JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 5, No. 4, December 2011, pp. 305-313

Intrusion Detection: Supervised Machine Learning

Ahmed H. Fares* **and Mohamed I. Sharawy**
Department of Electrical & Computer Engineering, Benha University, Cairo, Egypt
ahmed.fares@feng.bu.edu.eg, msharawy@bu.edu.eg

Hala H. Zayed
Department of Computer Science, Faculty of Computers, Benha University, Cairo, Egypt **hala.zayed@fci.bu.edu.eg**

Abstract
Due to the expansion of high-speed Internet access, the need for secure and reliable networks has become more critical. The sophistication of network attacks, as well as their severity, has also increased recently. As such, more and more organizations are becoming vulnerable to attack. The aim of this research is to classify network attacks using neural networks (NN), which leads to a higher detection rate and a lower false alarm rate in a shorter time. This paper focuses on two classification types: a single class (normal, or attack), and a multi class (normal, DoS, PRB, R2L, U2R), where the category of attack is also detected by the NN. Extensive analysis is conducted in order to assess the translation of symbolic data, partitioning of the training data and the complexity of the architecture. This paper investigates two engines; the first engine is the back-propagation neural network intrusion detection system (BPNNIDS) and the second engine is the radial basis function neural network intrusion detection system (BPNNIDS).The two engines proposed in this paper are tested against traditional and other machine learning algorithms using a common dataset: the DARPA 98 KDD99 benchmark dataset from International Knowledge Discovery and Data Mining Tools. BPNNIDS shows a superior response compared to the other techniques reported in literature especially in terms of response time, detection rate and false positive rate.

Categroy: Ubiquitous computing

Keywords: Intrusion detection systems; Machine learning; Denial of service; Neural networks; The Defense Advanced Research Projects Agency (DARPA)

I. INTRODUCTION

Intrusions can be defined as actions that attempt to bypass the security mechanisms of computer systems [1-3]. Intrusions may take many forms: attackers accessing a system through the Internet or insider attackers; authorized (official) users attempting to gain and misuse non-authorized privileges. So, we say that intrusions are any set of actions that threaten the integrity, availability, or confidentiality of a network resource. Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. Intrusion detection systems (IDS) raise the alarm when possible intrusions occur.

A lot of research into artificial neural networks (ANNs) has been undertaken. In [4], artificial neural networks and support vector machine (SVM) algorithms were applied to intrusion detection (ID), using a frequency-based encoding method, on the DARPA dataset. The authors use 250 attacks and 41,426 normal sessions and the detection rate (DR) varied from 100% to 43.6% with the false positive rate (FPR) ranging from 8.53% to 0.27% under different settings. In [5], the author concludes that the combination of a radial basis function (RBF) and self-organizing map (SOM) is useful as an intrusion detection model. He concludes that the "evaluation of human integration" is necessary to reduce classification errors. His experimental results showed that RBF−SOM achieves similar or even better results,

Open Access **http://dx.doi.org/10.5626/JCSE.2011.5.4.305** **http://jcse.kiise.org**

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received 25 November 2010, **Revised** 19 Septempber 2011, **Accepted** 11 November 2011
*Corresponding Author

Copyright © 2011. The Korean Institute of Information Scientists and Engineers pISSN: 1976-4677 eISSN: 2093-8020

compared to just an RBF. In [6], the authors use hierarchical (SOM) and conclude that the best performance is achieved using a two-layer SOM hierarchy, based on all 41-features in the KDD dataset and the 'Protocol' feature provides the basis for a switching parameter. The detector achieved an FPR and DR of 1.38% and 90.4% respectively. In [7], the authors use a hierarchical ID model using principal component analysis (PCA) neural network that gave a 97.1% DR and a 2.8% FPR. In [8], a critical study about the use of some neural networks (NNs) is used to detect and classify intrusions, the DR was 93.83% for the PCA approach, and the FPR was 6.16% for PCA. In [9], the authors present a biologically inspired computational approach to dynamically and adaptively learn signatures for network intrusion detection using a supervised learning classifier system. The classifier is an online and incremental parallel production rule-based system.

It should be noted that most of the previous systems concentrate on either detecting two categories (normal or attack) or detecting a certain category of attack. Also all of the previous work ignores the symbolic features of the KDD Cup 1999 data set, this adversely affects the accuracy of detection. This study suggested a back-propagation neural network intrusion detection system (BPNNIDS) and a radial basis function neural network intrusion detection system (RBFNNIDS). Both can perform either two category or multi-category detection and at the same time they do not ignore the symbolic features of the data.

This paper is organized as follows: Section II introduces the intrusion detection taxonomy, Section III explains the methodology and the proposed system architecture, Section IV presents the experimental results and finally Section V concludes the paper.

