A Novel Soft Computing Model Using Adaptive Neuro-Fuzzy Inference System for Intrusion Detection

Adel Nadjaran Toosi, Mohsen Kahani

Abstract—**The main purpose of this paper is to incorporate several soft computing techniques into the classifying system to detect and classify intrusions from normal behaviors based on the attack type in a computer network. Some soft computing paradigms such as neuro-fuzzy networks, fuzzy inference approach and genetic algorithms are investigated in this work. A set of neuro-fuzzy classifiers are used to perform an initial classification. The fuzzy inference system would then be based on the outputs of neuro-fuzzy classifiers, making decision of whether the current activity is normal or intrusive. As a final point, in order to attain the best result, a genetic algorithm optimizes the structure of the fuzzy decision engine. The experiments and evaluations of the proposed method were done with the KDD Cup 99 intrusion detection dataset.**

I. Introduction

With the widespread use of computer networks, the number of attacks has grown extensively, and many new hacking tools and intrusive methods have appeared. Using an intrusion detection system (IDS) is one way of dealing with suspicious activities within a network.

An intrusion detection system monitors the activities of a given environment and decides whether these activities are malicious (intrusive) or legitimate (normal) based on system integrity, confidentiality and the availability of information resources. The intrusion detection system collects information about the system being observed. This collected audit data is processed by the detector. The detector eliminates unnecessary information from the audit data and then makes a decision to evaluate the probability that these activities can be considered as a sign of an intrusion [1].

Soft computing is an innovative approach to construct a computationally intelligent system which parallels the extraordinary ability of the human mind to reason and learn in an environment of uncertainty and imprecision. Typically, soft computing consists of several computing paradigms, including neural networks, fuzzy sets, approximate reasoning, genetic algorithms, simulated annealing, and etc. Many soft computing approaches have been applied to the intrusion detection field [2], [3], [4], [5], [6], [7]. In this paper, a novel intrusion detection system based on the integration of a few soft computing methods including neuro-fuzzy, fuzzy and genetic algorithms is described. The key contribution of this work is the utilization of outputs of neuro-fuzzy network as linguistic variables which expresses how reliable current output is.

Fuzzy logic, as a robust soft computing method, has demonstrated its ability in intrusion detection systems [4], [5], [6]. Moreover, fuzzy systems have several important features which make them suitable for intrusion detection [5]. Most Fuzzy systems make use of human expert knowledge to create their fuzzy rule base and hence lack adaptation, though. Therefore, building fuzzy systems with learning and adaptation capabilities has recently received much attention [6]. Various methods have been suggested for automatic generation and adjustment of fuzzy rules without using the aid of human experts; the neural fuzzy [7] and genetic fuzzy are two most successful approaches in this regard.

From the view point of classification, the main work of building an intrusion detection system is to build a classifier that can categorize normal and intrusive event data from the original dataset. ANFIS as an Adaptive Neuro-Fuzzy Inference System [7] has the ability to construct models solely based on the target system sample data. This ability among others qualifies ANFIS as a fuzzy classifier for intrusion detection.

The proposed system has different layers which correspond to the needs in various modules of the proposed IDS system. First of all, several neuro-fuzzy classifiers use extracted features of the audit data to classify activities in the network. In this case fuzzy inference system- as a decision-making engine based on outputs of the classifiers of previous layer- makes the final decision on whether the current activity is normal or intrusive. Finally, genetic algorithms are employed to optimize the structure of fuzzy sets of the fuzzy decision-making engine.

In order to promote the comparison of different works in IDS area, the Lincoln Laboratory at MIT, under the Defense Advanced Research Project Agency (DARPA) and Air Force Research Laboratory (AFRL/SNHS) sponsorship, constructed and distributed the first standard dataset for evaluation of computer network IDS [8]. Afterward the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining with the purpose of

Manuscript received October 9, 2006. This work was supported in part by Iran Telecommunication Research Center (ITRC).

A. Nadjaran Toosi, was with Communication and Computer Research Laboratory, Ferdowsi University of Mashhad, Iran. He is now with the Department of Computer, Islamic Azad University, Mashhad Branch, Iran. (email: ad_na85@stu-mail.um.ac.ir, NadjaranToosi@mshd.iau).

