International Journal of Networking and Computing – www.ijnc.org, ISSN 2185-2847
Volume 10, Number 2, pages 213–226, July 2020

A Sequential Detection Method for Intrusion Detection System Based on Artificial Neural Networks

Zhao Hao
Department of Informatics, Kyushu University
744 Motooka Nishi-ku, Fukuoka, 819-0395, Japan
zhaohaoim@gmail.com

Yaokai Feng*
Faculty of Information Science and Electrical Engineering, Kyushu University
744 Motooka Nishi-ku, Fukuoka, 819-0395, Japan
(*corresponding author) fengyk@ait.kyushu-u.ac.jp

Hiroshi Koide
Faculty of Cybersecurity Center, Kyushu University
744 Motooka Nishi-ku, Fukuoka, 819-0395, Japan
koide@cc.kyushu-u.ac.jp

Kouichi Sakurai
Faculty of Information Science and Electrical Engineering, Kyushu University
744 Motooka Nishi-ku, Fukuoka, 819-0395, Japan
sakurai@inf.kyushu-u.ac.jp

Received: February 15, 2020
Revised: May 2, 2020
Accepted: June 9, 2020
Communicated by Yasuyuki Nogami

Abstract

With rapidly increasing cyber attacks, network security has become an important issue. To protect ourselves against cyber attacks, the Intrusion Detection System (IDS) has been introduced. In such systems, different kinds of machine learning algorithms play a more and more important role, such as support vector machine(SVM), artificial neural network(ANN), etc. False positive rate and false negative rate, in addition to accuracy, are widely used for the evaluation of IDSs. These indices, however, are often related to each other, which makes it is difficult for us to improve all the indices at the same time. For example, when we try to make the false negative rate decrease to prevent from missing attacks, more normal communications tend to be classified into attacks and the false positive rate may increase, and vice versa. In this study, we propose an ANN based sequential classifier method to mitigate this problem. We design each subclassifier with a low false positive rate, which may lead to high false negative rate. To decrease the false negative rate, the reported negative instances from the former subclassifier are sent to the next one to further check (reclassification). In this way, it can be expected that the false negative rate can also reach an acceptable level. The results of our experiment shows that our proposed method can bring lower false negative rate and higher accuracy, in the

mean time the false positive rate is kept at an acceptable level. We also investigated the effect of the number of subclassifiers on detection performance and found that the detection system performed best when using four subclassifiers.

Keywords: cyber security, intrusion detection, sequential detection, machine learning, false negative rate, false positive rate

1 Introduction

The popularity of computer networks has made us rely heavily on the Internet in almost all fields, including daily life, education, research, industry, governments, etc. It is not too much to say the Internet has become an indispensable part of us. At the same time, cyber attacks have brought us great troubles and losses and they have been a difficult problem for us to deal with. Tens of Thousands of new malware programs created each day. The number of new malicious files processed by Kaspersky Lab's in-lab detection technologies was about 70,000 a day in 2011. This figure, however, reached 360,000 a day in 2017, which is 11.5% more than the previous year and has grown five-fold from 2011 [1]. The purpose of cyber attacks has evolved from entertainment complacency and self-exhibition decades ago to the seizure of economic and political interests, the destruction of economic and political competitors, and the promotion of their own political and social claims, and so on. Darknet has also formed a huge scale, in which a lot of attack softwares/tools are being sold at a very low price, e.g. a few dollars and personal information (credit card information, email addresses, etc) stolen by cyber attacks can also be sold there.

Many attacks caused huge damages. For example, the billing service vendor AMCA (American Medical Collection Agency) disclosed in April, 2019, that its records were exposed to hackers between August 1, 2018 and March 30, 2019. Totally 25 million hosts were affected and up to 12 million records being compromised in the company Quest Diagnostics only. As a result, the parent company of AMCA filed bankruptcy, and others involved are facing several lawsuits and investigations [2]. For one more recent example, it was made public at May, 2019 that in the insurance company, First American, nearly 900 million records were compromised, which may be the second-biggest data breach in history behind Yahoo!'s hack in 2013 that impacted 3 billion accounts [2]. The ongoing data leak at First American reportedly involved mortgage documents dating back to 2003 and included personal identifying information, bank account numbers, driver's licenses, social security numbers, tax records and other stolen information.

Since governments, many enterprises and research institutes have been investing more and more money and manpower every year trying to deal with the problem of cyber attacks more effectively, why the frequency and loss of cyber attacks are still greatly increasing? Driven by the huge profit, the number of attackers is increasing and the attack technology is becoming more and more sophisticated. Besides, the current attack detection technologies still have fundamental deficiencies.