II. INTRUSION DETECTION TAXONOMY

In short, intrusions are generally classified into several categories [8]:

- Attack types that are classified as:
  - Denial of service (DoS)
  - Probe (PRB)
  - Remote to login (R2L)
  - User to root (U2R)
- Single network connections involved in attacks versus multiple network connections
- Source of computer attacks:
  - Single attack versus multiple attacks
- Network, host and wireless networks
- Manual attacks and automated attacks

In short, IDS are generally classified according to several categories as follows [5, 7, 10]:

- Work environments that can be classified as having host-based IDs or network-based IDs
- Analysis that can be classified as anomaly detection or misuse detection.
- Analysis that can be classified as real-time analysis or off-line analysis.
- Architecture that can be classified as single and centralized or distributed and heterogeneous.
- Activeness that can be classified as active reaction or passive reaction.
- Periodicity that can be classified as continuous analysis or periodic analysis.

The system proposed in this paper is considered to be network based, misuse with the ability to merge new attacks in one of the main four categories (DoS, PRB, U2R, and the R2L), offline and passive.

III. THE PROPOSED SYSTEM ARCHITECTURE

A. Overview

The proposed system architecture (Fig. 1) is primarily based on three stages: data pre-processing stage, building the neural network intrusion detection system (two engines) stage and metrics used to analyze the results stage. The three main stages of the proposed system architecture and their respective sub groupings are discussed in more detail in the following sections. According to a survey about the DARPA 98, KDD Cup 99 datasets, it should be noted that this dataset has been the target of the latest research [11, 12].

B. Data Pre-processing Stage

The data pre-processing stage consists of the following:

1) Adding Columns Headers: Since the KDD Cup 1999 dataset was retrieved unlabeled, one of the first important steps is to add columns headers to it. 41 columns headers are added that contain information such as duration, protocol_type, service, src_bytes, dst_bytes, flag, land, wrong_fragment, *ext*.

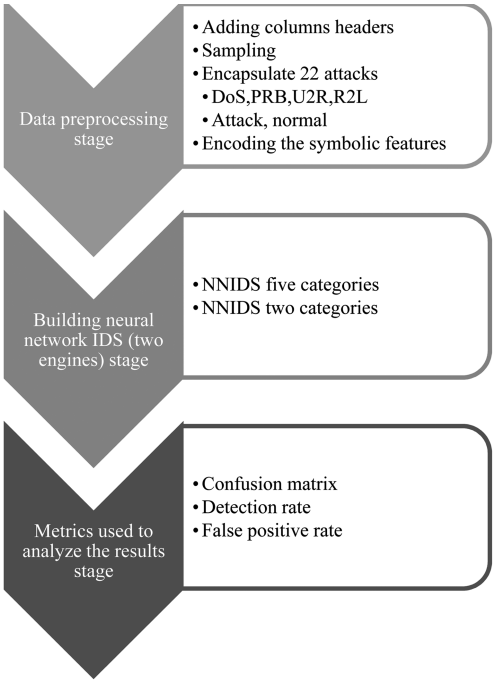

Fig. 1. Proposed system architecture. DoS: denial of service, PRB: Probe, U2R: user to root, R2L: remote to login, IDS: intrusion detection system, NNIDS: neural network intrusion detection system.

2) Sampling: This step is one of the most important steps in our system; the 10% KDD Cup 1999 dataset consists of 494,021 connections records as explained in Table 1. It can be easily calculated from the dataset that the percentage of normal, DoS, PRB, R2L, and U2R connections are 19.691066%, 79.239142%, 0.831341%, 0.227926%, and 0.010526% respectively.

As we can see in Fig. 2, the DoS represents the majority of the dataset followed by the other normal connections, where the rest of the categories represent less than 1% of the training dataset. Thus the neural network model will be over trained in the two major types, taking very long time in learning repeated data, at the same time it will consider minor types as noise due to their negligible percentages in the training dataset. To overcome this problem, the 10% version of the KDD Cup 1999 dataset is sampled in a way that the percentages of the various categories are of comparable value. The proposed system is trained using both the original and the sample dataset.

Table 1. 10% version of the KDD Cup 1999 dataset distributions

Category	Attack name/normal	No. of records	Percentage (%)
Normal	normal connection	97,278	19.691066
DoS (n = 391,458, 79.239142%)	smurf	280,790	56.84
	back	2,203	0.45
	land	21	0.00
	neptune	107,201	21.70
	pod	264	0.05
	teardrop	979	0.20
Probe (n = 4,107, 0.831341%)	ipsweep	1,247	0.25
	nmap	231	0.05
	satan	1,589	0.32
	portsweep	1,040	0.21
R2L (n = 1,126, 0.227926%)	ftp_write	8	0.00
	guess_passwd	53	0.01
	imap	12	0.00
	multihop	7	0.00
	spy	2	0.00
	phf	4	0.00
	warezclient	1,020	0.21
	warezmaster	20	0.00
U2R (n = 52, 0.010526%)	buffer_overflow	30	0.01
	loadmodule	9	0.00
	perl	3	0.00
	rootkit	10	0.00

DoS: denial of service, R2L: remote to login, U2R: user to root.

