A Novel Two-phase Cycle Algorithm for Effective Cyber Intrusion Detection in Edge Computing

Yiguang Gong (✉ yiguang-gong@nuist.edu.cn)

Nanjing University of Information Science and Technology <https://orcid.org/0000-0001-6517-7040>

Yunping Liu

Nanjing University of Information Science and Technology

Chuanyang Yin

Nanjing University of Information Science and Technology

Research

Keywords: Two-phase Cycle Algorithm, Edge Computing, Cyber Intrusion Detection

Posted Date: November 6th, 2020

DOI: <https://doi.org/10.21203/rs.3.rs-101252/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Version of Record: A version of this preprint was published at EURASIP Journal on Wireless Communications and Networking on July 10th, 2021. See the published version at <https://doi.org/10.1186/s13638-021-02016-z>.

RESEARCH

A Novel Two-phase Cycle Algorithm for Effective Cyber Intrusion Detection in Edge Computing

Yiguang Gong*, Yunping Liu and Chuanyang Yin

*Correspondence:
yiguang-gong@nuist.edu.cn
School of Automation, Nanjing
University of Information Science
& Technology, 210000 Nanjing,
China
Full list of author information is
available at the end of the article

Abstract

Edge computing extends traditional cloud services to the edge of the network, closer to users, and is suitable for network services with low latency requirements. With the rise of edge computing, its security issues have also received more and more attention. In this paper, a novel two-phase cycle algorithm is proposed for effective cyber intrusion detection in edge computing based on multi-objective genetic algorithm (MOGA) and modified back propagation neural network (MBPNN), namely TPC-MOGA-MBPNN. In the first phase, the MOGA is employed to build multi-objective optimization model that tries to find Pareto optimal parameter set for MBPNN. The Pareto optimal parameter set is applied for simultaneous minimization of average false positive rate (Avg FPR), mean squared error (MSE), and negative average true positive rate (Avg TPR) in the dataset. In the second phase some MBPNNs are created based on the parameter set obtained by MOGA and are trained to search for more optimal parameter set locally. The parameter set obtained in the second phase is used as the input of the first phase, and the training process is repeated until the termination criteria are reached. Benchmark dataset namely KDD cup 1999 is used to demonstrate and validate the performance of the proposed approach for intrusion detection. The proposed approach can discover a pool of MBPNN based solutions. Combining these MBPNN can significantly improve prediction performance, and a GA is used to find the optimal MBPNN combination. The result shows that the proposed approach could reach an accuracy of 98.81% and a detection rate of 98.23%, which outperform most systems of previous works found in the literature. In addition, the proposed approach is a generalized classification approach that is applicable to the problem of any field having multiple conflicting objectives.

Keywords: Two-phase Cycle Algorithm; Edge Computing; Cyber Intrusion Detection

1 INTRODUCTION

With the advance of edge computing, the number of edge services running on mobile devices grows explosively [1]. Edge computing can reduce processing times and improve application performance, but it also brings some new challenges. Edge computing is driving the need for more bandwidth across the network, 5G is introduced for offering sufficient communication bandwidth [2]. Privacy is a serious issue in edge computing as users data is collected, processed, transmitted, and shared over edge nodes, so it is becoming a necessity to secure user privacy before these data are integrated together [3]. The explosive growth and variety of information available on edge nodes frequently overwhelm users, recommender systems are a promising way for users to quickly find the valuable information that they are interested in from

massive data [4, 5, 6]. In addition to these challenges, edge computing introduces a scale of cyber security challenges regular data center operators may not be used to dealing with. An intrusion, which is one of the main cyber security challenges, is any set of actions intended to compromise the confidentiality, integrity, or availability of a resource [7]. Cyber intrusions are prevalent, increasingly sophisticated, and are adept at hiding from detection [8]. To counteract this ever-evolving threat, Network-based Intrusion Detection System (NIDS) has been considered to be one of the most promising methods.

Intrusion detection techniques have become a significant topic of research in recent years. Many researchers propose different algorithms in different categories, such as Artificial Neural Networks (ANNs)[9], SVM[10], k-nearest neighbor [11], random forest [12], deep learning approaches [13], Bayesian approaches [14], decision trees [15]. As a result, the performance of detection techniques is getting better and stronger.

