Performance Comparison between Backpropagation Algorithms Applied to Intrusion Detection in Computer Network Systems

Iftikhar Ahmad, M.A Ansari, Sajjad Mohsin
Department of Computer Sciences, Federal Urdu University, Islamabad &
COMSATS Institute of Information Technology, PAKISTAN.
{wattoohu@gmail.com, drmaansari@fuuastisb.edu.pk}

Abstract: - In this paper a topology of neural network intrusion detection system is proposed on which different backpropagation algorithms are benchmarked. The proposed methodology uses sampled data from KddCup99 data set, an intrusion detection attacks database that is a standard for the evaluation of intrusion detection systems. The performance of backpropagation algorithms implemented in batch mode, is addressed. A comparative analysis of algorithms is made and then the most optimum solution is selected with respect to mean square error.

Key-Words: - Intrusion Detection, Backpropagation, Neural Networks, Intrusion Detection System, Learning Algorithm, Dataset.

1.0 Introduction

Intrusion detection systems are the foremost tools for providing safety in computer and network system. The reliance of private and government organizations is increasing on their computer networks and defending theses system from attack is very much important. A single intrusion of a computer network can cause a heavy loss or the consistency of network became insecure [1]. Therefore we are presenting artificial neural network architecture to intrusion detection that uses online backpropagation, batch backpropagation and resilient backpropagation algorithms.

The online backpropagation, batch backpropagation and resilient backpropagation algorithms are used in learning and training phases of different neural networks. The backpropagation is the most extensively used algorithm for supervised learning with multilayered feed-forward networks. It was presented by Rumelhart et al. [2]. The resilient backpropagation is a novel learning scheme that performs a direct adaptation of the weight step based on local gradient information. It was presented by Riedmiller et al. [3]. This algorithm has also several implementations in different fields. This paper is divided into three prime areas. The first section provides a general idea of intrusion detection and neural networks. The second section presents our architecture of neural networks in the identification of attacks. Finally, a performance comparative analysis of results is made that are obtained by using the three algorithms. This shows an efficient and transparent adaptation step in the enhancement of network security.

2.0 Intrusion Detection

An illegal user that can access network resources and play some thing havoc is known as intruder. An Intrusion Detection System is used to sense illegal access to a computer or network system [4]. Intrusion Detection Systems (IDSs) are the most important tools for providing security in computer and networks. There are various methods of responding to a network intrusion, but they all require the exacting and suitable recognition of the attack [5]. Dr. Dorothy Denning proposed an intrusion detection model in 1987 which became a milestone in the research in this area. The model which she proposed forms the basic core of most intrusion detection methodologies in use today [6]. The intrusion detection systems can be classified into three categories as host based, network based and vulnerability assessment [7]. There are numerous approaches that are being used to accomplish the desirable essentials of an intrusion detection system like anomaly detection, misuse detection, combined anomaly/misuse detection, pattern recognition and networking monitoring [8].

2.1 Neural Networks

The first artificial neuron was produced in 1943 by the neurophysiologist Warren McCulloch and the logician Walter Pits [9]. An artificial neuron is a processing element with many inputs and one output. An artificial neural network consists of a group of processing elements that are greatly interconnected and convert a set of inputs to a set of preferred outputs. The result of the transformation is determined by the characteristics of the elements and the weights associated with the interconnections among them. By modifying the connections between the nodes the network is able to adapt to the desired outputs [10]. The application of neural network in intrusion detection is an on going area. It offers the potential to resolve a number of the problems encountered by the other current approaches to intrusion detection. Artificial neural networks are alternatives. The first advantage in the use of a neural network in the intrusion detection would be the flexibility that the network would provide. . A neural network would be capable of analyzing the data from the network, even if the data is incomplete or unclear. Similarly, the network would possess the ability to conduct an analysis with data in a non-linear fashion. Further, because some attacks may be conducted against the network in a coordinated attack by multiple attackers, the ability to process data from a number of sources in a non-linear fashion is especially important. The natural speed of neural networks is another advantage of this approach. However, the most significant advantage of neural networks in intrusion detection is the ability of the neural network to "learn" the characteristics of attacks and identify instances that have been observed before by the network [8, 11].

3.0 Architecture

The design of our system consists of one input, two hidden and one output layer. The Input layer takes input from the input file that contains data for training of the net. The hidden layers take inputs from the outputs of the input layer and apply its activation function. After this the output is sent to the output layer. The output layer allows a neural network to write output patterns in a file that are used for analysis of intrusion. The general architecture of our system is shown.

