Intrusion Detection in Computer Networks based on Machine Learning Algorithms

Alireza Osareh, Bita Shadgar

Computer Science Department, Faculty of Engineering, Shahid Chamran University, Ahvaz, Iran

Abstract

Network security technology has become crucial in protecting government and industry computing infrastructure. Modern intrusion detection applications face complex requirements; they need to be reliable, extensible, easy to manage, and have low maintenance cost. In recent years, machine learning-based intrusion detection systems have demonstrated high accuracy, good generalization to novel types of intrusion, and robust behavior in a changing environment. This work aims to compare efficiency of machine learning methods in intrusion detection system, including artificial neural networks and support vector machine, with the hope of providing reference for establishing intrusion detection system in future. Compared with other related works in machine learning-based intrusion detectors, we propose to calculate the mean value via sampling different ratios of normal data for each measurement, which lead us to reach a better accuracy rate for observation data in real world. We compare the accuracy, detection rate, false alarm rate for 4 attack types. The extensive experimental results on the KDD-cup intrusion detection benchmark dataset demonstrate that the proposed approach produces higher performance than KDD Winner, especially for U2R and U2L type attacks.

Key words:

Intrusion detection, KDD-cup dataset, Neural networks, Support vector machines, Anomaly detection

1. Introduction

Information held by IT products or systems is a critical resource that enables organizations to succeed in their mission. Additionally, individuals have a reasonable expectation that their personal information contained in IT products or systems remain private, be available to them as needed, and not be subject to unauthorized modification. IT products or systems should perform their functions while exercising proper control of the information to ensure it is protected against hazards such as unwanted or unwarranted dissemination, alteration, or loss. The term IT security is used to cover prevention and mitigation of these and similar hazards [1].

It is very important that the security mechanisms of a system are designed to prevent unauthorized access to system resources and data. However, completely preventing breaches of security appear, at present, unrealistic. However, we can try to detect these intrusion attempts so that action may be taken to repair the damage now or later. This field of research is called Intrusion Detection.

The goal of an Intrusion Detection System (IDS) is to identify occurrences of security breaches capable of compromising the integrity of resources or services. File integrity analyzers are a class of related tools that automatically verify the content of security-critical files. Frequently referred to as tripwires, they attempt to detect if files have been modified in unauthorized ways. Once suspicious modifications are detected by triggering the tripwire, the analyzer may alert a security administrator or invoke some type of automated response. Alternatively, file analyzers can provide guidance for damage control, such as identifying the modified files needing to be restored or hooks installed by the attacker to facilitate subsequent access [2, 3].

While introducing the concept of intrusion detection in 1980, we defined an intrusion attempt or a threat to be the potential of a deliberate unauthorized attempt to: access information, manipulate information, or render a system unreliable or unusable.

There are two ways to handle subversion attempts. One way is to prevent subversion itself by building a completely secure system [4]. Network administrator could, for example, require all users to identify and authenticate themselves; administrator could protect data by various cryptographic methods and very tight access control mechanisms. However this is not really feasible because:

1. In practice, it is not possible to build a completely secure system because bug free softwares are still a dream and no one wants to make the effort to try to develop such softwares.

Apart from the fact that users do not seem to be getting their money’s worth when they buy software, there are also security implications with their E-mail software. In addition, designing and implementing a totally secure system is thus an extremely difficult task.

Manuscript received November 5, 2008
Manuscript revised November 20, 2008

2. The vastly installed base of systems worldwide guarantees that any transition to a secure system (if it is ever developed) will be long in coming.

3. Cryptographic methods have their own problems. Passwords can be cracked; users can lose their passwords, and entire crypt-systems can be broken.

4. Ever a truly secure system is vulnerable to abuse by insiders who abuse their privileges.

5. It has seen that the relationship between the level of access control and user efficiency is an inverse one, which means that the stricter the mechanisms, the lower efficiency becomes.

We thus see that we are stuck with systems that have vulnerabilities for a while to come. If there were attacks on a system, we would like to detect them as soon as possible (preferably in real-time) and appropriate action. This is essential with what an IDS does. An IDS does not usually take preventive measures when an attack is detected, it is a reactive rather than pro-active agent [5, 6]. It plays the role of information rather than a police officer.