Artificial Neural Network (ANN), the computing paradigm that mimics the way neuron system of human brain work is widely used in cyber intrusion detection. Hodo E, Bellekens X, et al [16]. presents a multi-level perceptron, a type of supervised Artificial Neural Network (ANN) to detect Distributed Denial of Service (DDoS/DoS) attacks. The experimental results demonstrate 99.4% accuracy and can successfully detect various DDoS/DoS attacks. Chuan-Long Y, Yue-Fei Z, et al[17]. propose a deep learning approach for intrusion detection using recurrent neural networks (RNN-IDS). The experimental results show that RNN-IDS is suitable for modeling a classification model with high accuracy. Sheraz Naseer, Yasir Saleem[18] propose a deep convolutional neural network (DCNN) based intrusion detection system (IDS). The experimental results of proposed DCNN based IDS shows promising results for real world application in anomaly detection systems. Benmessahel I, Xie K, et al[19]. present an evolutionary neural network (ENN) which is a combination of ANN and evolutionary algorithm (EA), and experiments shows that this approach is effective for cyber intrusion detection. A. Arul Anitha, L. Arockiam[20] propose an Artificial Neural Network based IDS (ANNIDS) technique based on Multilayer Perceptron (MLP) to detect the attacks initiated by the destination oriented direct acyclic graph information solicitation (DIS) attack and version attack in IoT environment. Zichao Sun, Peilin Lyu[21] uses the LSTM neural network with long and short memory function to train the KDD99 dataset, and identify the DOS according to the trained model. This is a research process of the planned adjustment for the hyperparameters to find the optimal solution after processing the data. Shenfield, Day and Ayesh [22] present a novel approach to detecting malicious cyber traffic using artificial neural networks suitable for use in deep packet inspection based IDS. Results presented show that this novel classification approach is capable of detecting shell code with extremely high accuracy and minimal numbers of false identifications. Amruta and Talha [23] present a Denial of Service Attack Detection system using ANN for wired LANs. The proposed ANN classifier gives ninety six percent accuracy for their training data-set. Most of the systems have produced promising classification accuracy.

ANN has the ability to approximate an arbitrary function mapping and learn from examples much like the human brain. In many cases, ANN surpasses the conventional statistical method for the classification task in various fields of applications

[24]. However, designing an ANN is a difficult process. Its performance depends on the optimization of various design parameters such as choosing an optimal number of hidden nodes, suitable learning algorithm, learning rate and initial value of the weights, and some objectives are conflicting each other such as accuracy and complexity. Therefore, a multi-objective optimization (MOO) is considered to be a more realistic approach to the design of ANN, compared with the single-objective approach[25]. In addition, ANN has some shortcomings such as slow convergence speed, entrapment in local optimum, unstable network structure etc[26]. In contrast, the genetic algorithm(GA) exhibits its characteristic of global search and quick convergence ability, and is the most widely used technique in data mining and knowledge discovery [27]. On the other hand the method of Pareto-optimality has been widely used in MOO[28]. It offers a pool of non-inferior individual solutions and ensemble solutions instead of a single optimum, accordingly provides more degrees of freedom in selection of proper solutions. The multi-objective genetic algorithm (MOGA) [29] and the non-dominated sorting genetic algorithm (NSGA II [30, 31], NSGA III [32]) are two examples of GA based MOO which apply the concept of Pareto-optimality.

Some multi-objective genetic algorithms (MOGA) based approach is proposed for effective intrusion detection based on benchmark datasets. Elhag S , Altalhi A , et al[33]. propose a multi-objective evolutionary fuzzy system which can be trained using different metrics. The system obtains more accurate solutions, and allows the final user to decide which solution is better suited for the current network characteristics. M. Stehlík, A. Saleh, et al[34]. propose multi-objective evolutionary algorithms (NSGA-II and SPEA2 [35]) for intrusion detection parametrization, which focus on the impact of an evolutionary algorithm (and its parameters) on the optimality of found solutions, the speed of convergence and the number of evaluations. Kumar G., Kumar K[27] proposes a three phase MOGA based Micro Genetic Algorithm2 (AMGA2) [36] which considers conflicting objectives simultaneously like detection rate of each attack type, error rate, accuracy, diversity etc. In first phase, a Pareto front of non-inferior individual solutions is approximated. In second phase, entire solution set is further refined, and another improved Pareto front of ensemble solutions over that of individual solutions is approximated. In third phase, a combined method like majority voting method is used to fuse the predictions of individual solutions for determining prediction of ensemble solution. The experiments conducted on two benchmark datasets demonstrate that the proposed approach can discover individual solutions and ensemble solutions for intrusion detection.

This paper aims to develop a novel two-phase cycle training algorithm for intrusion detection. The MOGA based approach is used to find the Pareto optimal parameter set for the neural networks. A MBPNN set is created based on Pareto optimal parameter set and trained to find more optimal parameter set locally. The proposed approach can discover a pool of MBPNN based solutions to detect the intrusions accurately.

The rest of this paper is organized as follows: Section II presents an overview of the proposed methodology. Experimental results and discussion is presented in Section III. Finally, the concluding remarks of the study are provided in Section IV.

2 MATERIALS AND METHODS

2.1 The Proposed Approach

The TPC-MOGA-MBPNN includes training session, testing session and the combined method, and them are described below.

2.1.1 The training session

The training session as illustrated in Figure 1 is implemented by TPC-MOGA-MBPNN. In the first phase, a MOGA tries to find the Pareto optimal parameter set for the neural networks. The MOGA considers Avg TPR, Avg FPR and MSE on training dataset as the objectives to be minimized. Meanwhile, the weights, biases and gain factors of neural networks become the genotype to be evolved simultaneously by MOGA. In this phase we use the global search capability of MOGA to search for the initial parameter values of neural network, thereby avoid being trapped in local minima. In the second phase, neural network set is trained to find more optimal parameter set locally. The nondominated parameter set obtained from the first phase is considered as the input archive. A neural network set is generated by selecting excellent parameter set from the input archive, then back propagation is used to update the weights for the neurons, in order to bring the error function to a minimum. In this phase we use the local search and fast learning capabilities of the neural network to refine the parameter set from the first phase, and obtain a more optimal nondominated parameter set. The nondominated parameter set obtained in the second phase is used as the input of the first stage, and the training process of two-phase algorithm is repeated until the termination criteria are met. In addition, a new self-adaptive parameter adjustment strategy is also used in the training process of genetic algorithm and neural network.

