion detection based on cloning immune method. Int J Adv Comput Technol 5:2
10. Patel A, Qassim Q, Wills C (2010) A survey of intrusion detection and prevention systems. Inf Manag Comput Secur 18(4):277–290
11. Gaidhane R, Vaidya C, Raghuwanshi M (2014) Survey: Learning techniques for intrusion detection system (ids)
12. Gharibian F, Ghorbani AA (2007) Comparative study of supervised machine learning techniques for intrusion detection. In: Fifth annual conference on communication networks and services research, 2007. CNSR'07. IEEE, pp 350–358
13. Stolfo SJ, Fan W, Lee W, Prodromidis A, Chan PK (2000) Cost-based modeling for fraud and intrusion detection: results from the jam project. In: DARPA information survivability conference and exposition, 2000. DISCEX'00. Proceedings, vol 2. IEEE, pp 130–144
14. Garcia S, Grill M, Stiborek J, Zunino A (2014) An empirical comparison of botnet detection methods. Comput Secur 45:100–123
15. Chan PK, Lippmann RP (2006) Machine learning for computer security. J Mach Learn Res 7:2669–2672
16. Garcia-Teodoro P, Diaz-Verdejo J, Maciá-Fernández G, Vázquez E (2009) Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Secur 28(1):18–28
17. Davis L (1991) Handbook of genetic algorithms
18. Owais S, Snasel V, Kromer P, Abraham A (2008) Survey: using genetic algorithm approach in intrusion detection systems techniques. In: Computer information systems and industrial management applications, 2008. CISIM'08. 7th. IEEE, pp 300–307
19. Kim J, Bentley PJ, Aickelin U, Greensmith J, Tedesco G, Twycross J (2007) Immune system approaches to intrusion detection–a review. Nat Comput 6(4):413–466
20. Aickelin U, Bentley P, Cayzer S, Kim J, McLeod J (2003) Danger theory: The link between ais and ids? Artif Immune Syst 147–155
21. Aickelin U, Greensmith J (2007) Sensing danger: Innate immunology for intrusion detection. Inf Secur Tech Rep 12(4):218–227
22. Yang H, Li T, Hu X, Wang F, Zou Y (2014) A survey of artificial immune system based intrusion detection. Sci World J 2014
23. Devi S, Nagpal R (2012) Intrusion detection system using genetic algorithm-a review. Int J Comput Bus Ress
24. Dave MH, Sharma SD (2008) Improved algorithm for intrusion detection using genetic algorithm and snort
25. Siahmarzkooh AT, Tabarsa S, Nasab ZH, Sedighi F (2015) An optimized genetic algorithm with classification approach used for intrusion detection
26. Hoque MS, Mukit M, Bikas M, Naser A et al (2012) An implementation of intrusion detection system using genetic algorithm. arXiv:1204.1336
27. Jongsuebsuk P, Wattanapongsakorn N, Charnsripinyo C (2013) Real-time intrusion detection with fuzzy genetic algorithm. In: 2013 10th International conference on Electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON). IEEE, pp 1–6
28. Ireland E (2013) Intrusion detection with genetic algorithms and fuzzy logic. In: UMMC Sci senior seminar conference, pp 1–30
29. Kim DS, Nguyen H-N, Ohn S-Y, Park JS (2005) Fusions of ga and svm for anomaly detection in intrusion detection system. In: Advances in neural networks–ISNN 2005. Springer, pp 415–420
30. Stein G, Chen B, Wu AS, Hua KA (2005) Decision tree classifier for network intrusion detection with ga-based feature selection. In: Proceedings of the 43rd annual southeast regional conference-volume 2. ACM, pp 136–141
31. Tsang C-H, Kwong S, Wang H (2007) Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. Pattern Recogn 40(9):2373–2391
32. Kannan A, Maguire GQ, Sharma A, Schoo P (2012) Genetic algorithm based feature selection algorithm for effective intrusion detection in cloud networks. In: 20112 IEEE 12th international conference on data mining workshops (ICDMW). IEEE, pp 416–423
33. Dastanpour A, Ibrahim S, Mashinchi R (2014) Using genetic algorithm to supporting artificial neural network for intrusion detection system. In: The international conference on computer security and digital investigation (ComSec2014). The Society of Digital Information and Wireless Communication, pp 1–13
34. Aslahi-Shahri B, Rahmani R, Chizari M, Maralani A, Eslami M, Golkar M, Ebrahimi A (2015) A hybrid method consisting of ga and svm for intrusion detection system. Neural Comput Applic 1–8
35. Anil S, Remya R (2013) A hybrid method based on genetic algorithm, self-organised feature map, and support vector machine for better network anomaly detection. In: 2013 Fourth international conference on computing, communications and networking technologies (ICCCNT). IEEE, pp 1–5
36. Alazab M, Venkatraman S, Watters P, Alazab M (2011) Zero-day malware detection based on supervised learning algorithms of api call signatures. In: Proceedings of the Ninth Australasian data mining conference-volume 12. Australian Computer Society Inc., pp 171–182

37. Srinivasa K (2012) Application of genetic algorithms for detecting anomaly in network intrusion detection systems. In: Advances in computer science and information technology. Networks and communications. Springer, pp 582–591
38. Aziz ASA, Azar AT, Salama MA, Hassanien AE, Hanafy SE-O. (2013) Genetic algorithm with different feature selection techniques for anomaly detectors generation. In: 2013 Federated conference on computer science and information systems (FedCSIS). IEEE, pp 769–774
39. Amiri F, Yousefi MR, Lucas C, Shakery A, Yazdani N (2011) Mutual information-based feature selection for intrusion detection systems. J Netw Comput Appl 34(4):1184–1199