DOI: 10.21917/ijsc.2010.0005

INTRUSION DETECTION USING ARTIFICIAL NEURAL NETWORK WITH REDUCED INPUT FEATURES

P. Ganesh Kumar[1] and D.Devaraj[2]

[1]*Department of Information Technology, Anna University of Technology, Coimbatore, India*
E-mail: ganesh23508@gmail.com
[2] *Department of Electrical and Electronics Engineering, Kalasalingam University, India*

Abstract
Intrusion Detection is the task of detecting, preventing and possibly reacting to intrusion in a network based computer systems. This paper investigates the application of the Feed Forward Neural Network trained by Back Propagation algorithm for intrusion detection. Mutual Information based Feature Selection method is used to identify the important features of the network. The developed network can be used to identify the occurrence of various types of intrusions in the system. The performance of the proposed approach is tested using KDD Cup'99 data set available in the MIT Lincoln Labs. Simulation result shows that the proposed approach detects the intrusions accurately and is well suitable for real time applications.

Keywords:
Intrusion Detection, Artificial Neural Network, Feature Selection, Mutual Information

1. INTRODUCTION

Confidentiality, integrity and availability of the system resources are the major concerns in the development and exploitation of network based computer systems. Enlargements of computer infrastructure have raised the vulnerability of these systems to security threats, attacks and intrusions. Some specific examples of intrusions that concern system administrators are Attempted break-in, Masquerading or successful break-in, Penetration by legitimate user, Leakage by legitimate user, Inference by legitimate user, Trojan Horse, Virus and Denial-of-Service. Generally these intrusions would cause loss/damage to our system resources in terms of unauthorized modifications of system files, user files or information and any other system information in network components. Hence a system is needed that detects any unauthorized modification forced by an attacker and able to run continually with minimal human supervision.

An intrusion detection system (IDS) is one that inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. According to the detection principles there are two types of intrusion detection system: Misuse and Anomaly detection. In Misuse detection, attack patterns or the behavior of the intruder is modeled (attack signature is modeled). Here the system will signal the intrusion once a match is detected. In Anomaly detection system, the normal behavior of the system is modeled and the system will raise an alarm once the behavior of the network does not match with its normal behavior. According to the source of data, there are two types of intrusion detection: Network-based IDS (NIDS) and Host-based IDS (HIDS). A network based IDS captures all network traffic and analyzes the content of individual packets for malicious traffic where as a host-based IDS identifies intrusions by analyzing system calls, application logs, file system modifications (binaries, password files, capability/acl databases) and other host activities and state.

In the literature Statistical Techniques like Hidden Markov Model [1], Multivariate Adaptive Regression Splines [2], Bayesian Network and Classification and Regression Trees (CART) [3] have been applied to Intrusion detection. These statistical approaches usually results in an inflexible detection system that is unable to detect an attack if the sequence of events slightly different from the predefined profile. Rule-based systems have been proposed by Denning [4] for intrusion detection. Expert systems are the most common form of rule-based approaches. They permit the incorporation of an extensive amount of human expertise into a computer application that then utilizes that knowledge to identify activities that match the defined characteristics of misuse and attack. The constantly changing nature of network attacks requires a flexible defensive system that is capable of analyzing the enormous amount of network traffic in a manner which is less structured than rule-based systems. In [5] fuzzy logic approach has been combined with data mining techniques for discovering association rules which can be applied for detecting intrusions.

Recently, Artificial Neural Networks have been successfully applied for developing the IDS. ANN has the advantage of easier representation of nonlinear relationship between input and output and is inherent by fast. Even if the data were incomplete or distorted, a neural network would be capable of analyzing the data from a network. A Multilayer Perceptron (MLP) was used in [6] for misuse detection with a single hidden layer. A Similar approach was applied in [7] but generic keywords were selected to detect the attack preparations and actions after the break-in. The weakness of neural network based approaches is that if the dimension of the input data is very large then it is difficult for it to interpret the relationship between inputs and outputs.

