Proceedings of the 2002 IEEE
Workshop on Information Assurance
United States Military Academy, West Point, NY June 2001

Evolving Fuzzy Classifiers for Intrusion Detection

Jonatan Gomez and Dipankar Dasgupta

Abstract – The normal and the abnormal behaviors in networked computers are hard to predict, as the boundaries cannot be well defined. This prediction process usually generates false alarms in many anomaly based intrusion detection systems. However, with fuzzy logic, the false alarm rate in determining intrusive activities can be reduced, where a set of fuzzy rules is used to define the normal and abnormal behavior in a computer network, and a fuzzy inference engine can be applied over such rules to determine intrusions. This paper proposes a technique (genetic algorithm) to generate fuzzy rules (instead of manual design) that are able to detect anomalies and some specific intrusions. Experiments were performed with DARPA data sets [1], which have information on computer networks, during normal behavior and intrusive behavior. This paper presents some results and reports the performance of generated fuzzy rules in classifying different types of intrusions.

Index terms – Intrusion detection, fuzzy classification, rule generation, and genetic algorithms

I. INTRODUCTION

The number of intrusions into computer systems is growing because new automated hacking tools are appearing every day, and these tools along with various system vulnerability information are easily available on the web. The problem of intrusion detection has been studied extensively in computer security ([2], [3], [4], and [5]), and has received a lot of attention in machine learning and data mining ([6], [7], and [8]). Basically, there are two models of intrusion detection [5]:

Anomaly Detection: This model first build the normal profile that contains metrics derived from the system operation. While monitoring the system, current observation is compared with the normal profile in order to detect changes in the patterns of utilization or behavior of the system.

Signature or Misuse Detection: This technique relies on patterns of known intrusions to match and identify intrusions. In this case, the intrusion detection problem is a classification problem.

Intelligent Security System Research Lab, Division of Computer Sciences, Mathematical Sciences Department, The University of Memphis, Memphis, TN 38152. Jonatan Gomez is also a faculty of Universidad Nacional de Colombia.

ISBN 555555555/$10.00

However, there exist several approaches in implementing an intrusion detection system that use one or both the above-mentioned models:

1. Using data mining techniques over system audit data to extract consistent and useful patterns of program and user behavior, in order to build classifiers that can recognize anomalies [8]. The data mining techniques that are primarily used include classical association rules and the frequent episodes learning.

2. Using temporal association rules, in terms of multiple time granularities [9]. The temporal association rules technique generates fuzzy and classical rules [10].

3. Using short sequences of system calls performed by running programs as discriminators between normal and abnormal operating characteristics [11]. The discriminator uses the Hamming distance between short sequences of system calls. If the distance of a particular sequence to the normal sequences is higher than a threshold then the sequence is considered abnormal.

4. Distributing the detection task among multiple independent entities (autonomous agents) that work collectively [6]. The functionality of each agent is not defined but it can be simple or complex according to the specific detection task that an agent is assigned. If an agent detects some anomalies or intrusions, the agent sends messages to other agents to define, in a distributed manner, the action to take.

5. Using genetic programming to build autonomous agents that detect intrusions. The learning model uses feedback, and the process evolves agents over the scenario of intrusions and normal behavior [7].

6. Emulating mechanisms of the natural immune systems to detect anomalies in a distributed manner [12]. It combines two anomaly detection methods: using profiles of user behavior and correlation of user behavior with network statistical behavior. The decision support component uses an ART neural network and a Fuzzy Controller.

In this paper, we show the applicability of genetic algorithms to evolve a simple set of fuzzy rules (fuzzy classifier) that can solve some well-studied intrusion

detection problems. In this approach, genetic algorithms can find good and simple fuzzy rules to characterize intrusions (abnormal) and normal behavior of network systems. As the difference between the normal and the abnormal activities are not distinct, but rather fuzzy, fuzzy logic can reduce the false signal rate in determining intrusive activities.

