Feature Deduction and Ensemble Design of Parallel Neural Networks for Intrusion Detection System

Syed Muhammad Aqil Burney[†], **M. Sadiq Ali Khan**[††], **Dr.Tahseen A. Jilani**[†††]

Department of Computer Science, University of Karachi, Karachi-75270

Abstract

In this modern age of computer networks, there is an ultimate demand for development of reliable, extensible, easily manageable and have low maintenance cost solutions for intrusion detection. We have used KDD'99 dataset for experimental verifications of our proposed approach. With the features reduction step, it is possible to significantly reduce the number of input features so that the chance of over-fitting and data redundancy can be reduced. Then a multilayer Perceptron neural network classifier is applied on the selected feature space using one-against-one approach. For the training of neural network, each attack is trained with the normal dataset. Thus we have four neural networks working in parallel such as normal vs. probe, normal vs. DoS, normal vs. U2R and normal vs. R2L. After repeated simulations and bootstrapping, we have shown that our proposed approach has good results for Probe, DoS and R2L attacks and average results for U2R attack. A comparison with other intrusion detection systems is also presented.

Key words:

Intrusion Detection System (IDS), KDD-cup dataset, principal component analysis (PCA), neural networks (NN), ensemble of one-against-one approach

1. Introduction

Intrusion Detection systems are increasingly a key part of system defense both for protection of government organizations and industry computing infrastructures. In this modern age of computer networks, there is an ultimate demand for development of reliable, extensible, easily manageable and have low maintenance cost solutions for intrusion detection. In the last few years, machine learning algorithms have proven their significant results with high accuracies in many parts of intrusion detection.

1.1 Intrusion Detection Systems

The job of scrutinizing traffic and looking for distrustful or cynical action is performed by network intrusion detection system (NIDS). Intrusion detection system which is also referred to as 'intelligent firewalls', makes use of artificial intelligence, machine learning and data mining etc in order to keep an eye on patterns in network action and behaviors [11](P. Garcı´a-Teodoroa, J. Dı´az-Verdejoa, G. Macia´-Ferna´ndeza, E. Va´zquezb (2009)). Supervising the network traffic and checking for any mistrustful action can be done by an intrusion detection system (IDS). This system is proficient to vigilant the system or network supervisor. By taking certain actions, IDS is also able to reply to inconsistent or malevolent traffic, for example, congestion the user or internet protocol address of source by getting into the network. Such systems are termed as Intrusion Prevention systems (IPS). Server allocated for NIDS is capable to inspect files of the system and searches for illegitimate actions. The server is also able to inspect log files, uphold data and veracity of files as well. If any changes are made in the interior working of the server, the NIDS server is proficient to notice such changes. The chief security measures like firewalls, encryption, and several different verification approaches are not substituted by NIDS [12] (Matt Bishop (2003)).

Large diversity is found in intrusion detection system and numerous different techniques are available which can perceive Distrustful traffic. The three types of IDS are

- Network based intrusion detection system (NIDS)
- Host based intrusion detection system (HIDS)
- Distributed intrusion detection systems (DIDS)

Network based (NIDS) and host based intrusion detection systems (HIDS) are the two foremost classes of IDS. Some IDS are able to identify interruption by observing particular signatures of familiar threads, just like any antivirus software that identifies and guards against any malicious software. In addition, there are IDS which are able to identify disturbance by judging traffic designs opposed to a baseline and come across inconsistencies. Some IDS just do observation and make the supervisors attentive, while some IDS carry out actions in reply to a perceived threat see [10] Srilatha Chebrolua, Ajith Abrahama, Johnson P. Thomasa (2005). Host Intrusion Detection Systems (HIDS) are executed on each host on the set of connections. File system examining, log file supervisors, connection investigator and kernel based IDSs are the four main types of HIDS. The four diverse attributes of the above types of HIDS are features, unproblematic installation and support, approaches for dodging the IDS and tone down the causes of prevarication attempts by modifying the ways of implementations [17] [Network Security Architectures, 2004]. Signature based IDS searches for actions that seems equivalent to a predefined mold. This approach is not capable to notice

Manuscript received October 5, 2010
Manuscript revised October 20, 2010

new damages, whose signatures are unfamiliar. On the other hand, to detect attacks, signature based approaches are effective as they are devoid of numerous fake warnings [18][Wm. Arthur Conklin et. al. 2005].