Clustering can be performed to find hidden patterns in data and significant features for use in detection. Self-Organizing Map was applied to perform the clustering of network traffic and to detect attacks in [8]. A hybrid model of the SOM and the MLP was proposed in [9] to detect the dispersing and possibly collaborative attacks. In [10], the self-organizing map was combined with the Resilient Propagation Neural Network (RPROP) for visualizing and classifying intrusion and normal patterns. If the system is complex and input features are numerous, clustering the events can be a very time consuming task. Feature extraction methods like Principal Component Analysis (PCA) or Singular Value Decomposition (SVD) [11] can be an alternative solution but extraction of features will lead to a less accurate detection model.

Recently Feature selection is found to be more relevant to Intrusion detection System since the selected features retain their physical interpretation. In [12], a trial and error approach is employed for feature selection by deleting one feature at a time.

Each reduced feature set was then tested on Support Vector Machines and Neural Networks to rank the importance of input features. The reduced feature set that yielded the best detection rate in the experiments was considered to be the set of important features. Bayesian networks used in [3] not only classify the data, but also select features based on the Markov blanket of the target variables. The CART algorithm proposed in [3] classifies data by constructing a decision tree and identifies the important features based on predictor ranking (variable importance).

In general if a model which captures the relationship between different features or between different attacks and features the intrusion detection process would be simple and straightforward. In this paper we reported a Mutual Information [13] based Technique for selecting the important features and it is used as the input for a simple feed forward neural network trained by back propagation algorithm for detecting intrusions. Since the mutual information measures the arbitrary dependencies between random variables and is independent of the coordinates chosen they seem to be an appropriate approach for feature selection in ANN based Intrusion Detection.

The rest of this paper is organized as follows. In section 2, we give a brief description about the proposed model for intrusion detection. In section 3, we give a brief introduction about artificial neural network. Section 4 discusses the mutual information based feature selection for ANN based Intrusion Detection. In section 5, we present the simulation result for the developed two different ANN models, one with all features and another with selected features. Finally we present our conclusion in section 6.

2. PROPOSED MODEL FOR INTRUSION DETECTION

The proposed methodology for Intrusion Detection in Computer Networks is based on using Artificial Neural Network (ANN) for detecting the Normal and Abnormal conditions of the given parameters, which leads to various attacks. The neural network approach for this purpose has two phases; training and testing. During the training phase, neural network is trained to capture the underlying relationship between the chosen inputs and outputs. After training, the networks are tested with a test data set, which was not used for training. Once the networks are trained and tested, they are ready for detecting the intrusions at different operating conditions. The following issues are to be addressed while developing an ANN for Intrusion Detection [14]:

1. Data Collection
2. Data preprocessing, representation and Normalization
3. Dimensionality Reduction
4. Selection of Network Structure
5. Network Training and Testing

Fig.1 shows the schematic representation of the issues to be addressed while developing an ANN model for Intrusion Detection.

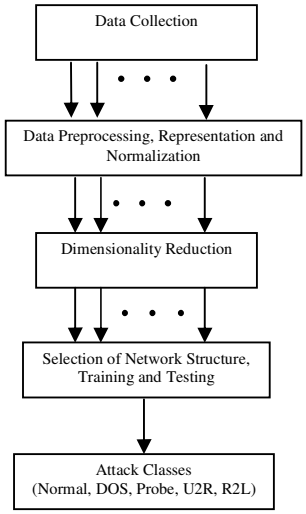

Fig.1. Schematic Representation of the proposed ANN Model for IDS

2.1 DATA COLLECTION

There are two ways to build IDS, one is to create our own simulation network, and collect relevant data and the other one is by using previously collected datasets. Issues like privacy, security, and completeness greatly restrict people from generating data. The advantage of using previously collected datasets is that the results can be compared with others in the literature. Some of the popularly used IDS datasets [15] are DARPA 1998 data set, DARPA 1999 data set and KDD Cup 1999 data set which are available in the MIT Lincoln Labs. In this work, we use KDD Cup 1999 data set for developing the IDS.