It is thus more important than ever before that since it seems obvious that administrators cannot prevent subversion, they should at least try to detect it and prevent similar attacks in the future. The following keywords are used in IDS:

Risk: Accidental or unpredictable exposure of information, or violation of operations integrity due to the malfunction of hardware or incomplete or incorrect software design.

Vulnerability: A known or suspected flaw in the hardware or software or operation of a system that exposes the system to penetration or its information to accidental disclosure.

Attack: A specific formulation or execution of a plan to carry out a threat.

Penetration: A successful attack, the ability to obtain unauthorized (undetected) access to files and programs or the control state of computer system.

Intrusion: A set of actions aimed to compromise the security goals, namely integrity, confidentiality, or availability, of a computing and
networking resource.

Intrusion detection: The process of identifying and responding to intrusion activities.

An IDS system aims to supervise and control all cases happening to computer system or network system, analyze any signal arising from related safety problems, send alarms when safety problems occur, and inform related personnel or units to take relevant measures to reduce possible risks [7]. Its framework includes three parts as follows:

1. Information collection: Data collection: the source of these collected data can be separated into host, network and application, according to the position.

2. Analysis engine: Analysis engine is able to analyze whether or not there are symptom of any intrusion.

3. Response: Take actions after analysis, record analysis results, send real-time alarm, or adjust intrusion detection system, and so on.

Generally speaking, there are two kinds of classification methods for intrusion detection system [7]:

1. According to different data sources, intrusion detection system includes host-based IDS and network-based IDS.

2. According to different analysis methods, intrusion detection system includes Misuse Detection and Anomaly Detection.

The following is to give a brief introduction on property, advantage and disadvantage of these intrusion detection systems.

(a) Classification based on different information source:

Host-based IDS: Its data comes from the records of various activities of hosts, including audit record of operation system, system logs, application programs information, and so on. Taking Windows NT operation system as an example, its event logs mechanism searches and collects three patterns of system events: Operation system event, safety event and application event; and examples of application program information are as follows: Database system, WWW servers, and so on. Its advantage and disadvantage are stated as follows [8]:

Advantage:

1. It can judge whether or not the host is intruded more accurately: Because its data comes form system audit records and system logs of hosts, comparing with network-based intrusion detection system, it can more accurately judge network attacks or intrusion on hosts.

2. It can detect attacks under encrypted network environment: Because the data comes from system files and transmitted encrypted data in network which are decrypted in hosts, thus the data is not affected.

3. It does not need additional hardware: It just needs monitoring system installed in specified hosts, without additional hardware.

Disadvantage:

1. Higher cost: Monitoring systems must be installed in each host; and because of different hosts, the audit files and log pattern are accordingly different, thus different intrusion detection systems are required in each host.

2. It may affect system efficiency of monitored hosts: Intrusion detection system in monitoring state may occupy system sources of hosts.

Network-based IDS [7]: Its data is mainly collected network generic stream going through network segments, such as: Internet packets. And its advantage and disadvantage are stated as follows:

Advantage:

1. Low cost: Only network-based IDS can detect all attacks in a LAN, and the cost is just for the device.
2. It can detect attacks that cannot be done by host-based IDS, such as denial of service.

Disadvantage:

1. The flux is large, and some packets may be lost, and it cannot detect all packets in network.
2. In large-scale network, it requires more rapid CPU and more memory space, to analyze bulk data.
3. It cannot deal with encrypted packets, and it may not receive attack information in encrypted packets accordingly.

(b) Classification based on different analysis method:

Misuse Detection [7]: It is also named signature-based detection, which can transform the information of attack symptom or policy disobeying into state transition-based signature or rule, and such information is stored in signature database. To judge whether or not it is attack, pre-treated case data should be first compared with the signature of signature database, and those conforming to attack signature data can be judged as attack. Its advantage is high detection rate and low false alarm rate for known attacks; however, its detection capacity is low for un-known detection methods, and attack database should be renewed on a regular basis.