1.2 Machine Learning and Intrusion Detection Systems

Machine learning (ML) permits our computers to be trained or learned itself through supervised or unsupervised approach. Apart from many other qualities which make machine learning superior, it can also be used to discover patterns of malevolent activities. Both for host based and network based intrusion detection. ML algorithms give improved precision along with minimum rate of bogus alarms, feasible performance enhancements and rapid event response time [19][Vojislav Kecman, 2001]. In the area of intrusion detection, the involvement of machine learning algorithms is not new-fangled at all. From both, training data and feature selection improving classification, the ML techniques can find out regular/irregular or consistent/ inconsistent patterns. Searching for the subset of features that best classifies the training data in order to identify attacks on computer systems is done for the improved classification of selected features. In order to get rid of redundancy and inappropriate features, a large number of attribute selection techniques have been set up. The reason for doing this is that correctness or strength of classification can be trimmed down by unprocessed features [1] M. Bahrololum, E. Salahi, M. Khaleghi (2009)]

There are several approaches for solving intrusion detection problems. [15] Yamada et al (2006) worked on an anomaly recognition system in which training data preparation was not needed. Refined training data was generated by their system that was appropriate to the learning by giving out warnings as a signature based IDS yields. The 1999 DARPA IDS estimation data and the protection scanner data were the two kinds of traffic that were exercised for the appraisal of the system. In order to scrutinize, network traffic traced from three sources, [16] Gunes et al(2005) worked on clustering and neural network algorithm. Two out of three traffic sources were unreal. This refers that the network traffic was producing in a restricted setting for infringement identification benchmarking. To make distinguish among unreal or artificial and real traffic, this research was carried out. [2] Xuren, Famei and Rongsheng (2006) presented an improved association rule discovering system under rough set theory framework of modeling IDS. The system makes association rule applicable in classifying fields. Compared to the best results of KDD'99 Contest [3] [W. Lee, S. J. Stolfo, and K. Mok, 1999] and [4][http://www.cs.ucsd.edu/users /elkan/ clresults.html, 1999], 99.50%, 83.32%, 97.12%, 13.16% and 08.40% of detecting Normal, Probing, DoS, U2R and R2L data respectively, their best detection accuracy, 99.58%, 74.89%, 96.83%, 3.8%, and 7.99% respectively. [5] Pan, Lian, Hu, and Ni (2005), presented an ID model based on neural network and expert system. The key idea is to aim at taking advantage of classification abilities of neural network for unknown attacks and the expert-based system for the known attacks using KDD'99 dataset. The experimental results for DoS and Probe are 96.6 percent, and less than 0.04 percent false alarm rate. [6] Faizal M. A., Mohd Zaki M., Shahrin S., Robiah Y, Siti Rahayu S., Nazrulazhar B. (2009), discussed a new technique for selecting static threshold value from a minimum standard features in detecting fast attack from the victim perspective. Guisong Liu and Xiaobin Wang (2008) [7] presented an integrated IDS scheme based on multiple neural networks. The approaches used in IIDS include principal component neural networks, growing neural gas networks and principal component self organizing map networks. By the abilities of classification and clustering analysis of the above methods, IIDS can be adapted to both anomaly and misuse detections for intrusive outsiders. Therefore, IIDS is able to detect the intrusions/attacks both from the outer Internet and an inner LAN. Experiments are carried out to illustrate the performance of the proposed IDS by using the KDD CUP 1999 Intrusion Detection Evaluation dataset. Alireza Osareh, Bita Shadgar (2008) [9], compared efficiency of ML methods in IDS, including ANN and support vector machine. Compared with other related works in ML based IDSs, they proposed 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. They compare the accuracy, detection rate, false alarm rate for 4 attack types. The extensive experimental results on the KDD-cup dataset demonstrated that their proposed approach produces higher accuracy, especially for U2R and R2L attacks. [8] S. Selvakani Kandeeban and Rengan S Rajesh (2010) investigated genetic algorithms and neural networks to model fast and efficient Intrusion Detection Systems. Using Genetic algorithm only eight of the input features were used for the rule generation for attack classification. The model was verified on KDD99 demonstrating higher detection rates than those reported by the state of art while maintaining low false positive rate.

