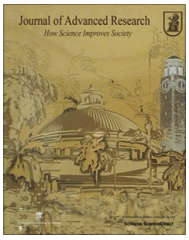
Journal of Advanced Research (2015) 6, 609–619

Cairo University

Journal of Advanced Research



ORIGINAL ARTICLE

A hybrid approach for efficient anomaly detection using metaheuristic methods

Tamer F. Ghanem [a,*], Wail S. Elkilani [b], Hatem M. Abdul-kader [c]

[a] Department of Information Technology, Faculty of Computers and Information, Menofiya University, Shebin El Kom, Menofiya, Egypt

[b] Department of Computer Systems, Faculty of Computers and Information, Ain Shams University, Cairo, Egypt

[c] Department of Information Systems, Faculty of Computers and Information, Menofiya University, Shebin El Kom, Menofiya, Egypt

ARTICLE INFO

Article history:
Received 20 October 2013
Received in revised form 26 February 2014
Accepted 27 February 2014
Available online 5 March 2014

Keywords:
Intrusion detection
Anomaly detection
Negative selection algorithm
Multi-start methods
Genetic algorithms

ABSTRACT

Network intrusion detection based on anomaly detection techniques has a significant role in protecting networks and systems against harmful activities. Different metaheuristic techniques have been used for anomaly detector generation. Yet, reported literature has not studied the use of the multi-start metaheuristic method for detector generation. This paper proposes a hybrid approach for anomaly detection in large scale datasets using detectors generated based on multi-start metaheuristic method and genetic algorithms. The proposed approach has taken some inspiration of negative selection-based detector generation. The evaluation of this approach is performed using NSL-KDD dataset which is a modified version of the widely used KDD CUP 99 dataset. The results show its effectiveness in generating a suitable number of detectors with an accuracy of 96.1% compared to other competitors of machine learning algorithms.

Introduction

Over the past decades, Internet and computer systems have raised numerous security issues due to the explosive use of networks. Any malicious intrusion or attack on the network may give rise to serious disasters. So, intrusion detection systems (IDSs) are must to decrease the serious influence of these attacks [1].

IDSs are classified as either signature-based or anomaly-based. Signature-based (misuse-based) schemes search for defined patterns, or signatures. So, its use is preferable in known attacks but it is incapable of detecting new ones even if they are built as minimum variants of already known attacks. On the other hand, anomaly-based detectors try to learn system's normal behavior and generate an alarm whenever a deviation from it occurs using a predefined threshold. Anomaly detection can be represented as two-class classifier which classifies each sample to normal or abnormal [2]. It is capable of detecting previously unseen intrusion events but with higher false

* Corresponding author. Tel.: +20 1004867003.
E-mail address: tamer.ghanem@ci.menofia.edu.eg (T.F. Ghanem).

Peer review under responsibility of Cairo University.

ELSEVIER Production and hosting by Elsevier

2090-1232 © 2014 Production and hosting by Elsevier B.V. on behalf of Cairo University.
http://dx.doi.org/10.1016/j.jare.2014.02.009

positive rates (FPR, events incorrectly classified as attacks) compared to signature-based systems [3].

Metaheuristics are nature inspired algorithms based on some principles from physics, biology or ethology. Metaheuristics are categorized into two main categories, single-solution-based and population-based metaheuristics [4]. Population-based metaheuristics are more appropriate in generating anomaly detectors than single-solution-based metaheuristics because of the need to provide a set of solutions rather than a single solution. Evolutionary Computation (EC) and Swarm Intelligence (SI) are known groups of population-based algorithms. EC algorithms are inspired by Darwin's evolutionary theory, where a population of individuals is modified through recombination and mutation operators. Genetic algorithms, evolutionary programming, genetic programming, scatter search and path relinking, coevolutionary algorithms and multi-start framework [5] are examples of EC algorithms. On the other hand, SI produces computational intelligence inspired from social interaction between swarm individuals rather than purely individual abilities. Particle swarm Optimization and Artificial Immune Systems are known examples of SI algorithms.

Genetic algorithms (GAs) are widely used as searching algorithm to generate anomaly detectors. It is an artificial intelligence technique that was inspired by the biological evolution, natural selection, and genetic recombination for generating useful solutions for problem optimization [6]. GAs use data as chromosomes that evolve through the followings: selection (usually random selection), cross-over (recombination to produce new chromosomes), and mutation operators. Finally, a fitness function is applied to select the best (highly-fitted) individuals. The process is repeated for a number of generations until reaching the individual (or group of individuals) that closely meet the desired condition. GAs are still being used up untill the current time to generate anomaly detectors using a fitness function which is based on the number of elements in the training set that is covered by the detector and also the detector volume [7,8].

Negative selection algorithm (NSA) is one of the artificial immune system (AIS) algorithms which inspired by T-cell evolution and self-tolerance in human immune system [9]. The principle is achieved by building a model of non-normal (non-self) data by generating patterns (non-self-detectors) that do not match an existing normal (self) patterns, then using this model to match non-normal patterns to detect anomalies. Despite this, self-models (self-detectors) could be built from self-data to detect the deviation from normal behavior [10]. Different variations of NSA have been used to for anomaly detection [11]. Although these newly developed NSA variants, the essential characteristics of the original negative selection algorithm [9] still remain, including negative representation of information, distributed generation of the detector set which is used by matching rules to perform anomaly detection based on distance threshold or similarity measure [12].

Generating anomaly detectors requires a high-level solution methods (metaheuristic methods) that provide strategies to escape from local optima and perform a robust search of a solution space. Multi-start procedures, as one of these methods, were originally considered as a way to exploit a local or neighborhood search procedure (local solver), by simply applying it from multiple random initial solutions. Some type of diversification is needed for searching methods which are based on local optimization to explore all solution space, otherwise, searching for global optima will be limited to a small area, making it impossible to find a global optimum. Multi-start methods are designed to include a powerful form of diversification [13].

Different data representation forms and detector shapes are used in anomaly detector generation. Input data are represented by either binary or real-valued [14]. Binary representation [15] is easy to use in finite problem space but it is hardly applicable to problems of real valued space [11]. As an alternative, real-valued representation [16] provides more expressiveness and scalability [17]. NSA detectors are formed with different geometric shapes such as hyper-rectangles, hyper-spheres, hyper-ellipsoids or multiple hyper-shapes [14]. The size and the shape of detectors are selected according to the space to be covered.

In this paper, a hybrid approach for anomaly detection is proposed. Anomaly detectors are generated using self- and non-self-training data to obtain self-detectors. The main idea is to enhance the detector generation process in an attempt to get a suitable number of detectors with high anomaly detection accuracy for large scale datasets (e.g., intrusion detection datasets). Clustering is used for effectively reducing large training datasets as well as a way for selecting good initial start points for detector generation based on multi-start metaheuristic methods and genetic algorithms. Finally, detector reduction stage is invoked so as to minimize the number of generated detectors.

The main contribution of this work is to prove the effectiveness of using multi-start metaheuristics methods in anomaly detector generation benefiting from its powerful diversification. Also, addressing issues arises in the context of detector generation for large scale datasets. These issues are related to the size of the reduced training dataset, its number of clusters, the number of initial start points and the detector radius limit. Moreover, their effect on different performance metrics is evaluated. Observations prove that performance improvement occurs compared to other machine learning algorithms.