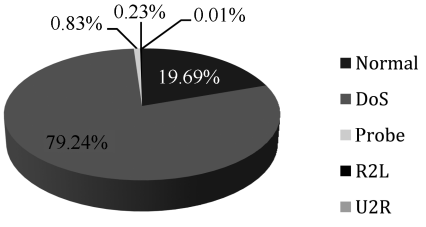

Fig. 2. 10% version of the KDD Cup 1999 dataset distributions. DoS: denial of service, R2L: remote to login, U2R: user to root.

It should be noted from Table 1 that, we have only three records of high value: normal, smurf and neptune. This should be taking into consideration in the sampling process, i.e., only these values should be decreased.

An iterative reduction algorithm is proposed and applied to the (10% version of the KDD Cup 1999 dataset) as follows:

1. Search for the top-3 major values and consider them set S1.
2. The rest of the values are considered as set S2.
3. The maximum value of connection V1 in S2 is selected.
4. The values in S1 are reduced to be comparable to V1 and at the same time their order is maintained.
5. The performance of the system is tested with the new values.
6. Steps 4 and 5 are repeated until optimum results are obtained.

The original KDD Cup 1999 dataset can be divided easily into two categories. The first category (S1) consists of 3 attacks with a massive amount of available samples. The second category (S2) consists of the rest of the attacks; each of them is represented with less than 1% of the dataset's samples. This could make the system treat them as noise and not recognize them as separate classes. So, the reduction algorithm takes a portion of the dataset where all categories are represented by a percentage that cannot be neglected by the classification system while keeping their relative abundance.

Reducing the top 3 records should take into consideration the other category records (S2). The 3 records shouldn't be reduced to values less than the maximum of the other category records (S2). So the record with maximum number of samples must be identified (V1) such that V1 $\in$ S2 where V1 = Max (S2).

The 3 chosen records (S1) will then be reduced iteratively keeping their values larger than V1 such that S1 (i) $\geq$ V1 where $1 \leq I \leq 3$. Also the algorithm tries to maintain the order and proportion of the reduced records compared to the original dataset size especially the "normal" record to keep the false positive alarm rate as low as possible. After each iteration, the performance of the proposed system is measured using the new values generated by the algorithm.

Optimum results are obtained when reducing the normal connections from 97,278 records to 4,000 records, the smurf records from 280,790 records to 4,406 and the Neptune records from 107,201 to 2203 as shown in Table 2.

Figs. 2 and 3 show the distributions of the normal and attack connections in the dataset before and after applying the proposed iterative reduction algorithm. Comparing these figures, it can be shown that the percentage of normal connections is approximately the same, this is important for reducing the FPR. The percentage of DoS records is reduced from 79% to 52% due to the reduction in smurf and neptune but it is still high compared to other categories. The percentages of both probe and R2L records have changed to significant values can now be seen in the dataset. This guarantees that the network will recognize these two types along with the other ones. Regarding the U2R category, this one is considered as host attack. This

Table 2. Our selected KDD Cup 1999 dataset distribution

Category	Attack name/normal	No. of records	Percentage (%)
Normal	Normal connection	4,000	20.66
DoS (n = 10,076, 52.04%)	smurf	4,406	22.76
	back	2,203	11.38
	land	21	0.11
	neptune	2,203	11.38
	pod	264	1.36
	teardrop	979	5.06
PRB (n = 4,107, 21.21%)	ipsweep	1,247	6.44
	nmap	231	1.19
	satan	1,589	8.21
	portsweep	1,040	5.37
R2L (n = 1,126, 5.82%)	ftp_write	8	0.04
	guess_passwd	53	0.27
	imap	12	0.06
	multihop	7	0.04
	spy	2	0.01
	phf	4	0.02
	warezclient	1,020	5.27
	warezmaster	20	0.10
U2R (n = 52, 0.27%)	buffer_overflow	30	0.15
	loadmodule	9	0.05
	perl	3	0.02
	rootkit	10	0.05

DoS: denial of service, PRB: Probe, R2L: remote to login, U2R: user to root.

