On the Efficiency and Efficacy of NNC-Trees for Intrusion Detection

Jhih-Siang Syu [1], Qiangfu Zhao [1] , Long-Sheng Chen [2]

[1]*University of Aizu, Japan*

{m5192102,qf-zhao}@u-aizu.ac.jp

[2]*Chaoyang University of Technology, Taiwan*

lschen@cyut.edu.tw

***Abstract*—In recent years, information and communication technology has rapidly developed, and it now brings many advantages in different fields. However, there are a lot of loopholes in the network, and they are attractting various attacks. Intrusion detection system has been developed to solve the problem. So far, different detection or analysis methods have been proposed for different data, but still there is no general model good for all cases. We focus on the anomaly detection in this study. We utilize nearest neighbour classifier trees (NNC-Trees) as the detector. To obtain a compact and effective detector, we tested NNC-Tree using different percentages of the training data. Results obtained with the KDD data set show that relatively good detectors can be obtained with only 10% of the data, and the implementation cost can be greatly reduced. The results can be useful for mobile device-based intrusion detection.**

***Keywords*— intrusion detection, nearest neighbor classifier trees, supervised learning.**

1. INTRODUCTION

Nowadays, information and communication technology (ICT) has become an important part in human life, no matter in governments, enterprises or other academic and medical organizations. But, too fast development also brings lots of security problems and crisis. Attack tools can even be found easily on the internet, and hackers are also trying various attacks using internet vulnerabilities [6,7,10].

Fig. 1 is a threat report of new malware from McAfee Labs in May 2015. From this figure (a), we can know that the number of new malware is increased from 2013 first quarter to 2015 first quarter, besides 2014 the fourth quarter. From figure (b), the total number of malware has been increasing all the time. It is so terrible that various malware may exist in our networks anywhere and anytime.

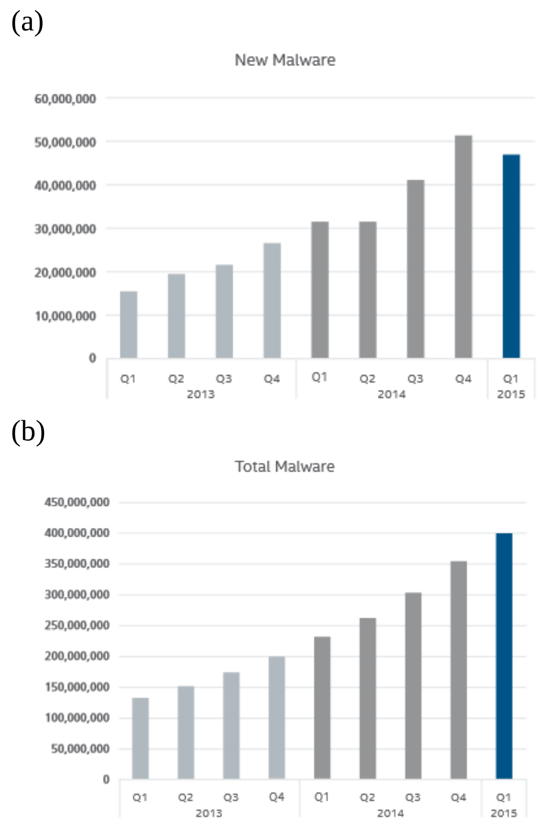


Fig. 1 Malware statistics (Source: 2015 McAfee Labs Threats Report)

Intrusion Detection Systems (IDS) have been developed by Anderson in 1980, hoping to resolve this network security problem. Since then, various attack identification techniques have been proposed, including rule-based, neural networks, support vector machines, and so on. But, 30 years passed, we still cannot resolve this problem. The main reason is the advancement of ICT, leading to the IDS need to be continuous improvement.

IDS could be divided into several different modes based on data type and data analysis methods[10]. For example, based on analysis method we may have misuse detection and anomaly detection. The first uses a known attack signature database, whenever the login data match any feature, IDS will give alarms. The second is a normal behavior model and then observes deviation of behavior. If the behavior is inconsistent with the model, IDS will give alarms. In this example, the accuracy will be high if we choose misuse detection to detect, but it cannot detect unknown attacks. If we use anomaly detection to detect, the false alarm rate will be higher than another, but it can detect unknown attacks [10]. So, there is no any IDS modes could be applicable to all situations in today's network environment.

