Comparison of variable learning rate and Levenberg-Marquardt back-propagation training algorithms for detecting attacks in Intrusion Detection Systems

Tummala Pradeep[1]
IV[th] Year Student, Department of CSE
Birla Institute of Technology (BIT), Mesra
Ranchi,Jharkhand,India
tummala_pradeep@hotmail.com

P.Srinivasu[2]
Associate Professor, Department of CSE
Anil Neerukonda Institute of Technology and Sciences (ANITS)
Visakhapatnam, Andhra Pradesh, India
ursrinivasu@gmail.com

P.S.Avadhani[3]
Professor, Department of CS&SE
Andhra University
Visakhapatnam, Andhra Pradesh, India

Y.V.S. Murthy[4]
Assistant Professor, Department of CSE
Anil Neerukonda Institute of Technology and Sciences (ANITS)
Visakhapatnam, Andhra Pradesh, India

{Corresponding author: **ursrinivasu@gmail.com, tummala_pradeep@hotmail.com**}

***Abstract*— This paper investigates the use of variable learning rate back-propagation algorithm and Levenberg-Marquardt back-propagation algorithm in Intrusion detection system for detecting attacks. In the present study, these 2 neural network (NN) algorithms are compared according to their speed, accuracy and, performance using mean squared error (MSE) (Closer the value of MSE to 0, higher will be the performance). Based on the study and test results, the Levenberg-Marquardt algorithm has been found to be faster and having more accuracy and performance than variable learning rate back-propagation algorithm.**

Keywords- KDD Dataset, Levenberg-Marquardt, Backpropagation, Intrusion detection, Artificial Neural Networks.

I. INTRODUCTION

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. Incidents have many causes, such as

1. Malware.
2. Attackers gaining unauthorized access to systems from the Internet

3. Authorized users of systems who misuse their privileges or attempt to gain additional privileges for which they are not authorized.

Network Behavior Analysis (NBA) is one of the most prominent technology deployed in intrusion detection systems. It examines network traffic to identify threats that generate unusual traffic flows, such as distributed denial of service (DDoS) attacks, certain forms of malware and policy violations. In order to determine attacks, the system must be trained to recognize normal system activity. Artificial Intelligence technique and neural networks system are 2 prominent methods for training IDSs. Most NBA sensors can reconstruct a series of observed events to determine the origin of threat. For example, if a worm infects a network, the sensor can analyze the worm's flow to determine the attacking host.

An intrusion detection system (IDS) is software that automates this intrusion detection process. IDS technologies use many methodologies to detect incidents. The primary classes of detection methodologies are

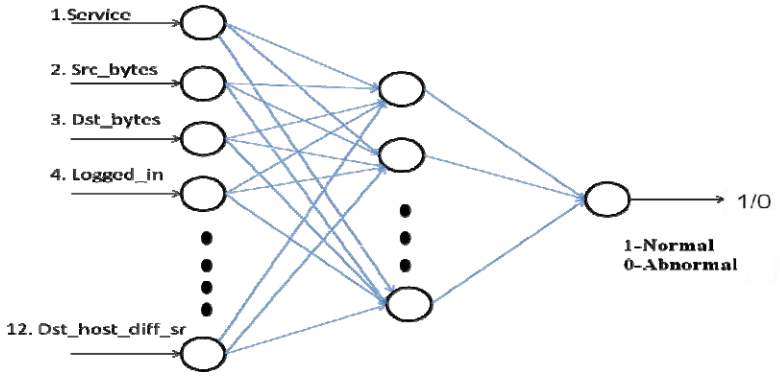
1. Signature-based detection
2. Anomaly-based detection
3. Stateful protocol analysis

Anomaly-based detection is primarily used in academic research due to its theoretical potential to detect attacks. Anomaly-based detection is the process of identify significant deviations by comparing definitions of what normal activity against observed events. An IDS using anomaly-based detection has profiles that represent the normal behavior of users, hosts, network connections, or applications. This approach is useful in detecting unknown threats [1].