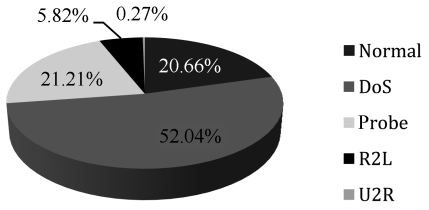

Fig. 3. Our selected KDD Cup 1999 dataset distribution. DoS: denial of service, R2L: remote to login, U2R: user to root.

research focuses only on network attacks, where U2R is considered as a noise in this research. The main advantage of this is reducing the training data from 494,021 records to 19,361 that greatly reduce straining time.

Finally, we will work using two datasets, a 10% version of the KDD Cup 1999 dataset and the proposed reduced dataset. We will compare the performance for both.

3) Encapsulating the 22 Attacks: The next step is to encapsulate the attack names to their categories, here two types of encapsulation are proposed. The first one is to encapsulate the 22 attack names to their four original categories DoS, PRB, R2L and U2L. The second one is to encapsulate the 22 attack names to the word attack.

We will work with two systems; the first one uses the first type of encapsulation and has five categories: DoS, PRB, R2L, U2L and normal. The second system uses the second type of encapsulation and has two categories: attack and normal.

Table 3. Translation of the PROTOCOL_TYPE symbolic feature

	TCP	UDP	ICMP
TCP translation	1	0	0
UDP translation	0	1	1
ICMP translation	0	0	1

TCP: Transmission Control Protocol, UDP: User Datagram Protocol, ICMP: Internet Control Message Protocol.

Table 4. The KDD Corrected testing set

	Normal	DoS	PRB	U2R	R2L
%	19.58	73.9	1.3	0.02	5.2
No.	60,593	229,853	4,166	228	16,189

DoS: denial of service, PRB: Probe, U2R: user to root, R2L: remote to login.

4) Encoding the Symbolic Features: The DARPA 98 KDD99 benchmark dataset has three symbolic features: PROTOCOL_TYPE, service and flag. These features are very important and shouldn't be ignored.

As an example, the Encoding of the PROTOCOL_TYPE feature is shown in Table 3. The rest of the symbolic features are translated in the same way.

C. Building Neural Network IDS Stage

Two systems are built; the first is a system that consists of five categories: DoS, PRB, R2L, U2L and normal. The second one is a system that consists of two categories: attack and normal. Two engines are used: back-propagation [13] and the radial basis function [14, 15]. The structure of the two systems will be explained in subsequent paragraphs. For the testing step, the KDD set was used (Table 4). The KDD corrected testing set contains 311,029 records, including records that describe 15 new attack types [16-19].

1) The Back Propagation Algorithm searches for weight values that minimize the total error of the network over a set of training examples (the training set). It consists of the repeated application of two passes: a forward pass and a backward pass. In the forward pass, the network is activated for one example and the error of each neuron of the output layer is computed. In the backward pass, the network error is used for updating the weights (a credit or blame assignment problem). This process is more complex, because hidden nodes are not directly linked to the error but are linked through the nodes of the next layer. Therefore, starting at the output layer, the error is propagated

backwards through the network, layer by layer. This is achieved by recursively computing the local gradient of each neuron.

The algorithm can be summarized by the following steps [14, 15]:

1. Initialize the weights of the network (often randomly).
2. Present a training sample to the neural network where, in our case, each pattern x consists of 115 features after the translation of the symbolic features.
3. Compare the network's output to the desired output from that sample. Calculate the error for each output neuron.
4. For each neuron, calculate what the output should have been, and a scaling factor i.e. how much lower or higher the output must be adjusted to match the desired output. This is the local error.
5. Adjust the weights of each neuron to lower the local error.

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

With Δw_{ji} computed using the (generalized) Delta rule.

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n)$$

α is the momentum constant $0 \le \alpha < 1$

δ_j is the local gradient of neuron j

$$\delta_j = \begin{cases} \varphi'(v_j)(d_j - y_j) & \text{IF j output node} \\ \varphi'(v_j) \sum_{k \text{ of next layer}} \delta_k w_{kj} & \text{IF j hidden node} \end{cases}$$

For sigmoid activation functions

$$\varphi'(v_j) = a y_j (1 - y_j)$$

where $v_j = \sum_i w_{ji} y_i$

with w_{ji} the weight of the link from node i

to node j and y_i the output of node i

6. Repeat the process from step 3 on the neurons at the previous level, using each one's "blame" as its error.

Feed forward neural networks have complex error surfaces (e.g. plateaus, long valleys etc.) with no single minimum. Adding the momentum term is a simple approach to deal with this problem.

There are two types of network training: incremental mode and batch mode. In incremental mode (on-line or per-pattern training), the weights are updated after presenting each pattern. In batch mode (off-line or per-epoch training), the weights are updated after presenting all the patterns. In the proposed system we used the incremental mode.