The detail implementation of the proposed algorithm is as follows:

Step 1: *Generate random initial population.* Create random initial population, and maintains it in a solution archive. The structure of the chromosomes that make up the entire population is illustrated in Figure 2.

The weight segment in chromosomes represents the weights between input layer and hidden layer, and also includes the weights between hidden layer and output layer. The bias segment is dedicated to representing the biases of hidden nodes and output nodes. The gain factors of hidden nodes and output nodes are represented by the gain factor segment. After studying multiple training results of neural network, the numerical range of neural network parameters is estimated, and the initial value of neural network parameters is limited between - 1000 and 1000.

Step 2: *Evaluate objective fitness functions.* Calculate the objective fitness function values for neural network corresponding to solution, then sort all solutions based on these values and update the solution archive. First, create neural networks based on the parameters represented by all chromosomes in the MOGA population. Then, these neural networks are used to classify the samples in the training dataset one by one. Next, calculate and record TPR and FPR for five output types. The TRP and FPR are the most important criteria for the classifier, at the same time, MSE is an important criterion for evaluating the performance of neural network classification. MSE is record and is calculated as follows.

$$MSE = \frac{1}{n} \sum_{l=1}^n (D_l - Y_l^o)^2 \quad (1)$$

Where D_l and Y_l^o represent the desired output and actual output for the l -th sample of the neural network respectively, and n is the number of samples in the training dataset.

After that calculate Avg TPR and Avg FPR for the types, and they together with MSE constitute three objective fitness functions of the MOGA. Using Avg TRP instead of TRP as objective fitness function of the algorithm can avoid bias against specific attack types, especially those with less training samples such as R2L and U2R. For the same reason, Avg FRP is also used as objective fitness function. Finally sort all chromosomes and update the solution archive. The chromosomes are sorted according to the following two rules [37]:

- (1) First, chromosomes are sorted according to non-inferior order, and the chromosomes with small non-inferior order values are ranked at the top of the solution archive.
- (2) Secondly, chromosomes with same non-inferior order are sorted according to crowded degree, and the less crowded chromosomes is closer to the top of the solution archive.

Among the above two rules, the first rule is made to find non-inferior solutions, and the second rule is set down to ensure that the distribution of non-inferior solutions is as dispersive as possible.

Step 3: *Stopping criteria or designated generation.* MOGA uses two different criteria to determine when to stop the solver. MOGA detects if there is no change in the best fitness value for some number of generations (stall generation limit). MOGA also stops when the designated number of generations is reached, by default this number is 300. If either of the two stopping criteria is reached, the algorithm will stop and go to step 5, otherwise the algorithm will go to step 4.

Step 4: *Select crossover and mutate.* MOGA uses selection, crossover, and mutation operators in the solution archives to generate new population. Add the new population into the solution archive, go back to step 2 and repeat the above steps.

The MOGA used in this paper is a variant of Stud GA [38]. First, some best solutions in the archive obtained from step 2 are moved to a stallion archive, and the rest solutions were moved to the other temporary archive. Then, the linear ranking selection is used to select a solution from the stallion archive as a stallion, and select the other solution from the temporary archive. Next, the selected two solutions are particular randomly for crossover and create two offspring by arithmetic crossover operator. After that all the chromosomes resulted from the crossover operation will go through a mutation process and consequently. Finally, the selection, crossover, and mutation operators are used repeatedly to generate new population until the maximum population size is reached.

Step 5: *Termination criteria.* The algorithm will terminate only if the following three conditions are met:

- (1) The Avg TPR value is greater than the designated value.
- (2) The Avg FPR value is smaller than the designated value.
- (3) The number of non-inferior individual solutions is greater than the designated value.

If termination criteria are satisfied, the algorithm will terminate, otherwise the algorithm will go to step 6.

Step 6: *Generate neural network set for training.* Select some best solutions from archive of nondominated solutions, and generate neural networks using the parameters represented by the solutions.

The MBPNN [39] which is used in our approach adds gain factors G to change the steepness of neural network activation function. During the learning process, gain factors change with the change of weights and biases, so as to speed up the convergence. The main modifications of the algorithm are as follows:

- (1) The neural network activation function is still sigmoid function, but the value range is changed to $[-0.5, +0.5]$. It can overcome the problem that the change of weights and biases does not change the calculation when learning zero value samples. The modified sigmoid is as follows:

$$f(x) = -0.5 + \frac{1}{1 + e^{-x}} \quad (2)$$

- (2) Let f be the neural network activation function, add gain factor G_j to the net input I_j , which is computed as the sum of the input features multiplied by the weights, so the output y_j is defined as follows:

$$y_j = f(G_j I_j) \quad (3)$$

- (3) The rule for gain factor update is the same as the rule for weights and biases, and the update value of gain factor ΔG_j can calculate as follows:

$$\Delta G_j = \eta \delta_j I_j / G_j \quad (4)$$

Where η is learning rate, δ_j represents error term at node j .