M. Kahani, is associate professor with the Department of Computer, Ferdowsi University of Mashhad, Iran (e-mail: Kahani@um.ac.ir).

demonstrating the learning contest, collected and generated TCP dump data provided by the aforementioned DARPA in the form of train-and-test sets whose features are defined for the connection records (a connection is a sequence of TCP packets starting and ending at some well-defined times). The main goal of the learning contest was to select classifiers with the best qualifications of recognizing normal and intrusive connections. The above dataset is named as KDD cup 99 dataset [9] here, and has been used for the experiments.

The subsequent parts of this paper are organized as follows: At first, in section 2, KDD cup 99 dataset on which the experiments are conducted briefly reviewed. Next, at section 3 and 4, the proposed system is explained and experimental results as well as evaluation of the proposed approach are discussed respectively. Finally, section 5 makes some concluding remarks and proposes further areas for future research.

II. KDD CUP 99 DATASET

The KDD cup 99 dataset includes a set of 41 features derived from each connection and a label which specifies the status of connection records as either normal or specific attack type. The list of these features can be found in appendix A. These features had all forms of continuous, discrete, and symbolic, with significantly varying ranges falling in four categories [9]: intrinsic features of a connection, the content features, the same host features and the similar same service features.

Likewise, attacks fall into four main categories [9]: DoS (Denial of Service), R2L (Remote to Local), U2R (User to Root) and Probe.

KDD dataset is divided into training and testing record sets. Total number of connection records in the training dataset is about 5 million records. This is too large for our purpose; as such, only concise training dataset of KDD, known as 10% training dataset, was employed here. The distribution of normal and attack types of connection records in this subset have been summarized in Table 1.

As it can be seen in table 1, sample distributions for different categories of attacks in training data differ significantly from each other. One of the main contributions of this work is to overcome this issue by using different classifier for each class of data.

The test data enjoys a different distribution. Moreover, the test data includes additional attack types not present in the training data which makes classifying more complicated. Table 2 summarizes the distribution of normal and attack types of connection records in the test dataset. And Table 3, based on major types of attack, shows the distribution of novelty in the test dataset.

TABLE I
THE SAMPLE DISTRIBUTIONS ON THE SUBSET OF 10% DATA OF KDD CUP 99 DATASET

Class	Number of Samples	Samples Percent
Normal	97277	19.69%
Probe	4107	0.83%
DoS	391458	79.24%
U2R	52	0.01%
R2L	1126	0.23%
	492021	100%

TABLE II
THE SAMPLE DISTRIBUTIONS ON THE TEST DATA WITH THE CORRECTED LABELS OF KDD CUP 99 DATASET

Class	Number of Samples	Samples Percent
Normal	60593	19.48%
Probe	4166	1.34%
DoS	229853	73.90%
U2R	228	0.07%
R2L	16189	5.20%
	311029	100%

TABLE III
THE NOVEL ATTACK SAMPLE DISTRIBUTIONS ON THE TEST DATA WITH THE CORRECTED LABELS OF KDD CUP 99 DATASET

Class	Number of Novel Attack Samples	Total Number of Samples	Samples Percent
Probe	1789	4166	43%
DoS	6555	229853	3%
U2R	189	228	83%
R2L	10196	16189	63%
	18729	250436	7.5%

III. PROPOSED SYSTEM

The proposed system is discussed in details in this section. First, the system architecture is explained. Then, data sources, selected from KDD for training the system, are introduced. Afterward, layers of proposed framework are presented in more details.

A. System Architecture

The proposed architecture for the Evolutionary Soft Computing Intrusion Detection System includes two layers. In the first layer, there are five ANFIS modules which are trained to explore the intrusive activity from the input data. Each ANFIS module belongs to one of the classes in the dataset each providing an output which specifies the degree of relativity of the data to the specific class. 1 shows total membership while -1 is used otherwise. (It is important to mention that the ANFIS structure has only one output). The most important motivation to using ANFIS in this way is that

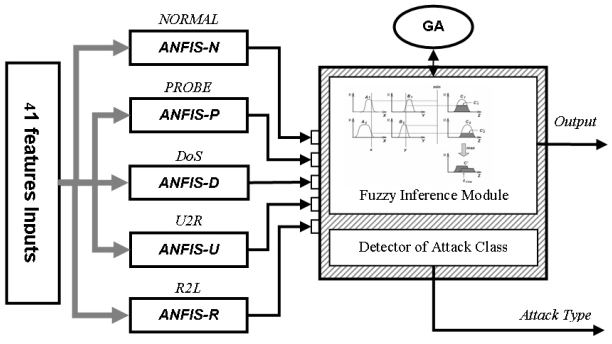

Fig. 1. System architecture block diagram

ANFIS is usually more appropriate as a binary classifier rather than a multi-classifier.