2. KDD-cup Data set

In the 1998 DARPA KDD-cup dataset [20] (R. Perdisci, g. Giacinto and F. Roli (2007)) 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 494022 records was used, of which about 20% represent normal patterns (Table 1). The four different categories of attack patterns are as follows. It is important to mention that in this paper, we have demonstrated the capability of the suggested learning method to detect abnormal behaviors via normal behaviors. The four types of attacks are [9][Aliraza and Shadgar (2008)]

Probing: Probing is a class of attacks where an attacker scans a network to gather information or find known vulnerabilities. An attacker can use the information about map of machines and available services for exploitation in a network. Some of the probe types abuse the computer's genuine features and some of them use social engineering techniques.

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 miss-configurations. DoS attacks are classified based on the services that an attacker makes unavailable to legal users.

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. Some of the common attacks of this type are regular buffer overflows.

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.

Table 1: Training (70%) and testing (30%) Dataset

Class	Class name	No. of Instances	%
0	Normal	97278	19.69
1	Probe	4107	0.83
2	DoS	391459	79.23
3	U2R	52	0.011
4	R2L	1126	0.22
Total		**494022**	100

In this work, machine learning algorithm i.e. neural network (NN) is tested against the KDD 10% 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.

Principal Component Analysis

Dimension of a huge data set can be trimmed down by using principal component analysis which is considered as one of the most prevalent and useful statistical method. This method transforms the original data in to new dimensions. The new variables are formed by taking linear combinations of the original variables of the form:

$$\mathbf{H_1} = b_1'K = b_{11}K_1 + b_{12}K_2 + \cdots + b_{1m}K_m$$
$$\mathbf{H_2} = b_2'K = b_{21}K_1 + b_{22}K_2 + \cdots + b_{2m}K_m$$
$$\ldots\ldots$$
$$\mathbf{H_p} = b_p'K = b_{p1}K_1 + b_{p2}K_2 + \cdots + b_{pm}K_m$$

In matrix style, we can write $\mathbf{H = B.K}$, where $b_{11}, b_{12}, \ldots, b_{pp}$ are known as the loading parameters. The new axes are attuned such that they are orthogonal to one another with utmost expand of information.

$$Var(\mathbf{H}_i) = b_i' \sum b_i \quad , \quad i = 1,2,\ldots,p$$

$$Cov(\mathbf{H}_i, \mathbf{H}_j) = b_i' \sum b_j \quad , \quad i = 1,2,\ldots,p$$

K_1 is the first principal component holding the prime variance. As the direct computation of matrix B is not achievable. So, in feature transformation, the first step is to ascertain the covariance matrix U which can be expressed as

$$\mathbf{U}_{m\times n} = \frac{1}{m-1}\left[\sum_{i=1}^{m}(K_i - \overline{K})'.(K_i - \overline{K})\right],$$

where $\overline{K} = \left(\frac{1}{m}\right)\sum_{i=1}^{m} X_i$

The next step is to determine the eigen values for the covariance matrix 'U'. Eventually, a linear transformation is defined by n eigen vectors match up to n eigen values from a m-dimensional space to n-dimensional space (n<m). Principal axes are also referred to as eigen vectors $E_1, E_2, \ldots, E_m$ correspond to eigen values $\lambda_1 + \lambda_2 + \cdots + \lambda_n$. Generally, the first few principal components hold most of the information. Analysis of variances' proportion represents the total number of principal components that should be retained from the dataset [21] (Kantardzic, 2005). In this paper, we have retained only those principle components that retain above 99% of the total data variability.