The rest of this paper is organized as follows: Section 2 presents some literature review on anomaly detection using negative selection algorithm. Section 3 briefly describes the principal theory of the used techniques. Section 4 discusses the proposed approach. Experimental results along with a comparison with six machine learning algorithms are presented in Section 5 followed by some conclusions in Section 6.

Related work

Anomaly detection approaches can be classified into several categories. Statistics-based approaches are one of these categories that identify intrusions by means of predefined threshold, mean and standard deviation, and probabilities [18,19]. Rule-based approaches are another category which use If–Then or If–Then–Else rules to construct the detection model of known intrusions [20,21]. In addition to these categories, state-based approaches exploit finite state machine derived from network behaviors to identify attacks [22,23]. The last category is heuristic-based approaches [24–26], which are inspired by biological concepts as mentioned in the previous section [1].

Statistical hybrid clustering approach was proposed for network volume anomaly detection [27]. A combination between K-Harmonic means (KHM) and Firefly Algorithm (FA) is used to make clustering for data signatures collected by Digital Signature of Network Segment (DSNS). This approach detects anomalies with trade-off between 80% true positive rate and 20% false positive rate. Another statistical hybrid approach was introduced by Assis et al. [19]. Anomaly detection is based on modeling the normal behavior of the analyzed network segments using four flow attributes. These attributes are treated by Shannon Entropy in order to generate four different Digital Signatures for normal behavior using the Holt-Winters for Digital Signature (HWDS) method.

Another work [22] introduces a finite state machine approach based on Hidden Markov Model (HMM). A framework is built to detect attacks early by predicting the attacker behavior. This is achieved by extracting the interactions between attackers and networks using Hidden Markov Model with the help of network alert correlation module.

As an example of rule-based approaches, a framework for intrusion detection is presented [20]. This framework combines anomaly and misuse detection in one module with the aim of raising the detection accuracy. Different modules are designed for different network devices according to their capabilities and their probabilities of attacks they suffer from. Finally, a decision-making module is used to integrate the detected results and report the types of attacks.

Negative selection algorithms (NSAs) are continuously gaining the popularity and various variations are constantly proposed. These new NSA variations are mostly concentrated on developing new detector generation scheme to improve the algorithm performance [12]. Most of the used algorithms in negative selection based detector generation are evolutionary computation and swarm intelligence algorithms, especially genetic [24,28] and particle swarm algorithms [29,30].

A genetic algorithm based on negative selection algorithm for detector generation was introduced [31]. They only focused on optimizing the non-overlapping of hyper-sphere detectors to obtain the maximal non-self-space coverage using fitness function based on detector radius. Additional research [8] uses genetic algorithm with deterministic crowding niching technique for improving hyper-sphere detector generation. Deterministic crowding niching is used with genetic as a way for improving the diversification to generate more improved solutions. In Ostaszewski et al. [32], hyper-rectangular detectors are tested in anomaly detection. Detectors are created using a niching genetic algorithm and enhanced by a coevolutionary algorithm.

Another work for detecting deceived anomalies hidden in the self-regions using boundary detectors is introduced [28]. These detectors are generated with the help of evolutionary search algorithm. Another research for intrusion data classification is proposed [30]. This approach uses rough set for feature selection along with a modified version of standard particle swarm intelligence called simplified swarm optimization for intrusion data classification.

As an improvement to hyper-spheres detectors, hyper-ellipsoid detectors are generated by evolutionary algorithm (EA) [33]. These detectors are more flexible because they can be stretched and reoriented the way that minimize the number of the needed detectors that cover similar non-self-space.

As far as we know, multi-start metaheuristic methods have gained no attention in negative selection based detector generation for anomaly detection. Its powerful diversification is much suitable for large domain space which is a feature of intrusion detection training datasets. Furthermore, most of previous research pays a great attention to detection accuracy and false positive rate, but no interest in studying the number of generated detectors and its generation time with different training dataset sizes. This paper introduces a new negative selection based detector generation methodology based on multi-start metaheuristic methods with the performance evaluation of different parameter values. Moreover, different evaluation metrics are measured to give a complete view of the performance of the proposed methodology. Results prove that the proposed scheme outperforms other competitors of machine learning algorithms.

Theoretic aspects of techniques

The basic concept of multi-start methods is simple: start optimization from multiple well-selected initial starting points, in hopes of locating local minima of better quality (which have smaller objective function values by definition), and then report back the local minimum that has the smallest objective function value to be a global minimum. The main challenges in multi-start optimization are selecting good starting points for optimization and conducting the subsequent multiple optimization processes efficiently.

In Ugray et al. [5], Multi-start framework is introduced with two phases, global phase and local phase. In global phase, scatter search [34] is used to intelligently perform a search on solution space [35]. This search aims to select good initial start points for being used in local phase. It operates on a set of solutions called the reference set or population. Elements of the population are maintained and updated from iteration to iteration. In local phase, nonlinear programming local solver is used with elements of the global phase reference set as a starting point input. Local solvers use values and gradients of the problem functions to generate a sequence of points that, under fairly general smoothness and regularity conditions, converge to a local optimum. The main widely used classes of local solver algorithms are successive quadratic programming (SQP) and generalized reduced gradient (GRG) [36].

Another work [37] introduces a multi-start approach that is based on the concept of regions of attraction to local minima. The region of attraction to a local minimum is a set of starting points from which optimization converges to that specific local minimum. A set of uniformly distributed points are selected as initial start points then evaluated using the objective function to construct regions of attraction. The goal is to start optimization exactly once from within the region of attraction of each local minimum, thus ensuring that all local minima are identified and the global minimum is selected. Local solver is invoked with each selected start point and then the obtained solution is used to update start points set. The process is repeated several times to obtain all local minima.

The proposed approach uses k-means clustering algorithm to identify good starting points for the detector generation based on a multi-start algorithm while maintaining their diversity. These points are used as an input to local solvers in hope to report back all local minima. K-means is one of the most widely used algorithm for geometric clustering [38]. It is a local search algorithm that partitions a set of observations (x_1, x_2,

…, x_n) into k clusters where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets ($k \leqslant n$), $S = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares as [39] follows:

$$\min_{S} \sum_{i=1}^{k} \sum_{x_j \in s_i} ||x_j - \mu_i||^2$$

where μ_i is the mean of points in cluster S_i. This can be reached by seeding with k initial cluster centers and assigning every data point to its closest center, then recomputing the new centers as the means of their assigned points. This process of assigning data points and readjusting centers is repeated until it stabilizes. K-means is popular because of its simplicity and observed speed [40].

Methodology

In this section, a new anomaly detector generation approach is proposed based on negative selection algorithm concept. As number of detectors is playing a vital role in the efficiency of online network anomaly detection, the proposed approach aims to generate a suitable number of detectors with high detection accuracy. The main idea is based on using k-means clustering algorithm to select a reduced training dataset in order to decrease time and processing complexity. Also, k-means is used to provide a way of diversification in selecting initial start points used by multi-start methods. Moreover, the radius of hyper-sphere detectors, generated using multi-start, is optimized later by genetic algorithm. Finally, rule reduction is invoked to remove unnecessary redundant detectors. Detector generation process is repeated to improve the quality of detection. The main stages are shown in Fig. 1 and a detailed description of each stage is presented below.