II. FUNDAMENTALS OF NEURAL NETWORKS

Artificial neural networks deal with the process of simulating the behavior of biological neurons to provide an effective means to handle classification problems. An Artificial Neural Network (ANN) is a highly parallel distributed network of connected processing units called neurons. It is motivated by the human cognitive process: the human brain is a highly complex, nonlinear and parallel computer. The network has a series of external inputs and outputs which take or supply information to the surrounding environment. Inter-neuron connections are called synapses which have associated synaptic weights. These weights are used to store knowledge which is acquired from the environment. Learning is achieved by adjusting these weights in accordance with a learning algorithm. It is also possible for neurons to evolve by modifying their own topology, which is motivated by the fact that neurons in the human brain can die and new synapses can grow. One of the primary aims of an ANN is to generalize its acquired knowledge to similar but unseen input patterns. Two other advantages of biological neural systems are the relative speed with which they perform computations and their robustness in the face of environmental and/or internal degradation. Thus damage to a part of an ANN usually has little impact on its computational capacity as a whole. This also means that ANNs are able to cope with the corruption of incoming signals[2].

A neuron is an information-processing unit that is fundamental to the operation of a neural network. A neuron has an input layer, a set of hidden layers and an output layer.



Basic steps in ANN

- Present the neural network with a number of inputs (vectors each representing a pattern).
- Check how closely the actual output generated for a specific input matches the desired output.
- Change the neural network parameters (weights) to better approximate the outputs.

III. ATTACK CLASSIFICATION

The input we are using here is KDD CUP'99 data set. KDD'99 has been the most widely used data set for the evaluation of anomaly detection methods. The data set is built based on the data captured in DARPA'98 evaluation program. The data set consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack [3]. The attacks fall in one of the following four categories:

1) **Denial of Service Attack (DoS)**: It is a type of attack in which an attacker denies legitimate users access to machines or makes computing resources too busy to handle requests.

2) **User to Root Attack (U2R)**: In U2R the attacker first accesses the system with a normal user account by sniffing passwords or social engineering and then gains root access to the system by exploiting some vulnerability.

3) **Remote to Local Attack (R2L)**: R2L occurs when a user without an account has the ability to send packets to a machine gains local access as a user of that machine.

4) **Probing Attack**: It is a method of gathering information about a network of computers with an intention of circumventing its security controls.

The datasets contain a total number of 24 training attack types, with an additional 14 types in the test data only.

Table 1: Attack Classification

Denial Of Service	Remote–to-Local	User to root	Probe
Smurf	Guess-Password	Imap	Ipsweep
Snmp	Warezmaster	Load module	Nmap
Back	ftp-write	Buffer_overflow	Portsweep
Land	Multihop	Rootkit	Saint
Neptune	phf		Satan

IV. NEURAL NETWORK TRAINING ALGORITHMS

Two different artificial NN algorithms, variable learning rate and Levenberg-Marquardt back-propagation, are compared in the present study. This was done with an intention of finding the algorithm with better accuracy and performance when deployed in intrusion detection systems. The objective of training is to reduce the global error defined as

$$E = \frac{1}{P}\sum_{P=1}^{P} E_P$$

Where P is the total number of training patterns and E_p is the error for training pattern.

E_p is calculated by the following by the formula:

$$E_p = \frac{1}{2}\sum_{i=1}^{N}(o_i - t_i)^2$$

Where N is the total number of output nodes, o_i is the network output at the i^{th} output node, and t_i is the target output at the i^{th} output node.

a. Variable Learning Rate Back-Propagation Algorithm

Back propagation is a neural network learning algorithm. Back propagation learns by iteratively processing a data set of training tuples. During the training session of the network, a pair of patterns is presented (X_k, T_k), where X_k is the input pattern and T_k is the target or desired output. The X_k pattern causes output responses at each neuron in each layer, hence, an actual output Ok at each layer. At the output layer, the difference between the actual and target outputs yields an error signal. This error signal depends on the values of the weights of the neurons in each layer. This error is minimized, and during this process new values of weights are obtained. The speed and accuracy of the learning process i.e. updating the weights, also depends on a factor known as the learning rate. The inputs to the nodes in the first layer are the normalized values of attributes selected. These are combined with the corresponding weights (randomly selected initially) along with the bias.

For the first run these values are propagated to the next layers where outputs are generated from them according to the formula of sigmoid function which is as follow

$$O_j = \frac{1}{1+e^{-I_j}}$$

Where I_j represents the weighted sum of all inputs at that node along with respective bias values. It is computed as follows

$$I_j = \sum w_{ij}O_i + \theta_j$$

Where w_{ij} is the weight of i^{th} node propagated to j^{th} node. O_i is the output of i^{th} node. θ_j is the bias at j^{th} node.