Secondly, a Fuzzy Inference module, based on empirical knowledge, is employed to make the final decision for recognition. The fuzzy inference module implements nonlinear mappings from the outputs of the neuro-fuzzy classifiers of the pervious layer to the final output space which specifies if the input data are normal or intrusive. Afterward, if the system recognizes that the current pattern is intrusive by nature, the classifier of the first layer, in which the output is the nearest value among all classifiers, specifies the class of the attack.

Finally, In order to attain the best results, genetic algorithm (GA) is used to optimize the structure of the fuzzy decision-making engine. The GA structure is discussed in more depth later. Figure 1 depicts the schematic block diagram of the proposed system architecture.

B. The Data Sources

All of the above features have been applied to the inputs of the five neuro-fuzzy classifiers. From the classification point of view, any system mainly consists of two phases: 1) the training of the parameters of the classifier according to the training dataset and 2) using the classifier to categorize a test dataset. Here, 10% of the training dataset was used as the source of the training dataset. Since the number of records in the 10% data set was still very large for our purposes, different subsets of the training and checking dataset were randomly selected from the subset of 10% of data, for the training phase. The basic idea behind using a checking dataset for model validation is that after a certain point in training, the model begins over fitting the training dataset. If over fitting does occur, we cannot expect the classifier to respond well to other independent datasets. In fact, if checking data is used for ANFIS training, the final FIS associated with the minimum checking error will be chosen.

Results of different machine learning algorithms show that anomaly detectors do better than signature-based detectors for KDD cup 99 dataset [15]. This might be because the testing data has substantial new attacks with signatures not correlated with similar attacks in the training data. On the other hand, the number of training samples for signature-based detectors seems not to be ample to develop classifiers to function as efficiently as possible. However, the attack samples in the testing dataset have rather enough deviation from normal or regular samples in the training dataset [13] [15].

Since each classifier in first layer of the system acts as a signature based classifiers and the goal is to select a good training and checking data set for the learning phase, training and checking data set has been selected, as shown on the tables 4 and 5, wherein numbers of samples in the normal class are approximately equal to the summation of the samples in the other classes. By this policy, in view of the fact that each classifier performs as binary classifier (current activity belongs to this class or not), each classifier somehow acts as an anomaly detector system.

The distribution of the samples in the two subsets that were used for the training is listed on Table 4 and 5. Selected subsets enjoy different numbers of samples, the smaller one contains a few number of samples to show the system is still capable, despite the fact that a small portion of the training data has been used. The other one in more number of samples enjoys more number of samples to illustrate efficiency of proposed system as much as possible.

Due to the reduction of random sampling effects, ten trails with the same distribution have been selected for each subset

TABLE IV
SAMPLE DISTRIBUTIONS ON THE *FIRST TRAINING AND CHECKING* DATA RANDOMLY SELECTED OF 10% DATA OF KDD CUP 99 DATASET

		Normal	Probe	DoS	U2R	R2L
ANFIS-N	Training	20000	4000	15000	40	1000
	Checking	2500	107	2000	12	126
ANFIS-P	Training	10000	4000	5000	40	1000
	Checking	1000	107	500	12	126
ANFIS-D	Training	25000	4000	20000	40	1000
	Checking	6000	107	5000	12	126
ANFIS-U	Training	200	50	50	46	50
	Checking	100	25	25	6	25
ANFIS-R	Training	4000	1000	2000	40	1000
	Checking	2000	500	1000	12	126

TABLE V
DISTRIBUTION OF SAMPLES ON THE *SECOND TRAINING AND CHECKING* DATA RANDOMLY SELECTED OF 10% DATA OF KDD CUP 99 DATASET