To protect against such attacks, intrusion detection systems (IDS) has been introduced [9]. An IDS works as a monitor and classifier for cyber attacks and normal networks traffics, for which statistical and machine learning methods of many types have been applied. Statistical methods always assume that normal or abnormal communication follows a certain distribution, but it is often not completely consistent with the actual situations and the determination of the parameters are not so easy. For classifiers by machine learning models, such as decision tree [32], artificial neural networks (ANN) [13], support vector machine(SVM) [4] and clustering [6] models, false positive rate and false negative rate, in addition to detection accuracy, also affect the performance greatly. However, whether the statistical methods or the methods based on machine learning, it is often nearly impossible for different indicators of the performance of detection systems to reach a satisfactory level at the same time because these indices are often related to each other. For example, while we try to decrease the false negative rate to prevent from missing attacks, more normal communications tend to be classified into attacks and the false positive rate may increase, and vice versa. Thus, the balance between different performance indicators is a very difficult and important thing.

In this study, we aim to mitigate this problem by proposing an ANN based sequential classifier method. That is, the detection system consists of multiple subclassifiers. Each classifier is trained

to have a low false positive rate, which may lead to a high false negative rate. Then, the reported negative instances from the former classifier are sent to the next one for reclassification. In this way, it is expected that both of the two performance indicators can reach an acceptable level. The main contributions of this paper are as follows.

1. A sequential system for detecting cyber attacks was proposed and implemented;
2. The detection performance of the proposed system was verified using a public dataset;
3. We investigated the effect of the number of subclassifiers on detection performance and found that four subclassifiers gave the best system performance.

The rest of the paper is structured as follows. In Section 2 we introduce the intrusion detection systems and artificial neural networks. In Section 3, we introduce the related existing researches of machine learning methods. Our proposed method of ANN based sequential classifier and the procedure are presented in Section 4, and the details of experiment results and evaluation are presented in Section 5. Finally in Section 6 we make the conclusion and talk about the future works.

2 Background

2.1 Intrusion Detection System

An intrusion detection system(IDS) works as a monitor of malicious activities for personal computers or the whole network. Based on the assumption that the behavior of intrusion events are different from that from normal ones [36], when an event happens in the system, the IDS analyses and compares it with the normal events. If the pattern of the new event is different from the normal ones, it will be classified as malicious activities and reported.

According to how it works, intrusion detection systems can be mainly divided into two types, signature-based IDS and anomaly-based IDS [20].

Signature-based IDS, or knowledge-based IDS, detects malicious activities by looking for specific patterns or known intrusion methods. The process of detecting for signature-based methods is to compare patterns against captured events for recognizing possible intrusions. The expert system, which has been developed from mid-1960s [11], can be one example of such systems. It works by classifying the data according to a set of rules. Although it is a simple and effective method to detect known attacks, it becomes ineffective to detect unknown attacks or the variants of known attacks. Also, maintaining the knowledge needed can be time consuming and it is hard to keep the signatures or patterns up to date.

Anomaly-based IDS, on the other hand, is based on the deviation of the system events to a known behavior and therefore capable of detecting unknown attacks. When anomaly-based methods are applied, a normal model are established according to the normal pattern of the system, and new events are compared against this model for the purpose of intrusion detection [19] .

There are many techniques have been used in anomaly-based systems for intrusion detection. Mainly they can be categorized into statistical techniques and machine learning techniques [10].

In statistical techniques, the network traffic data is captured and a profile representing its behavior is created, which is based on metrics such as the traffic rate, the number of packets from different protocols, the distribution of all IP addresses, and so on. According to how many metrics are used in this process, both univariate methods [8] and multivariate methods [38] have been proposed and further researched. Also, studies of time series models [35] were developed, in which the arriving order of events and the intervals between are used as metrics and traffic instances arriving in a time of low probability will be considered as malicious.

Many machine learning methods are applied for the purpose of obtaining a normal profile of object systems, as the statistical methods intend to, and therefore inspecting malicious activities, such as Bayesian networks, Markov models, Neural Networks, and so on. A further investigation of machine learning based methods in the field of intrusion detection will be made in Section 3.

2.2 Artificial Neural Networks

Inspired by the biological structure of human brains, artificial neural networks (ANN) was firstly proposed in 1940s [27], which shows good performance in detecting cyber attacks [33]. The training of multiple layer networks was achieved as the proposal of the backpropagation algorithm [14].