Step 7: *Train neural network.* Each created neural network from step 6 is trained on training dataset by back propagation algorithm, which reduces the error values and update weights, biases and gain factors, so that the actual output values and desired output values get close enough. The method to update gain factors is expressed as Equations (4).

Step 8: *Generate new population from trained neural network set.* When the learning process for neural networks is completed new chromosomes are constructed based on the parameters of each trained neural network, and then these chromosomes form a new population. Add the new population into the solution archive, go back to step 2 and repeated the above steps.

2.1.2 The testing session

The testing session is depicted in Figure 3. The testing session of the proposed approach integrates the predictions of several MBPNN classifiers to get prediction of the final ensemble. The majority voting method is used to determine which attack type a test sample is finally classified into. The final attack type of a test sample is the one which gets the most votes. The detail implementation of the testing session is as follows:

- Step 1: *Generate neural network set for testing.* Select some non-inferior solutions from archive of nondominated solutions obtained by genetic algorithm, and generate neural networks using the parameters represented by the solutions. The users can select several different solutions for combined prediction according to their own preferences. For example, choose a solution subset with large TPR values, or a solution subset with small FPR values, or a combined solution subset with large TPR values and small FPR values.
- Step 2: *Neural network predicting.* Each created neural network from step 1 is used to classify samples on test dataset by forward propagation algorithm of MBPNN, which is done and the neural network gives out prediction result.
- Step 3: *Combine prediction results of different neural network.* The prediction results from different MBPNN are combined by the majority voting method to give the final output of the ensemble. For each individual sample, the prediction result of each MBPNN will get one vote, and the final attack type of this sample is the one which gets the largest number of votes.

2.1.3 The combined classification method

In the testing session some non-inferior solutions are selected from archive of non-dominated solutions obtained in the training session, and generate a MBPNN set for predicting. Therefore, how to choose these non-inferior solutions is critical to the performance of combined prediction.

A genetic algorithm is used to find the optimal combination solution set. Each chromosome represents one combination solution set, the structure of the chromosomes is illustrated in Figure 4. Each gene of the chromosomes corresponds to a solution in the solution archive. When the gene value is 1, it means that the corresponding solution is selected into the combination solution set, and when it is 0, this corresponding solution is not selected. Since the number of solutions in the solution archive is too large, we choose a subset of the archive to form chromosomes. The Avg TPR segment in chromosomes represents some solutions with maximum Avg TPR values selected from the solution archive. The AVG_FPR segment represents some solutions with minimum AVG_FPR values, and the Accuracy segment represents some solutions with maximum Accuracy values. The GA takes Avg TPR, Avg FPR and Accuracy of combination solution set on training dataset as the optimization objectives. To simplify the calculation, the three objectives are combined into a single objective using a linear weighting method, as shown in the following formula.

$$\max F(l) = \sum_{k=1}^3 a_k * f_k(I) \quad (5)$$

Where $f_k(I)$ represents the k -th objective, which is one of the three objectives, namely Avg TPR, Avg FPR or Accuracy, a_k is the weight of the k -th objective.

2.2 The Intrusion Detection dataset

The performance of the proposed approach is measured based on that KDD cup 1999 [40] dataset which is the most widely used for validation of an IDS. Each record of the KDD dataset contains 41 feature attributes and 1 label attribute. The dataset contains five major types: Normal, Probe, Denial of Service (DoS), User-to-Root (U2R) and Remote-to-Local (R2L) attacks. The last four are attack types which can be subdivided into 39 different attack types. The dataset is very large, including 5 million training records and 2 million test records, so it's practically very difficult to use the whole dataset. In this study, we firstly remove records which have the same value for all features, then randomly select different records to form subsets which contain different proportions of normal and attack instances. The selected subsets used in our experiments is as depicted in Table 1. Validation set is created by extracting 10% of the records (14,558) from the train set. The test set has the same number of records as validation set, and has never been exposed to training session.

Data Transformation. We use one-hot encoding scheme to transform the three categorical features: protocol, service and state. A dimension is added for each new

Type	Train set	Test set	Validation set	Subtotal
Normal	79049	8783	8783	87832
Probe	1918	213	213	2131
DoS	49115	5457	5457	54572
U2R	47	5	5	52
R2L	899	100	100	999
Total	131028	14558	14558	145586

Table 1 Total instances of subsets of KDD cup 1999 dataset.

feature value found in the train set. The added dimensions indicate the presence or absence of certain category, represented by binary [0, 1]. Following this scheme, the resulting datasets are extended into 123-dimensional features.

Data Normalization. We use min-max normalization to scale feature values, since the range of numerical features differ considerably from one another. We scale feature values linearly into the range of [-0.5, 0.5] by using the following equation.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} - 0.5 \quad (6)$$

Where x' represents the normalized data, x represents the raw data, $\min(x)$ finds the minimum data value in the current feature values, and $\max(x)$ searches for the maximum data value for the current feature.

2.3 Experimental setup

To evaluate the proposed approach, an experimental application is implemented in C#. MBPNN is used as a basic classifier. The performance of the proposed technique is evaluated based on benchmark dataset of the KDD cup 1999 dataset. Avg TPR, Avg FPR and MSE are used as objective fitness functions of MOGA. Majority voting method is used to integrate the predictions of several basic classifiers to get combined prediction for the final ensemble. The results of experiments are computed on a Windows PC with Core i7-8550U, 1.80 GHz CPU and 16 GB RAM.