2) Radial Basis Function (RBF) Networks are nonlinear hybrid networks typically containing a single hidden layer of processing elements (PEs). This layer uses Gaussian transfer functions, rather than the standard sigmoidal functions employed by MLPs. The centers and widths of the Gaussians are set by unsupervised learning rules, while supervised learning is applied to only the output layer. The Gaussian function responds only to a small region of the input space where the Gaussian is centered [14, 15].

For standard RBF's, the supervised segment of the network only needs to produce a linear combination of the output at the unsupervised layer. Therefore having zero hidden layers is the default setting. Hidden Layers can be added to make the supervised segment a MLP instead of a simple linear perceptron.

It is impossible to suggest an appropriate number of Gaussians, because the number is problem dependent. We know that the number of patterns in the training set affects the number of centers (more patterns implies more Gaussians), but this is mediated by the dispersion of the clusters. If the data is very well clustered, then few Gaussians are needed. On the other hand, if the data is scattered, many more Gaussians are required for good performance.

3) Radial Basis Function (RBF) Algorithm:

- Centersare chosen randomly from the training set.
- Spreads are chosen by normalization.

$$\sigma = \frac{\text{Maximum distance between any 2 centers}}{\sqrt{\text{number of centers}}} = \frac{d_{max}}{\sqrt{m_1}}$$

$$[w_1 \ldots w_{m1}]^T = \Phi^+ [d_1 \ldots d_N]^T$$

- Weights: are computed by means of the pseudo-inverse method.

D. Metrics Used to Analyze the Results

We will use three performance metrics to analyze our results.

- DR is the ratio between the number of correctly detected attacks and the total number of attacks.
- FPR is the ratio between the number of normal connections that are incorrectly misclassified as attacks and the total number of normal connections.
- A confusion matrix (CM) is a visualization tool typically used in supervised learning.

IV. EXPERIMENTAL RESULTS

This section presents the experimental results detailing the performance of the BPNNIDS and the RBFNNIDS. Both systems are trained on a10% version of the KDD Cup 1999 dataset, and the sample we selected from the 10% version of the KDD Cup 1999 dataset. A Pentium 4 (2.33 GHz) laptop, with 2 GB of memory was used to implement the systems.

A. Five Category System using 10% Version of KDD Cup 1999 (FCS 10 KDD)

A four layer neural network (Fig. 4) was used. It has three hidden layers. The size of the input, hidden and output layer are 114, 50, 25, 13, and 5 respectively. 114 is the number of features used in the training and 5 is the number of categories.

We first tried a network architecture of two hidden layers, but this did not converge to a solution. Then we increased the hidden layers to three. The number of neurons in each hidden layer was chosen by trial and error. We started with a size of the first hidden layer at 41 neurons. This is the original number of features in the dataset. Then, this size was increased until optimum results were obtained. Similarly, the size of 2nd hidden and 3rd hidden layers were chosen. For the parameters, the mean square error (MSE) in the training step is 0.001, transfer sigmoid, learning rule momentum, step size 1.0 and momentum 0.7.

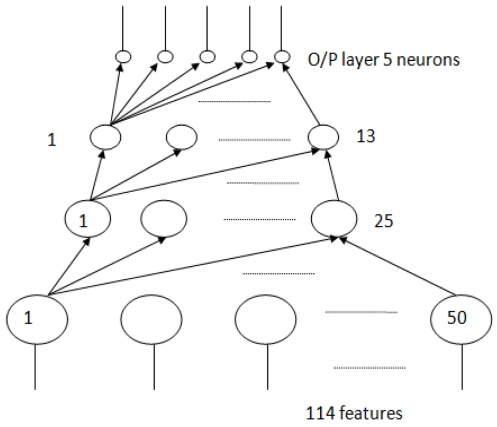

Fig. 4. Five categories system neural networks architecture.

Table 5. Confusion matrix for BPNNIDS trained by a 10% version of the dataset (five category system)

	PRB	R2L	DoS	U2R	Normal
PRB	97.17	0.00	2.02	0.02	0.80
R2L	4.02	70.02	15.04	0.00	10.92
DoS	0.00	0.00	100.00	0.00	0.00
U2R	0.00	38.46	0.00	0.00	61.54
Normal	0.01	0.02	2.05	0.05	97.91

BPNNIDS: back-propagation neural network intrusion detection system, PRB: Probe, R2L: remote to login, DoS: denial of service, U2R: user to root.

The suggested network architecture was trained using a 10% version of the KDD Cup1999 dataset (Table 1) in 40 minutes and 18 seconds, and then tested using the KDD Corrected testing set (Table 4).The resulting confusion matrix is shown in Table 5.