A typical artificial neural network is consisted of units called perceptrons, several perceptrons or nodes make up a layer and an artificial neural network is make up of several such layers, including the input layer, hidden layers and the output layer. The input layer is used for data input and the number of nodes of input layer corresponds to the number of features from data. In the output layer, the final result of the calculation is shown. In the case of binary classifications, there are two nodes in the output layer, corresponding to two types of results such as cyber attack or normal traffic. The structure of hidden layers, like the number of layers and the node number of each layer, is adjusted according to the performance of the model.

During the training procedure, training dataset is used for weight updating and the most commonly used method is called backpropagation. When each of data from the training dataset is read by the input layer and the training phase is finished, the updated weights are saved and used for testing or classification. During the classification, each of data from the testing dataset are read by the input layer, and according to the calculation, the each node of the output layer gives a result of probability, the highest of which corresponds to the classification result where one of the classes are predicted to be the right one.

3 Related Works

As introduced in Section 2, for the purpose of constructing anomaly based intrusion detection systems with the ability of finding new types of attacks, different methods have been proposed, of which machine learning based approaches have been widely applied especially in recent years.

3.1 Various Machine Learning Models

Support vector machine, or SVM, was first proposed by Vapnik [7] and has been a very popular machine learning method. When being applied, the input vector was transformed into a higher dimensional space and the optimal hyper-plane will be searched which can classify the data best. A hybrid method of support vector machine and genetic algorithm for intrusion detection is proposed by Aslahi-Shahri, et al. [5] and an accuracy of 97.3% was achieved on the KDD99 dataset. Horng S J, et al. [12] proposed a novel intrusion detection system based on hierarchical clustering and support vector machines and got the accuracy of 95.7% on KDD99 dataset. To solve the low detection rate problem of rare attacks in IDS, Pozi M S M, et al. [24] proposed a method based on support vector machine and genetic programming and achieved the accuracy of 89.28% for U2R attacks and 90.72% for R2L attacks on the NSL-KDD dataset. The false negative rate of 9% for DoS, 16% for probe, 2% for R2L, 11% for U2R and the false positive rate of 14% for DoS, 15% for probe, 25% for R2L, 12% for U2R were achieved.

A decision tree(DT) is a powerful and popular tool for classification problems and it uses tree-like graph to classify a sample through a sequence of decisions. The classification procedure starts from the root node to one of the end leaf nodes and each branch represents an outcome of the classification. Rahman C M, et al. [25] proposed a new learning algorithm for anomaly based network intrusion detection system using decision tree algorithm and got the experimental results on the KDD99 dataset that achieved 98% detection rate. Sahu S, et al. [29] used J48 decision tree algorithm to classify the network packet data from Kyoto2006+ dataset and achieved the accuracy of 97.2%. Jasmin Kevric, et al. [16] proposed a combining binary classifier based on tree algorithms for intrusion detection and the NSL-KDD dataset was used for evaluation. The detection accuracy of 89.24% was achieved using the combination of random tree and NBTree algorithms based on the sum rule scheme, and it outperformed the individual random tree algorithm.

For artificial neural networks(ANN), Gang Wang, et al. [37] proposed a method using neural networks and clustering algorithm, achieving the accuracy of 96.71% on the NSL-KDD dataset

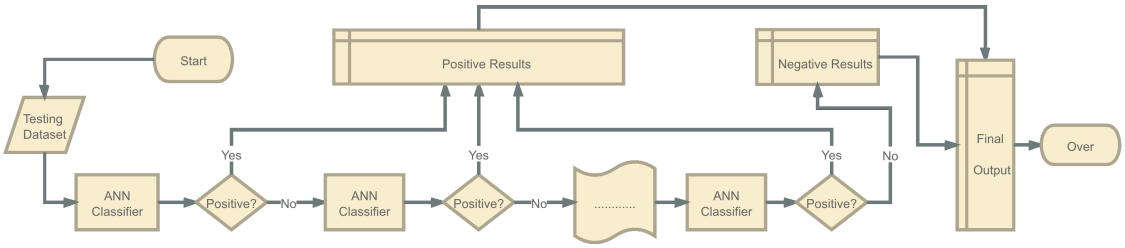

Figure 1: Proposed sequential detection system

and the false positive rate of 1% for normal, 4% for DoS, 20% for probe, 42% R2L, and 23% for U2R respectively. Recurrent neural network based intrusion detection was proposed by Mansour Sheikhan, et al. [30] and KDD dataset is used for performance evaluation, with the result of accuracy of 94.1%. Jihyun Kim, et al. [17] used long short term memory recurrent neural network (LSTM-RNN) as classifier for intrusion detection and the accuracy of 96.93% and the false positive rate of 10% was achieved on the KDD dataset. Deep learning based methods, which originally comes from the advancements of NN algorithms, have also been applied in the field of intrusion detection. Generally, there are three types of deep learning based method: generative, discriminative, and hybrid [18].