2.3.1 Performance Metrics

In order to evaluate the performance of the proposed IDS, we use the following widely known metrics: accuracy, true positive rate, false positive rate as defined as follows:

- 1) True Positive (TP) is the number of attack records classified correctly;
- 2) True Negative (TN) is the number of normal records classified correctly;
- 3) False Positive (FP) is number of normal records classified incorrectly;
- 4) False Negative (FN) is the number of attack records classified incorrectly.

True positive rate (TPR), also known as detection rate, recall or sensitivity, is the proportion of positive cases that are correctly identified and is calculated as follows:

$$TPR = \frac{TP}{TP + FN} \quad (7)$$

False positive rate (FPR), also known as false alarm rate (FAR), is the proportion of positive cases that are incorrectly identified and is calculated as follows:

$$FPR = \frac{FP}{FP + TN} \quad (8)$$

Accuracy is the proportion of the total number of predictions that is correct and is calculated as follows:

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \quad (9)$$

The closer the value of these metrics (except for FPR) to one is, the better the network topology is. The values of FPR should be close to zero for better topologies.

2.3.2 Design of Experiments

The proposed approach involves two algorithms: MOGA and MBPNN. The implementation of MOGA takes some parameters as depicted in table 2. Designated generation is used to set the maximum number of cycles for genetic algorithm. In order to find more solutions, the population size maintains a large value. Stallion population is created to keep few optimal solutions. Trained population size is the number of selected chromosomes which is used to created MBPNN for phase 2. The probability of crossover and mutation change with the quality of the obtained solutions, and bad solutions will increase the probability, but cannot exceed the set maximum value.

Parameter	Values
Designated generation	400
Population size	300
Stallion population size	10
Trained population size	20
Maximum value of crossover probability	0.4
Maximum value of mutation probability	0.1

Table 2 Configuration of MOGA.

The parameters of MBPNN is depicted in table 3. Designated training epochs number is used to set the maximum number of cycles. The number of input nodes is the number of cyber intrusion feature attributes after one-hot encoding. The number of hidden nodes keep small for less computation. The number of output nodes corresponds to five types of cyber intrusion. The learning rate increase when the network error is large, otherwise it will decrease. The learning rate cannot exceed the set maximum value. Maximum and minimum initial value is used to set the range of the initial parameter values.

Parameter	Values
designated training epochs number	100
Input nodes number	123
Hidden nodes number	3
Output nodes number	5
Maximum value of learning rate	0.5
Maximum initial value of neural network parameters	1000
Minimum initial value of neural network parameters	-1000

Table 3 Configuration of MBPNN.

The parameters of GA for combined prediction is depicted in table 4. Number of MBPNN means how many solutions are selected from the solution archive to participate in the optimization calculation of the genetic algorithm. Designated generation is set to 300. The population size is set to 30, and the number of elite

Figure 5 Training performance of the proposed approach

solutions is 3. The maximum value of crossover probability and mutation probability are set to 0.4 and 0.1 respectively.

Parameter	Values
Number of MBPNN	45
Designated generation	300
Population size	30
Elite population size	3
Maximum value of crossover probability	0.4
Maximum value of mutation probability	0.1

Table 4 Configuration of GA.

3 RESULTS AND DISCUSSION

Here, the two-phase cycle training algorithm is done to optimize parameter values for MBPNN. After that we select some non-inferior solutions obtained by the training algorithm to create MBPNN classifiers for the final ensemble. Finally, each created MBPNN is used to classify samples on test dataset, and the prediction results are combined by the majority voting method to give the final output of the ensemble.

The proposed approach is applied to KDD cup 1999 dataset that produces a set of non-inferior MBPNN based ensemble solutions. The performance of ensemble solutions for training data is depicted in Figure 5, and for test data is shown in Figure 6. A very clear Pareto front can be seen in Figure 5 and Figure 6, which exhibits the excellent optimization performance of the algorithm.

It is worth noting that although some individual non-inferior solutions with high DR and low FAR have good performance, the combination of these solutions can still significantly improve their performance. We choose one solution with largest Avg TPR value, one solution with smallest Avg FPR value, and one solution with smallest MSE value to carry out prediction respectively. At the same time, an experiment called Combined was performed, which classifies by combining some solutions selected by the genetic algorithm.

The detection rate on test dataset is shown in table 5. In the tables, for each major type, the highest DR and lowest FAR will be emphasized in bold. The shaded

cells represent the methods that have good trade off of DR and FAR. Based on DR and FAR, best performing classifiers are selected for local experts as follow: Avg TPR for Probe Detector, and Combined for Normal, DoS, U2R, and R2L Detectors. Obviously, the combined classification method performs better than the individual classification. table 6 compares the overall accuracy, detection rate and false alarm rate, and it also proves that the combined classification method is a feasible classification method with better performance for intrusion detection.