2.2 DATA PREPROCESSING, REPRESENTATION AND NORMALIZATION

Before training the neural network, the dataset should be preprocessed to remove the redundancy present in the data and the non-numerical attributes should be represented in numerical form suitably. During training of the neural network, higher valued input variables may tend to suppress the influence of smaller ones. Also, if the raw data is directly applied to the network, there is a risk of the simulated neurons reaching the saturated conditions. If the neurons get saturated, then the changes in the input value will produce a very small change or no change in the output value. This affects the network training to a great extent. To minimize the effects of magnitudes among inputs as well as to prevent saturation of the neuron activation function, the input data is normalized before being presented to the neural network. One way to normalize the data x is by using the expression:

$$x_n = \frac{(x - x_{\min}) \times range}{(x_{\max} - x_{\min})} + \text{starting value} \qquad (1)$$

where, x_n is the normalized value and $x_{\min}$ *and* $x_{\max}$ are the minimum and maximum values of the data.

2.3 DIMENSIONALITY REDUCTION

The ability to scale neural network applications to realistic dimension of intrusion detection problem is a major issue. The amount of audit data that an IDS needs to examine is very large and contains more number of variables even for a small network. If all the measured variables are used as inputs to neural network, it results in large size of the network and hence larger training time. To make the neural network approach applicable to large scale intrusion detection problems, some dimensionality reduction is mandatory. Also, networks involving too many input variables suffer from curse of dimensionality. A network with fewer inputs has fewer adaptive parameters to be determined, and these are more likely to be properly constrained by a data set of limited size, leading to a network with better generalization properties. There are two approaches to achieve dimensionality reduction: Feature Extraction and Feature Selection. In this work, feature selection is used for dimensionality reduction

2.4 SELECTION OF NETWORK STRUCTURE, NETWORK TRAINING AND TESTING

To make a Neural Network to perform some specific task, one must choose number of input neurons, output neurons, hidden neurons and how the neurons are connected to one another. For the best network performance, an optimal number of hidden-units must be properly determined using the trial and error procedure. The hidden layer neurons have tangent hyperbolic function as the activation function and the output have linear activation function. Once the appropriate structures of the network are selected, the ANN model is trained to capture the underlying relationship between the input and output using the training data. In this work, Back propagation algorithm is used to train the network, which propagates the error from the output layer to the hidden layer to update the weight matrix. After training, the networks are tested with the test data set to assess the generalization capability of the developed network.

3. REVIEW OF ARTIFICIAL NEURAL NETWORK

Artificial Neural Networks [16] can be viewed as parallel and distributed processing systems which consists of a huge number of simple and massively connected processors. The MLP architecture is the most popular paradigm of artificial neural networks in use today. Fig.1 shows a standard multilayer feed forward network with three layers. The neural network architecture in this class shares a common feature that all neurons in a layer are connected to all neurons in adjacent layers through unidirectional branches. That is, the branches and links can only broadcast information in one direction, that is, the "forward direction". The branches have associated weights that can be adjusted according to a defined learning rule.

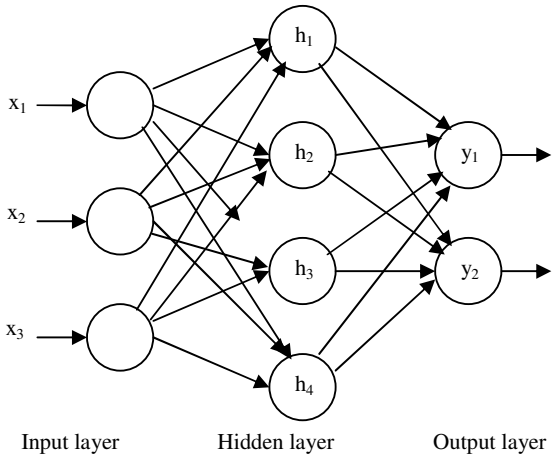

Fig.2. Architecture of feed forward neural network

Feed forward neural network training is usually carried out using the called back propagation algorithm. Training the network with back propagation algorithm results in a non-linear mapping between the input and output variables. Thus, given the input/output pairs, the network can have its weights adjusted by the back propagation algorithm to capture the non-linear relationship. After training, the networks with fixed weights can provide the output for the given input.