Preprocessing

In this step, training data source (DS) is normalized to be ready for processing by later steps as follows:

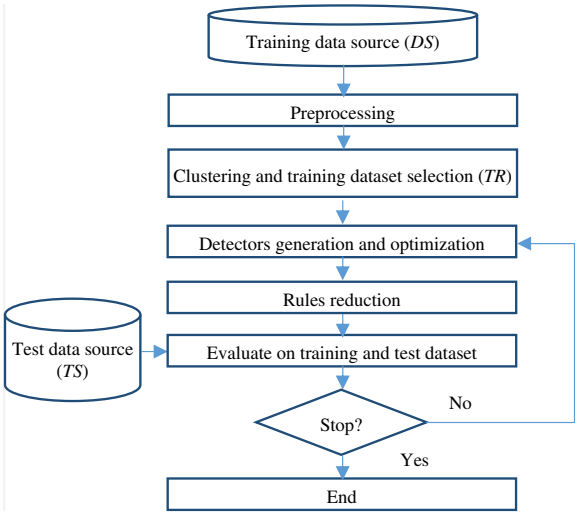


Fig. 1 The proposed approach main stages.

$$DS_{norm} = \begin{cases} (DS - \mu_{ds})/\sigma_{ds}, & \sigma_{ds} \neq 0 \\ DS - \mu_{ds}, & \sigma_{ds} = 0 \end{cases} \quad (1)$$

where

$$DS = \{x_{ij} | i = 1, 2, 3, \ldots, m \quad \text{and} \quad j = 1, 2, 3, \ldots, n\},$$

$$\mu_{ds} = \{\mu_j | \ j = 1, 2, 3, \ldots, n\},$$

$$\sigma_{ds} = \{\sigma_j | \ j = 1, 2, 3, \ldots, n\}.$$

DS is m samples with n column attributes, x_{ij} is the jth column attribute in the ith sample, μ_{ds} and σ_{ds} are $1xn$ matrix which are the training data mean and standard deviation respectively for each of the n attributes. Test dataset which is used to measure detection accuracy is also normalized using the same μ_{ds} and σ_{ds} as follows:

$$TS_{norm} = \begin{cases} (TS - \mu_{ds})/\sigma_{ds}, & \sigma_{ds} \neq 0 \\ TS - \mu_{ds}, & \sigma_{ds} = 0 \end{cases} \quad (2)$$

Clustering and training dataset selection

In order to decrease time complexity and number of detectors to be generated in later stages, small sample training dataset (TR) should be selected with a good representation of the original training dataset. So, k-means clustering algorithm is used to divide DS into k clusters. Then, TR samples are randomly selected and distributed over the labeled DS sample classes and clusters in each class to get a small number of TR samples (sz). The selection process is as follows:

Step 1: Count the number of DS samples in each class cluster (c). Let n is the number of available sample classes, k is the number of k-means clusters, then C_{ij} is the number of samples at the jth cluster in the ith class.
Step 2: Calculate the number of samples to be selected from each class cluster (CC).

$CC = 0$,
Loop:
  $step = (sz - \sum_{i=1}^{n}\sum_{j=1}^{k} C_{ij})/(n * k)$,
  $CC_{ij} = CC_{ij} + step, \quad \forall CC_{ij} < C_{ij}$
  If $CC_{ij} > C_{ij}$ *then* $CC_{ij}C_{ij}$
  If $sz < \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}$, *stop.*
end

- Step 3: Construct TR dataset from DS by randomly select a number of CC_{ij} samples from the jth cluster in the ith class.

Detector generation using multi-start algorithm

Multi-start searching algorithm focuses on strategies to escape from local optima and perform a robust search of a solution space. So, it is suitable for generating detectors which is used later to detect anomalies. Hyper-sphere detectors are used and defined by its center and radius. The idea is to use multi-start for solution space searching to get the best available hyper-spheres that cover most of the normal solution space. Multi-start parameters used in this work are chosen as follows:

Initial start points: the choice of this multi-start parameter is important in achieving diversification. So, an initial start number (*isn*) of points is selected randomly from normal *TR* samples and distributed over normal clusters.

Solution boundaries upper and lower bounds are important to limit the solution space. Let x_{ij} is the value of the ith sample at the jth column in the training data source DS_{norm} which is m samples with n column attributes, and detector radius $r = \{r \in \mathbb{R} | 0 < r \leq rrl\}$ where rrl is the hyper-sphere radius upper bound. So,

$u_j = max(x_{ij})\ where\ i = 1, 2, 3, \ldots, m,$

$l_j = min(x_{ij})\ where\ i = 1, 2, 3, \ldots, m,$

$$UB = (u_1, u_2, u_3, \ldots, u_n, rrl)$$

$$LB = (l_1, l_2, l_3, \ldots, l_n, 0),$$

where UB and LB are the upper and lower bounds for our solution space. The gotten solutions (detectors) $S = \{s_1, s_2, s_3, \ldots, s_{isn}\}$ are in the form of $S_i = (u_{i1}, u_{i2}, u_{i3}, \ldots, u_{in}, r_i)$ where hyper-sphere center is at $S_{center} = (u_{i1}, u_{i2}, u_{i3}, \ldots, u_{in})$ and hyper sphere radius is r_i.

Objective function

Generating detectors is controlled by fitness function which is defined as:

$$f(s_i) = \begin{cases} N_{abnormal}(s_i) - N_{normal}(s_i), & itr = 1 \\ N_{abnormal}(s_i) - N_{normal}(s_i) + old\_intersect(s_i), & itr > 1 \end{cases} \quad (3)$$

where *itr* is the iteration number of repetitive invoking detectors generation, $N_{abnormal}(s_i)$ is the number of abnormal samples covered by detector s_i, $N_{normal}(s_i)$ is the number of normal samples covered by detector s_i and $old\_intersect(s_i)$ is the percent of $N_{normal}(s_i)$ samples that are detected by detectors in previous iterations. The use of $old\_intersect(s_i)$ in next iterations is important to generate new detectors which are far as possible from the previously generated ones.

Anomaly detection is established by forming rules from the generated detectors. Each rule has the form of

$$if(dist(S_{center}, x) \leqslant r)\ then\ \{normal\}\ else\ \{abnormal\}$$

where r is the detector hyper-sphere radius and $dist(S_{center}, x)$ is the Euclidean distance between detector hyper sphere center S_{center} and test sample x.

Detector radius optimization using genetic algorithm

The previously generated detectors may cover normal samples as well as abnormal sample. So, further optimization is needed to adopt only detectors radius to cover the maximum possible number of only normal samples. Multi-objective genetic algorithm is used to make this adoption.

Initial population each detector radius is initialized to its value generated by multi-start algorithm.

Solution boundaries detector radius boundary is $r = \{r \in \mathbb{R} | 0 < r \leq rrl\}$, where rrl is the hyper-sphere upper bound.

Objective function fitness function which optimizes detector radius is defined as:

$$f(r_i) = N_{abnormal}(r_i) - N_{normal}(r_i) \quad (4)$$

where the number of abnormal samples covered by detector s_i is $N_{abnormal}(r_i)$ and $N_{normal}(s_i)$ is the number of normal samples covered by detector s_i using r_i as its radius.