The subsequent sections are organized as follows. Section 2 briefly describes the basic fuzzy logic and fuzzy classifiers concepts used in this paper, section 3 presents the proposed approach to solve some intrusion detection problems, section 4 describes experiments and analysis of results, and section 5 draws some conclusions.

II. FUZZY CLASSIFIERS FOR INTRUSION DETECTION

The intrusion detection problem (IDP) is a two-class classification problem: the goal is to classify patterns of the system behavior in two categories (normal and abnormal), using patterns of known attacks, which belongs to the abnormal class, and patterns of normal behavior. With fuzzy rules, the solution to classification problem is based on fuzzy logic concepts.

A. Fuzzy Logic

In fuzzy logic [13], fuzzy sets define the linguistic notions, and membership functions define the truth-value of such linguistic expressions. Table 1 shows the difference between classic sets and fuzzy sets.

FUZZY SETS	CLASSIC SETS
In fuzzy sets, an object can partially be in a set.	In classic sets, an object is entirely in a set or is not.
The membership degree takes values between 0 and 1.	The membership degree takes only two values, either 0 or 1.
1 means entirely in the set, 0 means entirely not in the set, other values mean partially in the set.	1 means entirely in the set, 0 means entirely not in the set. Other values are not allowed.

Table 1: Comparisson between fuzzy sets and classic sets

The degree of membership of an object in a fuzzy set is defined as a function where the universe of discourse (set of values that the object can take) is the domain, and the interval [0,1] is the range. Figure 1 shows an example of the membership function (triangular), which is in wide use.

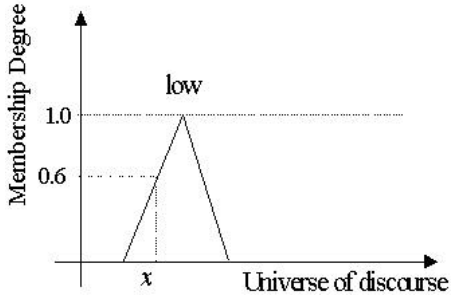

Figure 1: Triangular membership function

In figure 1, the object *x* has 0.6 degree of membership to the fuzzy set *low*, i.e., *x* belongs to the fuzzy set and does not belong to the fuzzy set at the same time. A collection of fuzzy sets, called fuzzy space, defines the fuzzy linguistic values or fuzzy-classes that an object can belong to. A standard fuzzy space is shown in figure 2.

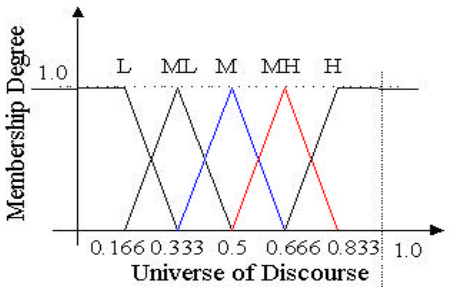

Figure 2: Fuzzy space with five fuzzy sets

With fuzzy spaces, fuzzy logic allows an object to belong to different classes at the same time. This concept is helpful when the difference between classes is not well defined. This is the case in the intrusion detection task, where the differences between the normal and abnormal classes are not well defined.

With these linguistic concepts, atomic and complex fuzzy logic expressions can be built. An atomic fuzzy expression is an expression:

parameter is [not] fuzzyset

Where, *parameter* is an object, and *fuzzyset* is a fuzzy set that belongs to the defined fuzzy space for the parameter. The truth-value (TV) of an atomic expression is the degree of membership of the parameter to the fuzzy set. Because TVs are expressed by numbers between 0 and 1, (0 means entirely false, 1 means entirely true, and others values means partially true), the fuzzy expression evaluation process is reduced to arithmetic operations. Also, for each classical logic operator (*and*, *or*, *negation*), there is a common fuzzy logic arithmetic operator (shown in table 2):

ISBN 555555555/$10.00

LOGIC OPERATOR	FUZZY OPERATOR
p AND q	*min{p, q}*
P OR q	*max{p, q}*
NOT p	*1.0-p*

Table 2: Fuzzy logic operators

Fuzzy rules have the form:
IF condition *THEN* consequent [weight]

Where,

- *condition* is a complex fuzzy expression, i.e., a logic expression that uses fuzzy logic operators and atomic fuzzy expressions
- *consequent* is an atomic expression, and
- *weight* is a real number that defines the confidence of the rule.