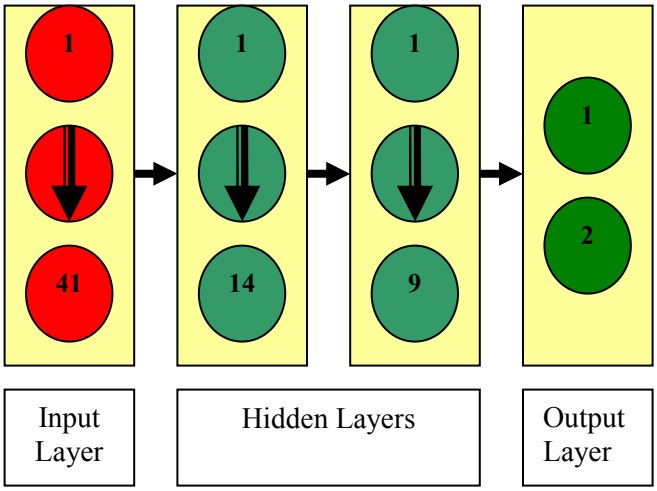

Fig.1: General Architecture

The design of our mechanism consists of one input, two hidden and one output layer with 41,14,9 and 2 neurons. The input layer consists of 41 neurons because the Kddcup data set contains 41 fields for a network packet to be used for intrusion detection training [1, 10, 12, 13, 14]. There is no exact formula for the selection of hidden layers so we can make it by comparison and select which one is best [15]. For choosing best set of hidden layers and its no. of neuron a comparison is made for many cases and best one is selected that is two hidden layers with 14 and 9 neurons as shown in the table below.

Sr.#	Hidden Layers	Neurons	RMSE
1	H1→H2→H3	14→9→3	0.1487
2	H1→H2→H3→H4	14→9→6→3	0.1523
3	H1→H2→H3	14→9→4	0.01486
4	H1→H2→H3	20→10→5	0.1942
5	H1→H2→H3	30→25→15	0.5632
6	H1→H2	14→9	0.00211
7	H1	30	0.1342

Table 1: A comparison for choosing best set of hidden layers and its no. of neuron

The architecture used in our system is feed forward. A feed forward neural net is composed of a number of consecutive layers/components, each one

connected to the next by a synapse/connection. Our design is also a FFNN with four layers connected with three synapses. Each layer is composed of a certain number of neurons, each of which have the same characteristics (transfer function, learning rate, etc) [1].

If the computed output of neural network is nearest to 0 and 1 then this packet will be considered as normal. If the output of neural network is nearest to 1 and 0 then this packet will be considered as bad means it involves some intrusion. The training architecture consists of five layers. These are input layer, two hidden layers, one output layer, and one teacher layer. All these are connected through synapses that serve as connection and transmission medium between them.

The training architecture is also consists of layers. The Layer contains units or nodes called neurons. A neuron is a processing element that has an activation function to produce its output. The layers of the net are interconnected by synapse. The Layer object is the basic element that forms the neural network. It is composed of neurons, all having the same characteristics. This component transfers the input pattern to the output pattern by executing a transfer function. The output pattern is sent to a vector of Synapse objects attached to the layer's output. It is the active element of a neural net. The Layer applies a sigmoid transfer function to its input patterns.

Transfer Function

$$y = \frac{1}{1+e^{-x}}$$

4.0 Algorithms

A neural network learns to resolve a problem simply by modifying its internal connections (biases and weights) by back-propagating the difference between the current output of the neural network and the desired response. In order to obtain that, each bias/weight of the network's components (both layers and synapses) is adjusted according to some specific algorithm. There is not just one algorithm to change the internal weights of a neural network but we need a flexible mechanism in order to be able to set the training algorithm suitable for a determined problem. There is no any optimal algorithm that is good for whatever problem. We need to try several of them in order to find the best one for our specific application that identifies intrusions in network systems by separating bad packets from normal ones.

A- The basic On-Line BackProp algorithm

This is the most familiar algorithm used in training. It adjusts the Layers' biases and the Synapses' weights according to the gradient calculated by the TeacherSynapse and back-propagated by the backward-transportation mechanism. It is called 'On-Line' because it adjusts the biases and weights after each input pattern is read and elaborated, so each new pattern will be elaborated using the new weights/biases calculated during the previous cycles. The algorithm searches for an optimal combination of net's biases/weights until a good minimum is found. The algorithm uses two parameters to work: the learning rate and the momentum. Both these parameters must be set before learning, and good values can be found only through several trials. The momentum can be set to 0, whereas the learning rate must be always set to a value greater than zero, otherwise the network cannot learn[19].