Anomaly Detection: It may establish a profiles for normal behavior of users, which comes from statistics data of users in the former period; when detection is performed, the profiles is compared with actual users' data, if the offset is below threshold value, user's behavior can be considered normal, and it has no intention of attacks; if the offset is above threshold value, user's behavior can be considered abnormal. Anomaly detection is based on an assumption that intruder's behavior is different from normal users' behavior. Detection rate of the method is high, and it is more likely to detect un-known attacks, but mis-judgment rate is also high.

Hybrid: The advantage of misuse detection is low mis-judgment rate, as well as low detection capacity for unknown attacks; comparatively, anomaly detection owns the capacity of detecting unknown attacks, but with high mis-judgment rate. If these said two methods are combined for detection, they can supply disadvantage of each other, such as: MINDS [8].

IDS appears like internet supervision and alarm device, to observe and analyze whether the internet attacks may occur, timely send alarm before risks are caused by attacks, execute corresponding response measures, and reduce occurrence of bigger losses. Moreover, some technologies are based on pattern check, with low misjudgment rate, but the pattern-based should be upgraded on a regular basis, such technologies do not possess enough detection capacity for unknown and renewed attack manners.

Machine learning is widely applied in various areas currently, such as: medical diagnosis, Biological signature differentiation, search engine, pronunciation and handwritten identification and so on. Recently, many researches have applied the state of the art machine learning and data mining algorithms to intrusion detection, which can analysis bulk data, and such technologies own better detection capacity for unknown attacks. Though some research achievements have been scored, there is a lot of development potential.

Under such circumstance with most same conditions, how is the efficiency of different machine learning methods applied in intrusion detection. Besides the said manners, what methods are there? Therefore, this research intends to compare the efficiency of two well-known machine learning methods i.e. artificial neural networks and support vector machines applied in intrusion detection with the hope of providing possible suggestion for improvement, as the reference for building intrusion detection system.

2. Related Work

There are several approaches for solving intrusion detection problems. Lee et al. [9] built an intrusion detection model by used association rule and frequent episode techniques on system audit data. Axis attribute(s) as a form of item constraints are used only to compute relevant patterns and an iterative level-wise approximate mining procedure is used to uncover the low frequency patterns in semi-automated way.

NIDES system performs anomaly detection by using statistical approaches [10]. It generates profiles by using statistical measurements that tip into activity of subjects and profile generation. In general, there four types of statistical measurements: activity intensity, audit record distribution, categorical and ordinal.

Neural networks are trained to detect intrusion systems. An n-layer network is constructed and abstract commands are defined in terms of sequence of information units, the input to the neural in the training data. Each command is considered with pre-defined *w* commands together to predict the next coming command expected from the user. After training, the system has the profile of the user. At the testing step, the anomaly is said to occur as the user deviates from the expected behavior [11]. Short sequences of system calls carry out the prediction process. In this system, Hamming distance comparison with a threshold is used to discriminate the normal sequence from the abnormal sequence [12].

Natural immune system is another proposed method to deal with the intrusion detection problem in distributed manner. Distributed positive and negative detectors are used to distinguish self and non-self behaviors [13].

According to the work described in Balasubramaniyan et al. [14] a multi-agent architecture detects the intrusion of multiple independent entities by autonomous agents working collectively. Another multi-agent architecture consisting of autonomous agents that are built on genetic programming method is also proposed in Crosbie et al. [15]. Agents exploiting the learning power of genetic programming are evaluated with their performance and agents having highest performance are chosen to detect intrusions. Clustering techniques were applied on unlabeled data in order to discover anomalies in the data [16].

Evolving fuzzy classifiers have been studied for possible application to the intrusion detection problem [17]. System audit training data is used to extract rules for each normal and abnormal behavior by the genetic algorithm. Rules are represented as complete expression tree with identified operators, such as conjunction, disjunction and not.

3. Intrusion Dataset

In the 1998 DARPA (KDD-cup dataset) [18] intrusion detection evaluation programme, an environment was set up to get raw TCP/IP dump data for a network by simulating a typical US Air Force LAN. The LAN was operated like a real environment, but was blasted with several attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. Of this database, a training subset of 494014 records was used, of which about 20% represent normal patterns (Table 1). Indeed, the test set was composed of 311029 data records (see Table 2).