The standard back propagation algorithm for training the network is based on the minimization of an energy function representing the instantaneous error. In other words, we desire to minimize a function defined as

$$E(m) = \frac{1}{2}\sum_{q=1}^{q}\left[d_q - y_q\right]^2 \qquad (2)$$

where d_q represents the desired network output for the q^{th} input pattern and y_q is the actual output of the neural network. Each weight is changed according to the rule:

$$\Delta w_{ij} = -k\frac{dE}{dw_{ij}} \qquad (3)$$

where, k is a constant of proportionality, E is the error function and w_{ij} represents the weights of the connection between neuron *j* and neuron *i*. The weight adjustment process is repeated until the difference between the node output and actual output are within some acceptable tolerance.

4. FEATURE SELECTION FOR ANN - BASED INTRUSION DETECTION

Feature selection improves classification by searching for the subset of features, which best classifies the training data. Feature selection leads to savings in measurement cost and the selected features retain their original physical interpretation. Hence, feature selection is more relevant to Intrusion Detection System. The application of "mutual information" [17] between the input variables and the output provides the basis for feature selection. If the information regarding a certain system variable results in significant reduction in the system entropy, then this variable must have significant impact on the task of detecting attacks. Therefore, this variable will be selected as a feature for intrusion

detection. On the other hand, the system variables which result in minor reduction in the system entropy will be regarded as having minor effects on the task of detecting attacks and will not be selected as feature.

Once the mutual information value of input variables is evaluated, the variables are ranked, with the variable having the high mutual information value at the top and so on. The optimum number of features can be selected by consequent training of the neural networks using a progressively increasing number of features until the minimum required accuracy is obtained.

4.1 DEFINITION OF MUTUAL INFORMATION

Consider a stochastic system with input X and output Y. Let the discrete variable X has N_x possible values and Y has N_y possible values. Now the initial uncertainty about Y is given by the entropy H(Y) which is defined as,

$$H(Y) = -\sum_{j=1}^{Ny} P(y_j).\log(P(y_i)) \quad (4)$$

Where $P(y_j)$ are the probabilities for the different values of Y. The amount of uncertainty remaining about the system output Y after knowing the input X is given by the conditional entropy H(Y/X) which is defined as,

$$H(Y/X) = -\sum_{i=1}^{N_x} P(x_i)\left(\sum_{j=1}^{N_y} P(y_j|x_i).\log\left(P(y_j|x_i)\right)\right) \quad (5)$$

Where $P(y_j|x_i)$ is the conditional probability for output y_j given the input vector x_i. Now the difference $H(Y)-H(Y|X)$ represents the uncertainty about the system output that is resolved by knowing the input. This quantity is called the mutual information between the random variables X and Y. Denoting it by I(Y;X), we may thus write,

$$I(Y;X) = H(Y) - H(Y|X) \quad (6)$$

The mutual information is therefore the amount by which the knowledge provided by X decreases the average uncertainty about the random experiment represented by the variable Y. Mutual information is a symmetrical measure. That is, the amount of information gained about Y after observing X is equal to the amount of information gained about X after observing Y. For the intrusion detection problem under consideration, X corresponds to the set of input features and Y corresponds to the intrusions that may indicate normal or abnormal behavior.

4.2 MUTUAL INFORMATION FOR FEATURE SELECTION

For feature selection first the mutual information between each variable and the model output is calculated using (4)-(6). If a variable has high value of mutual information with respect to the output, then this variable must have significant effect on the output value which is to be estimated. Therefore, this variable is selected as a feature for the neural network. On the other hand, those variables which have low values of mutual information will be regarded as having minor effects on the output and are not selected for network training. Next, the mutual information among the selected input variables is calculated. If any two input variables have high value of mutual information between them, then they will have similar effect on the output and hence one is considered for network training discarding the other one.