The training of the neural networks is stopped at 20 epochs, with a minimum MSE of 8.63806E-05

The values at the diagonal of the matrix in Table 5 represent the correct detected records. So, the detection rate can be calculated from Tables 4 and 5 by using the following equation:

$$DR = \frac{\text{correct detected attack}}{\text{total number of attack}}$$

$$= \frac{0.97*4166+0.7*16189+229853+0}{4166+16189+229853+228}\%$$

$$= 97.92\%$$

Similarly, the values in the last row of the confusion matrix (Table 5) show the records detected to be normal. The first four values are obviously misclassified as an attack. The false positive rate is calculated as follow:

$$FPR = 0.01 + 0.02 + 2.05 + 0.05 = 2.13\%.$$

B. Five Category System using the Proposed Reduced Dataset (FCS P KDD)

The suggested network architecture is trained using the sample we selected from a 10% version of the KDD Cup 1999 dataset (Table 2) in 3 minutes and 27 seconds, and tested using the KDD Corrected testing set (Table 4).The resulting confusion matrix is shown in Table 6.

Table 6. Confusion matrix for BPNNIDS trained by our selected KDD Cup 1999 dataset (five category system)

	PRB	R2L	DoS	U2R	Normal
PRB	99.85	0.00	0.15	0.00	0.00
R2L	0.40	92.10	6.50	0.00	0.00
DoS	0.00	0.22	99.54	0.00	0.24
U2R	7.69	53.85	26.92	0.00	11.54
Normal	0.00	0.00	0.40	0.00	99.60

BPNNIDS: back-propagation neural network intrusion detection system, PRB: Probe, R2L: remote to login, DoS: denial of service, U2R: user to root.

Table 7. Performance under the two datasets (five category system)

	DR (%)	FPR (%)	Training time
10% KDD	97.92	2.13	40 min 18 sec
The paper dataset	98.97.	0.4	3 min 27 sec

DR: detection rate, FPR: false positive rate.

The training of the neural network is stopped at 22 epochs, with minimum MSE of 0.000531693

Similarly, the DR and FPR can be calculated from Table 4 and Table 6 to be:

$$DR = 98.97\%.$$

$$FPR = 0.4\%.$$

Comparing Performance under the two sets (FCS)

By comparing the two results, we can conclude that the the dataset we selected has an excellent training time of only 3 minutes and 27 seconds (Table 7) with a better detection and false positive rate.

C. Two Category System using 10% Version of Kdd Cup 1999 (Tcs 10 Kdd)

A four layer neural network was used. It has three hidden layers. The sizes of the input, hidden and output layers are 114, 50, 25, 15, and 2 respectively. 114 is the number of features used in the training and 2 is the number of categories. We chose the number of hidden layers and the number of neurons in each layer in a way similar to that used in the five category system.

For the parameters, the mean square error (MSE) in the training step is 0.001, transfer function is the sigmoid, learning rule is the momentum, the step size is 1.0 and momentum is 0.7.

The suggested network architecture is trained using a 10% version of the KDD Cup 1999 dataset (Table 1) in 48 minutes, and tested using the KDD Corrected testing set (Table 4).The resulting confusion matrix is shown in Table 8.

The training of the neural network was stopped at 21 epochs, with minimum MSE of 0.000204713.

Table 8. Confusion matrix for BPNNIDS trained by 10% version of dataset (two category system)

	Attack	Normal
Attack	99.96	0.0398
Normal	2.124	97.876

BPNNIDS: back-propagation neural network intrusion detection system.

Similarly, the DR and FPR can be calculated from Tables 4 and 8 to be:

DR = 99.96%.

FPR = 2.124%.

D. Two Category System using the Proposed Reduced Dataset (TCS P KDD)

The suggested network architecture is trained using the sample we selected from the 10% version of the KDD Cup 1999 dataset (Table 1) in 3 minutes 4 seconds, and tested using the KDD Corrected testing set (Table 4).The resulting confusion matrix is shown in Table 9.

The training of the neural networks is stopped at 22 epochs, with minimum MSE of 0.000556344.

Similarly, the DR and FPR can be calculated from Tables 4 and 9 to be:

DR = 99.91%

FPR = 1.5997%.

Comparing Performance under the two sets (TCS)

By comparing the two results, we can conclude that the dataset we selected has an excellent training time of only 3 minutes and 4 seconds (Table 10) with a better detection and false positive rate.

E. Five Category System using the Proposed Reduced Dataset and RBF (FCS P KDD)

A four layer neural network is used. It has one RBF layer, two hidden layers and one output layer. The size of the input , RBF, hidden and output layers are 114, 114, 25, 13, and 5 respectively.

We first tried a network architecture with the number of centers less than the number of features. However, this did not converge to a solution. Then, we increased the number of centres to be equal to the number of features. The number of neurons in each hidden layer was chosen by trial and error. We started with the size of the first hidden layer to be 20 neurons. Then, this size was increased until optimum results were obtained. Similarly, the sizes of the two hidden layers were chosen.