3.2 Sequential Methods Towards Performance Improving

Soe Y N, et al. [31] proposed a sequential detection method based on attack specific neural networks for IoT devices and experimented on the N-BaIoT dataset for evaluation. According to this method, up 99% of accuracy was achieved in detecting ten types of attacks by using ten different attack specific neural networks.

Many types of machine learning algorithm based methods have been proposed and the main focus is the detection rate or the accuracy of IDS. However, for classifiers of cyber attacks false positive rate and false negative rate should also be considered.

A sequential classifier combination method is proposed by Phetlasy S, et al. [23] to reduce the false negative rate. In this paper, five different types of classifiers, respectively based on Random Tree, Decision Tree J48, K Nearest Neighbor, Multilayer Perceptron and Naive Bayes classifier are applied. The false negative rate of 9.4%, the false positive rate of 9.92% and the accuracy of 90.37% were achieved on the NSL-KDD dataset.

4 Proposed Method

Our proposed method is shown as Figure 1, five four-layer ANNs are sequentially connected to each other. Mainly there are four steps in how this system works.

1. Firstly, all traffic data is classified by the first artificial neural network, and the result will be separated into positives and negatives, according to each of data is classified as an attack or normal network traffic.

2. Secondly, the negative results, which may contain undetected malicious traffic, will be reclassified by the next subclassifier, while the positive results will be directly sent into the final output.

3. After that, the negative results of second subclassifier will be used as the input data of third subclassifier and the same procedure happens in latter subclassifiers, too.

4. The last subclassifier will finish the classification, and both the positives and negatives will be sent into the final output.

Table 1: Number of normal and abnormal data

Dataset	Normal	Abnormal	Total
KDDTrain+	67342	58627	125969
KDDTest+	9711	12832	22543

Table 2: Attacks in NSL-KDD dataset

Denial of Service	User to Root	Remote to Local	Probing
Back	Perl	FTP write	IP sweep
Ping of death	Buffer overflow	Guess password	NMAP
Neptune	Load module	IMAP	Port sweep
Smurf	Rootkit	Multi HOP	Satan
Land		Phf	
Teardrop		SPY	
		Wareclient	
		Warezmaster	

As a result, the output is consisted of two parts. One is the positive results from all five subclassifiers and the other one is the negative result from the last subclassifier. As for the trade-off between false negative rate and false positive rate, more ANN classifiers can actually be added. To decide how many classifiers to use, two main problems need to be considered.

- The false positive rate must be acceptable.
- As few as possible classifiers should be used.

In our experiment, five classifiers are used at first. As discussed later in Section 5, the case of four ANNs seems to bring the best performance.

5 Experiment

5.1 Dataset

For the purpose of intrusion detection, the KDD99 dataset, which was proposed in 1999, has been widely used [3]. It was built based on the data captured in DARPA'98. As an improved version, NSL-KDD dataset was proposed by Tavallaee, et al. [22], in 2009. In this dataset, four data files are provided for model training and testing: KDDTrain+, KDDTrain+20Percent, KDDTest+ and KDDTest-21.

Among these datasets, KDDTrain+ is an improved version training dataset with a total number of 125,969 records and KDDTrain+20Percent provides a smaller scale of records for the purpose of training faster. KDDTest+ is used as the standard testing dataset, provided with a total amount of 22,543 records. KDDTest-21, on the other hand, is provided as a testing dataset consisted of records harder to classify by different kinds of classifiers. Table 1 shows the number of normal and abnormal data in KDDTrain+ and in KDDTest+.

In NSL-KDD dataset, four main types of attacks are included: Denial of Service (DoS) Attack, User to Root (U2R) Attack, Remote to Local (R2L) Attack and Probing (Probe) Attack, each of which is consisted of a group of similar attacks. Table 2 shows the attacks in NSL-KDD Dataset.

In the perspective of features, there are 41 features, of which 34 are numeric features and seven are categorical features. They are divided into four groups, 10 of basic features, 12 of contents features, nine of time based traffic features and 10 of host based traffic features, which are shown in Table 3.

Table 3: Features from NSL-KDD dataset

Feature	No.
Basic features	1-10
Contents features	11-22
Time based traffic features	23-31
Host based traffic features	32-41

5.2 Procedure

5.2.1 Data pre-processing

During the this procedure, three pre-processing actions were taken.