5. SIMULATION RESULT

This section presents the details of the simulation study carried out on KDD Cup 1999 Dataset [18] using the proposed method. This data set was collected by simulating a typical U.S Air force local area network (LAN), operated like a real environment and being blasted with multiple attacks. Each KDD records contains 41 input features which is given in table 1 and one output that is labeled as either normal or as an attack, with exactly one specific attack type (DOS, Probe, U2R, R2L).

The 41 input features are divided into four feature subsets. They are Basic or Intrinsic features, Content features, Time-based features and Host-based features. Basic features are features to every network connection like duration of connection, service requested, bytes transferred between source and destination machine, etc. Content features are collected by using domain knowledge of U2R and R2L attacks since these attack categories did not contain any frequently occurring patterns. E.g. logged in flag, number of failed logins, number of root commands, number of compromised conditions, hot indicators, etc. Time-based features are collected by observing various connections in “two-second” time window with respect to current connection. E.g. SYN error rates, Rejection rates, number of different services requested etc. Host based features are collected based on the past 100 connections similar to the one under consideration.

Table.1 Detail of the Input Features

Label	Feature Name
F1	duration
F2	protocol-type
F3	service
F4	flag
F5	src_bytes
F6	dst_bytes
F7	land
F8	wrong_
F9	urgent
F10	hot
F11	num_failed_logins
F12	logged_in
F13	num_compromised
F14	root_shell
F15	su_attempted
F16	num_root
F17	num_file_creations
F18	num_shells
F19	num_access_files
F20	num_outbound_cmds
F21	is_host_login
F22	is_guest_login
F23	count
F24	srv_count

F25	serror_rate
F26	srv_serror_rate
F27	rerror_rate
F28	srv_rerror_rate
F29	same_srv_rate
F30	diff_srv_rate
F31	srv_diff_host_rate
F32	dst_host_count
F33	dst_host_srv_count
F34	dst_host_same_srv_rate
F35	dst_host_diff_srv_rate
F36	dst_host_same_src_port_rate
F37	dst_host_srv_diff_host_rate
F38	dst_host_serror_rate
F39	dst_host_srv_serror_rate
F40	dst_host_rerror_rate
F41	dst_host_srv_rerror_rate

The original data contain 744MB data with 4,940,000 records. A ten percent subset of this data contain 75MB with 4,94,021 records which approximately contain 20% normal patterns and the rest 80% of patterns are with attacks belonging to four categories (DOS, Probe, U2R and R2L). Among them we have selected 12,723 records randomly for developing the Neural Network. The details of the records selected for training and testing the Neural Network is given in Table 2.

Among the 41 input features, 32 features are continuous variables and 9 features are discrete variables. Suitable integer numbers are assigned to these discrete variables. For example, for the discrete variable protocol_type which describes the type of the protocol we have assigned 1 for tcp, 2 for udp, 3 for http and so on. Accordingly suitable integer numbers are assigned to other discrete variables also. The output attack label is represented as [0 0 0 0] for Normal, [0 0 0 1] for DOS, [0 0 1 0] for Probe, [0 1 0 0] for R2L and [1 0 0 0] for U2R. Two different ANN models were developed for intrusion detection, one with all features and another with reduced features.

Table.2 Distribution of Data

Total Number of Samples: 12,723					
Data Distribution	**Normal**	**DOS**	**Probe**	**U2R**	**R2L**
Training: 6,363	2500	1500	1500	20	843
Testing: 6,360	2500	1500	1500	19	841

The neural network model is developed using MATLAB 6.5 Neural Network Toolbox in Pentium 4 with 2.40 GHz processor with 256 MB of RAM. The details of the model developed are given below:

5.1 CASE (I): ANN MODEL WITH ALL FEATURES

Initially all the 41 input features are given as input to the neural network without any feature selection. Trial and error procedure was followed to identify the optimal number of hidden nodes. There are about forty one neurons in the input layer that corresponds to the all the 41 input features and four neurons in the output layer in which all neurons set to zero corresponds to Normal and one in each neuron corresponds to any one of the four attacks (DOS, Probe, R2L and U2R). The number of hidden-units is directly related to the capabilities of the network. The network is trained with least means square algorithm until it reaches the mean square error of 0.001. The mean square error achieved during training is 9.9975e-004. With ten hidden nodes, the network took 257.7030 seconds to reach the error goal.