4. Neural Networks

In machine learning algorithms, artificial neural networks are simulated natural machines that can perform a number of different operations that range from vision, hearing to sensing. They perform learning through experience using a highly complex, nonlinear and parallel information processing system (Rumelhart D., G. Hinton and R Williams, 1986)[23]. It has the capability to organize its structural constituents, known as neurons, so as to perform certain computations like pattern recognition, pattern matching, classification and forecasting. Neural networks form a parallel and distributed structure. A neuron is the fundamental information-processing unit of a NN that consists of, a set of synaptic links, an adder and an activation function. Mathematically, the function of kth neuron in a neural network can be defined as

$$u_k = \sum_{j=0}^{m} w_{kj}\, x_i \ \text{with} x_0 = 1, b_k = w_{k0}$$

$$\text{and} \quad y_k = f(u_k)$$

where $x_1, x_2, .., x_m$ are the input signals and $W_{k1}, W_{k2}, .., W_{km}$ are the synaptic weighs of neuron k; u_k is the linear combination output due to the input signals; b_k is the bias parameter; $f(.)$ is the activation function; and y_k is the output signal of the neuron. The bias b_k is an external parameter and shows an affine transformation to the output u_k .

4.1 Error-correction learning

Consider the simple case of a neuron k constituting the only computational node in the output layer of a feed forward neural network. Neuron k is driven by a signal vector $x(n)$ produced by one or more layers of hidden neurons, which are themselves driven by an input vector (stimulus), applied to the input layer (i.e. source node) of the neural network. The argument n denotes discrete time, or more precisely, the time step of an iterative process involved in adjusting the synaptic weights of neuron k. The output signal of neuron k is denoted by $y_k(n)$. This output signal, representing the only output of the neural network, is compared to a desired response or target output, denoted by $d_k(n)$. Consequently, an error signal, denoted by $e_k(n)$ is produced, given by

$$e_k(n) = d_k(n) - y_k(n)$$

The step-by-step adjustments to the synaptic weights of neuron k are continued until the system reaches a steady state. At that point the learning process is terminated.

In particular, minimization of the cost function leads to a learning rule commonly referred to as the delta rule or Widrow-Hoff rule, named in honor of its originators (Kecmann, 2001) [19]. Let $w_{kj}(n)$ denote the value of synaptic weight of output neuron k excited by element $x_j(n)$ of the signal vector $x(n)$ at time step n. According to the delta rule, the adjustment

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$

where η is the learning rate parameter. Then the update value of synaptic weight is $w_{kj}(n)$ determined by

$$W_{kj}(n+1) = W_{kj}(n) + \Delta W_{kj}(n)$$

4.2 Multilayer Perceptron Neural Network (MPNN)

In a MPNN, hidden neurons play a critical role in the operations of a multilayer Perceptron with back-propagation learning because they act as feature detectors. As the learning process progresses, the hidden neurons begin to gradually discover the salient features that characterize the training data through a nonlinear transformation on the input data onto a new space called the hidden space, or feature space. In the new space, the classes of interest in a pattern-classification task may be more easily separated from each other than in the original space. The true-negative and false-positive rates determine the accuracy of a MPNN model.

4.3 Back propagation Algorithm

The error signal at the output neuron j at iteration n (i.e., presentation of the nth training example) is defined as

$$e_j(n) = d_j(n) - y_j(n)$$

We define the instantaneous value of the error signal for neuron j as $\frac{1}{2} e_j^2(n)$. Correspondingly, the instantaneous value $E(n)$ of the total energy is obtained by

$$E(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n)$$

where the set C includes all the neurons in the output layer of the network. If there are N training input patterns then the average squared error energy is obtained

$$E_{av} = \frac{1}{N} \sum_{n=1}^{N} E(n)$$

where $\frac{1}{N}$ is the normalizing factor. The objective of the study is to minimize E_{av} using LMS algorithm. Network free parameters are adjusted on a pattern-by-pattern basis until one epoch that is complete presentation of the entire set, has been dealt with. The adjustments to the weights are made in accordance with the respective error computed for each pattern presented to the network. Based on LMS

method, the back-propagation formula for the local gradient $\delta_k(n)$ is described:

$$\delta_k(n) = f_j'(u_j(n)).\sum_k \delta_k(n).w_{kj}(n) \quad (1)$$

where neuron j is hidden neuron. The factor $f_j'(u_j(n))$ involved in the computations of induced local field in **(1)**, depends solely on the activation function associated with the hidden neuron j . Thus the general expression of back-propagation algorithm is:

$$\text{Weight Correction}\Delta W_{ji}(n) =$$

$$(\text{learning rate parameter } \eta).(\text{local gradient}) *$$

$$(\text{input signal of neuron} j y_i(n))$$

5. Analysis and Evaluation

5.1 Data Preprocessing

Attributes in the KDD datasets had all forms - continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most of the classification algorithms are not able to handle both continuous and symbolic data at a time. Hence preprocessing was required before pattern classification models could be built. Preprocessing consisted of two steps: first step involved mapping symbolic-valued attributes to numeric-valued attributes and second step implemented scaling. Attack names (like buffer_overflow, guess_passwd, etc.) were first mapped to one of the five classes, 0 for Normal, 1 for Probe, 2 for DoS, 3 for U2R, and 4 for R2L. The categorical features like protocol_type, service and flag were mapped to integer values ranging from 0 to n-1 where n is the number of categories in a feature. Then each of these features was linearly scaled to the range [0.0, 1.0]. Features having integer value ranges like duration, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, su_attempted, num_root, num_file_creations, num_shells, num_access_files, count, srv_count, dst_host_count, and dst_host_srv_count were also scaled linearly to the range [0.0, 1.0]. Two features spanned over a very large integer range, namely src_bytes and dst_bytes. Logarithmic scaling (with base 10) was applied to these features to reduce the range to [0.0, 9.14]. All other features were either Boolean, like logged_in, having values (0 or 1), or continuous, like diff_srv_rate, in the range [0.0, 1.0]. Hence scaling was not necessary for these features. The software tool SPSS-17 is used to simulate pattern recognition and machine learning model. All simulations were performed on dual microprocessors, running at 2.00 GHz with 2048MB of RAM and WINDOWS XP operating system.

5.2 Data Reduction Step

Using principle component analysis, the preprocessed database is processed through PCA to reduce the redundancy by removing features having very small contribution in the total variability of the dataset. We have selected a threshold of 0.5 for change of eigenvalue in PCA. Using normal vs. attack approach, we have applied PCA for each case. The following table shows the feature reduced from the database for each case alongwith the standard deviation for each feature.

5.3 Neural Network Simulation

Multilayer Perceptron (MLP) is one of most commonly used neural network classification algorithms. The architecture used for the MLP during simulations with KDD dataset consisted of a three layer feed-forward neural network: one input, one hidden, and one output layers. Unipolar sigmoid transfer functions were used for each neuron in the hidden layer and threshold activation function in the output layer. The learning algorithm used was stochastic gradient descent with mean squared error function. Based on features extraction through PCA, we have different number of neurons in the input, hidden layer and 2 neurons (normal or attack) in the output layer. Multiple simulations were performed with number of hidden layer nodes varying from 10 to 50 in increments of 5. Further for each simulation a constant learning rate (one of the four values 0.1, 0.2, 0.3, and 0.4) was used along with 0.6 as the weight change momentum value. Different simulations had different learning rates that varied from 0.1 to 0.4 in steps of 0.1.

Randomly selected initial weights were used that were uniformly distributed in the range [-0.1, 0.1]. Initially a total of 5000 epochs were performed on the training dataset. Other simulations studied the effect of changing the number of training epochs: number of epochs was varied to 500, 1000, 2000 and 5000.

This work intends to apply PCA based ANN against KDD-cup dataset. This dataset is over large and various data is distributed unevenly. Therefore, this research work will sample training dataset for both training and testing purposes. 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. We have applied one-against-one approach for PCA-ANN based IDS. The input data is fed to four neural networks namely normal vs. Probe, normal vs. DoS, normal vs. U2R and normal vs. R2L.