1. Format transformation In the NSL-KDD dataset, files of txt and arff format are provided. They are transformed into csv format for faster calculation. In the experiment, KDDTrain+ file is read and used for training and for the performance evaluation KDDTest+ file is used. As an overview, the number of normal and abnormal data of the two datasets is shown in Table 1.

2. Data Numeralization Among the 41 features of the dataset, there are three nonnumerical features (protocol_type, service and flag) which need to be transformed into numeric form. In this step, they are mapped into integer values ranging from 0 to N-1, where N stands for the number of such nonnumerical features.

3. Data Normalization Then data normalization is conducted according to the equation below for a higher calculation efficiency. See Equation (1).

$$x_{std} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

5.2.2 Classifier model training

During the training process, the structure of classifiers are to be decided. In this experiment, we apply an input layer of 41 nodes since the are 41 features from the dataset. Also, since the classification is binary for the purpose of detecting whether it is an attack or normal network traffic, an output layer of two nodes are applied and the Softmax function is used as the activation function.

FNR (False Negative Rate), FPR (False Positive Rate) and Accuracy are used for evaluating the system performance, which are calculated as follows.

- FNR=FN/(TP+FN)*100%;
- FPR=FP/(FP+TN)*100%;
- Accuracy=(TP+TN)/(TP+FP+TN+FN)*100%,

where FN is the number of the false negatives; FP is the number of the false positives; TP is the number of the true positives and TN is the number of the true negatives.

In the perspective of hidden layers, we use a two-layer structure with 30 nodes of each layer. For the number of neurons in each hidden layer, it plays an important role in the overall performance of the neural networks. Using too few neurons in the hidden layers will cause underfitting problem, which means no adequate capability of learning and detecting in a complicated dataset. Also, using too many neurons in the hidden layer may result in the overfitting problem, in which case the neural network cannot be trained enough by the limited number of data from the dataset so that all the neurons are trained in the hidden layers. In addition, it may take a large mount of time of training for such neural networks with too many neurons.

To choose an acceptable number of neurons for the classifier, we used the rule of thumb method [28] to decide the size of hidden layer as following.

Table 4: Determining the number of hidden layers

Hidden Layer	Computation capabilities
none	Only capable of representing linear separable functions or decisions
1	Can approximate any function that contains a continuous mapping from one finite space to another
2	Can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy

Table 5: Best cases of different structures

No. of Hidden Layer	FPR	FNR	Accuracy
Single hidden layer	1.46%	45.54%	73.45%
Two hidden layers	1.53%	39.02%	77.13%
Three hidden layers	1.58%	44.75%	73.85%
Four hidden layers	1.80%	42.27%	75.16%
Five hidden layers	2.00%	42.94%	74.69%

1. The number of hidden neurons should be in the range between the size of the input layer and the size of the output layer.

2. The number of hidden neurons should be 2/3 of the input layer size, plus the size of the output layer.

3. The number of hidden neurons should be less than twice the input layer size.

Since there are 41 neurons in the input layer and two neurons in the output layer, we decide the size of hidden layers to 30 according to the rule of thumb method.

Another important thing is to decide how many hidden layers are needed. A neural network with no hidden layer is only capable of representing linear separable functions or decisions and thus not enough for our problem. Although it has been shown that a single layer neural network containing a finite number of neurons can approximate continuous functions despite of the choice of the activation function [15], the universal approximation theorem does not specify how easy it will be for that neural network to actually learn something. On the other hand, neural networks with two hidden layers is capable of representing an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy. Since generally two or fewer layers will often suffice with simple datasets, which in our case has 41 features and around 125,000 records, we consider a structure of two hidden layers suitable theoretically. Table 4 shows the determination of the number of the hidden layers.

In the combination of the theoretical analysis, we experimented and compared the best cases of single subclassifier from one hidden layer to five hidden layers in dealing with our problem and as the results show in Table 5, more than two hidden layers brings no further performance improvement for a single ANN classifier. Also, based on the comparison between single hidden layer and two hidden layers, the latter shows lower false negative rate and higher accuracy and shares a similar level of false positive rate with single hidden layer, therefore we use a two layer structure for ANN classifiers. As for the activation function, the ReLU function is chosen for both two hidden layers.

In conclusion, a 41-30-30-2 structure is used for each subclassifier. As for the optimizer RMSprop is applied and the learning rate is set to 0.001. Each subclassifier is trained separately with all 41 features. During the training procedure, as described in Section 2, the backpropagation algorithm is used for the weight updating of the subclassifiers.