The following is an example of a fuzzy rule:

R: IF x is HIGH and y is LOW THEN
pattern is normal [0.4]

The TV of the fuzzy rule is calculated as the product of the condition truth-value by the weight, i.e.:

*TV(R) = TV(condition) * weight*

For the previous example, if the degree of membership of the parameter *x* to the fuzzy set *HIGH* is *0.2*, the degree of membership of *y* to *LOW* is *0.4* and the weight is *0.4* then the truth-value of the fuzzy rule is:

*TV(R) = TV(x is HIGH and y is LOW) * 0.4*
*= min{0.2, 0.4} * 0.4 = 0.2 * 0.4 = 0.08*

B. Fuzzy classifiers and the intrusion detection problem

In intrusion detection, there are *m+1* classes where every object should be classified. Among these classes, there is one special class called the normal class and *m* classes called the abnormal classes (based on known intrusions or attacks). The data set used by the learning algorithms consists of a set of objects, each object with *n+1* attributes. The first *n* attributes define the object characteristics (monitored parameters) and the last attribute defines the class that the object belongs to (the classification attribute).

Accordingly, fuzzy classifier system for solving intrusion detection problem should have a set of *m+1* rules, one for the normal class and *m* for the abnormal classes, where the condition part is defined by the monitored parameters and the consequent part is an atomic expression for the classification attribute. For example,

R_{Normal} *: IF x is HIGH and y is LOW THEN*
pattern is normal [0.4]
$R_{Abnormal\text{-}1}$ *: IF x is MEDIUM and y is HIGH THEN*
pattern is abnormal$_1$ *[0.6]*
. . .
$R_{Abnormal\text{-}m}$ *: IF x is LOW THEN*
pattern is abnormal$_m$ *[0.6]*

There are several techniques to determine the class that an object belongs to. One of these techniques is the maximum technique, which classifies the object as the class in the consequent part of the rule that has the maximum truth-value, i.e.:

$$class = \begin{cases} N & if\ TV(R_N) > TV(R_{A\_i}) \forall i = 1..m \\ A_i & if\ TV(R_N) < TV(R_{A\_i}) \wedge \\ & TV(R_{A\_j}) < TV(R_{A\_i}) \forall j = 1..m\ j \neq i \end{cases}$$

Where,
N represents the normal class,
A_i the i-th abnormal class,
R_N is the rule for the normal class, and
$R_{A\text{-}i}$ is the rule for the i-th abnormal class

If *k* number of rules produced have the same maximum truth-value, then one of these rules can be picked randomly.

III. EVOLVING FUZZY CLASSIFIERS

We used a genetic algorithm to generate fuzzy classifiers for intrusion detection using datasets with patterns of the system behavior during normal and under intrusive (abnormal) conditions.

In a genetic search, a set of chromosomes (population), each chromosome codifying a possible solution for the given problem, is evolved using a set of genetic operators (mutation, crossover, selection). Each chromosome has probability to be modified by one of these genetic operators, and this probability depends on the adaptability of the chromosome. The following steps outline the basic structure of the genetic algorithm (GA):

1. **[Start]** Generate random population of *n* chromosomes
2. **[Fitness]** Evaluate the fitness *f(x)*
3. **[New population]** Create a new population
4. **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
5. **[Crossover]** With a crossover probability swap genetic material between the parents to form a new

ISBN 555555555/$10.00

offspring (children). If no crossover was performed, offspring is an exact copy of parents.

6. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
7. **[Accepting]** Place new offspring in a new population
8. **[Replace]** Use new generated population for a further run of algorithm
9. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
10. **[Loop]** Go to step **2**

There are several approaches to evolve fuzzy classifiers with GAs [14], [15], [16]. We used the approach proposed by Gomez in [17], where, an adaptive-parameter genetic algorithm with special operators (gene addition, gene deletion) is used for each class (normal and abnormal classes). Figure 3 illustrates the steps to produce a fuzzy rule for the class k using a GA.

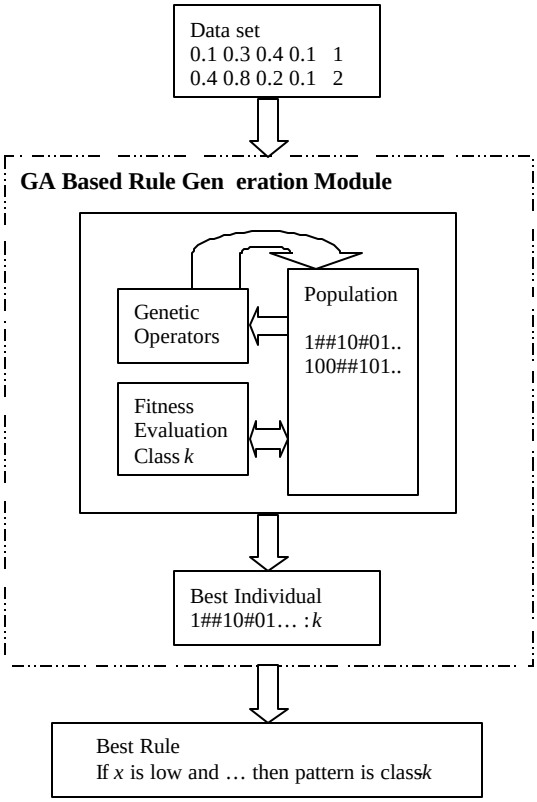

Figure 3: Steps to generate a fuzzy rule for class k using a GA

Because, the variable segment in this fuzzy rule is the condition part (the consequent and the confidence weight are fixed), only this segment is encoded as a linear chromosome with variable length that uses complete expression trees.

For example, the expression:

$$(x \text{ is } C \text{ Ù } w \text{ is not } D) \text{ Ú } z \text{ is } E$$

can be represented without parenthesis and using complete expression trees as:

$$x \text{ is } C \text{ Ú } z \text{ is } E \text{ Ù } w \text{ is not } D$$

With complete expression tree, the chromosome is defined as a set of *n* genes, each gene is composed of an atomic condition <variable> is [not] <set> and a logic operator, as is shown in figure 4.

Gen_1				...	Gen_n				Gen_{n+1}			
ac_1			op_1	...	can			op_n	ac_{n+1}			*
var_1	ro_1	set_1		...	var_n	ro_n	set_n		var_{n+1}	ro_{n+1}	set_{n+1}	*

Figure 4: Representation of the condition part of a fuzzy rule using operator precedences.
var_i, ro_i, set_i, and op_i are the variable, relational operator, set and logic operator of the i-th gene.

An example expression encoded in the chromosome using operator priority is as follows:

Gen_1				Gen_2				Gen_3				
Ac_1			op_1	Ac_2			op_2	ac_3			op_3	
X	YES	C	∨	Z	YES	E	∧	W	NOT	D	*	*

Figure 5: Codification of the expression X is C or Z is E and W is not D

Gomez used the fuzzy confusion matrix to calculate the fitness of a chromosome [17]. In the fuzzy confusion matrix, the fuzzy truth degree of the condition represented by the chromosome and the fuzzy negation operator are used directly. In our case, the fitness of a chromosome for a specific class is evaluated according to the set of following equations:

$$TP = \sum_{i=1}^{p} predicted(class\_data_i)$$

$$TN = \sum_{i=1}^{q} [1 - predicted(other\_class\_data_i)]$$

$$FP = \sum_{i=1}^{q} predicted(other\_class\_data_i)$$

$$FN = \sum_{i=1}^{p} [1 - predicted(class\_data_i)]$$

$$sensitivity = \frac{TP}{TP + FN},$$

$$specificity = \frac{TN}{TN + FP}$$

$$length = 1 - \frac{chrom\_length}{10},$$

$$fitness = w_1 * sensitivity + w_2 * specificity + w_3 * length$$

ISBN 555555555/$10.00

Here,

TP, *TN*, *FP*, and *FN* are the true positive, true negative, false positive, and false negative values for the codified rule respectively,

predicted is the fuzzy value of the condition part of the codified rule,

p is the number of samples of the evolved class and *q* is the number of samples of the remaining classes samples in the training data set, that are used by each chromosome in the genetic algorithm,

w_1, w_2, and w_3 are the assigned weights for each rule characteristic respectively,

$class\_data_i$ is a element of the subset of training samples of the evolved class, and, $other\_class\_data_i$ is a element of the subset of remaining classes (not evolved by the genetic algorithm) in training samples.

Accordingly, the best chromosome in the population is chosen and the fuzzy rule: *if* <*condition*> *then pattern is* <*class*>, is added to the fuzzy classifier.

IV. EXPERIMENTATION

A. Test Data Set

Tests were conducted using ten percent of the kdd-cup'99 data set [1], in order to evaluate the performance of the proposed approach to generate comprehensible fuzzy classifiers for intrusion detection problems. This data set is a version of the 1998 DARPA intrusion detection evaluation data set prepared and managed by MIT Lincoln Labs [18]. In this data set, forty-two attributes (or fields) that usually characterize network traffic behavior compose each record. Also, the number of records in the 10% data set is very large (492021).

This data set contains 22 different types of attacks that can be classified in four main intrusion classes, as is shown in table 3. Also, the proportion of samples per class is not uniform, for example from class U2R the number of samples in the training data set is 59 while from class DOS the number of samples is 391458.

CLASS	SUB-CLASSES	SAMPLES
Normal		95278 (19.3%)
U2R	buffer_overflow, loadmodule, multihop, perl, rootkit	59 (0.01%)
R2L	ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster	1119 (0.23%)
DOS	back, land, Neptune, pod, smurf, teardrop	391458(79.5%)
PRB	ipsweep, nmap, portsweep, satan	4107 (0.83%)

Table 3: Classes in the 10 % of the KDDCup 99 data set

B. Preprocessing

We normalized the original 10% KDD-cup99 data set, where each numerical value in the data set is normalized between 0.0 and 1.0 according to the following equation:

$$x = \frac{x - MIN}{MAX - MIN}$$

Where,

x is the numerical value, *MIN* is the minimum value for the attribute that x belongs to, and *MAX* is the maximum value.

For each numerical attribute we assign the fuzzy space as shown in figure 6.

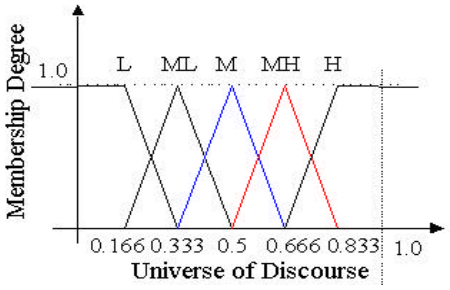

Figure 6: Fuzzy space for numerical attributes in the KDD-cup99data set

For non-numerical attributes like logged-in we used the categorical values as crisp sets (fuzzy sets that do not overlap each other). Figure 7 shows the fuzzy space associated to the logged-in attribute. For example, a value of false for this attribute has a degree of membership to the crisp set FALSE equal to 1.0 and degree of membership to the fuzzy set TRUE equal to zero.

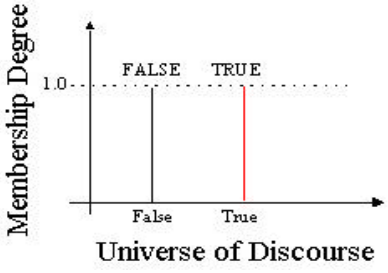

Figure 7: Fuzzy space for the non-numerical attribute logged-in

C. Experimental Setting

We randomly selected a 20% of the total set for testing and the remaining 80% for training. The testing data set was built using a 20% of all the samples of each class in

ISBN 555555555/$10.00

the data set. The score of the trained classifier was calculated as the classification accuracy over the testing set.