The four different categories of attack patterns are as follows [19]. It is important to mention that in this paper, we have demonstrated the capability of the suggested learning method to detect abnormal behaviours via normal behaviours.

3.1. Probing

Probing is a class of attacks where an attacker scans a network to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use the information to look for exploits. There are different types of probes: some of them abuse the computer's legitimate features; some of them use social engineering techniques. This class of attacks is the most commonly heard and requires very little technical expertise.

3.2. Denial of service (DOS) attacks

DoS is a class of attacks where an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, thus denying legitimate users access to a machine There are different ways to launch DoS attacks: by abusing the computer's legitimate features; by targeting the implementations bugs; or by exploiting the system's mis-configurations. DoS attacks are classified based on the services that an attacker renders unavailable to legitimate users.

3.3. User to root (U2R) attacks

User to root exploits are a class of attacks where an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Most common exploits in this class of attacks are regular buffer overflows, which are caused by regular programming mistakes and environment assumptions.

3.4. Remote to user (R2L) attacks

A remote to user attack is a class of attacks where an attacker sends packets to a machine over a network, then exploits machine's vulnerability to illegally gain local access as a user. There are different types of R2U attacks: the most common attack in this class is done using social engineering.

Table 1: Training data set

Class	Class name	No. of instances	%
0	Normal	97271	19.6
1	Probe	4107	0.83
2	DOS	391458	79.2
3	U2R	59	0.01
4	R2L	1119	0.22

Table 2: Test data set

Class	Class name	No. of instances	%
0	Normal	60593	19.4
1	Probe	4166	1.33
2	DOS	231455	74.4
3	U2R	88	0.02
4	R2L	14727	4.73

4. System Structure

In this work, two distinct machine learning algorithms i.e. neural network (NN) and support vector machines (SVM) are tested against the KDD dataset. An overview of how optimum models of these algorithms were identified as well as their intrusion detection performance on the KDD testing dataset follows next. The proceeding flow of the research is shown in Fig. 1.

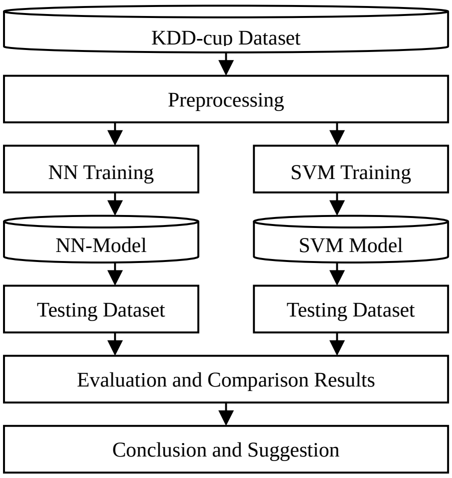

Fig. 1 Architecture of proposed system

4.1. Neural Networks

The Multilayer Perceptrons (MLP) [20] neural networks have been very successful in a variety of applications, producing results, which are at least competitive and often exceed other existing computational learning models. They are capable of approximating, to arbitrary accuracy, any continuous function as long as they contain enough hidden units. This means that such models can form any classification decision boundary in feature space and thus act as non-linear discriminant function.

When the NN is used for pattern classification, there is one input node for each element of the feature vector. There is usually one output node for each class to which a feature may be assigned (Figure 2). The hidden nodes enable internal representation of the data to be developed by the NN during learning.

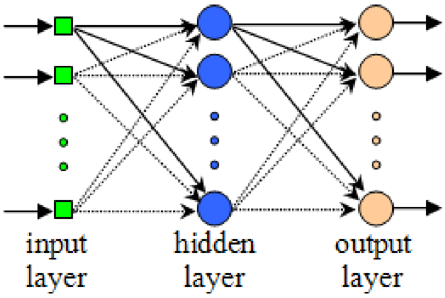

Fig. 2 A typical MLP neural network structure

One learning algorithm used for MLP is called back-propagation rule. This is a gradient descent method and based on an error function that represents the difference between the network's calculated output and the desired output. This error function is defined, based on the Mean Squared Error (MSE). Thus the error for pattern *i* is written as:

$$E^i = \frac{1}{2}\sum_{p=1}^{k}\left(y_p^i - o_p^i\right)^2 \qquad (1)$$

where y^i_p is the true output of the *p*th output node of the network when the *i*th feature vector is fed to the network and *k* represents the number of neurons of the output layer. Similarly the o^i_p is the desired output of the *p*th output node. Consequently the MSE can be summed over the entire training set.