After training, the generalization performance of the network is evaluated with the 6,360 test data. The trained Neural Network classified 6,038 data correctly which shows an overall detection rate of 94.93%. During testing the Mean Square Error achieved by the network is 0.0097. The performance of the network during testing is presented in Table 3.

Table.3 Testing Performance without MI

Attack Classes	**No. of Correctly identified attack**	**Detection rate**
Normal	2494	99.76%
DOS	1500	100%
Probe	1500	97.8%
R2L	570	67.77%
U2R	7	36.84%
TOTAL	6,038	94.93%

From table III it is inferred that if all the 41 features are used as input to the neural network then some of the features that are redundant and containing false correlation hinder the process of detecting intrusions and therefore we get very poor detection rate especially for the R2L and U2R attack. Hence we in the second case we employ Mutual Information to remove unnecessary features and select only an informative features to be used as input to the developed ANN model.

5.2 CASE (II): ANN WITH REDUCED FEATURES

In this case, the network is trained with reduced features. Hence the number of output neurons is kept constant while the number of input neurons is varied depending on the features selected by Mutual Information. For selecting the input features, the training data set is arranged in the ascending order based on the output. Then, the output quantity is divided into three groups and the initial entropy is calculated using Equation (4). The input variables are divided into ten levels and their conditional entropies are evaluated using Equation (5). Next, the mutual information of each variable with respect to the output is computed using Equation (6). The mutual information between the input variables and the output classes is shown in Fig.3. From this figure, it is evident that only a few variables are having significant information about the output quantity and the remaining variables have very less amount of information only.

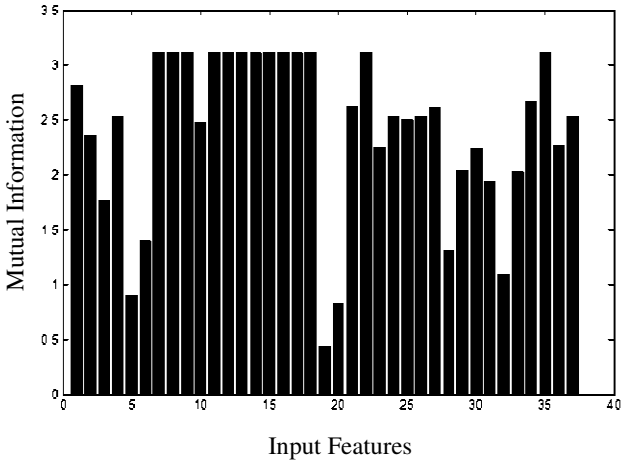

Fig.3. Mutual Information for the input features

To select the optimum number of features for the neural network, the input variables are ranked based on their mutual information value and the top 20 features are used to train the network after normalization along with the output and this number is increased progressively until the maximum required accuracy is reached. The network has shown satisfactory performance with 29 features. The name of the selected features is given in Table 4.

Table.4 Details of Selected Input Features

Label	Feature Name
F1	duration
F2	protocol-type
F4	flag
F7	land
F8	wrong_
F9	urgent
F10	hot
F11	num_failed_logins
F12	logged_in
F13	num_compromised
F14	root_shell
F15	su_attempted
F16	num_root
F17	num_file_creations
F18	num_shells
F21	is_host_login
F22	is_guest_login
F23	count
F24	srv_count
F25	serror_rate
F26	srv_serror_rate
F27	rerror_rate
F29	same_srv_rate
F30	diff_srv_rate
F33	dst_host_srv_count
F34	dst_host_same_srv_rate
F35	dst_host_diff_srv_rate
F36	dst_host_same_src_port_rate
F37	dst_host_srv_diff_host_rate

The mean square error achieved by the network during training is 9.9979e-004. With ten hidden nodes, the network took 249.7030 seconds to reach the error goal. The performance of network during training is shown in Fig.4.