We used adaptive parameter settings in genetic algorithm as proposed by Gomez in [19], where each genetic search was initialized with a random population of individuals, with length between one and six genes. We performed several tests with different values of fitness function weights.

Because the combined proportion of samples from the normal class and the DOS class are almost 99% of the data set, we evolved a fuzzy rule for the DOS attack independently using only the normal and the DOS records in the training data set. Each individual in the population uses a number of samples from the data set that is proportional to the number of samples of that class present in the dataset. The proportion of samples from each class is given by the proportion of samples of each class in the training dada set.

For each of the remaining classes (U2R, R2L, PRB, and normal), we evolved one fuzzy rule using the training set without the DOS class samples. In this case, however the numerical proportion in which each class is present in the dataset is not used to choose the number of samples used for that class. Rather a uniform number of samples is considered for each of these classes.

The reported results were obtained using a population size of 200 and a maximum of 200 iterations. Each individual used only 2000 records from the training dataset and a proportional random assignment of the fitness function weights as is explained by Gomez in [17]. These values showed good performance in the evolution of fuzzy classifiers for the intrusion detection problem.

D. Results and Analysis

The proposed approach was able to generate simple fuzzy rules (the longest fuzzy rule contains only five atomic expression). The following are some fuzzy rules that were evolved in a sample run:

if (dst_host_srv_count is not low or
protocol_type is not tcp) and protocol_type is not icmp **then** record_type is normal [1.0]
if dst_host_srv_count is low and flag is not S0 and
protocol_type is not icmp and
dst_host_srv_rerror_rate is not level-4
then record_type is U2R [1.0]
if (dst_host_srv_count is low or is_guest_login is true) and flag is not REJ and dst_host_same_srv_rate is not low and duration is not level-4 **then** record_type is R2L [1.0]
if count is not low or same_srv_rate is low
then record_type is DOS [1.0]
if dst_host_same_srv_rate is low and flag is not SF or
protocol_type is icmp
then record_type is PRB [1.0]

There are two elements that define the cost function of an intrusion detection system: the false alarm (FA) rate (the system produces an alarm in a normal condition), and the detected attacks rate (DR) (the system considers an abnormal behavior as normal). The performance of the proposed approach: Evolving Fuzzy Rules for Intrusion Detection (EFRID) is shown in table 4.

Algorithm	FA %	DR %	Complexity
EFRID	**7.0**	**98.95**	**O(n)**
RIPPER-Artificial Anomalies [20]	2.02	94.26	$O(n*\log^2 n)$
SMARTSIFTER [21]	-	82.0	$O(n^2)$

Table 4: Comparison of the proposed approach with RIPPER and SMARTSIFTER

Thus table 4 compares the performance of EFRID against other methods (RIPPER, SMARTSIFTER). Our approach resulted in DR of 98.95 compared to 94.26 in R.

In EFRID, each individual in the population used less than 1% of the data-set, the number of individuals per iteration is 200, the number of iterations is 200, and the genetic algorithm is run 5 times (one per each class). Then the number of times that the data set is used is bounded by 2000 times, but because there are some individuals per iteration that are not feasible (the result of the genetic operator can be a not valid fuzzy rule), this value is less than 2000. We calculated the average number of times that the data set is accessed (875.4 times). Therefore EFRID is a linear algorithm respect to the size of the data set.

The classification accuracy of EFRIDS is shown in table 5 along with the performance achieved by the winner group in the KDDCup' 99 contest [22]. However, these results are not comparable as different data sets were used for testing. So, this information is for reference only.