In Shameli-Sendi et al. research, a defense of network intrusion includes four stages. Respectively, prevention, surveillance, detection and mitigation. In prevention stage, calculator and other related equipment need to appropriate places in different locations in order to ensure that the data and services running. In surveillance stage, set monitor system to collect useful data of host computer and network information. In detection stage, running IDS to detect and analysis. The common way is using anomaly detection, through a pre-established model of normal behavior. IDS will give alarms if deviation. The final stage is mitigation. Make a strategic decision by IDS and restore the system [14]. Ben-Asher and Gonzalez (2015) think that, make sure the network security is a difficult task. In addition to relying on the professional knowledge, but also cognitive the possible attacks from a huge number of network data. Therefore, many data mining methods were successfully applied to the IDS.

Because the range of IDS is wide, the major objective of this study focuses on the classification of anomaly detection, hoping to improve the performance. Kuang et al. (2014) pointed out that “the intrusion detection can be seen as essentially a classification problem to distinguish normal activities and abnormal activities”. Therefore, we utilize the Nearest Neighbor Classifier-Trees (NNC-Trees) to classify with supervised learning. Due to the network packet is a type of big data, we hope to design a good detector using as few data as possible. If we can use minimal data to obtain the same result, it is a good point for NNC-Trees in intrusion detection. To verify the efficiency and efficacy of the NNC-Tree, we use KDD data set and conducted 10 times 10 fold validation.

2. LITERATURE REVIEW

In this chapter, we will briefly describe the Intrusion Detection Systems and Nearest Neighbor Classifier-Trees.

2.1. Intrusion Detection Systems

IDS was first starting from 1980 by Anderson, it has been more than 30 year history. Those unauthorized activities which have been designed to access system resources or data are called “intrusion” [10]. The main purpose of IDS is to detect those attack activity, and provide network administrator to do corresponding treatment. The existing types of IDS include several oriented. Such as sources of information to distinguish and analytical methods to distinguish. The former have network-based (NIDS), host-based (HIDS) and so on. The latter contain misuse-based, anomaly-based and hybrid [5].

In more than 30 year history of IDS, rule-based early detection module has been firstly developed and become the mainstream [8]. After then, different algorithms based detection methods, such as genetic algorithm (GA) [11], Bayes [4], neural networks (NN) [15], and support vector machine (SVM) [6,13] have been constructed. Shameli-Sendi et al. (2014) presented a taxonomy of Intrusion Response Systems (IRS) and Intrusion Risk Assessment (IRA), These two also are important parts of IDS.

Till now, according to the rapid development of ICT, single one type of IDS mode is not enough. So, the hybrid approaches are gradually becoming the mainstream. However, the main purpose of this study is not to build a complete IDS, but how to improve the classification performance. Thus, We choose NNC-Trees method and conduct experiments with supervised learning, hoping to test efficiency and efficacy.

2.2. Nearest Neighbor Classifier-Trees

NNC-Tree have been proposed by Zhao in 2006, and it is a decision tree (DT) with each non-terminal node containing a nearest neighbor classifier (NNC). It is also a heuristic method for defining the teacher signals (group labels), and its basic idea is to partition the data based both on the class labels and on the neighborhood information.

In this tree, every node contains an NNC, besides the terminal node. Fig. 2 shows the basic structure of an NNC-Trees. The best advantage of NNC-Trees are more comprehensible, because the prototypes can be considered as the precedents and can be interpreted easily. Thus, an NNC-Tree can be a very promising model for improving the generalization ability while preserving the comprehensibility of the DTs [16].

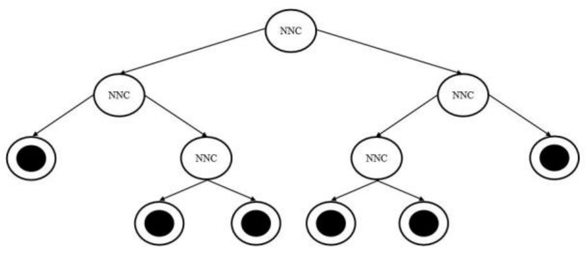


Fig. 2 Structure of an NNC-Tree

NNC-Tree also is a hierarchical clustering method. But difference to others model is that information used in the learning process. Since the label information is not available for clustering, the system obtained is more complex than that obtained by supervised learning. So, NNC-Trees may more suitable with semi-supervised learning [16]. This is the reason why we choose it. If we want to combine semi-supervised learning in the future, the first step is to confirm the effects of NNC-Trees with supervised learning.