For the unsupervised learning parameters, the number of centers is 144, the maximum epochs are 100, termination– weight change is 0.001 and learning rate is varied from 0.01 to 0.001.

For the other parameters, the MSE in the training step is 0.001, transfer function is the sigmoid, learning rule is the momentum, the step size is 1.0 and momentum is 0.7.

The suggested network architecture was trained using the paper sample we selected from the 10% version of the KDD Cup 1999 dataset (Table 2) in 36 minutes 14 seconds, and tested using the KDD Corrected testing set (Table 4), the resulting confusion matrix is shown in Table 11.

By comparing Tables 7 and 11, we can conclude that the proposed BPNNIDS has an excellent training time of only 3 minutes and 4 seconds (Table 10) with a better detection and false positive rate.

We compare the performance of the paper proposed BPNNIDS with some of the other neural-network-based approaches, such as K-means NN, SVM, SOM, and PCA. For this purpose, we use the published results in (6, 8, 10, 20). We compare the %DR and %FPR. Table 12 shows the experimental results. Some incomplete items in the published results are represented by '_'. The proposed back-propagation neural network intrusion

Table 9. Confusion matrix for BPNNIDS trained by the our selected KDD Cup 1999 dataset (two category system)

	Attack	Normal
Attack	99.91	0.0929
Normal	1.5997	98.4003

BPNNIDS: back-propagation neural network intrusion detection system.

Table 10. Performance under the two sets (two category system)

	DR (%)	FPR (%)	Training time
10% KDD	99.96	2.124	48 min
The paper dataset	99.91	1.5997	3 min and 4 sec

Table 11. Confusion matrix for RBF trained by the ourselected KDD Cup 1999 dataset (five category system) without stopping

	PRB	R2L	DoS	U2R	Normal
PRB	71.01	0.00	28.99	0.00	0.00
R2L	0.42	36.58	63.00	0.00	0.00
DoS	0.05	0.00	96.95	0.00	3.00
U2R	0.00	0.00	100	0.00	0.00
Normal	0.04	0.00	14.90	0.00	85.06

RBF: radial basis function, PRB: Probe, R2L: remote to login, DoS: denial of service, U2R: user to root.

Table 12. Comparison with the previous work

Technique	DR (%)	FPR (%)	Time
K-means neural network [20]	92	6.21	28 min 21 sec
Support vector machine [10]	98	10	-
Principal component analysis [8]	93.83	6.16	26 min 56 sec
Self-organizing map [6]	90.14	1.4	-
BPNNIDS-FCS-10%	97.92	2.13	40 min 18 sec
BPNNIDS-FCS-P	98.97	0.4	3 min 27 sec

BPNNIDS: back-propagation neural network intrusion detection system, FCS: five category system

detection system (BPNNIDS) achieves a higher DR and lower FPR than all the other listed systems in less time.

V. CONCLUSIONS

In this paper, a supervised learning approach to the intrusion detection problem is investigated and demonstrated on the International Knowledge Discovery and Data Mining Tools Competition intrusion detection benchmark (the KDDCUP 99 dataset). To do so this we investigated two architectures; the first engine is a back-propagation neural network intrusion detection system (BPNNIDS) and the second is a RBFNNIDS. The two engines work under two basic data sets; one is limited to 19361 connections (records), which is the set we selected from the 10% version of the KDD Cup 1999 dataset whereas the other contains 494021 connections (records), which is the 10% version of the KDD Cup 1999 dataset.

The significance of this paper's iterative reduction algorithm is to reduce the training time from 40 minutes 18 seconds to 3 minutes 27 seconds in the five category system. It also reduces the training time from 48 minutes to 3 minutes and 4 seconds in the two category system.

Our systems include two types of encapsulation. The first one encapsulates the 22 attack types to their four original categories DoS, PRB, R2L, and U2L. The second one encapsulates the 22 attack types to the word attack. Two systems are introduced; the first one uses the proposed five category encapsulation: DoS, PRB, R2L, U2L, and normal. The second system uses the proposed two category encapsulation: attack and normal.

The proposed iterative reduction algorithm, encoding the symbolic features and the complexity architecture of the proposed neural networks had a great effect on ensuring a high DR with a low FPR. The DR was 98.97% and FPR was 0.4% in the five category system that used the back-propagation neural networks engine with the reduced dataset. The DR was 97.92% and FPR was 2.13% in the five category system that used the back-propagation neural networks engine with the 10% version of the KDD Cup 1999 dataset. The DR was 99.91% and FPR was 1.5997% in the two category system that used the back-propagation neural networks engine with the reduced dataset. The DR was 99.96% and FPR was 2.124% in the two category system that used the back-propagation neural networks engine with the 10% version of the KDD Cup 1999 dataset.