Attack type		Avg TPR	Avg FPR	MSE	Combined
Normal	DR	97.92	99.18	99.26	99.28
	FAR	2.68	1.7	1.97	1.7
DoS	DR	95.6	98.41	98.26	98.36
	FAR	0.11	0.18	0.19	0.19
Probe	DR	97.65	92.96	95.31	96.71
	FAR	0.82	0.35	0.26	0.23
U2R	DR	100	100	80	100
	FAR	0.66	0.07	0.07	0.09
R2L	DR	83	88	84	86
	FAR	0.46	0.08	0.04	0.03

Table 5 Comparison of detection rate.

Attack type	Avg TPR	Avg FPR	MSE	Combined
Accuracy	96.94	98.72	98.74	98.81
DR	97.27	98.3	98.02	98.23
FAR	3.26	1	0.8	0.8

Table 6 Comparison of overall accuracy, Detection rate, and false alarm rate.

More experiments are done to validate the performance of the proposed approach. One of experimental results of TPC-MOGA-MBPNN are compared with the results of some well-known models reported in the literatures as shown in table 7. It is evident that the proposed TPC-MOGA-MBPNN outperforms the cited approaches in case of consistently good overall accuracy and DR.

Method	Normal	DoS	Probe	U2R	R2L	Accuracy	DR	FAR
Multi-level SVM-ELM[41]	98.13	99.54	87.22	21.93	31.39	95.75	95.17	1.87
NFC[36]	98.2	99.5	84.1	14.1	31.5	N/A	95.2	1.9
Genetical algorithm[41]	69.5	99.4	71.1	18.9	5.4	90	94.95	30.46
SVM+BIRCH clustering[41]	99.3	99.5	97.5	19.7	28.8	95.7	N/A	0.7
MOGFIDS[41]	98.36	97.2	88.6	15.79	11.01	93.2	91.96	1.6
Association rules[41]	99.47	96.6	74.8	3.8	1.21	92.4	N/A	0.53
Multiclass SVM[41]	99.6	96.8	75	5.3	4.2	92.46	90.74	0.43
Winning the KDD99[41]	99.5	97.1	83.3	13.2	8.4	93.3	91.81	0.55
t-SNE SVM[42]	99.85	99.62	98.89	86.8	96.51	N/A	N/A	N/A
Hybrid classifier[43]	96.8	98.66	93.4	46.97	71.43	96.77	96.77	3
DT-SVM[44]	99.7	99.92	98.57	48	37.8	N/A	N/A	N/A
GA FPM[45]	N/A	98.85	98.05	78.7	98.7	N/A	N/A	N/A
Proposed approach(Combined)	99.28	98.36	96.71	100	86	98.81	98.23	0.8

Table 7 Comparison of proposed model with other methods by detection rate, accuracy, and false alarm rate.

4 Conclusion

In this paper, a novel TPC-MOGA-MBPNN algorithm based on MOGA and MBPNN is proposed for effective intrusion detection. The proposed approach is capable of producing a pool of non-inferior individual solutions which exhibit classification trade-offs for the user. By using certain heuristics or prior domain knowledge, a user can select an ideal solution or combined solution as per application specific requirements. The proposed approach attempts to tackle the issues of low DR, high FPR and lack of classification trade-offs in the field of intrusion detection. The proposed approach consists of encoding of chromosomes that provides optimized parameter values of MBPNN. MOGA is employed to build multi-objective optimization model that generates Pareto optimal solutions with simultaneous consideration of Avg TPR, Avg FPR and MSE in the dataset. A two-phase cycle training algorithm based approach can rapidly generate numerous non-inferior solutions. In the first phase, a MOGA tries to find the Pareto optimal parameter set for the neural networks. In the next phase some selected MBPNN based on chromosomes obtained by MOGA are trained to find more optimal parameter set locally. The nondominated parameter set obtained in the second phase is used as the input of the first stage, and the training of two-phase algorithm is repeated until the termination criteria are reached.

KDD cup 1999 dataset for intrusion detection are used to demonstrate and validate the performance of the proposed approach. The proposed approach exhibits the excellent optimization performance of the algorithm and a very clear Pareto front has been obtained. The optimized set of MBPNN exhibits the classification tradeoffs for the users. The user may select an ideal solution as per application specific requirements. We also demonstrate that combining a few MBPNN to classify is a feasible classification method and has better performance than using an individual MBPNN to classify. A genetic algorithm is used to find the optimal MBPNN combination, and can discover an optimized set of MBPNN with good accuracy and detection rate from benchmark datasets. The result shows that the proposed approach could reach an accuracy of 98.81% and a detection rate of 98.23%, which outperform most systems of previous works found in the literature. The result of this work has also provided an alternative in the issue of selecting an optimal solution among the non-dominated Pareto-optimal solutions.

The major issue in the proposed approach is that MOGA takes long time to compute fitness functions in various generations. It may be overcome by computing the function values in parallel, or limiting the population size. The proposed approach only uses a small subset of benchmark dataset for validation, and its applicability can be validated by more experiments on real cyber traffic in the field of intrusion detection.