B- The Batch BackProp algorithm

This is a variant of the on-line algorithm, because it works exactly like that one, except that the biases/weights adjustments are applied only at the end of each epoch (i.e. after all the input patterns of an entire epoch have been elaborated). It works by storing in a separate array all the changes calculated for each pattern, and by applying them only when the current epoch is finished. In this manner each pattern belonging to the same epoch will be elaborated using an unmodified copy of the weights/biases. This causes more memory to be consumed by the network, but in some cases the batch algorithm converges in less epochs. This algorithm uses, beside the same parameters of the on-line version, also another parameter named batch size. It indicates the number of input patterns during which we want to use the batch mode, before to apply the on-line modification of the biases/weighs. This parameter, normally, is set to the number of training patterns, but by setting it to a smaller value, we can train our network also in mixed-mode [20].

C- The Resilient BackProp algorithm

This is an improved adaptation of the batch backprop algorithm, and for numerous problems it converges very quickly [15, 16]. It uses only the sign of the backpropagated gradient to change the biases/weights of the network, instead of the magnitude of the gradient itself. This because, when a

Sigmoid transfer function is used the gradient can have a very small magnitude, causing small changes in the weights and biases, even though the weights and biases are far from their optimal values. This modified algorithm is a batch training algorithm, and uses only the batch size property. The value of the learning rate and the momentum properties doesn't affect the calculus of the Rprop algorithm.

5.0 Data set

The data set chosen for the experiment is the KDD Cup 99 data set. This data set is available at [11, 12]. The DARPA data set is the most widely used data set for testing IDS [6, 7, 8, 9, 21, 22]. This DARPA project was prepared and executed by the Massachusetts Institute of Technology (MIT) Lincoln Laboratory. One of the reasons for choosing this data set is that the data set is standard. This will make it easy to compare the results of our work with other similar works. Another reason is that it is difficult to get another data set which contains so rich a variety of attacks as the one used here.

Usage of the data set

This section describes how the data set is used for our experiment. The 80% from the samples are used for training while others are used for testing the net. The data set is preprocessed so that it may be able to give it as an input to our developed system. This data set consists of numeric and symbolic features and we converted it in numeric form so that it can be given as inputs to our neural network. We replaced symbolic with specific numeric, comma with semicolon, normal with 0, 1 and attack with 1, 0. Now this modified data set is ready to be used as training and testing of the neural network.

6.0 Implementation

The design and the implementation are modular. The programming language used for the implementations of the algorithms and the experiment is Java and its JOONE API [13, 14]. As an example of object oriented programming language, Java supports modularity. Furthermore Java is an efficient and flexible programming language. The efficiency is an important issue in our research because of the large size of the data set. The architecture of the system is composed of two modules: the training module, and the evaluation module. The interface of the application is shown below.

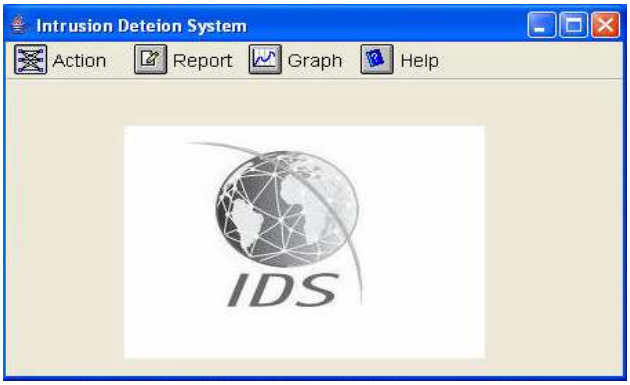

Fig.2: An Interface of IDS

The training module

This involves the training of the neural networks. It is necessary for accurate detection of intrusions in the computer network systems. The accuracy of the system depends on the correctness of data and its training algorithms. We trained the system on preprocessed data using online, batch and resilient backpropagation. A sample of training pattern of POD attack and its normal is shown below.