Now we calculate the error at the output node to be the difference between the expected and observed output values at the node.

$$Err_j = O_j(1 - O_j)(T_j - O_j)$$

We propagate this error backwards from a node to the nodes connected to it from the previous layer and update the synaptic weights and the bias values.

$$Err_j = O_j(1 - O_j)\sum_k Err_k w_{jk}$$

$$\Delta w_{ij} = \mu * Err_j O_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

Where μ is the learning rate of the neural network

Once the error is adjusted among the different nodes, another run is required to check if the error has been properly adjusted. Now the training is done with several sample inputs and once training is done, the neural network is to be validated by using the validation dataset to see that the network is under fitted not over fitted [4].

b. Levenberg-Marquardt Back-Propagation Algorithm

Levenberg-Marquardt algorithm was designed to approach second-order training speed without having to compute the Hessian matrix [5]. The Hessian matrix can be approximated as

$$H = J^T J$$

When the performance function has the form of a sum of squares.

The gradient can be computed as $g = J^T e$

Where **J** is the Jacobian matrix computed through a standard backpropagation and **e** is a vector of network errors.

The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix using the following update:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}]^{-1}\mathbf{J}^T\mathbf{e}$$

When the scalar μ is zero, this is just Newton's method which is faster and more accurate near an error minimum. So the aim is to shift toward Newton's method as quickly as possible. Performance function is always reduced at each iteration of the algorithm by decreasing μ after each successful step and is increased only when a tentative step would increase the performance function[6][7].

V. TESTING

a. Software And Input Dataset

The algorithm has been implemented in MATLAB 2010 since it increases flexibility when computing large number of records. The requirement for this software is Windows XP.

We have selected 12 important features from the total 41 features of KDD CUP'99 in the actual dataset

Table 2: Input Dataset

S.No	Name of the Feature	Description	Data type
1	Service	Destination service	D
2	Source bytes	Bytes sent fromsource to destination	C
3	Destination Bytes	Bytes sent from destination to source	C
4	Logged in	1 if successfully logged in;0 otherwise	D
5	Count	No of connections to the same host as the current connection in the past 2 seconds	C
6	Srv count	No of connections to the same service as the current connection in past 2 seconds	C
7	Serror rate	No of connections that have "SYN" error	C
8	Srv rerror rate	No of connection that have "REJ" error	C
9	Srv diff host rate	No of connection to different host	C
10	Dst host count	Count of connection having same dest hot	C
11	Dst host srv count	Count of connection having same dest hot and using the same service	C
12	Dst host diff srv rate	No of different service on the current host	C

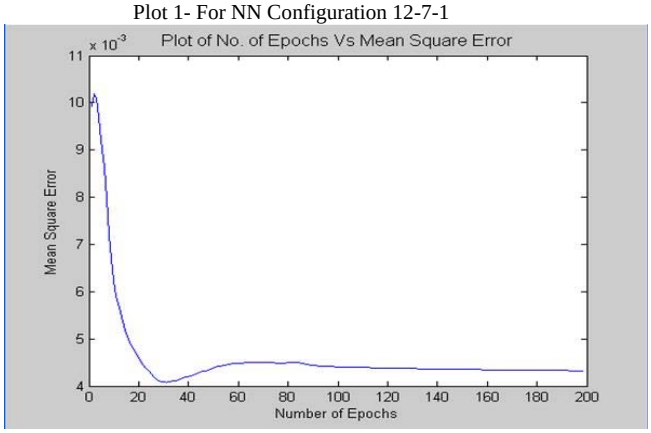
The original data set contains attributes such as "protocol type" with values "UDP","TCP" and Feature "Flag" with values "SF","REJ","RSTR" etc. Since we feed these attributes to the real coded backpropagation algorithm, there must be a way of representing the string values with real numerical values. We have also encoded the service attribute [8].