		Normal	Probe	DoS	U2R	R2L
ANFIS-N	Training	1500	500	500	52	500
	Checking	1500	500	500	0	500
ANFIS-P	Training	1500	500	500	52	500
	Checking	1500	500	500	0	500
ANFIS-D	Training	1500	500	500	52	500
	Checking	1500	500	500	0	500
ANFIS-U	Training	1500	500	500	46	500
	Checking	1500	500	500	6	500
ANFIS-R	Training	1500	500	500	52	500
	Checking	1500	500	500	0	500

of trainings (Training sets in table 4 and 5). Therefore, all the evaluation results in the latter parts of the paper have been computed over these ten trials except those explicitly mentioned.

Before concluding this subsection, it should be mentioned that to be fair, we did not have any access to the data-test during the training and optimization phase. Moreover, the standard conditions of the KDD cup competition have been deployed.

C. The Neuro-Fuzzy Classifiers

The subtractive clustering method with r_a=0.5 (neighborhood radius) has been used to partition the training sets and generate an FIS structure for each ANFIS. For further fine-tuning and adaptation of membership functions, training sets were used for training ANFIS. Each ANFIS trains at 50 epochs of learning and final FIS that is associated with the minimum checking error has been chosen. All the MFs of the input fuzzy sets were selected in the form of Gaussian functions with two parameters.

D. The Fuzzy Decision Module

The fuzzy inference module has five inputs, obtained from the output values of each ANFIS classifiers. The fuzzy inference module, based on these inputs, determines whether the current connection record is an attack or not.

A five-input, single-output of Mamdani fuzzy inference system with centroid of area defuzzification strategy was used for this purpose. Each input fuzzy set includes two MFs and all the MFs are Gaussian functions which are specified by four parameters. The proposed fuzzy inference module uses the rules shown in the fuzzy associative memory in Table 6. The output of the fuzzy inference engine, which varies between -1 and 1, specifies how intrusive the current record is, 1 to show completely intrusive and -1 for completely normal. Records with positive intrusive values are selected as intrusive patterns. After an attack is detected, its class is selected based on the ANFIS module class which returns the highest value.

E. Genetic Algorithm Module

The genetic algorithm repeatedly modifies a population (a set of individual) by a set of genetic operators including mutation, crossover, and selection. It selects individuals evolving toward an optimal solution from the current population and uses them to produce children of the next generation. The algorithm stops when the stopping criterion is met. In the proposed system, each individual (chromosome) has genes codifying parameters of the MFs of the input fuzzy set of the fuzzy decision engine. A chromosome consists of 320 bits of binary data. Each 8 bits of a chromosome determines one parameter out of the four parameters of an MF. Figure 2 illustrates the decoding process of each individual chromosome.

TABLE VI
FUZZY ASSOCIATIVE MEMORY FOR THE PROPOSED FUZZY INFERENCE RULES

Normal	PROBE	DoS	U2R	R2L	Output
High	-	-	-	-	Normal
-	¬High	¬High	¬High	¬High	Normal
-	High	-	-	-	Attack
-	-	High	-	-	Attack
-	-	-	High	-	Attack
-	-	-	-	High	Attack
Low	-	-	-	-	Attack
-	Low	Low	Low	Low	Normal

TABLE VII
THE SAMPLE DISTRIBUTIONS ON THE SELECTED SUBSET OF 10% DATA OF KDD CUP 99 DATASET FOR THE OPTIMIZATION PROCESS WHICHI IS USED BY GA

	Normal	Probe	DoS	U2R	R2L
Number of Samples	200	104	200	52	104

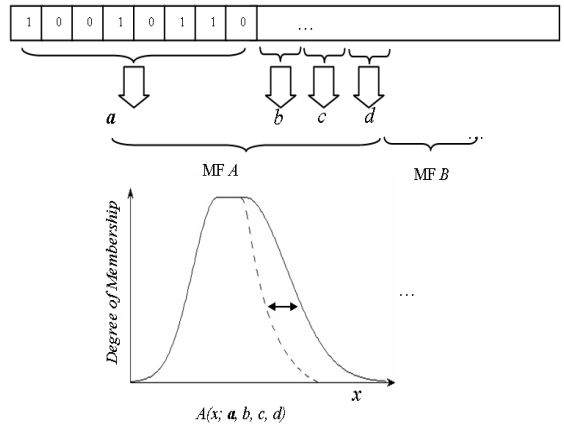

Fig. 2. Schematic decoding process of the individual chromosome

The genetic algorithm, which is used here to optimize the input MFs of the fuzzy decision-making module, uses a subset selected from 10% of KDD dataset for the optimization process. The distribution of samples for this subset is shown in Table 7.