In order to successfully learn, the network's true output should approach the desired output by continuously reducing the value of this error. The back-propagation rule calculates the error for a particular input and then back-propagates the error from one layer to the previous one. The connection weights, between the nodes, are adjusted according to the back-propagated error so that the error is reduced and the network learns.

4.2. Support Vector Machines

Support Vector Machines [21, 22] have become an increasingly popular tool for machine learning tasks involving classification and regression. The SVMs demonstrate various attractive features such as good generalisation ability compared to other classifiers. Indeed, there are relatively few free parameters to adjust and the architecture does not require to be found experimentally.

Given a training set of instance-label pairs (x_i, y_i), i=1, …, l where $x_i \in R^n$ and $y \in \{1, -1\}^l$, the SVMs require the solution of the following optimization problem:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C\sum_{i=1}^{l}\xi_i \qquad (2)$$

subject to $y_i\left(w^T\phi(x_i)+b\right) \geq 1-\xi_i, \quad \xi_i \geq 0.$

Here training vectors x_i are mapped into a higher (maybe infinite) dimensional space by the function *φ*. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space (Figure 3). *b* determines an offset of the discrimination hyperplane from origin. Slack variables ξ_i are introduced to measure the amount of violation of the constraints. The penalty *C* is a user defined positive regularisation parameter (setting *C* = ∞ leads back to the linearly separable case) which controls a trade-off between the wide margin and a small number of margin failures (*soft margins*).

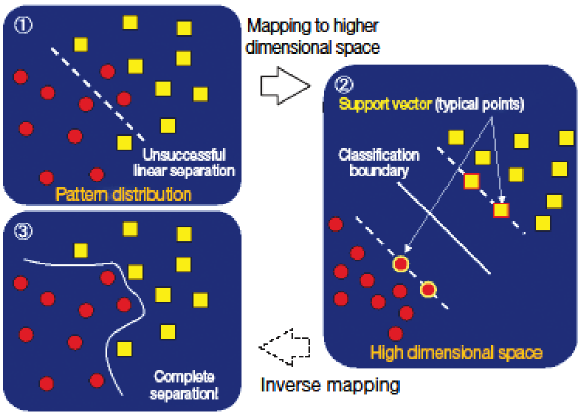

Fig. 3 SVM non-linear separable cases

Furthermore, $K(x_{i,}\ x_j) = \varphi(x_i)^T\ \varphi(x_j)$ is called the kernel function. There are many kernels that can be used such as Gaussian radial basis functions (RBF) [22]:

$$K(x_{i,}x_j)=\exp\left(-\frac{\|x_i-x_j\|^2}{2\sigma^2}\right) \qquad (3)$$

where $\sigma > 0$ is a constant that defines the kernel width. Another kernel function is the polynomial (of degree d):

$$K(x_i,x_j)=(1+x_i.x_j)^d \qquad (4)$$

where $d > 0$ is a constant that defines the kernel order. The associated parameters, order d or Gaussian σ are determined within the training phase.

4.3. Analysis and Evaluation

This work intends to compare the efficiency of neural networks and support vector machines against KDD-cup dataset. This dataset is over large and various data is distributed unevenly. Therefore, this research work will sample training dataset and test dataset. In fact, based on the normal proportion, we select each 10000 group of data where normal proportion is 10%, 20%, …, 90% in training and test datasets and make remaining data, namely attack data, even and sample them. Having done pre-position modification of data, training and test can begin.

Detection and identification of attack and non-attack behaviors can be generalized as the follows:

(a) True positive (TP): the amount of attack detected when it is actually attack.

(b) True negative (TN): the amount of normal detected when it is actually normal.

(c) False positive (FP): The amount of attack detected when it is actually normal, namely false alarm.

(d) False negative (FN): The amount of normal detected when it is actually attack, namely the attacks which can be detected by intrusion detection system.