b. Test Results

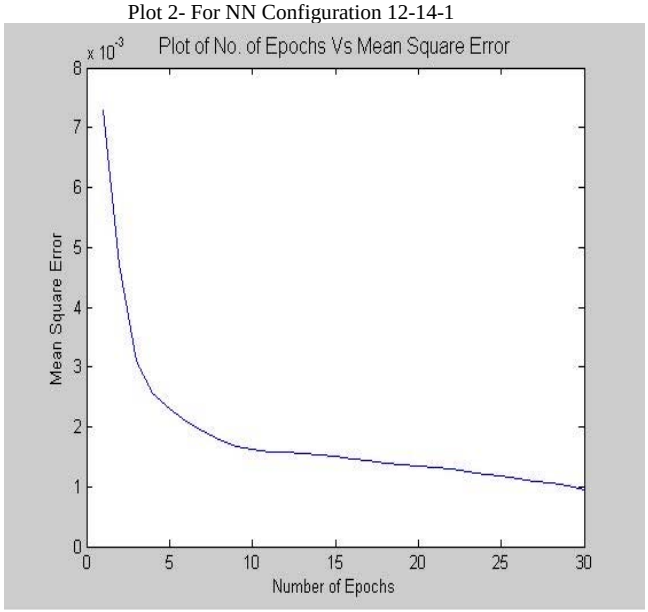
1. Variable Learning Rate Backpropagation Algorithm

The plot of number of epochs vs mean squared error for various configurations is as follows

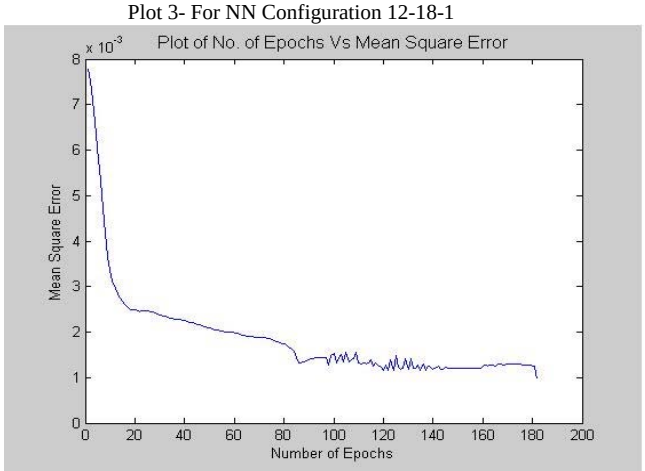
Plot 1- For NN Configuration 12-7-1



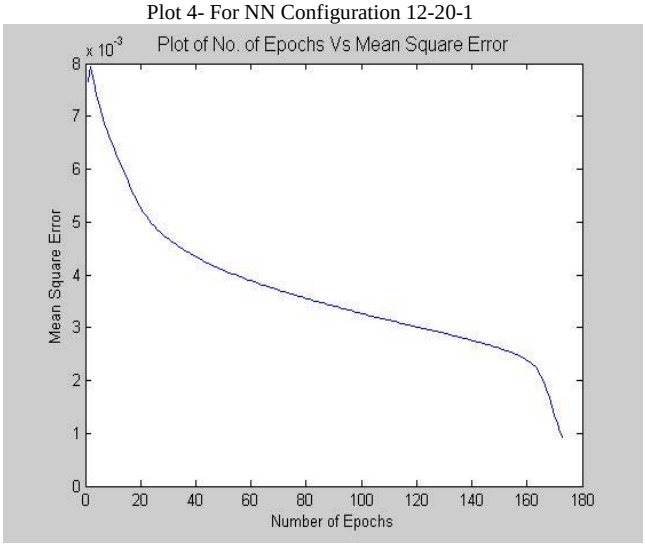
Plot 2- For NN Configuration 12-14-1



Plot 3- For NN Configuration 12-18-1



Plot 4- For NN Configuration 12-20-1



Using VLRB algorithm explained in section IV (a), the following results were obtained,

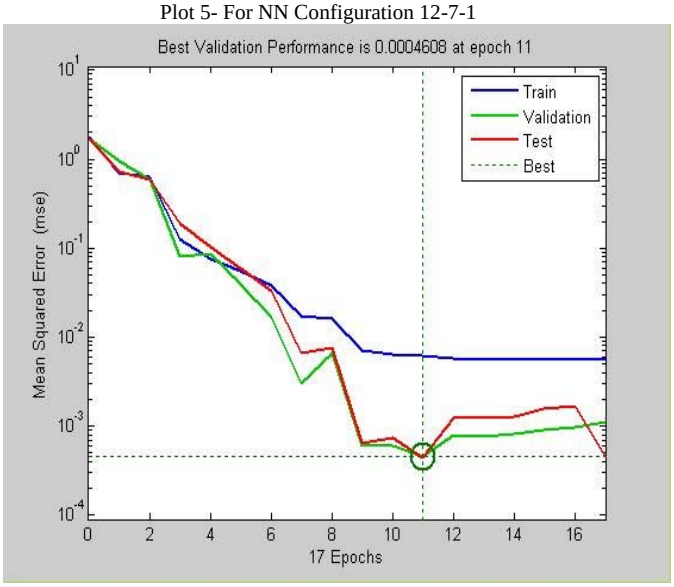
Table 3: Results for each configuration

NN Configuration	Epochs	Mean Square Error	Accuracy
12-7-1	200	0.0042	80.5%
12-14-1	30	0.0014	88.5%
12-18-1	182	0.000965	94.9%
12-20-1	173	0.0009	98.3%