5 Abbreviations

GA:genetic algorithm;MOGA:multi-objective genetic algorithm;MBPNN:modified back propagation neural network;Avg FPR:average false positive rate;MSE:mean squared error;Avg TPR:average true positive rate;NIDS:Network-based Intrusion Detection System;ANNs:Artificial Neural Networks;ANN:Artificial Neural Network;DDoS/DoS:Distributed Denial of Service;RNN-IDS:intrusion detection using recurrent neural networks;DCNN:deep convolutional neural network;IDS:intrusion detection system;ENN:evolutionary neural network;EA:evolutionary algorithm;ANNIDS: Artificial Neural Network based intrusion detection system;MLP:Multilayer Perceptron;DIS:direct acyclic graph information solicitation;MOO:multi-objective optimization;NSGA II:non-dominated sorting genetic algorithm;AMGA2:Micro Genetic Algorithm2;U2R:User-to-Root;R2L:Remote-to-Local;TP:True Positive;TN:True Negative;FP:False Positive;FN:False Negative;TPR:True positive rate;FPR:False positive rate;FAR:false alarm rate.

6 Declarations

6.1 Availability of data and materials

The KDD cup 1999 dataset is used to demonstrate and validate the performance of the proposed approach for intrusion detection. The dataset can be accessed from the website: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>

6.2 Competing interests

The authors declare that there are no conflicts of interest regarding the publication of this paper.

6.3 Acknowledgements

This work was supported by National Key Research and Development Program of China[Grant no. 2018YFC1405700], Industry University Research Cooperation Project of Jiangsu Province [Grant no. BY2019005].

6.4 Authors' contributions

Yiguang Gong, Yunping Liu, Chuanyang Yin conceived and designed the review. Yiguang Gong wrote the paper. All authors reviewed and edited the manuscript. All authors read and approved the final manuscript.

References

1. Weiyi Zhong, Xiaochun Yin, Xuyun Zhang, Shancang Li, Wanchu Dou, Ruili Wang, and Lianying Qi. Multi-dimensional quality-driven service recommendation with privacy-preservation in mobile edge environment. *Computer Communications*, 157, 04 2020.
2. X. Xu, X. Zhang, X. Liu, J. Jiang, L. Qi, and M. Z. A. Bhuiyan. Adaptive computation offloading with edge for 5g-envisioned internet of connected vehicles. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10, 2020.

3. L. Qi, C. Hu, X. Zhang, M. R. Khosravi, S. Sharma, S. Pang, and T. Wang. Privacy-aware data fusion and prediction with spatial-temporal context for smart city industrial environment. *IEEE Transactions on Industrial Informatics*, pages 1–1, 2020.
4. L. Qi, X. Wang, X. Xu, W. Dou, and S. Li. Privacy-aware cross-platform service recommendation based on enhanced locality-sensitive hashing. *IEEE Transactions on Network Science and Engineering*, pages 1–1, 2020.
5. L. Wang, X. Zhang, T. Wang, S. Wan, G. Srivastava, S. Pang, and L. Qi. Diversified and scalable service recommendation with accuracy guarantee. *IEEE Transactions on Computational Social Systems*, pages 1–12, 2020.
6. Lina Wang, Xuyun Zhang, Ruili Wang, Chao Yan, Huaizhen Kou, and Lianyong Qi. Diversified service recommendation with high accuracy and efficiency. *Knowledge-Based Systems*, 204:106196, 07 2020.
7. R Heady, G Luger, A Maccabe, and M Servilla. The architecture of a network level intrusion detection system. 8 1990.
8. Melissa Michael. The state of cyber security 2017, 2017.
9. Akashdeep, Ishfaq Manzoor, and Neeraj Kumar. A feature reduced intrusion detection system using ann classifier. *Expert Systems with Applications*, 88:249 – 257, 2017.
10. Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. *Computers & Security*, 77:304 – 314, 2018.
11. L. Li, Y. Yu, S. Bai, Y. Hou, and X. Chen. An effective two-step intrusion detection approach based on binary classification and k -nn. 6:12060–12073, 2018.
12. Nabila Farnaaz and M.A. Jabbar. Random forest modeling for network intrusion detection system. *Procedia Computer Science*, 89:213 – 217, 2016.
13. R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman. Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7:41525–41550, 2019.
14. Alma Cemerlic, Li Yang, and Joseph M Kizza. Network intrusion detection based on bayesian networks. In *SEKE*, pages 791–794, 2008.
15. Z. Cataltepe, U. Ekmekci, T. Cataltepe, and I. Kelebek. Online feature selected semi-supervised decision trees for network intrusion detection. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 1085–1088, 2016.
16. Elike Hodo, Xavier Bellekens, Andrew Hamilton, Pierre-Louis Dubouilh, Ephraim Iorkyase, Christos Tachtatzis, and Robert Atkinson. Threat analysis of iot networks using artificial neural network intrusion detection system. pages 1–6, 2016.
17. C. Yin, Y. Zhu, J. Fei, and X. He. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, 5:21954–21961, 2017.
18. S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han. Enhanced network anomaly detection based on deep neural networks. *IEEE Access*, 6:48231–48246, 2018.
19. Ilyas Benmessahel, Kun Xie, and Mouna Chellal. A new evolutionary neural networks based on intrusion detection systems using multiverse optimization. *Applied Intelligence*, 48(8):2315–2327, 2018.
20. A Arul Anitha and L Arockiam. Annids: Artificial neural network based intrusion detection system for internet of things.
21. Zichao Sun and Peilin Lyu. Network attack detection based on neural network lstm.
22. Alex Shenfield, David Day, and Aladdin Ayesh. Intelligent intrusion detection systems using artificial neural networks. *ICT Express*, 4(2):95–99, 2018.
23. Nitin Talhar. Effective denial of service attack detection using artificial neural network for wired lan. In *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pages 229–234. IEEE, 2016.
24. Mukta Paliwal and Usha A Kumar. Neural networks and statistical techniques: A review of applications. *Expert systems with applications*, 36:2–17, 2009.
25. Fadzil Ahmad, Nor Ashidi Mat Isa, Zakaria Hussain, and Siti Noraini Sulaiman. A genetic algorithm-based multi-objective optimization of an artificial neural network classifier for breast cancer diagnosis. *Neural Computing and Applications*, 23(5):1427–1435, 2013.
26. Xin-Yan Cao, Hong-Li Yu, and Ying-Yong Zou. Character recognition based on genetic algorithm and neural network. In *Proceedings of the 2012 International Conference on Information Technology and Software Engineering*, pages 915–923. Springer, 2013.
27. Gulshan Kumar and Krishan Kumar. A multi-objective genetic algorithm based approach for effective intrusion detection using neural networks. In *Intelligent methods for cyber warfare*, pages 173–200. Springer, 2015.
28. Hussein A Abbass. Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization. In *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.*, volume 3, pages 2074–2080. IEEE, 2003.
29. Carlos M Fonseca, Peter J Fleming, et al. Genetic algorithms for multiobjective optimization: Formulation and discussion and generalization. In *Icga*, volume 93, pages 416–423. Citeseer, 1993.
30. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.
31. X. Xu, R. Mo, F. Dai, W. Lin, S. Wan, and W. Dou. Dynamic resource provisioning with fault tolerance for data-intensive meteorological workflows in cloud. *IEEE Transactions on Industrial Informatics*, 16(9):6172–6181, 2020.
32. X. Xu, B. Shen, X. Yin, M. R. Khosravi, H. Wu, L. Qi, and S. Wan. Edge server quantification and placement for offloading social media services in industrial cognitive iot. *IEEE Transactions on Industrial Informatics*, 2020.
33. Salma Elhag, Alberto Fernández, Abdulrahman Altalhi, Saleh Alshomrani, and Francisco Herrera. A multi-objective evolutionary fuzzy system to obtain a broad and accurate set of solutions in intrusion detection systems. *Soft Computing*, 23(4):1321–1336, 2019.
34. Martin Stehlik, Adam Saleh, Andriy Stetsko, and Vashek Matyas. Multi-objective optimization of intrusion detection systems for wireless sensor networks. In *Artificial Life Conference Proceedings 13*, pages 569–576.