As intrusion detection systems require high detection rate and low false alarm rate, thus we compare *accuracy*, *detection rate* and *false alarm rate*, and present the comparison results of various attacks.

Accuracy refers to the proportion of data classified an accurate type in total data, namely the situation TP and TN, thus the accuracy can be defined as follows:

Accuracy = (TP+TN/TP+TN+FP+FN)*100% (5)

Table 3 summarizes the results measured by original class label classification.

Table 3: Accuracy results of NN and SVM classifiers against KDD-cup dataset

Percentage of normal data	NN (%)	SVM (%)
10	42.3	38.0
20	44.5	42.1
30	53.9	49.3
40	58.6	56.2
50	65.0	64.5
60	74.3	73.8
70	80.6	82.9
80	87.1	89.2
90	93.7	95.3
Average (overall)	66.6	65.7

To assess and analyze the behavior of NN and SVM classifiers in terms of accuracy criterion and throughout a whole range of normal data values, the curves shown in Fig. 4 are produced. As it can be seen from Table 1 and Fig. 4, there is not significant difference between accuracy of the two methods; however, NN could achieve better accuracy than SVM when the proportion of normal information is small. On the other hand when the proportion of normal information is about 50% the NN and SVM accuracy is approximately equal and for the values>70% SVM outperformed NN. According to the overall accuracy, NN classifiers are slightly better than SVMs.

As another useful measurement, *detection rate* refers to the proportion of attack detected among all attack data, namely, the situation of TP, thus detection rate is defined as follows:

Detection Rate = (TP/TP+FN) *100% (6)

Table 4 present the detection rate results measured based on NN and SVM classifiers.

Table 4: Detection rate results of NN and SVM classifiers against KDD-cup dataset

Percentage of normal data	NN (%)	SVM (%)
10	71.6	77.7
20	69.0	75.1
30	68.1	76.5
40	69.5	75.0
50	67.4	73.6
60	65.2	72.0
70	64.0	71.3
80	63.8	70.2
90	63.5	68.7
Average (overall)	66.9	73.3

Figure 5 shows the detection rate results for different normal data percentages. In detection rate, both NN and SVM classifiers results decline as the percentage of normal data rises. Overall, SVMs outperform NNs and in terms of average value, SVMs surpasses NNs by about 6.4%.

False alarm rate refers to the proportion that normal data is falsely detected as attack behavior, namely, the situation of FP, thus false alarm rate is defined as follows:

$$\text{False alarm rate} = (FP/FP+TN) *100\% \qquad (7)$$

Table 5 summarizes results of false alarm rate between NN and SVM classifiers.

Table 5: False alarm rate results of NN and SVM classifiers against KDD-cup dataset

Percentage of normal data	NN (%)	SVM (%)
10	3.11	2.10
20	2.24	1.75
30	2.15	1.50
40	1.04	0.75
50	1.00	0.84
60	0.90	0.45
70	1.24	0.40
80	1.54	0.32
90	1.39	0.25
Average (overall)	1.62	0.92

To assess and analyze the behavior of NN and SVM classifiers in terms of alarm rate criterion and throughout a whole range of normal data values, the curves shown in Fig. 6 are produced.

It is evident from Table 5 and Fig. 6 that in terms of false alarm rate, SVM outperformed the NN in all the cases. Thus, SVM could provide less average false alarm rate (0.92) than the NN (1.62).

4.4. Accuracy Comparison between Different Attacks

Table 6 summarizes comparison results of accuracy (refers to the proportion that the type of data is corrected classified) of 4 different attacks i.e. Probe, Dos, U2R, R2L based on NNs and SVMs. It is evident from this Table that:

(a) For Probe attack: Accuracy of NN is better than that of SVM when the proportion of normal information is less than 50% and in other circumstances, SVM outperform NN. For this type of attack the average accuracy of NN and SVM classifiers are 82.5% and 83.2% respectively.

(b) For Dos attack: SVM classifiers outperform their NN counterparts in all cases except when the proportion of normal data is 70%. For this type of attack the average accuracy of NN and SVM classifiers are 58.6% and 62.5% respectively.