1. One data from KDDTrain+ is used as the input data and the output is calculated using the default weights.

2. The comparison is made between the calculation results and the real label value and error of them are calculated.

3. The error of each layer is calculated from the output layer to the input layer.

4. The weights of every two nodes in connection is calculated using the error and the next data is read.

And the training of one epoch is finished after the whole training data is used. All 15 epochs of training is performed for each subclassifier for a better generalization capability.

5.2.3 Classifier model testing

The results of each subclassifier are collected and the confusion matrixes are made for performance evaluation, according to which the accuracy, false positive rate and false negative rate are calculated and to be compared.

5.3 Results

The experiment environment is MacBook Pro with 2.8GHz Intel Core i7 processor, 16GB 1600MHz DDR3 of memory and 64bit operation system. Python is used for programming and Keras and Tensorflow are used for model training procedure and the testing procedure.

5.3.1 Confusion matrix

The effect of the number of subclassifiers on the detection performance of the whole system is investigated and the test results (confusion matrix) of the different cases from the case of single subclassifier to the case of five subclassifiers are shown as Table 6.

Table 6: Confusion matrix in different cases

Different cases	Confusion Matrix	
Single subclassifier (ANN1)	TP1=7707 FN1=5125	FP1=294 TN1=9417
Two subclassifiers (ANN1, 2)	TP2=541 FN2=4548	FP2=14 TN2=9403
Three subclassifiers (ANN1, 2, 3)	TP3=705 FN3=3879	FP3=186 TN3=9217
Four subclassifiers (ANN1, 2, 3, 4)	TP4=269 FN4=3610	FP4=39 TN4=9178
Five subclassifiers (ANN1, 2, 3, 4, 5)	TP5=15 FN5=3595	FP5=5 TN5=9173

5.3.2 Comparison on FPR, FNR and Accuracy

The negative results, including true negative(TN) and false negative(FN), of the former subclassifier is treated as the input data of the next subclassifier. When applying the proposed method, we record and compare the results of each subclassifier for performance evaluation. The results of FPR, FNR and Accuracy are shown in Table 7.

5.3.3 Comparison on ROC

The performance in perspective of the ROC is also investigated and shown in Figure 2, which shows how much each model is capable of distinguishing between classes and the higher the area under curve (AUC), the better the model is at classifying attack and normal network traffic.

Table 7: FPR, FNR and Accuracy in different cases

Different cases	FPR	FNR	Accuracy
Single subclassifier (ANN1)	3.03%	39.94%	75.96%
Two subclassifiers (ANN1, 2)	3.17%	35.72%	78.30%
Three subclassifiers (ANN1, 2, 3)	5.09%	30.23%	80.60%
Four subclassifiers (ANN1, 2, 3, 4)	5.49%	28.13%	81.62%
Five subclassifiers (ANN1, 2, 3, 4, 5)	5.54%	28.06%	81.67%

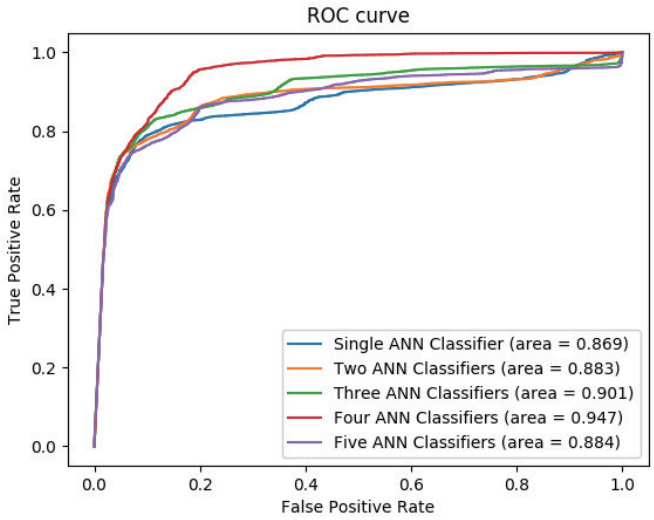

Figure 2: ROC curves of the different cases. "area" here means the area under curve (AUC)

5.3.4 Comparison on elapsed time

Also, we recorded the time consumed in both training and testing procedure, as shown in Table 8.

Table 8: Elapsed time for training and testing

Different cases	Elapsed time (training)	Elapsed time (testing)
Single subclassifier (ANN1)	50.59s	0.34s
Two subclassifiers (ANN1, 2)	106.18s	0.53s
Three subclassifiers (ANN1, 2, 3)	159.65s	0.71s
Four subclassifiers (ANN1, 2, 3, 4)	218.66s	0.89s
Five subclassifiers (ANN1, 2, 3, 4, 5)	280.03s	1.05s

5.4 Observations

From the above experimental result, we can observe that

1. Our sequential method can improve the detection performance clearly. As more subclassifiers are added, false negative rates decrease and the accuracy also grows, with the increasing to some extent of false positive rate.