In view of the fact that GA optimization process does not always provide an absolute response, the optimization phase was performed three times and the average of the experiments results was computed for each attained structures. Also, due to the reduction of the effects of randomly sampling, five different trails of subsets- not overlapping with each other- have been used for this phase.

The fitness function evaluates the fitness value for each individual. Fundamentally, the fitness function is the function that should be optimized. This works considers two different fitness functions.

Before discussing more about the fitness functions, it seems necessary to talk about standard metrics that has been developed for evaluating network intrusion detections. *Detection rate* and *false alarm rate* are the two most famous metrics that have already been used. Detection rate is computed as the ratio between the number of correctly detected attacks and the total number of attacks, while false alarm (false positive) rate is computed as the ratio between the number of normal connections that is incorrectly misclassified as attacks and the total number of normal

connections. Another metric used here is the *classification rate*. Classification rate for each class of data is defined as the ratio between the number of test instances correctly classified and the total number of test instances of this class.

For the purpose of classifier algorithm evaluation, another comparative measure is defined which is "Cost per Example" (CPE) [12].

CPE is calculated using the following formula:

$$CPE = \frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{m} CM(i,j) * C(i,j) \qquad (1)$$

Where *CM* and *C* are confusion matrix and cost matrix, respectively, and *N* represents the total number of test instances, *m* is the number of the classes in classification. A confusion matrix is a square matrix in which each column corresponds to the predicted class, while rows correspond to the actual classes. An entry at row *i* and column *j*, *CM*(*i, j*), represents the number of misclassified instances that originally belong to class *i*, although incorrectly identified as a member of class *j*. The entries of the primary diagonal, *CM(i,i)*, stand for the number of properly detected instances. Cost Matrix is similarly defined, as well, and entry *C(i,j)* represents the cost penalty for misclassifying an instance belonging to class *i* into class *j*.

Cost Matrix values employed for the KDD'99 Classifier Learning Contest are shown in Table 8(a) [9]. Lower values for cost per example measure show better classification for the intrusion detection system.

1) Fitness Functions

This work considers two different fitness functions. The First fitness function considered here, represents the baseline case in which a Cost per Example with equal misclassification costs (Table 8(b)) is employed. The genetic algorithm used to minimize the cost per examples is calculated in this way. Using the mentioned fitness function resolves the trade-off between detection rate and false alarm rate and leads to maximizing the overall detection rate and classification rate with low false alarm rate.

Another fitness function is employed based on the cost per examples used for evaluating results of the KDD'99 competition [9]. Using the Cost Matrix values employed for the KDD'99 Classifier Learning Contest attained the best classification rate with respect to weighed misclassification cost.

TABLE VIII
CHARACTERISTICS OF COST MATRIX; THE COLUMNS CORRESPOND TO PREDICTED CLASSES, ROWS CORRESPOND TO ACTUAL CLASSES. (A)COST MATRIX VALUES FOR THE KDD'99 CLASSIFIERS LEARNING CONTEST. (B)COST MATRIX VALUES WITH EQUAL MISCLASSIFICATION COSTS

		Predicted				
		Normal	PROBE	DoS	U2R	R2L
Actual	Normal	0	1	2	2	2
	PROBE	1	0	2	2	2
	DoS	2	1	0	2	2
	U2R	3	2	2	0	2
	R2L	4	2	2	2	0

(A)

		Predicted				
		Normal	PROBE	DoS	U2R	R2L
Actual	Normal	0	1	1	1	1
	PROBE	1	0	1	1	1
	DoS	1	1	0	1	1
	U2R	1	1	1	0	1
	R2L	1	1	1	1	0

(B)

IV. RESULTS

All samples of correctly labeled test dataset of KDD cup 99 dataset (Table 2) as the testing data to evaluate the classifiers.