Attack	0;18;21;20;1480;0;0;1;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;1;1;0.00;0.00;0.00;0.00;1.00;0.00;0.00;1;1;1.0;0.00; 1.00; 0.00; 0.00; 0.00; 0.00; 0.00; 1; 0
Normal	0;18;21;20;1480;0;0;1;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;2;4;0.00;0.00;0.00;0.00;1.00;0.00;0.50;2;4;1.00;0.00;1.00;0.50;0.00;0.00;0.00;0.00;0;1

Table 2: A POD attack and its normal pattern

The evaluation module

This involves the testing of the trained net. After training, the net only involves computation of the feed forward phase. The output of the net is saved in file that will be used for intrusion analysis. If the global error value is nearest to 0 and 1 then it is normal packet otherwise consider as attack.

7.0 Results

First of all we make a performance comparison between above described algorithms. We tested the above implemented algorithms on preprocessed data of the kddcup data set for different epochs on same parameters (alpha=0.1, m=0.9). The computed results by our application are shown below.

Iteration	Epochs	Online	Batch	Resilient
1	25	0.19084	0.19025	0.19109
2	50	0.18312	0.18614	0.18639
3	75	0.15116	0.14189	0.14297
4	100	0.106163	0.109336	0.0928
5	125	0.04202	0.005732	0.03373

Table 3: First five iterations

Fig. 3: Performance of algorithms

Iteration	Epochs	Online	Batch	Resilient
1	150	0.001579	0.00283	0.002769
2	175	0.001077	0.001	9.82E-04
3	200	6.55E-04	6.21E-04	6.65E-04
4	225	5.34E-04	5.73E-04	4.49E-04
5	250	3.56E-04	2.77E-04	3.79E-04

Table 4: Next five iterations

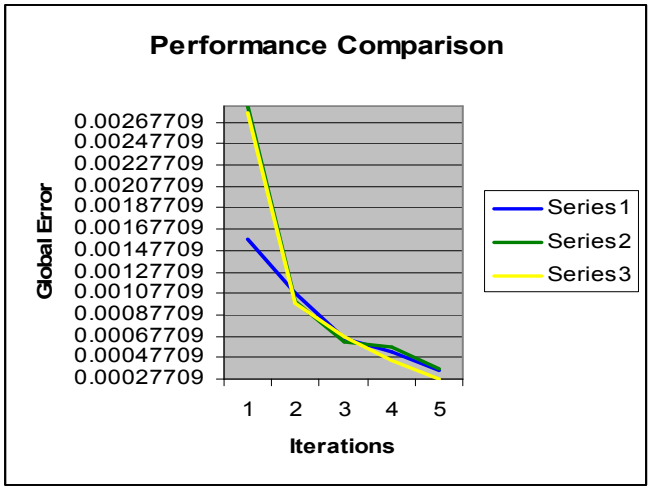

Fig. 4: Performance of algorithms

Initially online and batch backpropagation algorithms converge more quickly but as the time passes the RPROP algorithm shows its performance best. It uses the sign of the backpropagated gradient to change the weights of the network, instead of the magnitude of the gradient itself. This is because the gradients have a very small magnitude causing small changes in the weights even though the weights are far from their optimal values. Teaching neural network using RPROP requires fewer steps as compared to other algorithms. Moreover it requires similar computational effort. Another feature of RPROP is its robustness which is explained by the ability of escaping flat areas of the error surface much faster than the other methods. This characteristics make the less dependent on the initialization of the weights and also more capable of avoiding local minima. Now for more satisfaction we perform another test of eight iterations on different epochs as shown below.

Iteration	Epochs	Online	Batch	RPROP
1	1000	0.0226	0.0177	0.0228
2	2000	0.0225	0.0225	0.0176
3	3000	0.0176	0.0176	0.014
4	4000	0.0125	0.0176	0.0176
5	5000	0.01761	0.0125	0.0124
6	6000	0.00761	0.0176	0.00769
7	7000	0.01761	0.0176	0.01254
8	8000	0.01761	0.01234	0.0123

Table 5: Iterations at different epochs

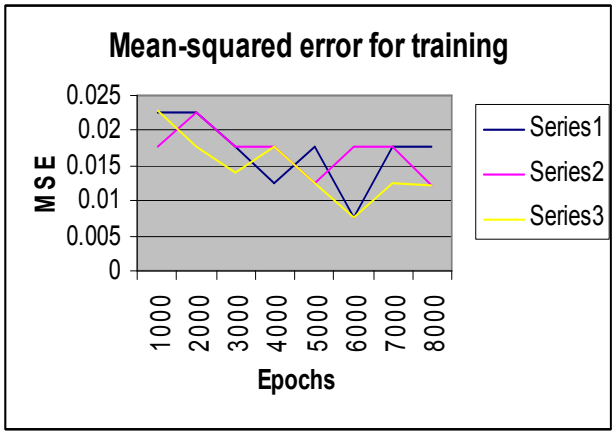

Fig. 5: Mean-squared error

The comparative analysis of algorithms attests the performance and effectiveness of RPROP. By

the above investigation of the results, it was verified that RPROP had a better performance to the application.