Table 2: Accuracy comparison of NN classifiers against 4 kinds of attacks

	Normal vs. Probe		**Normal vs. DoS**		**Normal vs. U2R**		**Normal vs. R2L**	
	Feature Name	S.D.	Feature Name	S.D.	Feature Name	S.D.	Feature Name	S.D.
1	A	0.003	A	0.007	A	0.003	A	0.003
2	B	0.000	C	0.004	B	0	B	0.000
3	C	0.01	D	0.009	C	0.01	C	0.012
4	D	0.021	E	0.007	D	0.021	D	0.035
5	E	0.014	F	0.008	E	0.022	E	0.017
6	F	0.017	G	0.009	F	0.017	F	0.017
7	G	0.021	I	0.036	G	0.024	G	0.022
8	I	0.081	J	0.000	I	0.081	I	0.082
9	J	0.000	K	0.000	J	0	J	0.000
10	K	0.000	L	0.028	K	0	K	0.000
11	L	0.06	S	0.06	L	0.062	L	0.083
12	M	0.047	O	0.026	M	0.028	M	0.03
13	N	0.061	R	0.091	N	0.027	N	0.028
14	O	0.086			T	0.092	U	0.092
15	P	0.047			O	0.050	O	0.051
16	Q	0.057			P	0.029	P	0.031
17					Q	0.016	Q	0.018
Features Reduction (in %)	**39%**		**32%**			**41%**		**41%**

wrong_fragment =B; Urgent=C; num_failed_logins= D; root_shell=E; su_attempted= F ; num_shells=G; num_access_files=I ; num_outbound_cmds=J; is_host_login=K ; is_guest_login=L; serror_rate=M; srv_serror_rate=N; dst_host_srv_diff_host_rate=O; dst_host_serror_rate=P; dst_host_srv_serror_rate=Q; num_file_creations=R; diff_srv_rate=S; same_srv_rate=T; same_srv_rate=U

In the testing phase, if the test data is a attack then it will produce '1' at any one of the four neural networks placed in parallel. This ensemble approach will finally produce a soft-max output from the four parallel neural networks so that a decision surface is formed between attacks and non-attacks data packets. 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. Detection rate refers to the proportion of attack detected among all attack data, namely, the situation of TP. *False alarm rate* refers to the proportion that normal data is falsely detected as attack behavior, namely, the situation of FP. Table 3-5 summarizes the results of accuracy rate, detection rate and false alarm rate.

$$\text{Accuracy} = \left(\frac{TP + TN}{TP + TN + FP + FN}\right) \times 100\%$$

$$\text{Detection Rate} = \left(\frac{TP}{TP + FN}\right) \times 100\%$$

$$\text{False Alarm Rate} = \left(\frac{FP}{FP + TN}\right) \times 100\%$$

Table 3: Accuracy results (in %) of PCA-ANN classifier against KDD-cup dataset

Percentage of normal data	Probe	DoS	U2R	R2L
10	99.86%	99.71%	100%	99.45%
20	99.96%	99.82%	100%	99.52%
30	99.96%	99.83%	100%	99.55%
40	99.94%	99.80%	100%	99.63%
50	99.91%	99.99%	100%	99.77%
60	99.83%	99.82%	100%	99.82%
70	99.81%	99.60%	67%	99.81%
80	99.17%	99.78%	100%	99.85%
90	99.69%	99.65%	100%	99.97%
Average (overall)	**99.79%**	**99.78%**	**96.21%**	**99.71%**

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

Percentage of normal data	Probe	DoS	U2R	R2L
10	99.82%	99.57%	85.43%	99.98%
20	99.92%	99.91%	44.24%	99.74%
30	99.92%	99.46%	45.31%	99.61%
40	99.70%	99.54%	80.14%	98.86%
50	99.29%	99.71%	28.67%	96.39%
60	98.34%	99.85%	3.01%	97.98%
70	98.12%	99.57%	0.10%	96.28%
80	96.98%	99.79%	18.92%	97.46%
90	95.21%	99.75%	3.03%	93.77%
Average (overall)	**98.59%**	**99.68%**	**34.32%**	**97.79%**

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

Percentage of normal data	Probe	DoS	U2R	R2L
10	0.11%	1.94%	0.03%	0.42%
20	0.03%	1.71%	0.01%	0.00%
30	0.02%	1.52%	0.00%	0.01%
40	0.03%	1.46%	0.02%	0.01%
50	0.03%	0.00%	0.01%	0.07%
60	0.11%	0.31%	0.00%	0.19%
70	0.10%	0.11%	3.33%	0.25%
80	0.20%	0.21%	0.02%	0.22%
90	0.10%	0.47%	0.00%	0.02%
Average (overall)	**0.08%**	**0.86%**	**0.38%**	**0.13%**

Table 6: Accuracy, Detection Rate and False Alarm rate results (in percentage) of PCA-ANN classifier against KDD-cup dataset

Percentage of normal data	Probe	DoS	U2R	R2L
Accuracy Rate	99.79%	99.78%	96.21%	99.71%
Detection Rate	98.59%	99.68%	34.32%	97.79%
False Alarm Rate	0.08%	0.86%	0.38%	0.13%

5.4 Accuracy Comparison between Different Attacks

Above 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 PCA-ANNs. In table 7, a comparison between our results and other researches is given. The detection rate for Probe and DoS are comparable with the other good results. Detection rate for U2R is very poor because only 0.01% of the total data contains U2R attacks. On the other side, R2L detection results are the best as compared to other results.