Detectors reduction

Reducing the number of detectors is a must to improve effectiveness and speed of anomaly detection. Reduction is done over S which is the combination between recently generated detectors and previously generated detectors if exist and is done as follows:

- Step 1: First level reduction is as follows:

$$if\ N_{abnormal}(s_i) > thr_{maxabnormal}\ or\ N_{normal}(s_i) < thr_{minnormal}\ then$$

$$remove\ detector\ s_i, \forall s_i, \in S,$$

where $thr_{maxabnormal}$ is the maximum allowed number of abnormal samples to be covered by detector s_i, $thr_{maxabnormal}$ is set to 0. $thr_{minnormal}$ is the minimum allowed number of normal samples to be covered by detector s_i.

- Step 2: Another level of reduction intends to remove any detector s_i, if its N_{normal} is covered by one or more bigger detectors with a percent equal or more than $thr_{intersect}$. The more $N_{normal}(s_i)$), the bigger the detector is. $thr_{intersect}$ is set to 100% so as to remove any detector that is totally covered by one or more repeated or bigger detectors.

Repetitive evaluation and improvements

Anomaly detection performance is measured at each iteration by applying the reduced detectors $S_{reduced}$ from previous stage on the original training dataset at the first iteration TR_{org}. If improvement in accuracy is noticed, new training dataset *TR* is created to work on it in later iterations. New *TR* is a combination between all normal samples not covered N_{normal_nc} by $S_{reduced}$ plus all abnormal samples in the original training dataset TR_{org}. If no improvement in accuracy, then use $S_{reduced}$ and new *TR* of previous iteration as if they are the current. Also, new *isn* is computed as $isn_{new} = N_{normal\_nc} * isp$ where $\{isp \in \mathbb{R} | 0 < isp < 1\}$.

Steps 3–6 are repeated for a number of iterations. Different conditions can be invoked to stop the repetitive improvement process, i.e. a maximum number of iterations are reached, maximum number of consecutive iterations without improvement occurs or a minimum percent of training normal samples coverage exists.

Results and discussion

Experimental setup

In this experiment, NSL-KDD dataset is used for evaluating the proposed anomaly detection approach. This dataset is a modified version of KDDCUP'99 which is the mostly widely used standard dataset for the evaluation of intrusion detection

Table 1 Distribution of different classes in train (*DS*) and test dataset (*TS*).

Class	Train (*DS*)	Test (*TS*)
Normal	812,814	47,911
Dos	247,267	23,568
Probe	13,860	2682
U2R	999	2913
R2l	52	215
Total size	1,074,992	77,289

systems [41]. This dataset has a large number of network connections with 41 features for each of them which means it is a good example for large scale dataset to test on. Each connection sample belongs to one of five main labeled classes (Normal, DOS, Probe, R2L, and U2R). NSL-KDD dataset includes training dataset *DS* with 23 attack types and test dataset *TS* with additional 14 attack types. Distribution of connections over its labeled classes for training and test dataset is tabulated in Table 1. Our experiments ran on a system with 3.0 GHz Intel® Core™ i5 processor, 4 GB RAM and Windows 7 as an operating system.

Based on NSL-KDD training dataset, clustering is used to select different sample training dataset (*TR*) sizes (*sz*) with different cluster numbers (*k*) for each of them. The distribution of *TR* samples to its labeled classes in is stated in Table 2.

Results

In this section, performance study of our approach is held using different parameters values. The results are obtained using matlab 2012 as a tool to apply and carry out the proposed approach. Multi-start searching method [5] and genetic algorithm parameters are as default except the mentioned parameters in Table 3. Also, four parameters are selected to study its effect on performance which are stated in this table. The different values given to these parameters are dependent on the selected NSL-KDD dataset and need further study in future work to be chosen automatically. Performance results are averaged over five different copies of each sample training dataset *TR* along with the different values given to the studied parameters. Performance evaluation is measured based on number of generated detectors (rules), time to generate them, test accuracy and false positive rate during each repetitive improvement iteration using NSL-KDD test dataset. Classification accuracy and false positive rate (FBR) are calculated as follows:

$$Classification\,accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5}$$

$$False\,positive\,rate = \frac{FP}{TN + FP} \tag{6}$$

where true positive (*TP*) is normal samples correctly classified as normal, false positive (*FP*) is normal samples incorrectly classified as abnormal, true negative (*TN*) is abnormal samples correctly classified as abnormal and false negative (*FN*) is abnormal samples incorrectly classified as normal.

To study the effect of each one of the selected four parameter, a certain level of abstraction should be done by averaging the results over other parameters next to the studied parameter in (*sz,isn,rrl,k*). Fig. 2 shows the overall performance results of the proposed approach averaged over (*isn,rrl,k*) using training dataset sizes (*sz* = 5000,10,000,20,000,40,000,60,000) at different iterations (*itr* = 1,2,3,4,5). It is noted that, performance measures are gradually increased as increasing the number of iterations and become consistent at *itr* > 1. The reason behind this is that the generated detectors at early iterations try to cover most of the volumes occupied by normal samples inside

Table 2 Distribution of different classes in reduced sample train dataset (*TR*).

sz	*k*	Normal	Dos	Probe	U2R	R2l	True size
5000	100	2802	844	1061	999	52	5758
	200	3150	789	932	999	52	5922
	300	3241	709	822	999	52	5823
	400	3365	688	756	999	52	5860
10,000	100	5873	1750	2011	999	52	10,685
	200	6448	1629	1683	999	52	10,811
	300	6721	1472	1546	999	52	10,790
	400	6951	1402	1439	999	52	10,843
20,000	100	12,085	3550	3680	999	52	20,366
	200	13,124	3313	3186	999	52	20,674
	300	13,691	2997	2872	999	52	20,611
	400	14,194	2879	2645	999	52	20,769
40,000	100	25,361	7255	6391	999	52	40,058
	200	26,746	6813	5824	999	52	40,434
	300	27,974	6068	5394	999	52	40,487
	400	28,781	5789	4818	999	52	40,439
60,000	100	39,207	11,165	8581	999	52	60,004
	200	41,002	10,342	7847	999	52	60,242
	300	42,757	9185	7252	999	52	60,245
	400	43,954	8866	6463	999	52	60,334

Table 3 The settings of parameters used for my approach.

Parameter	Value
Multi-start searching method	
Minimum distance between two separate objective function values	10
Minimum distance between two separate points	0.001
isp	0.1
Genetic searching algorithm	
Population size	20
Number of generations	20
Detectors reduction	
$thr_{minnormal}$	10
$thr_{miaxabormal}$	0
$thr_{intersect}$	100%
Parameters under study	
Number of *TR* samples (*sz*)	5000, 10,000, 20,000, 40,000, 60,000
Multi-start initial start points (*isn*)	100, 200, 300
Detector radius upper bound (*rrl*)	2, 4, 6
Number of clusters (*k*)	100, 200, 300, 400

the training dataset and leave the remaining small volumes coverage to the later iterations. Therefore, much more increasing is observed in test accuracy at $itr = 1, 2$ compared to slow increasing at $itr > 2$. At the same time, an increasing in number of detectors (rules) and generation time is noted due to the need for more iterations to generate more detectors to cover the remaining normal samples in training dataset. False positive rate (FPR) follows the same increasing behavior because of the generation of some detectors to cover the boundaries between normal and abnormal training samples. So, a chance to misclassify abnormal samples to normal at testing dataset increases as more iteration number is invoked. As a trade of between these different performance measures, results should be chosen at $itr = 2$ as stability of these measures begins.