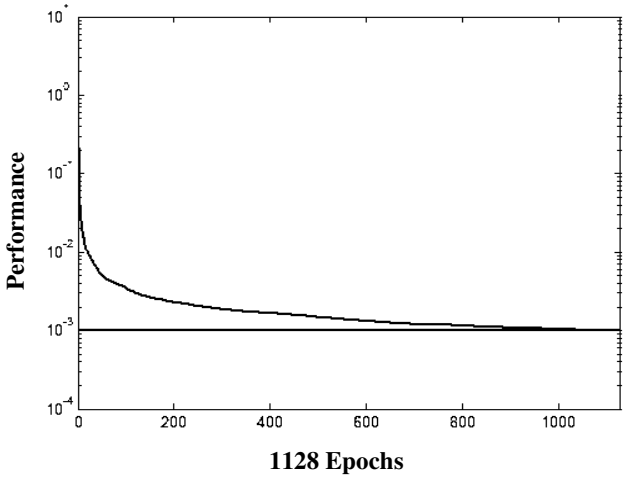

Fig.4. Training Performance of the network

After training, the generalization performance of the network is evaluated with the test data. During testing the Mean Square Error achieved by the network is 4.2758e-004. The performance of the network during testing is presented in Table 5.

Table.5 Testing Performance with MI

Attack Classes	No. of Correctly identified attack	Detection rate
Normal	2500	100%
DOS	1500	100%
Probe	1500	100%
R2L	841	100%
U2R	18	94.73%
TOTAL	6359	99.98%

The performance of the proposed Mutual Information based Feature Selection for the Artificial Neural Network Model has been compared with the other approaches and it is presented in the Table 6 and 7. Table 8 shows the comparison of the results obtained by using ANN in which features are selected by feature ranking method (deleting one feature at a time) proposed in [12] and our approach.

Table.6 Performance comparison with Sung *et al.* [12]

Approaches	ANN – MI	SVM – PBRM	SVM – SVDFRM
Features Selected	29	30	23
Normal	100%	99.51%	99.55%
DOS	100%	99.22%	99.20%
Probe	100%	99.67%	99.71%
U2R	94.73%	99.87%	99.87%
R2L	100	99.87%	99.78%

Table.7 Performance comparison with Ensemble

	ANN – MI	Ensemble Approach[3]	BN [3]	CART [3]
Features Selected	29	17	19	19
Normal	100%	99.64%	99.57%	95.50%
DOS	100%	100%	99.02%	94.31%
Probe	100%	100%	96.71%	96.85%
U2R	94.73%	72%	56%	84%
R2L	100	99.29%	97.87%	97.69%

Table.8 Results Comparison with ANN-FR

Performance Metrics	ANN – MI	ANN – FR [12]
Features Selected	**29**	**34**
Accuracy	**99.98%**	**81.57%**
False Positive rate	**0**	**18.19**
False negative rate	**0.05**	**0.25**
Number of epochs	**1128**	**27**

From these tables, it is found that the trained neural network shows 100% detection rate for the Normal, DOS, Probe and R2L attacks and 94.73% for U2R attack with just 29 features selected by Mutual Information. The overall performance of the network is found to be 99.98% detection rate in just 1128 epochs. This shows that the trained network is able to identify the different type of attack accurately with zero false positive and less negative rates when compared with other approaches reported in the literature.