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 Aliraza and Shadgar (2008) [9]	82.5	58.6	65.4	14.6
SVM Aliraza and Shadgar (2008) [9]	83.2	62.5	65.5	14.7
Decision Tree (Peddabachigaria,2007) [11]	99.86	96.83	68	84.19
SVM (Peddabachigaria,2007)[11]	99.57	99.92	40	33.92
Hybrid Decision tree-SVM (Peddabachigaria,2007) [11]	98.57	99.92	48	37.8
Kandeeban,2010 [8]	86.1	86.7	79.2	81.2
PCA-ANN	**98.59**	**99.68**	**34.32**	**97.79**

6. Conclusion and Suggestions

6.1 Conclusion

The research work compares accuracy, detection rate, false alarm rate and accuracy of the four types of attacks under different proportion of normal information. KDD Cup 99 dataset is current benchmark dataset in intrusion detection; however, it is not evenly distributed, so error may occur if only one set is used. Therefore, following the approach of Aliraza and Shadgar (2008) [9], 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 PCA-ANN, we find that PCA-ANN is superior to Aliraza and Shadgar (2008) [9], and KDD winner results in detection; in false alarm rate and in accuracy for Probe, Dos and R2L attacks, while PCA-ANN is not working well for U2R attacks.

6.2 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.

We further aim to apply genetic algorithm and neuro-fuzzy logic for intrusion detection and intrusion prevention systems.

References

[1] M. Bahrololum, E. Salahi, M. Khaleghi: Machine Learning Techniques for Feature Reduction in Intrusion Detection Systems: A Comparison. Convergence Information Technology, Fourth International Conference on Computer Sciences and Convergence Information Technology. 2009, pp. 1091-1095

[2] Wang Xuren, He Famei, Xu Rongsheng: Modeling Intrusion Detection System by Discovering Association Rule in Rough Set Theory Framework. IEEE- International Conference on Computational Intelligence for Modeling Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC), 2006

[3] W. Lee, S. J. Stolfo, and K. Mok. Data mining in work flow environments: Experiences in intrusion detection. In Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD99), 1999

[4] Results of the KDD'99 Classifier Learning Contest, http://www.cs.ucsd.edu/users /elkan/ clresults.html, 1999

[5] Zhi Song PAN, Hong LIAN, Gu Yu HU, and Gui Qiang NI: An Integrated Model of Intrusion Detection Based on Neural Network and Expert System. Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2005

[6] Faizal M. A., Mohd Zaki M., Shahrin S., Robiah Y, Siti Rahayu S., Nazrulazhar B.: Threshold Verification Technique for Network Intrusion Detection System. (IJCSIS) International Journal of Computer Science and Information Security, Vol. 2, No. 1, 2009

[7] Guisong Liu and Xiaobin Wang: An Integrated Intrusion Detection System by Using Multiple Neural Networks, 978–1–4244–1674–5/08/$25.00 _c 2008 IEEE CIS, 2008

[8] S Selvakani Kandeeban and Rengan S Rajesh: Integrated Intrusion Detection System Using Soft Computing, International Journal of Network Security, Vol.10, No.2, 2010, pp.87-92