Furthermore, the bigger the size of the training dataset, the bigger the number of rules and generation time values. This is reasonable because more detectors are needed to achieve more coverage of normal training samples, which requires more processing time. On the other hand, increasing training dataset size has a smaller bad effect on test FPR and test accuracy especially at $itr > 2$. As an explanation, detectors generated at later iterations are pushed by the proposed approach to be as far as possible from the older ones. This means it tends to cover boundaries between normal and abnormal samples in training dataset which may have bad effect when testing them on unseen test dataset. So, as a tradeoff between different performance metrics, small training dataset (*TR*) sizes are preferable.

Performance evaluation at $itr = 2$ of different numbers of initial start points ($isn = 100, 200, 300$) averaged over (*rrl*,*k*) is shown in Fig. 3. At each training dataset size, increasing the number of initial start points gives multi-start method the opportunity to give best solutions with more coverage to normal samples at early iterations even though applying rule reduction at later stages. As a result, performance measures increase in general as increasing the number of initial start points (*isn*) with a small effect on FBR with lower number of rule and processing time. As increasing *sz* values, more detectors are needed to cover normal samples and hence, more processing time. Also, more boundaries between normal and abnormal samples exist which rise the false positive rate (FBR) and stop the growing of test accuracy at bigger training dataset sizes. Therefore, higher number of initial start points ($isn = 300$) is preferable.

Fig. 4 shows the performance of different detector radius upper limits ($rrl = 2,4,6$) at $itr = 2$, $isn = 300$ and averaged over (*k*). At each training dataset size, it is obvious that small values will generated more detectors to cover all normal samples while increasing the accuracy as a result of more detectors will fit into small volumes to achieve the best coverage. Lower values of ($rrl = 2$) along with small *TR* sizes could be a good

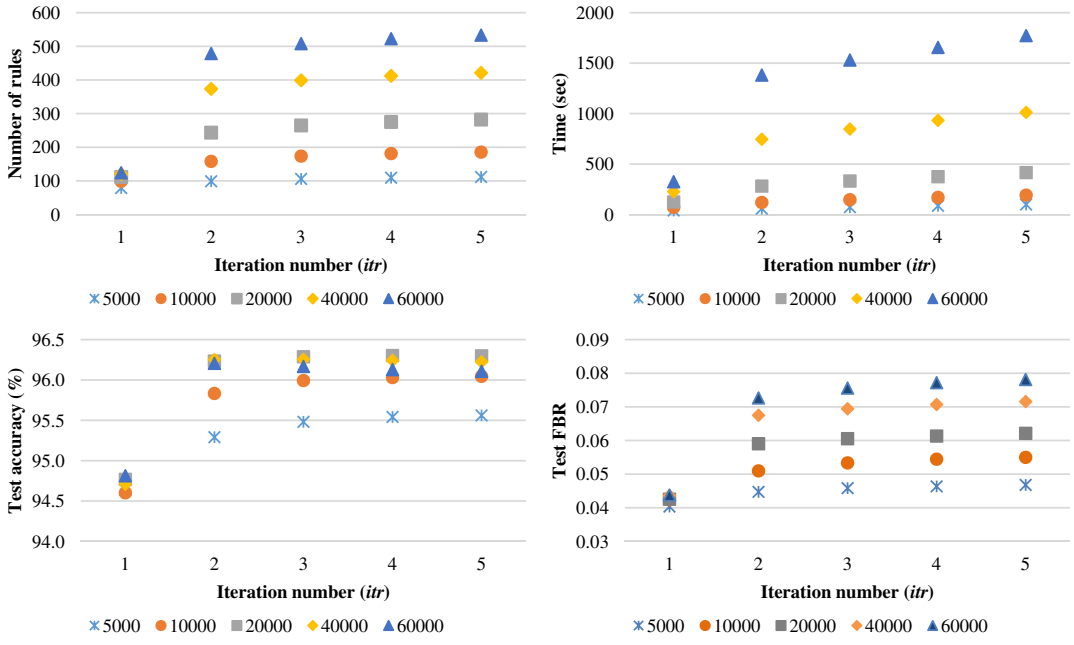


Fig. 2 Overall performance results for different training dataset sizes (*sz*) averaged over (*isn*,*rrl*,*k*).

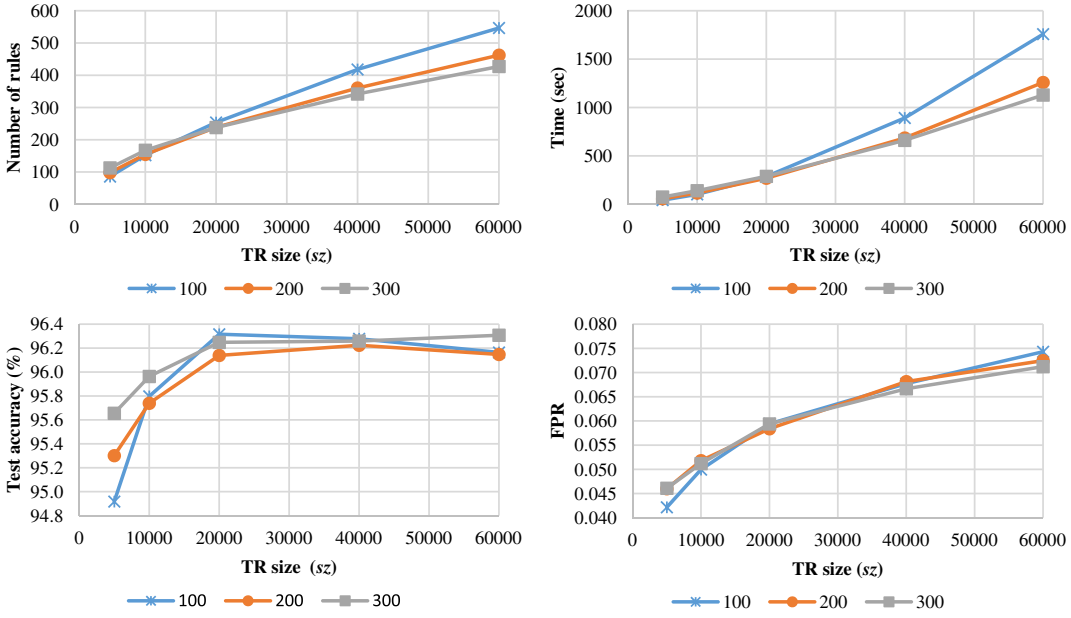


Fig. 3 Performance results for different initial start points numbers (*isn*) averaged over (*rrl*,*k*), *itr* = 2.

choice to have higher accuracy and lower FBR with small extra number of detectors and processing time.

According to results shown in Fig. 5, with higher number of clusters (k), there is a tendency to generate more detectors with higher FBR and slight variance in accuracy and. This is because of the distributed selection of training dataset (TR) samples over more clusters which gives more opportunity to represent smaller related samples found in training data source (DS_{norm}), hence, more rules are needed to cover these volumes. This distribution of samples increases the interference between normal and abnormal samples inside TR as increasing clusters number which badly affect FBR value, We can notice that medium value of (k = 200) is an acceptable tradeoff between different performance metrics.