3. METHODOLOGY

The main goal of this study is to find the minimum percentage of data to produce a good and compact NNC-Tree detector. The procedure for conducting the research is shown in Fig. 3. The 5 steps will be discussed in detail as follows.

Step 1: Data Collection

The employed “KDD10%” data were taken from KDD CUP’ 99 data set [9]. KDD CUP’ 99 is a well-known public data set for intrusion detection.

The original data file of KDD10% has almost 500,000 data. This amount of data is too huge for our equipment. So, for each category we extracted 1% of the data based on a study of Adel et al. (2015). This approach not only reduce the amount of data, but also maintain the ratio of data from all categories. After extraction, we have nearly 5,000 data. We want to know if increasing the amount of data will affect the result. Thus, we also created another data set by randomly extracting 2% of the data. So we have

- Data set A containing about 5,000 data; and
- Data set B containing about 10,000 data.

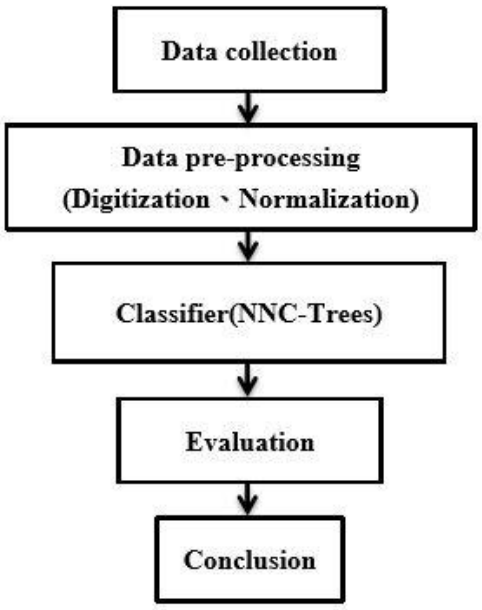


Fig. 3 The implement procedure of this work

Step 2: Data pre-processing

In KDD10% data set, the attacks have been categorized into 4 main types. Moreover, every type could be divided into more detailed sub-types, and totally there are 23 types.

In our study, we conducted experiments with two-class classification and multi-class classification. In two-class classification, we all attacks are put together to form the "attack" class; and non-attacks belong to the “normal” class. In a multi-class classification, we divided all attacks into 4 classes. Adding the normal class we have 5 classes. Digitization and normalization are performed for all data.

Step 3: Induction of the Classifier

In this study, we choose NNC-Trees to be our classifier (detector). We use the training data to construct the classifier, and then input the test data to validate the performance. Moreover, we conducted 10 fold cross validation of 10 times.

Step 4: Evaluation

Criteria used for evaluation include the tree size, the average NNC size, the test error and the time used for training.

Step 5: Conclusion

Based on the results, we can make conclusions.

4. Experimental Results

4.1. Data pre-processing

In this work, we employ KDD10% file from KDD [9]. In this data set, the attack type has been categorized in Table I. Table II and Table III show the data size and class distribution information, respectively, for the two-class and five-class problems.

We first conducted experiments using 1% to 10% of the training data. Then we changed the percentage from 10% to 100%, with a 10% stepsize.

TABLE I

ATTACK TYPESES IN KDD DATA SET

Attack Type	Attack detailed information
U2R	buffer_overflow, loadmodule, multihop, perl, rootkit
R2L	ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster
DOS	back, land, neptune, pod, smurf, teardrop
Probe	Ipsweep, nmap, portsweep, satan

TABLE II

CLASS DISTRIBUTION OF BINARY

Data set	Class	Data distribution	Data size
A	Normal	973	4947
	Attack	3974	
B	Normal	1946	9894
	Attack	7948	

TABLE III

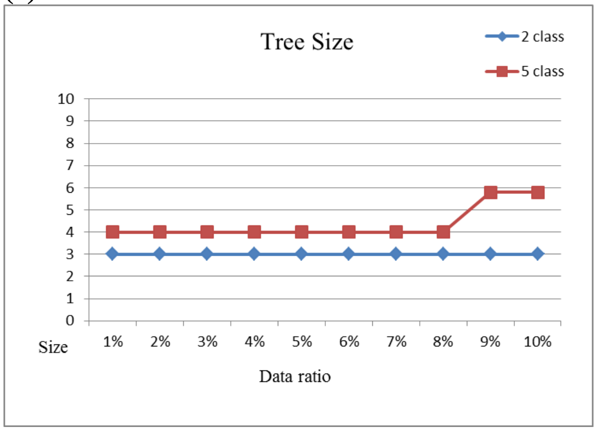
CLASS DISTRIBUTION OF 5-CATEGORY

Data set	Class	Data distribution	Data size
A	Normal	973	4947
	U2r	5	
	R2l	13	
	Dos	3915	
	Probe	41	
B	Normal	3974	9894
	U2r	10	
	R2l	26	
	Dos	7830	
	Probe	82	