[9] Alireza Osareh, Bita Shadgar: Intrusion Detection in Computer Networks based on Machine Learning Algorithms. IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.11, 2008
[10] Srilatha Chebrolua, Ajith Abrahama, and Johnson P. Thomasa: Feature deduction and ensemble design of intrusion detection systems. Computers and Security, vol. 24, 2005, pp.295-307
[11] P. Garcı´a-Teodoroa, J. Dı´az-Verdejoa, G. Macia´-Ferna´ndeza, E. Va´zquezb: Anomaly-based network intrusion detection: Techniques, systems and challenges. Computer and Security, 28, 2009
[12] Matt Bishop: Computer Security: Art and Science, Pearson Education, India. 2003
[13] Sandhya Peddabachigaria, Ajith Abrahamb, Crina Grosanc, and Johnson Thomasa: Modeling intrusion detection system using hybrid intelligent systems. Journal of Network and Computer Applications, vol. 30, pp. 114-132
[14] M. Saniee Abadeh, J. Habibi and C. Lucas: Intrusion detection using a fuzzy genetics-based learning algorithm. Journal of Network and Computer Applications, Volume 30, Issue 1, January 2007, pp. 414-428
[15] A. Yamada, Y. Miyake, K. Takemori, T. Tanaka: Intrusion Detection System to Detect Variant Attacks Using Learning Algorithms with Automatic Generation of Training Data. 2006
[16] H. Güneş Kayacık, Nur Zincir-Heywood, "Analysis of Three Intrusion Detection System Benchmark Datasets Using Machine Learning Algorithms", 2005 [http://users.cs.dal.ca/~zincir/bildiri/isi05-gn.pdf]
[17] Network Security Architectures- Expert guidance on designing secure networks, Pearson Education. 2004
[18] Wm. Arthur Conklin, Dwayne Williams, Gregory B. White, Roger L. Davis, and Chuck Cothern: Principles of Computer Security, Security+ and Beyond, McGraw hill Technology Education. 2005
[19] Vojislav Kecman: Learning and Soft Computing, Support Vector Machines, Neural Networks and Fuzzy Logic Models, Pearson Education. 2001
[20] R. Perdisci, g. Giacinto and F. Roli: Alarm Clustering for Intrusion detection systems in computer networks. Engineering applications of artificial intelligence, vol. 19, (2007), pp. 429-438
[21] Kantardzic M., Data Mining: Concepts, Models, Methods, and Algorithms. John Wiley & Sons 2003, chapter 5
[22] R. Perdisci, G. Giacinto and F. Roli: Alarm clustering for intrusion detection systems in computer networks. Engineering Applications of Artificial Intelligence, vol. 19, (2006), pp. 429–438
[23] 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, MIT Press, (1986), pp. 318-362
[24] Aqil Burney S.M., Jilani A. Tahseen and Saleemi: Approximate Knowledge Extraction using MRA for TYPE-I Fuzzy Neural Networks. (2006)

Dr.S.M.Aqil Burney is the Meritorious Professor and approved Supervisor in Computer Science and Statistics by the Higher Education Commission, Govt of Pakistan. He is also the Director & Chairman of Computer Science Department, University of Karachi. He is also member of various higher academic boards of different universities of Pakistan. His research interest includes AI, Soft Computing, Neural Network, Fuzzy Logic, Data Mining, Statistics, Simulation and Stochastic Modeling of Mobile Communication system and Networks, Network Security and MIS in health services. Dr.Burney is also referee of various journals and conferences proceedings, nationally & internationally. He is member of IEEE(USA), ACM(USA) and fellow of Royal Statistical Society, United Kingdom. He has vast education management experience at the university level. Dr.Burney have been awarded best IT academician in the country in 2003 by NCR (Pak).

M.Sadiq Ali Khan received his BS & MS Degree in Computer Engineering from SSUET in 1998 and 2003 respectively. Since 2003 he is serving Computer Science Department, University of Karachi as an Assistant Professor. He has about 12 years of teaching experience and his research areas includes Data Communication & Networks, Network Security, Cryptography issues and Security in Wireless Networks. He is member of CSI, PEC and NSP.

Tahseen A. Jilani received the B.Sc., first class second M.Sc. (Statistics) and M.A. (Economics) from Karachi University in 1998, 2001 and 2003 respectively. He was awarded Ph.D in 2007. He is currently working as an Assistant Professor in Department of Computer Science University of Karachi. He is member of IEEE-Computational Intelligence Society. His research interest includes AI, neural networks, soft computing, fuzzy logic, Statistical data mining and simulation. He is teaching since 2002 in the fields of Statistics, Mathematics and Computer Science.