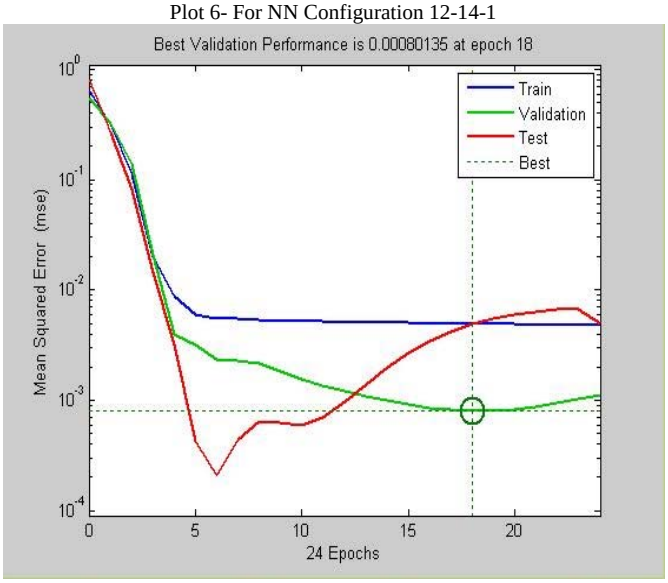
2. Levenberg-Marquardt Backpropagation Algorithm

The plot of number of epochs vs. mean squared error for various configurations is as follows

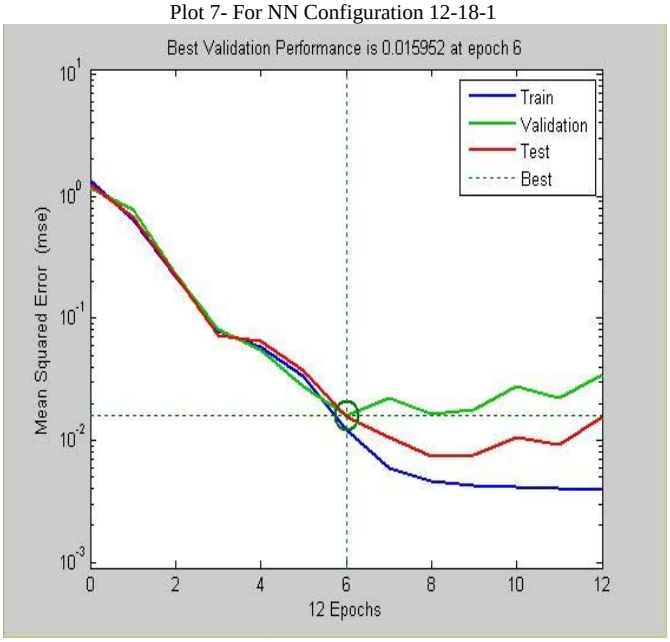
Plot 5- For NN Configuration 12-7-1



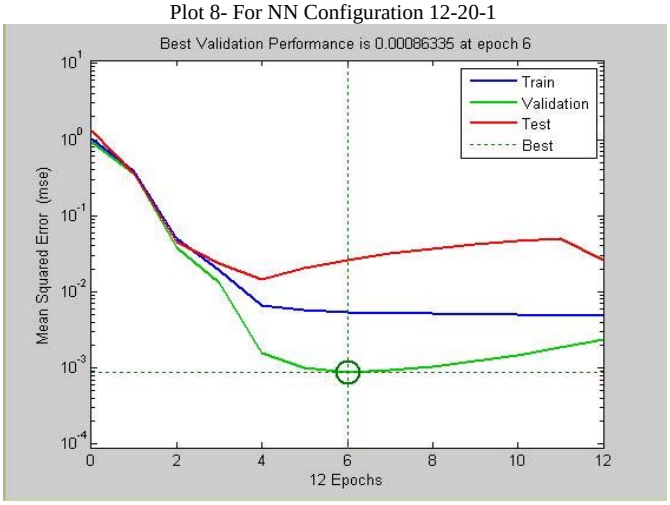
Plot 6- For NN Configuration 12-14-1



Plot 7- For NN Configuration 12-18-1



Plot 8- For NN Configuration 12-20-1



Using LM Backpropagation algorithm explained in section 3.2, the following results were obtained,