CLASS	EFRID	WINNER ENTRY
Normal	92.78%	94.50%
U2R	88.13%	13.2%
R2L	7.41%	8.4%
DOS	98.91%	97.10%
PRB	50.35%	83.30%

Table 5: Accuracy of EFRID and the winner entry in the KDDCup' 99 contest

In the simplest form, intrusion detection problem can be considered as a two-class classification problem, where the positive class is the abnormal class and the negative class is the normal class. We have applied the Receiver Operating Characteristic (ROC) analysis to evaluate the performance of the evolved classifiers [23]. In the ROC

ISBN 555555555/$10.00

analysis, for classifier systems that produce a continuous output with respect to some parameter α, the coordinate point $(FP, TP)_\alpha$ is plotted in the coordinate system. Here, TP is the true positive rate (the percentage of abnormal behavior classified as abnormal) and FP is the false positive rate (the rate of false alarms).

To generate the ROC curve for the intrusion detection problem from the evolved fuzzy classifier, we varied α between 0.0 and 1.0 and assigned to the normal rule the confidence weight $1-\alpha$ and to the abnormal rules the confidence weight α.

The plotted points define the ROC curve for the given classifier. This ROC curve can be used to determine when a classifier is good or bad. If the ROC curve of a classifier *A* dominates the ROC curve of the classifier *B* then classifier *A* is better than classifier *B* [24]. The ROC curve for an intrusion detection classifier shows how the fuzzy rule confidence value affects the rate of alarms and the rate of detected abnormal activities. The ROC curve for the evolved fuzzy classifier systems is shown in figure 7.

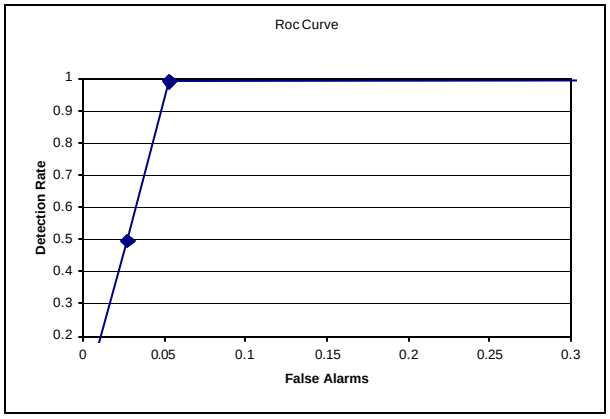

Figure 3: ROC curve for the evolved fuzzy classifier

According to the ROC curve it is possible to reduce the false alarm rate to 5% with a detection rate of 98.5% if the confidence value of the abnormal class rule is set to 0.1 and the confidence value of the normal rule is set to 0.9.

V. CONCLUSIONS

Our experiments showed that the proposed approach works well in detecting different attacks. The accuracy of fuzzy classifiers was good and comparable to those reported in the literature. Also, the accuracy can further be improved applying specific strategies to generate the fuzzy space for each monitored parameter.

The evolved fuzzy rules are not complex as no more than five attributes are used in each rule. It allows characterization of the normal and abnormal behaviors in a simple way. Simpler fuzzy rules have a clear advantage in real applications. First, they yield rules that are easier to interpret, hence score high on *interpretability*. Second, they yield a classifier rule that is *faster* in deployment. This is especially *crucial* for data involving a *large number of attributes*.

The importance of fast response is even more crucial in *network security* applications: In automatic intrusion detection systems, a *fast* classifier can mean all the difference between being able to *detect and stop* the intruder's behavior *in time*, as compared to detecting the user's behavior, but responding *too late* because of a *slow* detection process. We emphasize that training is the most time consuming part of our system, it is a linear time algorithm over the data set size but it is one of the most efficient (see table 4).

The main contribution of the present work is the design of a classification process for the intrusion detection problem. It allows application of fuzzy logic and genetic algorithms for the detection of various types of attacks.

VI. ACKNOWLEDGES

This work was supported by the Defense Advanced Research Projects Agency (no. F30602-00-2-0514) and National Sciences Foundation (no. NSF-EIA-9818323).