Before discussing the result, it should be mentioned that to perform the experiments, the structures obtained from 10 subsets of training data for both series were used for the classifiers. The genetic algorithm was performed three times, each time for one of the five series of selected subsets. Totally 150 different structures were used and the result is the average of the results of this 150 structures.

In the rest of this section, the performance of the proposed Evolutionary Soft Computing Intrusion Detection System (ESC-IDS) using two different training datasets (Tables 4 and 5) and two different fitness functions is compared. Two different training datasets for training the classifiers and two different fitness functions to optimize the fuzzy decision-making module were used. Table 9 shows the notation used for the special versions of ESC-IDS.

Table 10 shows results for the versions of ESC-IDS having structured based on the training set 2 with total training samples around 30000 patterns, which already contains repeated samples and far less than total number of samples in the original training set.

The performance of the ESC-IDS has been compared with some other machine learning methods tested on the KDD dataset and is shown in Table 11. The proposed method demonstrates better performances in a number of attacks categories and an unprecedented cost per examples of 0.1579. Based on the results shown in the Table 11, it can be easily seen that the proposed approach has a good performance for detecting intrusion in computer networks. Also, this method is flexible and can be adjusted for special situations using different fitness functions.

It should be noted that some values of Table 11 can be misleading. Parzen-window and RSS-DSS are anomaly detection methods that only detect if a connection record is intrusive or not, and do not have any information regarding the attack type. For such systems that do not classify intrusions in some specific groups of attacks, correct classification concept is different from others. In classifying

TABLE IX
ABBREVIATIONS USED FOR OUR APPROACHES

Abbreviation	Approach
ESC-KDD-1	First Training set with fitness function of KDD
ESC-EQU-1	First Training set with fitness function of equal misclassification cost
ESC-KDD-2	Second Training set with fitness function of KDD
ESC-EQU-2	Second Training set with fitness function of equal misclassification cost

TABLE X
CLASSIFICATION RATE, DETECTION RATE (DTR), FALSE ALARM RATE (FA) AND COST PER EXAMPLE OF KDD (CPE) FOR THE DIFFERENT APPROACHES OF ESC-IDS ON THE TEST DATASET WITH CORRECTED LABELS OF KDD CUP 99 DATASET

Model	Normal	Probe	DoS	U2R	R2L	DTR	FA	CPE
ESC-KDD-1	98.2	84.1	99.5	14.1	31.5	95.3	1.9	0.1579
ESC-EQU-1	98.4	89.2	99.5	12.8	27.3	95.3	1.6	0.1687
ESC-KDD-2	96.5	79.2	96.8	8.3	13.4	91.6	3.4	0.2423
ESC-EQU-2	96.9	79.1	96.3	8.2	13.1	88.1	3.2	0.2493

TABLE XI
CLASSIFICATION RATE, DETECTION RATE (DTR), FALSE ALARM RATE (FA) AND COST PER EXAMPLE OF KDD (CPE) FOR THE DIFFERENT ALGORITHMS PERFORMANCES ON THE TEST DATASET WITH CORRECTED LABELS OF KDD CUP 99 DATASET (N/R STANDS FOR NOT REPORTED)

Model	Normal	Probe	DoS	U2R	R2L	DTR	FA	CPE
ESC-IDS	98.2	84.1	99.5	14.1	31.5	95.3	1.9	0.1579
RSS-DSS[2]	96.5	86.8	99.7	76.3	12.4	94.4	3.5	n/r
Parzen-Window[14]	97.4	99.2	96.7	93.6	31.2	n/r	2.6	0.2024
Multi-Classifier [13]	n/r	88.7	97.3	29.8	9.6	n/r	n/r	0.2285
Winner of KDD [10]	99.5	83.3	97.1	13.2	8.4	91.8	0.6	0.2331
Runner Up of KDD [11]	99.4	84.5	97.5	11.8	7.3	91.5	0.6	0.2356
PNrule [12]	99.5	73.2	96.9	6.6	10.7	91.1	0.4	0.2371

system with attack type recognition, while a record has been corrected recognized as an intrusion, misclassification among attack groups is considered as an error. Looking at results, shows that the proposed system has correctly identified an intrusive record, while might have had problem classifying it.