   (a) The AUC in the ROC increases to 0.947 in the case of four subclassifiers from 0.869 in the case of single subclassifer;

(b) False negative rates are decreased from 39.94% in the case of single subclassifier by 11.81% to 28.13% in the case of four subclassifiers;

(c) False positive rates are changed from 3.03% in the case of single subclassifier by 2.46% to 5.49% in the case of four subclassifers, which is much more less than the decrease of the false and we think is at an acceptable level.;

(d) Also, the accuracy is increased from 75.96% in the case of single subclassifier by 5.66% to 81.62% in the case of four subclassifiers.

2. The case of four subclassifiers leads to the best performance.

   From the ROC shown in Figure 2, we can know that the case of four subclassifiers has the largest area (AUC). From four subclassifiers (ANN1, 2, 3, 4) to five subclassifiers (ANN1, 2, 3, 4, 5), all three metrics show almost no improvement, of which only 0.07% for the false negative rate; 0.05% for the false positive rate and 0.05% for the accuracy is changed.

   As we can see, as more classifiers are added, they contribute fewer and fewer to the performance improvement. Since the main purpose of the latter subclassifiers is to reduce the number of false negatives by classifying the negative results from the previous one, with more of them successfully classified it becomes harder and harder for latter subclassifiers to make a contribution and according to our experiment results, subclassifiers after the fourth makes little contribution to decrease the false negative results. Considering the cost of time and calculation, we think the case of four subclassifiers is the best choice.

3. The time consumed for detection is acceptable in the case of four subclassifiers. As shown in Table 8, the testing time consumed (0.89s) in the case of four subclassifiers is about 2.5 times that (0.34s) in the case of single subclassifier, although this figure becomes about four times for training. Since the time consumed for testing is much more important than that for training, we think the time consumed for testing is acceptable.

6 Conclusion and Future Work

In this paper, based on the fact that, for a single classifier, it is difficult for us to improve the different indices (e.g. FPR and FNR) of detection performance at the same time, we proposed a sequential method consisting of multiple subclassifiers in order to mitigate this problem. In our system, each subclassifier is trained to have lower FPR, which often lead to a higher FNR. The negative instances output from the former subclassifier are sent to the next one for reclassification. As a result, the performance of the whole system has a lower FNR with an acceptable FPR and a higher accuracy than the case of single subclassifiers, which was verified by experiment using a public dataset. The case of four subclassifiers was found to perform best in our experiment.

Obviously, contrary to our proposed method, it is also possible to train each subclassifier having a low FNR, which perhaps makes the FPR higher. After that, all the positive instances reported by the former subclassifier are sent to the next one for further checking. This will be investigated in our future work. Also, we will consider ways of ruling out some negatives at each level to make the additional subclassifiers cheaper to run. Moreover, more datasets will be used for evaluating our proposal.

Acknowledgment

This work was partially supported by JSPS KAKENHI Grant Numbers JP17K00187 and JP18K11295. This work is also partially supported by Strategic International Research Cooperative Program, Japan Science and Technology Agency (JST). We would also like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript.

References

[1] Kaspersky Lab detects 360,000 new malicious files daily: https://www.kaspersky.com/about/press-releases/2017_kaspersky-lab-detects-360000-new-malicious-files-daily (accessed on April 29, 2020).

[2] 5 Biggest Cyberattacks of 2019 and Lessons Learned: https://www.gflesch.com/blog/biggest-cyberattacks-2019 (accessed on April 29, 2020).

[3] Ozgur A and Erdem H. A review of kdd99 dataset usage in intrusion detection and machine learning between 2010 and 2015[j]. *PeerJ Preprints*, 4, 2016.

[4] Shams E A and Rizaner A. A novel support vector machine based intrusion detection system for mobile ad hoc networks. *Wireless Networks*, 2018.

[5] B M Aslahi-Shahri et al. A hybrid method consisting of ga and svm for intrusion detection system. *Neural computing and applications*, 27(6):1669–1676, 2016.

[6] Lin W C and Ke S W. An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-based systems*, 2015.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] Denning D and Neumann P G. Requirements and model for ides-a real-time intrusion-detection expert system[m]. *SRI International*, 1985.

[9] D E Denning. An intruison-detection model. *IEEE Symposium on Security and Privacy*, 1986.