This paper introduced two types of machine learning in a supervised environment. The first one depends on back-propagation neural networks and the second one depends on the radial basis function. By comparing the training time, the detection rate and the false positive rate it can be concluded that the engine with the back-propagation neural networks produced better results than the one using the radial basis function.

REFERENCES

1. R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "1999 DARPA off-line intrusion detection evaluation," *Computer Networks*, vol. 34, no. 4, pp. 579-595, 2000.
2. D. Anderson, T. Frivold, and A. Valdes, Next-Generation Intrusion Detection Expert System (NIDES): a summary. SRI Technical Report No.: SRI-CSL-95-07, Menlo Park, CA: SRI International Computer Science Laboratory, 1995.
3. K. R. Kendall, "A database of computer attacks for the evaluation of intrusion detect systems," MS Thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999.
4. W. H. Chen, S. H. Hsu, and H. P. Shen, "Application of SVM and ANN for intrusion detection," *Computers and Operations Research*, vol. 32, no. 10, pp. 2617-2634, 2005.
5. T. Horeis, "Intrusion detection with neural networks combination of self-organizing maps and radial basis function networks for human expert integration," http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.191&rep=rep1&type=pdf.
6. H. Gunes Kayacik, A. Nur Zincir-Heywood, and M. I. Heywood, "A hierarchical SOM-based intrusion detection system," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 4, pp. 439-451, 2007.
7. G. Liu, Z. Yi, and S. Yang, "A hierarchical intrusion detection model based on the PCA neural networks," *Neurocomputing*, vol. 70, no. 7-9, pp. 1561-1568, 2007.
8. R. Beghdad, "Critical study of neural networks in detecting intrusions," *Computers and Security*, vol. 27, no. 5-6, pp. 168-175, 2008.
9. K. Shafi and H. A. Abbass, "An adaptive genetic-based signature learning system for intrusion detection," *Expert Systems with Applications*, vol. 36, no. 10, pp. 12036-12043, 2009.
10. E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: detecting intrusions in unlabeled data," *Applications of Data Mining in Computer Security. Advances in Information Security Vol. 6*, D. Barbara and S. Jajodia, Eds., Boston, MA: Kluwer Academic Publishers, 2002, pp. 77-101.
11. M. S. Mok, S. Y. Sohn, and Y. H. Ju, "Random effects logistic regression model for anomaly detection," *Expert Systems with Applications*, vol. 37, no. 10, pp. 7162-7166, 2010.
12. G. Wang, J. Hao, J. Mab, and L. Huang, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6225-6232, 2010.
13. A. H. Fares, M. I. Sharrawy, and H. H. Zayed, "A fast intrusion detection technique based on machine learning," *Proceedings of the 35th International Conference for Statistics and Computer Science and Its Applications*, Cairo, Egypt, April 11-22, 2010.
14. S. S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd ed., Upper Saddle River, NJ: Prentice Hall, 1999.
15. K. Mehrotra, C. K. Mohan, and S. Ranka, Elements of Artificial Neural Networks, Cambridge, MA: MIT Press, 1997.
16. Information Systems Technology Group, "The 1998 intrusion detection off-line evaluation plan information," http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html.
17. The KDD cup 1999 data set, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
18. S. Hettich, S. D. Bay, "The UCI KDD archive," http://kdd.ics.uci.edu.
19. KDD data set task and features categories details, http://kdd.ics.uci.edu/databases/kddcup99/task.html.
20. K. M. Faraoun and A. Boukelif, "Neural networks learning improvement using the K-means clustering algorithm to detect network," *International Journal of Computational Inteligence*, vol. 3, no. 2, pp. 161-168, 2006.

Hala H. Zayed

Hala H. Zayed received the BSc in electrical engineering (with honor degree) in 1985, the MSc in 1989 and PhD in 1995 from Zagazig university (Benha Branch) in electrical and communication engineering. She is now an associate professor at faculty of computers and informatics, Benhauniversity. Her areas of research are pattern recognition, content based image retrieval, biometrics and image processing.

Mohamed I. Sharawy

Mohamed I. Sharawy received BSc in Engineering – Electronics and communications in 1986, MSc in Engineering – Electronics and communications in 1990, PhD in Engineering – Electronics and Computer Engineering in 1998. He is now an associate professor in Shoubra Faculty of Engineering – Benha University.

Ahmed H. Fares

Ahmed H. Fares received the BSc in electrical engineering- computer system engineering (with honor degree) in 2005 from Benha University, the Mscin computer system engineering in 2010 from Benha University. He is now an assistant lecturer in Shoubra Faculty of Engineering – Benha University.