4.2. Results for Data Set A

Fig. 4 shows the tree size of the NNC-Trees. In this figure, we can see the trend for the 2-class and 5-class problems. Fig. 4 (a) shows the results obtained using 1% to 10% of the training data, and Fig. 4 (b) shows the results using 10% to 100% of the data. We can see from these figures that the tree size of 2-class have no change even if we use more and more data for training. For the 5-class problem, however, the tree size increases from 3 to 9. From this difference, we may guess that the more the number of classes to classify, the more difficult the problem is, and thus the more complex the detector must be.

(a)



(b)

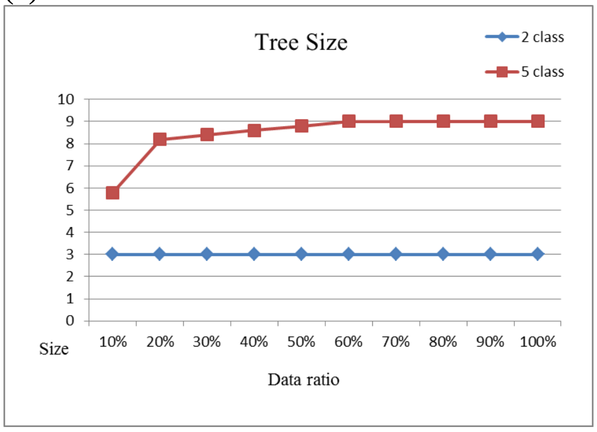
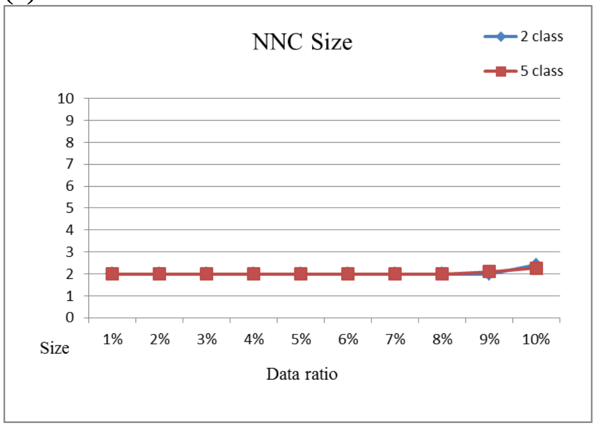


Fig. 4 Tree sizes of the NNC-Tree detectors obtained for the class-2 and class-5 problems

Fig. 5 provides the average NNC size of NNC-Trees. From this figure we can see that when the percentage of data used for training is small, the NNC size almost keeps the same. But, from Fig. 5 (b) we can see that the NNC size increases 2 to 8 for the 2-class problem, and from 2-3 for the 5-class problem. Combining Fig. 4 and Fig. 5 we can see that the "total cost" (i.e. the average NNC size times the tree size) of the NNC-Tree always increases when the percentage of data used for training is increased.

(a)



(b)