(c) For U2R attack: Generally speaking, accuracy of SVM is better than that of NN when the proportion of normal information is less than 60% and in other circumstances, SVM outperform NN. For this type of attack the average accuracy of NN and SVM classifiers are 65.4% and 65.5% respectively.

(d) For R2L attack: According to the average value, these two methods are similar in accuracy. When the proportion of normal data is 10%, 40%, 50% and 70%, SVM is better, and NN is better otherwise. For this type of attack the average accuracy of NN and SVM classifiers are 14.6% and 14.7% respectively.

Finally, average results which are achieved in this work are compared with the results obtained through KDD Cup 99 winner, shown in Table 7. As it can be seen the accuracy of KDD Winner is very high in Dos attack, but it is far worse than NN and SVM in U2R and R2L.

Table 6: Accuracy comparison of NN and SVM classifiers against 4 kinds of attacks

Data	Probe(%)		Dos(%)		U2R(%)		R2L(%)	
	NN	SVM	NN	SVM	NN	SVM	NN	SVM
10	75.0	70.3	55.3	60.8	58.3	62.4	8.3	14.5
20	81.2	72.5	56.0	61.2	60.1	66.0	16.8	12.3
30	83.4	78.9	56.4	61.9	67.2	70.9	17.4	12.9
40	80.1	78.5	54.2	56.7	72.0	74.1	10.6	16.8
50	85.8	90.1	53.1	58.0	70.3	72.2	18.0	19.2
60	82.0	86.4	60.7	65.1	59.2	64.1	20.1	17.6
70	84.0	89.0	66.5	66.2	65.7	61.3	11.3	13.7
80	85.3	91.2	64.0	67.4	67.4	58.9	12.5	10.0
90	85.7	92.5	61.2	65.5	69.1	60.0	16.8	15.9
Average	**82.5**	**83.2**	**58.6**	**62.5**	**65.4**	**65.5**	**14.6**	**14.7**

Table 7: Detection rate average results for various attacks through KDD Winner.

	Probe	Dos	U2R	R2L
KDD Winner	83.3	97.1	13.2	8.4
NN	82.5	58.6	65.4	14.6
SVM	83.2	62.5	65.5	14.7

5. Conclusions and suggestions

5.1. Conclusions

The research work compares accuracy, detection rate, false alarm rate and accuracy of other attacks under different proportion of normal information. KDD Cup 99 dataset is current benchmark dataset in intrusion detection; however, its data is not distributed evenly, error may occur if only one set is used. Therefore, in comparison, the research applies different normal data proportion for training and test, finally get one average value, and expect to obtain more objective results.

For comparison results of NN and SVM, we find that SVM is superior to NN in detection; in false alarm rate and in accuracy for Probe, Dos and U2R and R2Lattacks, while NN could outperform the SVM only in accuracy.

5.2. Future work

The KDD Cup 99 dataset which is utilized in this work is popularly used as a benchmark dataset in several different research works. However, since 1999 network technology and attack methods changes greatly, thus this dataset may not be able to reflect real network situation nowadays. Therefore, if newer information can more accurately reflect current network situation.

Through our test and comparison, the accuracy of NN is higher than that of SVM, but false alarm and detection rate of SVM is better; if we combine the two methods, overall accuracy can be increased greatly.

In sampling, this research supposes that the distribution of attack data other than normal data is even, which cannot surely get optimal results, and this should be improved and validated in future work.

References

[1] CCIMB, Common Crireria for Information Technology Security Version 2.1: Part 1: Introduction and General Model, CCIMB-99-031, Evaluation 1999.

[2] R. DeMara and A. Rocke, “Migration of network tempering using dynamic dispatch of mobile agents,” Computers and Security, vol. 23, pp. 31-42, 2004.

[3] Y. Fyodor, “SNORTNET”- A distributed intrusion detection system, 2000 (Avaiable from http://snornet.scorpions.net/snortnet.pdf)

[4] J. Viega and G. McGraw, Building Secure Software: How to avoid Security Problems the Right Way, Addison Wesley, 2002.

[5] S. Cho, “Incorporating soft computing techniques into a probabilistic intrusion detection system,” IEEE Transactions on Systems, Man and Sybernetics, vol. 32, no. 2, pp. 154-160, 2002.