It can be stated that all the machine learning algorithms tested on the KDD'99 dataset offered an acceptable level of detection performance only for DoS and PROBE attack categories and demonstrated poor performance on the U2R and R2L categories[15]. The proposed method shows improvement in these two classes (U2R and R2L).

V. CONCLUSION

In this paper, an evolutionary soft computing approach for intrusion detection was introduced and was successfully demonstrated its usefulness on the training and testing subset of KDD cup 99 dataset. The ANFIS network was used as a neuro-fuzzy classifier for intrusion detection. ANFIS is capable of producing fuzzy rules without the aid of human experts. A fuzzy decision-making engine was developed to make the system more powerful for attack detection, using the fuzzy inference approach. At last, this paper proposed a method to use genetic algorithms to optimize the fuzzy decision-making engine. Experimentation results showed that the proposed method is effective in detecting various intrusions in computer networks. Our future work will focus on reducing features for the classifiers by methods of feature selection. Also, the work will be continued to study the fitness function of the genetic algorithm to manipulate more parameters of the fuzzy inference module, even concentrating on fuzzy rules themselves.

REFERENCES

[1] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Department of Computer Engineering, Chalmers University, Tech. Rep. 99-15, 2000.

[2] D. Song, M.I. Heywood, A.N. Zincir-Heywood, Training Genetic Programming on Half a Million Patterns: An Example from Anomaly Detection. *IEEE Transactions on Evolutionary Computation*, 9(3), 225-239, 2005.

[3] J. Gomez, D. Dasgupta, "Evolving Fuzzy Classifiers for Intrusion Detection," in *Proceeding of 2002 IEEE Workshop on Information Assurance, United States Military Academy*, West Point NY, USA, 68-75, 2001.

[4] K. Shah, N. Dave, S. Chavan, S. Mukherjee, A. Abraham and S. Sanyal, "Adaptive Neuro-Fuzzy Intrusion Detection System," in *Proceeding of IEEE International Conference on Information Technology: Coding and Computing (ITCC' 04),* USA, 5-7 April 2004, IEEE Computer Society, Vol. 1, 70-74, 2004.

[5] J. E. Dickerson, J. Juslin, O. Koukousoula,J. A. Dickerson, "Fuzzy Intrusion Detection," in *proceeding of IFSA World Congress and 20th North American Fuzzy Information Processing Society* (NAFIPS) International Conference. North American Fuzzy Information Processing Society (NAFIPS), Vancouver, Canada, 25-28 July 2001, IEEE Computer Society, Vol. 3, 1506–1510, 2001.

[6] M. S. Abadeh, J. Habibi, C. Lucas, Intrusion detection using a fuzzy genetics-based learning algorithm. *Journal of Network and Computer Applications*, 2005.

[7] J.-S. R. Jang, ANFIS: Adaptive-Network-based Fuzzy Inference Systems, *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3), 665-685, 1993.

[8] DARPA Intrusion Detection Evaluation: http://www.ll.mit.edu/SSt/ideval/result/result_index.html.

[9] KDD Cup 1999 Intrusion detection dataset: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[10] Pfahringer B., Winning the KDD99 Classification Cup: Bagged Boosting. *SIGKDD explorations*, 1(2), 65-66, 2000.

[11] I. Levin, KDD-99 Classifier Learning Contest LLSoft's Results Overview. *SIGKDD Explorations*, *ACM SIGKDD*, 1(2) 67-75, 2000.

[12] R. Agarwal, M. V. Joshi, 2000. "PNrule: A New Framework for Learning classifier Models in Data Mining," Department of Computer Science, University of Minnesota, Report No. RC-21719.

[13] M. R. Sabhnani, G. Serpen "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context," in *Proceedings of International Conference on Machine Learning: Models, Technologies, and Applications*, Las Vegas, Nevada, USA, 209-215, 23-26 June 2003.

[14] D.Y. Yeung, C. Chow, "Parzen-window Network Intrusion Detectors," in *Proceeding of 16th International Conference on Pattern Recognition*, 11-15 IEEE Computer Society, Vol. 4, 385-388, August, 2002.

[15] M. R. Sabhnani, G. Serpen, Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set, *Journal of Intelligent Data Analysis*, 8(4), 403-415, 2004.