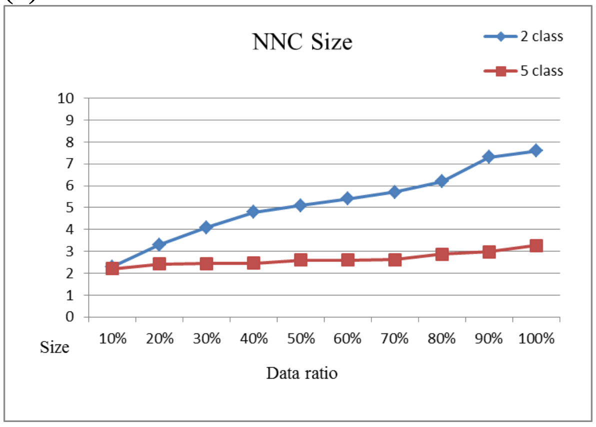
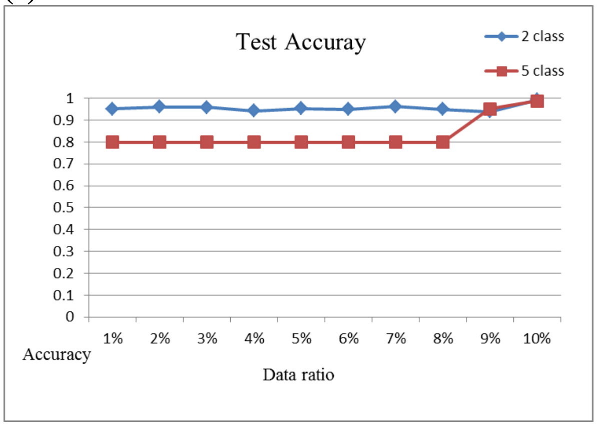


Fig. 5 NNC size of 2-class and 5-class problems

Fig. 6 shows the results of test accuracy. In figure Fig. 6 (a), the performance of 2-class is very good, even it only uses 1% ratio data. But, the performance of 5-class is not good before 8% ratio data. In figure Fig. 6 (b), the performance of both of them are good. In any case, if we use about 10% of the data for training, we are guaranteed to have a very good detector.

Table IV to VII are detailed numerical results for the data set A. We can see more detail results from these tables, and we find that if you use more data to design the NNC-Tree, we will spend more time. So, In this dataset, if we use 1% data to conduct 2-class experiments, we can have the best performance. If we use 9% data to conduct 5-class experiment, we will have the best performance.

(a)



(b)

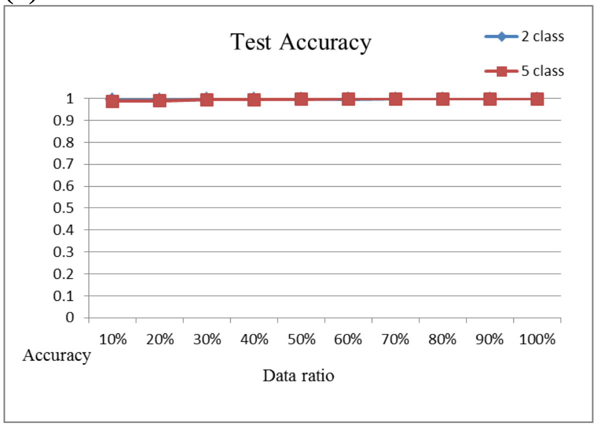


Fig. 6 Accuracy of 2-class and 5-class problems

TABLE IV

THE RESULT OF CLASS-2 WITH A DATA SET AMONG 1% TO 10%

Ratio	Tree Size	NNC Size	Train Error	Test Error	Used Time
1%	3.00 (0)	2.00 (0)	0.00 (0)	0.05 (0.03)	0.14 (0.01)
2%	3.00 (0)	2.00 (0)	0.00 (0)	0.04 (0.02)	0.29 (0.02)
3%	3.00 (0)	2.00 (0)	0.00 (0)	0.04 (0.02)	0.39 (0.13)
4%	3.00 (0)	2.00 (0)	0.00 (0)	0.06 (0.03)	0.59 (0.06)
5%	3.00 (0)	2.00 (0)	0.00 (0)	0.05 (0.03)	0.73 (0.05)
6%	3.00 (0)	2.00 (0)	0.00 (0)	0.05 (0.02)	0.99 (0.36)
7%	3.00 (0)	2.00 (0)	0.00 (0)	0.04 (0.02)	1.03 (0.07)
8%	3.00 (0)	2.00 (0)	0.00 (0)	0.05 (0.02)	1.30 (0.5)
9%	3.00 (0)	2.00 (0)	0.00 (0)	0.06 (0.03)	1.46 (0.54)
10%	3.00 (0)	2.40 (1.26)	0.00 (0)	0.01 (0.01)	2.14 (1.12)

Note: The number 3.00 (0) in this table means Mean (Standard Deviation), respectively.