- MIT Press, 2013.
35. X. Xu, X. Liu, Z. Xu, F. Dai, X. Zhang, and L. Qi. Trust-oriented iot service placement for smart cities in edge computing. *IEEE Internet of Things Journal*, 7(5):4084–4091, 2020.
 36. Santosh Tiwari, Georges Fadel, and Kalyanmoy Deb. Amga2: improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization. *Engineering Optimization*, 43(4):377–401, 2011.
 37. et al Fei ye, Li Nannan. Multiobjective optimization method based on pareto solution and its application. *Lifting and transportation machinery*, 2006(9):13–15, 2006.
 38. Wael Khatib and Peter J Fleming. The stud ga: a mini revolution? In *International Conference on Parallel Problem Solving from Nature*, pages 683–691. Springer, 1998.
 39. Zhu Jianying. Non-classical mathematical methods for intelligent systems.
 40. KDD Cup. <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>. *The UCI KDD Archive*, 1999.
 41. Wathiq Laftah Al-Yaseen, Zulaiha Ali Othman, and Mohd Zakree Ahmad Nazri. Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system. *Expert Systems with Applications*, 67:296–303, 2017.
 42. Yasir Hamid and M Sugumaran. A t-sne based non linear dimension reduction for network intrusion detection. *International Journal of Information Technology*, 12(1):125–134, 2020.
 43. Cheng Xiang, Png Chin Yong, and Lim Swee Meng. Design of multiple-level hybrid classifier for intrusion detection system using bayesian clustering and decision trees. *Pattern Recognition Letters*, 29(7):918–924, 2008.
 44. Sandhya Peddabachigari, Ajith Abraham, Crina Grosan, and Johnson Thomas. Modeling intrusion detection system using hybrid intelligent systems. *Journal of network and computer applications*, 30(1):114–132, 2007.
 45. Priya Uttam Kadam and Manjusha Deshmukh. Real-time intrusion detection with genetic, fuzzy, pattern matching algorithm. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 753–758. IEEE, 2016.

7 Figure Title and Legend Section

Figure 1. Training session of the proposed methodology.

Figure 2. The chromosome represents the parameters of neural network.

Figure 3. Test session of the proposed methodology.

Figure 4. The chromosome represents which solutions are selected in the combination solution set.

Figure 5. Training performance of the proposed approach.

Figure 6. Test performance of the proposed approach.

Figures

Figure 1

Training session of the proposed methodology

Figure 2

The chromosome represents the parameters of neural network

Figure 3

Test session of the proposed methodology

Figure 4

The chromosome represents which solutions are selected in the combination solution set

Figure 5

Training performance of the proposed approach

Figure 6

Test performance of the proposed approach