Table 4: Results for each configuration

NN Configuration	Epochs	Mean Square Error	Accuracy
12-7-1	17	0.00046	99.1%
12-14-1	29	0.00068	99.1%
12-18-1	12	0.0159	98.8%
12-20-1	12	0.00086	98.5%

VI. PERFORMANCE ANALYSIS

Using variable learning rate (VLR) backpropagation algorithm, the average mean squared error obtained is 0.00187. Though the maximum accuracy possible is 98%, the speed is found to be slower taking more than 100 epochs in most of the cases. In comparison, the average mean squared error obtained using Levenberg-Marquardt (LM) backpropagation algorithm is 0.0008925 indicating performance of LM is higher than VLR algorithm. Also, LM is much faster than VLR with accuracy around 99% in most of the configurations. Comparing the three parameters – speed, accuracy and performance, Levenberg-Marquardt is found to be better than Variable learning rate in all three parameters.

VII. CONCLUSION

This paper has compared the speed, accuracy and performance of Levenberg-Marquardt and Variable learning rate Backpropagation algorithm. Results show Levenberg-Marquardt to be better in all parameters. So, it is a favorable tool for designing intrusion detection system. If a hybrid intrusion detection system is designed using Levenberg-Marquardt and Genetic Algorithm, the accuracy and performance can be improved further. Future enhancements include working on the algorithm to analyze and reduce false positives and false negatives. The process used cannot differentiate between various types of attacks, it can only identify normal and attack data. Thus, we can modify the process to identify the attack types.

REFERENCES

[1] Karen Scarfone, Peter Mell, “A guide to Intrusion Detection and Prevention Systems,” NIST.
[2] S.Rajasekaran, G.A Vijayalakshmi Pai, “Neural Networks, Fuzzy Logic, and Genetic Algorithms”,PHI
[3] A Detailed Analysis of the KDD CUP 99 Data Set, *CISDA 2009*
[4] J. C. Dunn, “Comparison of three Back-Propagation Training Algorithms,” Indian Journal of Engineering & Material Sciences, Volume 12, pp. 434-442, 2005.
[5] More J J, in Numerical Analysis, edited by Watson G A, Lecture Notes in Mathematics 630, (Springer Verlag, Germany) 1997,105-116.
[6] Hagan M T & Menhaj , IEEE Trans Neural Networks, 5(6) (1994) 989-993.
[7] S. A. Dudani, “The Distance-Weighted k-NN Rule,” IEEE Transactions on Systems, Man and Cybernetics, Volume 6, Number 4, pp. 325-327, 1976.
[8] KDD’99 archive: The Fifth International Conference on Knowledge Discovery and Data Mining. URL: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

AUTHORS PROFILE

Mr. Tummala Pradeep, is pursuing his Final Year in Computer Science and Engg., in Birla Institute of Technology, Ranchi, Jharkhand, INDIA. He has 2 research paper on Intrusion detection system as co-author. His research areas include intrusion detection, soft computing, network security and crowd computing.

Mr. Pakkurthi Srinivasu, received his M.Tech(CST) from Andhra University , Visakhapatnam, Andhra Pradesh, India. Presently he is working as Associate Professor in Computer Science and Engineering in Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam Dist, AP, India. His research area includes Intrusion Detection, Network Security, Neural Network, Data Mining and fuzzy logic.

Prof. P. S. Avadhani, did his Masters Degree and PhD from IIT, Kanpur. He is presently working as Professor in Dept. Of Computer Science & Systems Engineering in Andhra University college of Engineering in Visakhapatnam. He has more than 70 papers published in various National & International journals and conferences. His research areas include Cryptography, Data Security, Algorithms, and Computer Graphics.

Mr. Y. V. Srinivasa Murthy, who is working as Assistant Professor for the CSE department in ANITS. He is awarded with his masters degree in Computer Science & Technology. An excellent command over programming language adds to his skills. He is expertized in accomplishing the papers and developing them with the thoughts ignited by the senior faculty members in a successful manner.