TABLE V

THE RESULT OF CLASS-2 WITH A DATA SET AMONG 10% TO 100%

Ratio	Tree Size	NNC Size	Train Error	Test Error	Used Time
10%	3.00 (0)	2.30 (0.48)	0.00 (0)	0.01 (0.01)	2.34 (1.02)
20%	3.00 (0)	3.30 (1.49)	0.00 (0)	0.01 (0)	6.95 (2.7)
30%	3.00 (0)	4.10 (1.2)	0.00 (0)	0.00 (0)	12.58 (1.15)
40%	3.00 (0)	4.80 (0.63)	0.00 (0)	0.00 (0)	17.57 (0.77)
50%	3.00 (0)	5.10 (0.74)	0.00 (0)	0.00 (0)	21.92 (1.15)
60%	3.00 (0)	5.40 (0.97)	0.00 (0)	0.00 (0)	26.85 (1.26)
70%	3.00 (0)	5.70 (1.16)	0.00 (0)	0.00 (0)	32.74 (2.06)
80%	3.00 (0)	6.20 (0.63)	0.00 (0)	0.00 (0)	39.23 (2.35)
90%	3.00 (0)	7.30 (0.95)	0.00 (0)	0.00 (0)	46.23 (4.41)
100%	3.00 (0)	7.60 (1.08)	0.00 (0)	0.00 (0)	53.51 (8.11)

Note: The number 3.00 (0) in this table means Mean (Standard Deviation), respectively.

TABLE VI

THE RESULT OF CLASS-5 WITH A DATA SET AMONG 1% TO 10%

Ratio	Tree Size	NNC Size	Train Error	Test Error	Used Time
1%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.20 (0.07)
2%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.38 (0.1)
3%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.55 (0.17)
4%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.76 (0.27)
5%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.91 (0.31)
6%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	0.93 (0.32)
7%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	1.23 (0.45)
8%	4.00 (1.05)	2.00 (0)	0.00 (0)	0.20 (0)	1.43 (0.51)
9%	5.80 (1.40)	2.10 (0.16)	0.00 (0)	0.05 (0.02)	1.67 (0.9)
10%	5.80 (1.40)	2.27 (0.34)	0.00 (0)	0.01 (0.01)	2.07 (0.74)

Note: The number 4 .00(1.05) in this table means Mean (Standard Deviation), respectively.

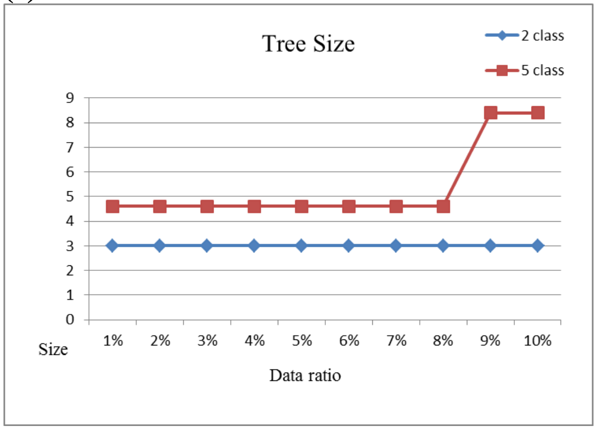
TABLE VII

THE RESULT OF CLASS-5 WITH A DATA SET AMONG 10% TO 100%

Ratio	Tree Size	NNC Size	Train Error	Test Error	Used Time
10%	5.80 (1.4)	2.20 (0.23)	0.00 (0)	0.01 (0.01)	2.04 (0.67)
20%	8.20 (1.03)	2.42 (0.31)	0.00 (0)	0.01 (0.01)	5.55 (2.49)
30%	8.40 (0.97)	2.45 (0.25)	0.00 (0)	0.00 (0)	10.08 (4.64)
40%	8.60 (0.84)	2.47 (0.3)	0.00 (0)	0.01 (0)	15.72 (5.81)
50%	8.80 (0.63)	2.61 (0.16)	0.00 (0)	0.00 (0)	23.28 (7.47)
60%	9.00 (0)	2.60 (0.29)	0.00 (0)	0.00 (0)	31.18 (8.54)
70%	9.00 (0)	2.63 (0.21)	0.00 (0)	0.00 (0)	39.33 (7.87)
80%	9.00 (0)	2.88 (0.27)	0.00 (0)	0.00 (0)	49.33 (5.53)
90%	9.00 (0)	2.98 (0.4)	0.00 (0)	0.00 (0)	53.04 (7.8)
100%	9.00 (0)	3.28 (0.18)	0.00 (0)	0.00 (0)	64.04 (5.04)

Note: The number 5.80 (1.4) in this table means Mean (Standard Deviation), respectively.