[6] G. Shipley, Chapter 12, “Intrusion Detection Systems (IDSs)”, in Shelley Johnston Markunday (Ed.), Maximum Security: A Hacker's Guide to Protecting Your Internet Site and Network, Third edition, Sams Publication, 2001.

[7] A. Abraham, R. Jain and J. Thomas, “D-SCIDS: Distributed soft computing intrusion detection system,” Journal of Network and Computer Applications, vol. 30, pp. 81-98, 2007.

[8] L. Ertoz, E. Eilerson and A. Lazareviv, “The MINDS – Minnesota intrusion detection system, next generation data mining,” MIT Press, 2004.

[9] W. Lee and J. Salvatore, “Mining audit data to build intrusion detection models,” Proceedings of ACM SIGKDD international conference on knowledge discovery and data mining, pp. 66-72, 1998.

[10] T. Lunt, “Detecting intruders in computer systems,” Proceedings of auditing and computer technology conference, pp. 23-30, 1999.

[11] J. Ryan, M. Lin and R. Miikkulainen, “Intrusion detection with neural networks. In: Advances in neural information processing systems,” vol. 10, MIT Press, 1998.

[12] S. Bridges, and R. Vaughn, “Fuzzy data mining and genetic algorithms applied to intrusion detection,” Proceedings of the national information systems security conference, pp. 8-15, 2000.

[13] S. Hofmeyr, S. Forrest and A. Somayaji, “Intrusion detection using sequences of system calls,” Journal of Computer Security, vol. 6, pp.151-180, 1998.

[14] J. Balasubramaniyan, J. Fernandezm, D. Isacoff D and E. Spafford, “An architecture for intrusion detection using autonomous agents,” Proceedings of the annual computer security applications conference, pp. 13-24, 1998.

[15] M. Crosbie, “Applying genetic programming to intrusion detection,” Proceedings of AAAI fall symposium series, pp. 45-52, 1995.

[16] K. Sequeira and M. Zaki, “ADMIT: anomaly-base data mining for intrusions,” Proceedings of ACM SIGKDD international conference on knowledge discovery and data mining, pp. 45-56, 2002.

[17] J. Gomez, D. Dasgupta and O. Nasraoui, “Complete expression trees for evolving fuzzy classifiers systems with genetic algorithms and application to network intrusion detection,” Proceedings of the NAFIPS-FLINT joint conference, pp. 469-474, 2002.

[18] R. Perdisci, G. Giacinto and F. Roli, “Alarm clustering for intrusion detection systems in computer networks,” Engineering Applications of Artificial Intelligence, vol. 19, pp. 429–438, 2006.

[19] M. Saniee Abadeha, J. Habibia and C. Lucas, “Intrusion detection using a fuzzy genetics-based learning algorithm,” Journal of Network and Computer Applications, vol. 30, pp. 414–428, 2007.

[20] D. Rumelhart, G. Hinton and R Williams, “Learning internal representations by back-propagating errors,” Parallel Distributed Processing: Explorations in the Microstructure of Cognition, D. Rumelhart and J. McClelland editors, vol. 1, pp. 318-362, MIT Press, 1986.

[21] V. Vapnik, The Nature of Statistical Learning Theory, Springer-Verleg, 1995.

[22] J. Burges, “A tutorial on support vector machines for pattern recognition,” Data Mining and Knowledge Discovery, vol. 2, pp. 121-167, 1998.

Alireza Osareh received the M.Sc. degree in Artificial Intelligence and Robotics from Shiraz University, Iran, in 1997 and a Ph.D. degree in Computer Science from Bristol University, UK, in 2004. He is currently an assistant professor in Computer Science Department at Shahid Chamran University, Ahvaz, Iran. His research interests include medical and biomedical engineering, machine learning and pattern recognition.

Bita Shadgar received the M.Sc. degree in Software from Ferdowsi University, Iran, in 1999 and a Ph.D. degree in Computer Science from Bristol University, UK, in 2003. She is currently an assistant professor in Computer Science Department at Shahid Chamran University, Ahvaz, Iran. Her research interests include Software engineering, databases, semantic web and grid computing.