As shown in Table 4 there are a tradeoff between getting low number of rules with small generation time and having high accuracy with low false positive rate. This table states a sample performance comparison between the results of best selected parameters values chosen earlier (at table rows 5–8) and other parameters values (at table rows 1–4, 9–12). At the first four rows, high accuracy with low false positive rate is obtained at itr >1, but with higher number of rules and generation time compared to the results stated at rows 5–8. On the other hand, rows 9–12 have lower rules number and less gen-

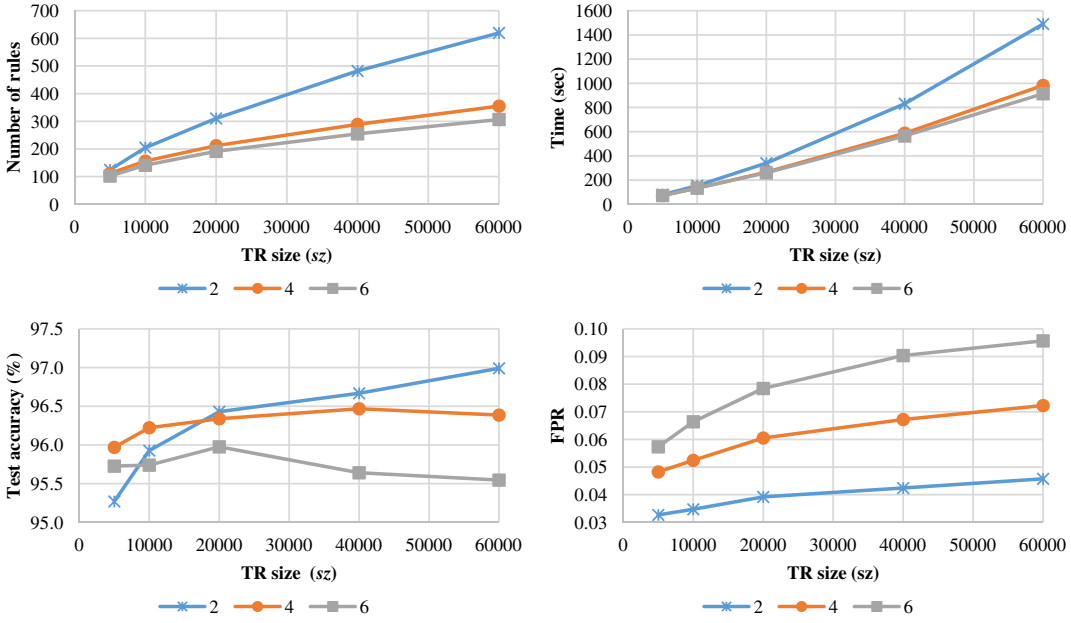


Fig. 4 Performance results for different radius upper bound values (*rrl*) averaged over (k), *itr* = 2, *isn* = 300.

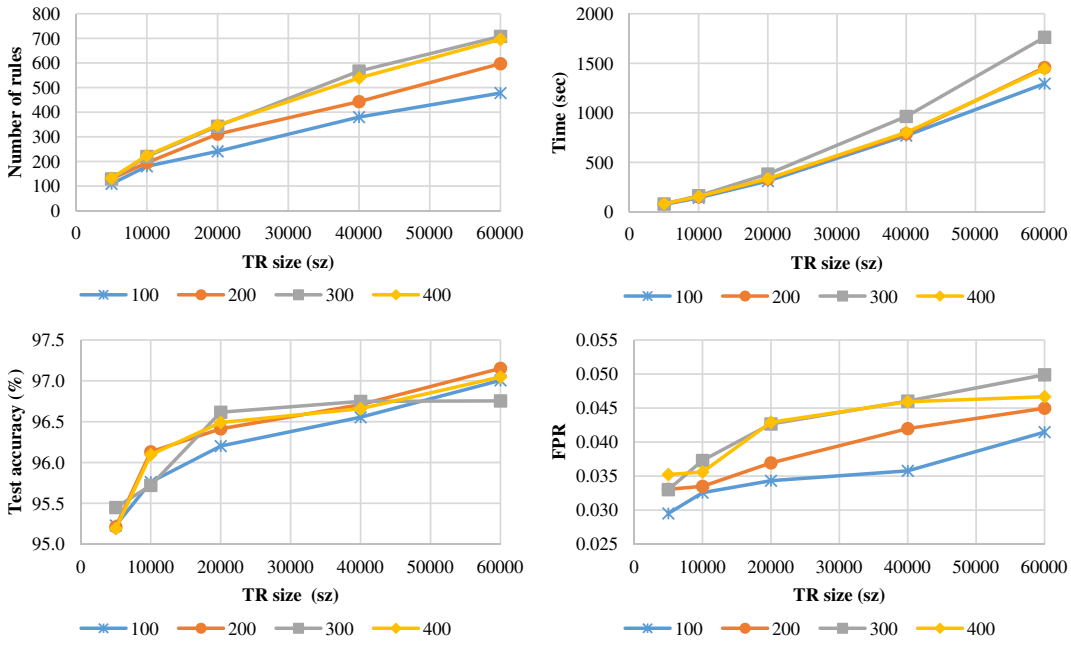


Fig. 5 Performance results for different clustering values (k) at $isn = 300$, $rrl = 2$.

eration time at itr >1, but with lower accuracy and higher FPR compared to the selected parameters values at rows 5–8. So, from these results, we can distinguish that results shown in bold at (isn = 300, rrl = 2, k = 200, itr = 2) are an acceptable trade of between different performance metrics as mentioned in the earlier discussion. With regard to other machine learning algorithms used for intrusion detection problems, Performance comparison between the proposed approach with best selected parameters values and six of these algorithms, Bayes Network (BN), Bayesian Logistic Regression (BLR), Naive Bayes (NB), Multilayer Feedback Neural Network (FBNN), Radial Basis Function Network (RBFN), and Decision Trees (J48) is shown in Fig. 6. Weka 3.6 is used as a tool to get performance results of these machine learning algorithms. These machine learning classifiers are trained by using our generated TR datasets. Results show that the proposed approach outperforms other techniques with higher accuracy, lower FBR and acceptable time.