(a)



(b)

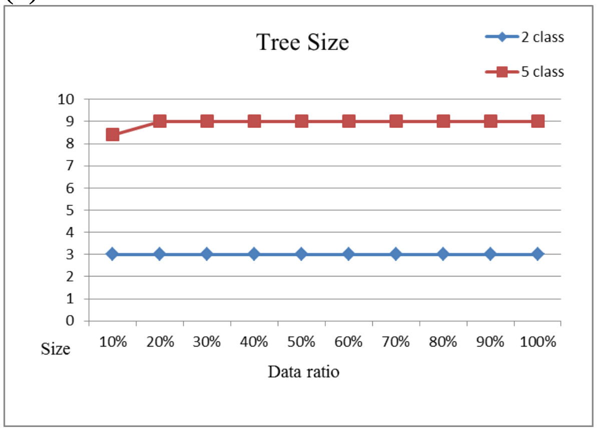


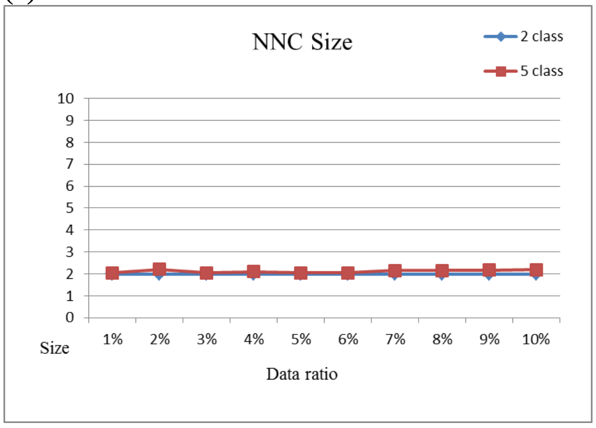
Fig. 7 Tree size of class-2 and class-5 with B data set

4.3. Results for Data Set B

After the experiments with data set A, we want to know if we use more data to test whether have the same result. Therefore, we use data set B to conduct the same experiments.

The results are shown in Fig. 7 to Fig. 9. Basically, the results between datasets A and B have no major change. Except the result of NNC size have different performance in 10% to 100%. But, we still have the same conclusion.

(a)



(b)

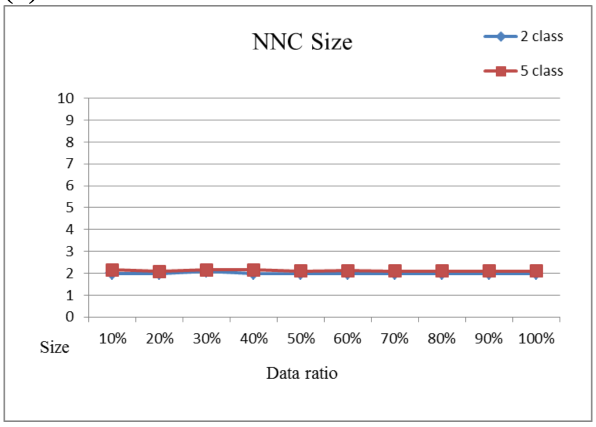


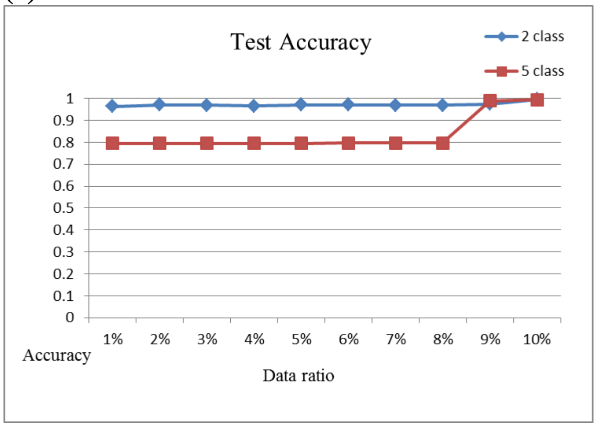
Fig. 8 NNC size of 2-class and 5-class problems with data set B

5. Conclusions

In this study, we have tried to utilize NNC-Trees for intrusion detection. We have conducted a lot of experiments using different percentages of the training data. Comparing results obtained using datasets A and B dataset, we can see that all criteria have the same trends. That is, we can obtain very good NNC-Tree detectors using around 10% of the data, and the total cost for implement (i.e. the average NNC size times the tree size) can be greatly reduced.