6. CONCLUSION

The bottleneck of the ANN model for Intrusion Detection is the size and dimensionality of the data set considered because the amount of the data that an IDS needs to examine is very large even for a small network and it contains extraneous features which is very much harder to detect suspicious behavior patterns. This paper proposes a mutual information based feature selection for intrusion detection using a simple feed forward neural networks trained by the back propagation algorithm. The performance of the network was tested using ten percent of the KDD cup 1999 dataset which is available in the UCI KDD Archive and is compared with other approaches. Simulation result shows that the proposed approach detects the intrusions accurately and is well suitable for real time applications.

REFERENCES

[1] A. Zhong and C.F. Jia. 2004, "Study on the applications of hidden Markov models to computer intrusion detection," in *Proceedings of the Fifth World Congress on Intelligent Control and Automation WCICA*, Vol. 5, pp. 4352-4356.

[2] M.Analoui, A.Mizaei, and P.Kabiri. 2005, "Intrusion detection using multivariate analysis of variance algorithms," in *Third International Conference on Systems, Signals & Devices SSD05*, Vol. 3.

[3] Chebrolu, S., A. Abraham and J.P. Thomas. 2005. Feature deduction and ensemble design of intrusion detection system. *Computers & Security*. Vol.24, No.4: pp.295-307.

[4] Denning, D. E., 1987. An Intrusion-Detection Model. *IEEE Transactions on Software Engineering*. Vol.13, No.2: pp.222-232.

[5] Luo J., S. M. Bridges. 2000. Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection. *International Journal of Intelligent Systems.* Vol.15, No.8: pp.687-704.

[6] J. Cannady, 1998, "Artificial Neural Networks for Misuse Detection," *Proceedings, National Information Systems Security Conference (NISSC'98)*, pp.443-456.

[7] Lipmann, R.P., and R.K Cunningaham. 1999. Improving Intrusion Detection Performance using keyword selection and neural networks. Computer Networks. Vol.34, No.4: pp.597-603.

[8] M.Ramadas, S.Ostermann, and B.Tjaden, 2003, "Detecting anomalous network traffic with self organizing maps," in *Recent Advances in Intrusion Detection, 6th International Symposium,(RAID-2003)*, pp. 36-54.

[9] J.M. Bonoficio, 1998. "Neural Networks Applied in Intrusion Detection Systems," *IEEE World Congress on Computational Intelligence*, Vol.1, pp. 205-210.

[10] C.Jirapummin, N.Wattanapongsakorn and P.Kanthamanon, 2002, "Hybrid Neural Networks for Intrusion Detection System," *Proceedings of the International Technical Conference on Circuits / Systems, Computers and Communications (ITC-CSCC 2002)*, pp.928-931.

[11] Kabiri, P., and A. A. Ghorbani. 2005. Research on Intrusion Detection and Response: A Survey. *International Journal of Network Security.* Vol.1, No.2: pp.84-102.

[12] A. H. Sung, S. Mukkamala, 2003, "Identifying important features for intrusion detection using support vector machines and neural networks," in *Proceedings of International Symposium on Applications and the Internet (SAINT 2003),* pp. 209-17.

[13] R. Battiti. 1994. Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transaction on Neural Networks.* Vol.5, No.4: pp.537-550.

[14] P.GaneshKumar, D.Devaraj, V.Vasudevan, 2006, "Artificial Neural Network for Misuse Detection in Computer Network," *Proceedings of the International Conference on Resource Utilisation and Intelligent Systems (INCRUIS-2006)*, pp.889-893.

[15] DARPA Intrusion Detection Evaluation – MIT Lincoln Laboratory – (http://www.ll.mit.edu/IST/ideval)

[16] Devaraj, D., R.UmaRani and J. P. Roselyn. 2007. Artificial Neural Network Model for Voltage Security Based Contingency Ranking. Applied Soft Computing Journal Vol.7, No.3: pp.722-727.

[17] D.Devaraj and B. Yegnanarayana, 2001, "Performance of neural network based contingency selection with reduced input features", *Proceedings of Intelligent system Applications to Power Systems (ISAP-2001) Conference*.

[18] KDD-cup dataset, http://kdd.ics.uci.edu/ databases/ kddcup99/ kddcup99.htm.