5.0 Conclusion

The focus of our research was to compare performance of backpropagation algorithms in intrusion detection and select the most optimum solution that may be applied in intrusion detection systems.

References:

[1]. Iftikhar Ahmad, Sami Ullah Swati, Sajjad Mohsin, Intrusion Detection Mechanism by Resilient Back Propagation (RPROP), EUROPEAN JOURNAL OF SCIENTIFIC RESEARCH, Volume 17, No. 4 August 2007 (pp.523-530).

[2]. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representation by error propagating" in Parallel Distributed Processing: Exploration in the Microstructure of Cognition (D.E. Rumelhart and J.L. McClelland,eds.), ch.8, MIT Press,Cambridge, MA, 1986.

[3]. M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, 1993, pp. 586--591.

[4]. http://www.stevespace.net/ids/index.html

[5]. Cannady J. Artificial neural networks for misuse detection. National Information Systems Security Conference; 1998. p. 368–81.

[6]. Denning, Dorothy. (February, 1987). An Intrusion-Detection Model. IEEE Transactions on Software Engineering, Vol. SE-13, No. 2.

[7] Jean-Philippe, Information Security, SANS Institute,2001.

[8] Shahbaz Pervez, Iftikhar Ahmad, Adeel Akram, Sami Ullah Swati A Comparative Analysis of Artificial Neural Network Technologies in Intrusion Detection Systems, WSEAS TRANSACTION ON COMPUTERS, Issue 1, Volume 6, January 2007 (pp.175-180).

[9] Christos Stergiou and Dimitrios Siganos, NEURAL NETWORKS, SURPRISE 96 Journal volume 4, 180 Queen's Gate, London SW7 2BZ, UK.

[10]. M. Amini , R. Jalili, Network-Based Intrusion Detection Using Unsupervised Adaptive Resonance Theory (ART), Fourth International ICSC Symposium on ENGINEERING OF INTELLIGENT SYSTEMS (EIS 2004) in collaboration with the University of Madeira Island of Madeira, Portugal February 29 – March 2, 2004.

[11] Cannady, J., & Harrell, J.R. (1996). A Comparative Analysis of Current Intrusion Detection Technologies. Proceedings of Technology in Information Security Conference (TISC) '96, 212-218.

[12]. The 3rd International Knowledge Discovery and Data Mining Tools Competition, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 2003.

[13]. M. Sabhnani and G. Serpen, "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context", http://www.eecs.utoledo.edu/~serpen/professional/Research/Publication/MLMTA 2003 Manuscript Submission Version.pdf, July 2003.

[14]. Aleksandar Lazarevic, Levent Ertoz, Vipin Kumar, Aysel Ozgur, Jaideep Srivastava, A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection, SIAM International Conference on Data Mining (2003) http://www.siam.org/meetings/sdm03.

[15]. www.ieeexplore.ieee.org/iel3/1723/4574 /00181220.pdf

[16]. KDD cup 99. Located at: htttp://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[17]. DARPA. Located at: http://www.ll.mit.edu/IST/ideval/

[18] Rhodes, B., Mahaffey, J., & Cannady, J. (2000, October). Multiple Self-Organizing Maps for Intrusion Detection. Proceedings of the 23rd National Information Systems Security Conference.

[19]. JOONE API. Located at: http://www.joone.org

[20]. JAVA .Located at: .http://java.sun.com/

[21].A Resilient Backpropagation Neural Network based Phase Correction System for Automatic Digital AC Bridges Dutta, M.; Chatterjee, A.; Rakshit, A.; Precision Electromagnetic Measurements Digest, 2004 Conference on June 2004 Page(s):374 – 375.

[22] Parameter Incremental Learning Algorithm for Neural Networks Sheng Wan; Banta, L. E.; Neural Networks, IEEE Transactions on Volume 17, Issue 6, Nov. 2006 Page(s):1424 – 1438