Next step, we would like to try a different data set to verify the efficiency and efficacy of the NNC-Tree detector. In addition, we will also try the NNC-Trees for semi-supervised learning. It is believed that semi-supervised learning is good to fully use information contained in un-labeled data. We will try to improved the algorithm for training NNC-Trees, and compare the results with existing algorithms.

(a)



(b)

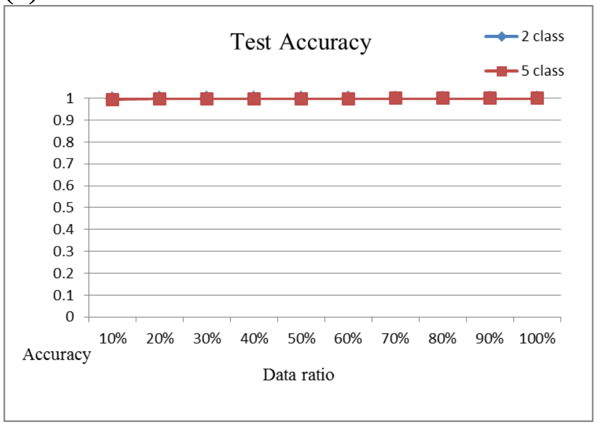


Fig. 9 Test accuracy of 2-class and 5-class problems with data set B

References

[1] Adel, S., E., Zeynep, O., and Adnan, M., A., B., A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems, Expert Systems with Applications, 42 (2015), 2670 – 2679.

[2] Anderson, P., Computer Security Threat Monitoring and Surveillance, Box 42 Fort Washington, Pa. 19034, 215(1980), 646–4706.

[3] Ben-Asher, N., Gonzalez, C., Effects of cyber security knowledge on attack detection, Computers in Human Behavior 48 (2015) 51–61.

[4] Benferhat, S., Boudjelida, A., Tabia, K., and Drias, H., An intrusion detection and alert correlation approach based on revising probabilistic classifiers using expert knowledge, Applied Intelligence, 38 (4) (2013), 520–540.

[5] Chen, L-S., Syu, J-S., Local kernel-principal component analysis based method to detect network intrusions, International Conference on Internet Studies, July 18-19, 2015, Tokyo, Japan..

[6] Feng, W., Zhang, Q., Uu, G., and Huang, J. X., Mining network data for intrusion detection through combining SVMs with ant colony networks, Future Generation Computer Systems, 37 (2014), 127–140.

[7] Govindarajan, M., and Chandrasekaran, RM., Intrusion Detection using an Ensemble of Classification Methods, Proceedings of the World Congress on Engineering and Computer Science WCECS (2012).

[8] Han, S. J., and Cho, S. B., Rule-based integration of multiple measure-models for effective intrusion detection Systems, IEEE International Conference 1. 6(2003), 120-125.

[9] Hettich, S., and Bay, S. D., The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science (1999).

[10] Hubballi, N., and Suryanarayanan, V., False alarm minimization techniques in signature-based intrusion detection systems : A survey, Computer Communications, 49(2014), 1–17.

[11] Kuang, F., Xu, W., and Zhang, S., A novel hybrid KPCA and SVM with GA model for intrusion detection, Applied Soft Computing, 18(2014), 178–184.

[12] Lee, W., Stolf, S. J., and Mok, K. W., A data mining framework for building intrusion detection models. Proceedings of the 1999 IEEE Symposium Security and Privacy 13(1999), 120–132.

[13] Mohammed, M. N., and Sulaiman, N., Intrusion Detection System Based on SVM for WLAN, Procedia Technology 1(2012), 313–317.

[14] Shameli-Sendi, A., Cheriet, M., Hamou-Lhadj, A., Taxonomy of intrusion risk assessment and response system, Comp u t e r s & S e c u r i t y 4 5 (2 0 1 4) 1–1 6.

[15] Thomas, C., and Balakrishnan, N., Performance enhancement of intrusion detection systems using advances in sensor fusion, in: Fusion'08: Proceedings of the 11th International Conference on Information Fusion, (2008), 1671–1677.

[16] Zhao, Q., Inducing NNC-Trees with the R4-Rule, IEEE Transactions on Systems, Man, And Cybernetics—Part B: Cybernetics, Vol. 36, No. 3, June 2006.