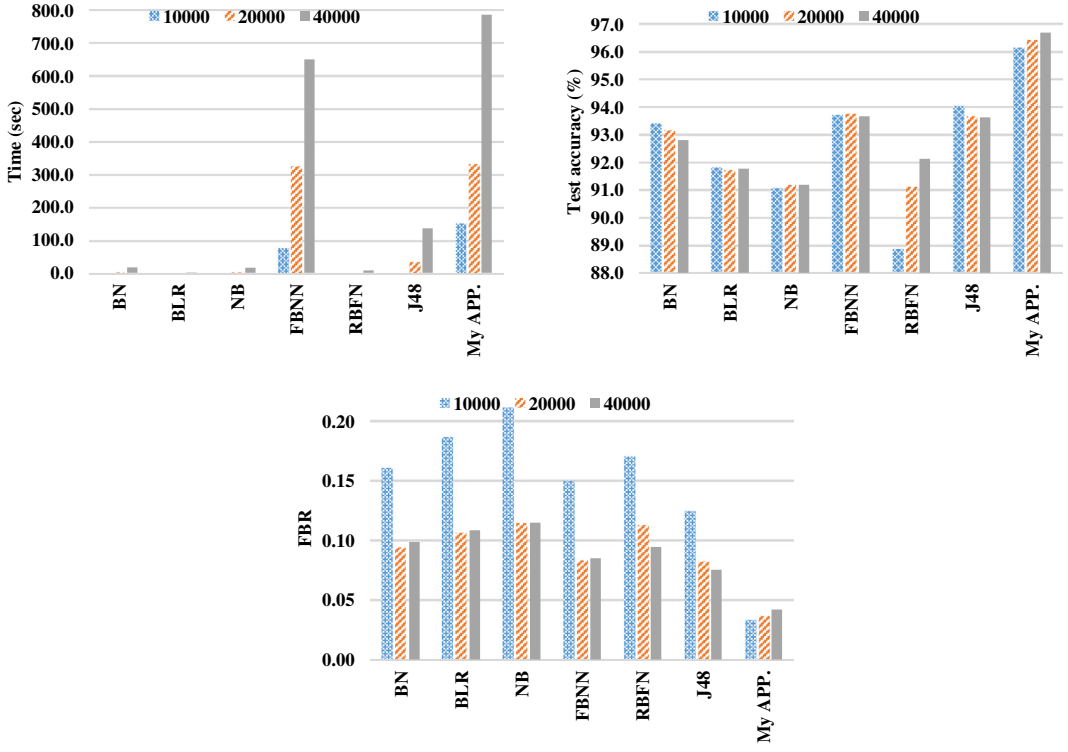


Fig. 6 Test accuracy, FBR, and time comparison between different machine learning algorithms and the proposed approach.

Table 4 Performance results of best selected parameters value and other parameters values.

sz	*isn*	*rrl*	*k*	Number of rules					Time (s)					Test accuracy (%)					Test FPR (%)				
				itr					itr					itr					itr				
				1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
10,000	200	2	400	123	202	218	249	259	70	136	181	225	258	94.8	95.5	95.8	96.4	96.4	0.032	0.037	0.038	0.038	0.038
20,000	100	2	200	69	345	369	381	391	63	317	391	452	509	91.4	97.0	97.0	97.0	97.1	0.015	0.035	0.035	0.035	0.035
40,000	100	2	300	77	634	673	689	700	105	1197	1346	1469	1584	92.0	97.1	97.1	97.1	97.1	0.029	0.044	0.047	0.048	0.048
60,000	100	2	300	75	861	902	919	932	160	2367	2576	2758	2929	92.6	96.6	96.6	96.6	96.6	0.029	0.051	0.052	0.053	0.053
10,000	**300**	**2**	**200**	**149**	**195**	**211**	**218**	**224**	**105**	**152**	**188**	**221**	**253**	**95.7**	**96.1**	**96.2**	**96.2**	**96.2**	**0.031**	**0.033**	**0.034**	**0.035**	**0.035**
20,000	**300**	**2**	**200**	**178**	**311**	**349**	**365**	**376**	**182**	**332**	**409**	**473**	**531**	**95.8**	**96.4**	**96.5**	**96.5**	**96.6**	**0.032**	**0.037**	**0.038**	**0.039**	**0.039**
40,000	**300**	**2**	**200**	**193**	**443**	**497**	**519**	**538**	**344**	**788**	**949**	**1086**	**1203**	**95.9**	**96.7**	**96.8**	**96.9**	**96.9**	**0.031**	**0.042**	**0.042**	**0.043**	**0.043**
60,000	**300**	**2**	**200**	**198**	**597**	**653**	**681**	**700**	**480**	**1457**	**1703**	**1901**	**2092**	**96.2**	**97.2**	**97.2**	**97.2**	**97.2**	**0.033**	**0.045**	**0.045**	**0.046**	**0.047**
10,000	300	6	100	103	119	124	129	132	107	128	144	159	173	95.3	95.9	95.9	96.0	96.0	0.055	0.057	0.057	0.059	0.059
20,000	300	6	100	118	152	168	175	181	192	247	283	314	342	95.7	96.1	96.1	96.0	96.0	0.057	0.067	0.070	0.074	0.077
40,000	300	6	100	129	209	236	246	256	344	545	632	697	760	95.6	95.8	95.8	95.8	95.7	0.062	0.078	0.084	0.085	0.087
60,000	300	6	100	143	258	283	300	310	481	872	1002	1105	1192	95.4	95.7	95.6	95.6	95.6	0.058	0.084	0.089	0.091	0.093

Conclusions

This paper presents a hybrid approach to anomaly detection using a real-valued negative selection based detector generation. The solution specifically addresses issues that arise in the context of large scale datasets. It uses k-means clustering to reduce the size of the training dataset while maintaining its diversity and to identify good starting points for the detector generation based on a multi-start metaheuristic method and a genetic algorithm. It employs a reduction step to remove redundant detectors to minimize the number of generated detectors and thus to reduce the time needed later for online anomaly detection.

A study of the effect of training dataset size (*sz*), number of initial start pointers for multi-start (*isn*), detector radius upper limit (*rrl*) and clustering number (*k*) is stated. As a balance between different performance metrics used here, choosing results at early iterations (*itr* = 2) using small training dataset size (*sz* = 10,000), higher number of initial start points (*isn* = 300), lower detector radius (*rrl* = 2) and medium number of clusters (*k* = 200) are preferable.

A comparison between the proposed approach and six different machine learning algorithms is performed. The results show that our approach outperforms other techniques by 96.1% test accuracy with time of 152 s and low test false positive rate of 0.033. Although, the existence of offline processing time overhead for the proposed approach which will be considered in future work, online processing time is expected to be minimized. The reason behind this is that a suitable number of detectors will be generated with high detection accuracy and low false positive rate. As a result, a positive effect on online processing time is expected.

In future, the proposed approach will be evaluated on other standard training datasets to ensure its high performance. Moreover, its studied parameter value should be chosen automatically according to the used training dataset to increase its adaptability and flexibility. In addition, detector generation time should be decreased by enhancing the clustering and detector radius optimization processes which will have a positive impact on the overall processing time as we expected. Finally, the whole proposed approach should be adapted to learn from normal training data only in order to be used in domains where labeling abnormal training data is difficult.

Conflict of interest

The authors have declared no conflict of interest

Compliance with Ethics Requirements

This article does not contain any studies with human or animal subjects.

References

[1] Liao HJ, Lin CHR, Lin YC, Tung KY. Intrusion detection system: a comprehensive review. J Netw Comput Appl 2013;36(1):16–24.

[2] Patcha A, Park JM. An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput Netw 2007;51(12):3448–70.