[10] Pedro Garcia-Teodoro et al. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28:1–2, 2009.

[11] Liao S H. Expert system methodologies and applications—a decade review from 1995 to 2004[j]. *Expert systems with applications*, 28(1):93–103, 2005.

[12] Chen Y H Horng S J, Su M Y et al. A novel intrusion detection system based on hierarchical clustering and support vector machines[j]. *Expert systems with Applications*, 38(1):306–313, 2011.

[13] Kang M J and Kang J W. Intrusion detection system using deep neural network for in-vehicle network security. *PloS one*, 2016.

[14] Werbos P J. The roots of backpropagation: from ordered derivatives to neural networks and political forecasting[m]. *John Wiley & Sons*, 1994.

[15] Hornik K. Approximation capabilities of multilayer feedforward networks[j]. *Neural networks*, 4(2):251–257, 1991.

[16] Jukic S Kevric J and Subasi A. An effective combining classifier approach using tree algorithms for network intrusion detection[j]. *Neural Computing and Applications*, 28(1):1051–1058, 2017.

[17] Thu H L T Kim J, Kim J et al. Long short term memory recurrent neural network classifier for intrusion detection. *Platform Technology and Service (PlatCon), International Conference on. IEEE*, 2016.

[18] Aminanto M E Kim K and Tanuwidjaja H C. Network intrusion detection using deep learning: A feature learning approach[m]. *Springer*, 2018.

[19] Buczak A L and Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 2016.

[20] Lin C H R Liao H J and Lin Y C. Intrusion detection system: A comprehensive review[j]. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[21] Ting K M. Confusion matrix[j]. *Encyclopedia of Machine Learning and Data Mining*, pages 260–260, 2017.

[22] W Lu M Tavallaee, E Bagheri and A Ghorbani. A detailed analysis of the kdd cup 99 data set. *IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009.

[23] Wu C Phetlasy S, Ohzahata S et al. A sequential classifiers combination method to reduce false negative for intrusion detection system[j]. *IEICE Transactions on Information and Systems*, 102(5):888–897, 2019.

[24] Mustapha N Pozi M S M, Sulaiman M N et al. Improving anomalous rare attack detection rate for intrusion detection system using support vector machine and genetic programming. *Neural Processing Letters*, 2016.

[25] Harbi N Rahman C M, Farid D M et al. Attacks classification in adaptive intrusion detection using decision tree[j]. 2010.

[26] Loia V Rathore S, Sharma P K et al. Social network security: Issues, challenges, threats, and solutions[j]. *Information sciences*, 421:43–69, 2017.

[27] McCulloch W S and Pitts W. A logical calculus of the ideas immanent in nervous activity[j]. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[28] Panchal F S and Panchal M. Review on methods of selecting number of hidden nodes in artificial neural network[j]. *International Journal of Computer Science and Mobile Computing*, 3(11):455–464, 2014.

[29] Sahu S and Mehtre B M. Network intrusion detection system using j48 decision tree[c]. *International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE*, 2015:2023–2026, 2015.

[30] Zahra Jadidi Sheikhan, Mansour and Ali Farrokhi. Intrusion detection using reduced-size rnn based on feature grouping. *Neural Computing and Applications*, 21:6, 2012.

[31] Santosa P I Soe Y N, Feng Y et al. A sequential scheme for detecting cyber attacks in iot environment[c]. *IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress. IEEE*, pages 238–244, 2019.

[32] Wu A S Stein G, Chen B et al. Decision tree classifier for network intrusion detection with ga-based feature selection. *Proceedings of the 43rd annual Southeast regional conference. ACM*, 2, 2005.

[33] Biswas S Subba B and Karmakar S. A neural network based system for intrusion detection and attack classification[c]. *Twenty Second National Conference on Communication (NCC). IEEE*, pages 1–6, 2016.

[34] Case D U. Analysis of the cyber attack on the ukrainian power grid[j]. *Electricity Information Sharing and Analysis Center (E-ISAC)*, 2016.

[35] Me L Viinikka J, Debar H et al. Processing intrusion detection alert aggregates with time series modeling[j]. *Information Fusion*, 10(4):312–324, 2009.

[36] Stallings W. *Cryptography and network security: principles and practices[J].* 2003.

[37] Gang Wang et al. A new approach to intrusion detection using artificial neural networks and fuzzy clustering. *Expert systems with applications*, 37:9, 2010.

[38] Chen Q Ye N, Emran S M et al. Multivariate statistical analysis of audit trails for host-based intrusion detection[j]. *IEEE Transactions on computers*, 51(7):810–820, 2002.