[3] García-Teodoro P, Díaz-Verdejo J, Maciá-Fernández G, Vázquez E. Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Secur 2009;28(1–2):18–28.
[4] Boussaïd I, Lepagnot J, Siarry P. A survey on optimization metaheuristics. Inf Sci 2013;237:82–117.
[5] Ugray Z, Lasdon L, Plummer J, Glover F, Kelly J, Martí R. Scatter search and local NLP solvers: a multistart framework for global optimization. Informs J Comput 2007;19(3):328–40.
[6] Genetic algorithm. [Online]; 2013 [cited 2013. <http://en.wikipedia.org/wiki/Genetic_algorithm>.
[7] Li W. Using genetic algorithm for network intrusion detection. In: Proceedings of the United States Department of Energy Cyber Security Group 2004 Training Conference; 2004. p. 24–7.
[8] Aziz ASA, Salama M, ella Hassanien A, El-Ola Hanafi S. Detectors generation using genetic algorithm for a negative selection inspired anomaly network intrusion detection system. In: FedCSIS proceedings of federated conference on computer science and information systems; Wroclaw: IEEE; 2012. p. 597–602.
[9] Forrest S, Perelson AS, Allen L, Cherukuri R. Self-nonself discrimination in a computer. In: Proceedings of the 1994 IEEE symposium on security and privacy; Oakland, USA: IEEE Computer Society; 1994. p. 202–12.
[10] Stibor T, Mohr P, Timmis J, Eckert C. Is negative selection appropriate for anomaly detection? In: Beyer HG, editor. GECCO '05 Proceedings of the 2005 conference on genetic and evolutionary computation. New York, NY, USA: ACM; 2005. p. 321–8.
[11] Ji Z, Dasgupta D. Revisiting negative selection algorithms. Evol Comput 2007;15(2):223–51.
[12] Dasgupta D, Yu S, Nino F. Recent advances in artificial immune systems: models and applications. Appl Soft Comput 2011;11(2):1574–87.
[13] Marti R, Moreno-Vega JM, Duarte A. Advanced multi-start methods. In: Gendreau M, Potvin JY, editors. Handbook of Metaheuristics. Springer US; 2010. p. 265–81.
[14] Wu SX, Banzhaf W. The use of computational intelligence in intrusion detection systems: a review. Appl Soft Comput 2010;10(1):1–35.
[15] Esponda F, Forrest S, Helman P. A formal framework for positive and negative detection schemes. IEEE Trans Syst Man Cybernetics, Part B 2004;34(1):357–73.
[16] Gonzalez F, Dasgupta D, Kozma R. Combining negative selection and classification techniques for anomaly detection. In: CEC '02. Proceedings of the 2002 congress on evolutionary computation. Honolulu, HI, USA; 2002. p. 705–10.
[17] Gonazalez FA, Dasgupta D. Anomaly detection using real-valued negative selection. Gen Program Evol Mach 2003;4(4):383–403.
[18] Xu X. Sequential anomaly detection based on temporal-difference learning: principles, models and case studies. Appl Soft Comput 2010;10(3):859–67.
[19] Assis MVOd, Rodrigues JJPC, Jr. MLP. A hybrid approach for anomaly detection on large-scale networks using HWDS and entropy. In: 21st international conference on software, telecommunications and computer networks (SoftCOM 2013). Split-Primosten, Croatia; 2013. p. 18–20.
[20] Wang SS, Yan KQ, Wang SC, Liu CW. An integrated intrusion detection system for cluster-based wireless sensor networks. Exper Syst Appl 2011;38(12):15234–43.
[21] Kartit A, Saidi A, Bezzazi F, El Marraki M, Radi A. A new approach to intrusion detection system. JATIT 2012;36(2):284–90.
[22] Shameli Sendi A, Dagenais M, Jabbarifar M, Couture M. Real time intrusion prediction based on optimized alerts with hidden Markov model. JNW 2012;7(2):311–21.
[23] García-Teodoro P, Díaz-Verdejo J, Maciá-Fernández G, Vázquez E. Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Sec 2009;28:18–28.
[24] Abadeh MS, Mohamadi H, Habibi J. Design and analysis of genetic fuzzy systems for intrusion detection in computer networks. Expert Syst Appl 2011;38(6):7067–75.
[25] Gong M, Zhang J, Ma J, Jiao L. An efficient negative selection algorithm with further training for anomaly detection. Knowl-Based Syst 2012;30:185–91.
[26] de Assis MVO, Carvalho LF, Rodrigues JJPC, Lemes Proenca M. Holt-Winters statistical forecasting and ACO metaheuristic for traffic characterization. In: IEEE International Conference on Communications (ICC). Atlanta, GA, USA; 2013. p. 2524–8.
[27] Adaniya MHAC, Lima MF, Rodrigues JJPC, Abraão T, Jr. MLP. Anomaly detection using DSNS and firefly harmonic clustering algorithm. In: IEEE international conference on communications (IEEE ICC 2012). Ottawa, Canada; 2012. p. 10–5.
[28] Wang D, Zhang F, Xi L. Evolving boundary detector for anomaly detection. Expert Syst Appl 2011;38(3):2412–20.
[29] Kalyani S, Swarup KS. Classifier design for static security assessment using particle swarm optimization. Appl Soft Comput 2011;11(1):658–66.
[30] Chung YY, Wahid N. A hybrid network intrusion detection system using simplified swarm optimization (SSO). Appl Soft Comput 2012;12(9):3014–22.
[31] Gao XZ, Ovaska SJ, Wang X. Genetic algorithms-based detector generation in negative selection algorithm. In: SMCals/06 Proceedings of IEEE mountain workshop on adaptive and learning systems. Utah, Logan, USA: IEEE; 2006. p. 133–7.
[32] Ostaszewski M, Seredynski F, Bouvry P. Immune anomaly detection enhanced with evolutionary paradigms. In: GECCO '06, Proceedings of the 8th annual conference on Genetic and evolutionary computation. New York, NY, USA: ACM; 2006. p. 119–26.
[33] Shapiro JM, Lamont GB, Peterson GL. An evolutionary algorithm to generate hyper-ellipsoid detectors for negative selection. In: Beyer HG, editor. GECCO '05. Proceedings of the 2005 conference on Genetic and evolutionary computation. New York, NY, USA: ACM; 2005. p. 337–44.
[34] Laguna M, Marti R. Scatter search: methodology and implementations in C. 1st ed. Boston, MA: Kluwer Academic Publishers; 2003.
[35] Glover F. A template for scatter search and path relinking. In: Hao JK, Lutton E, Ronald E, Schoenauer M, Snyers D, editors. Artificial evolution. Nimes, Berlin Heidelberg: Springer; 1998. p. 1–51.
[36] Edgar TF, Himmelblau DM, Lasdon LS. Nonlinear programming with constraints. In: Glandt ED, Klein MT, Edgar TE, editors. Optimization of chemical processes. New York: McGraw-Hill; 2001. p. 264–350.
[37] Hughes E, Huang T, Xu Y. Parallel multistart nonlinear optimization with PROC OPTMODEL. In: SAS SUGI proceedings: Operations Research, SAS Institute Inc. 158(Paper 158); 2013. p. 158 1–12.
[38] Jain AK. Data clustering: 50 years beyond K-means. Pattern Recogn Lett 2010;31(8):651–66.
[39] k-means clustering. [Online]; 2013 [cited 2013. <http://en.wikipedia.org/wiki/K-means_clustering>.
[40] Vattani A. K-means requires exponentially many iterations even in the plane. Discrete Comput Geomet 2011;45(4):596–616.
[41] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: CISDA 2009 Proceedings of IEEE symposium on computational intelligence for security and defense applications; Ottawa, Canada: IEEE Press; 2009. p. 1–6.