VII. REFERENCES

[1] KDD-cup data set. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[2] R. Heady, G. Luger, A. Maccabe, and M. Sevilla. The Architecture of a Network-level Intrusion Detection System, Technical report, CS90-20. Dept. of Computer Science, University of New Mexico, Albuquerque, NM 87131.
[3] E. Amoroso, "Intrusion detection", Intrusion.net Books, January 1999.
[4] J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner, "State of the practice of intrusion detection technologies", Technical Report CMU/SEI99 -TR-028, ESC-99-028, Carnegie Mellon, Software Engineering Institute, Pittsburgh, Pennsylvania, 1999.
[5] S. Axelsson, "Intrusion detection systems: A survey and taxonomy", Technical Report No 99-15, Dept. of Computer Engineering, Chalmers University of Technology, Sweden, March 2000.
[6] J. Sundar, J. Garcia-Fernandez, D. Isaco, E. Spafford, and D. Zamboni, "An architecture for intrusion detection

ISBN 555555555/$10.00

using autonomous agents", Tech. Rep. 98/05, Purdue University, 1998.
[7] M. Crosbie, "Applying genetic programming to intrusion detection", In Proceedings of the AAAI 1995 Fall Symposium series, November 1995.
[8] W. Lee, S. J. Stolfo, and K. W. Mok, "Mining audit data to build intrusion detection models", Proc. Int. Conf. Knowledge Discovery and Data Mining (KDD'98), pages 66-72, 1998.
[9] Y. Li, N. Wu, S. Jajosia, and X. S. Wang, "Enhancing profiles for anomaly detection using time granularities", Center for secure information systems. To appear in Journal of Computer Security, 2002.
[10] S. Bridges and R. Vaughn, "Fuzzy data mining and genetic algorithms applied to intrusion detection", Proceedings twenty third National Information Security Conference, October 1-19, 2000.
[11] S. A. Hofmeyr, A. Somayaji, and S. Forrest, "Intrusion detection using sequences of systems call", Journal of Computer Security, 6:151-180, 1998.
[12] D. Dasgupta and H. Brian, "Mobile security agents for network traffic analysis", Published by the IEEE Computer Society Press in the proceedings of DARPA Information Survivability Conference and Exposition II (DISCEX-II), June 12-14, 2001, Anaheim, California.
[13] Zadeh, L.A., "Fuzzy sets" in Information and Control, 8: 338-352, 1965
[14] C.E. Bojarczuk, H.S. Lopes, and A.A. Freitas "Discovering comprehensible classification rules using genetic programming: a case study in medical domain". Proceedings Genetic and Evolutionary Computation Conference GECCO99, 1999.
[15] H. Ishibuchi and T. Nakashima "Linguistic rule extraction by genetic-based machine learning". Proceedings Genetic and Evolutionary Computation Conference GECCO00, 2000.
[16] J. Liu and J. Kwok "An extended genetic rule induction algorithm". Proceedings of the Congress on Evolutionary Computation Conference, 2000.
[17] J. Gomez, D. Dasgupta, O. Nasraoui, and F. Gonzalez, and, "Complete Expression Trees for Evolving Fuzzy Classifier Systems with Genetic Algorithms", To appear in the proceedings of the North American Fuzzy Information Processing Society Conference NAFIPS-FLINTS 2002. June 2002.
[18] Lincoln Laboratory MIT. http://www.ll.mit.edu/
[19] J. Gomez, and D. Dasgupta, "Using Competitive Operators and a Local Selection Scheme in Genetic Search", Submitted as Late-breaking paper to the Evolutionary Computation Conference GECCO02, 2002
[20] W. Fan, W. Lee, M. Miller, S. J. Stolfo, and P. K. Chan, "Using artificial anomalies to detect unknown and know network intrusions", Proceedings of the First IEEE International Conference on Data Mining, 2001.
[21] K. Yamanishi, Jun-ichi Takeuchi and G. Williams, "On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms", Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 320-324, 2000.
[22] Results of the KDD' 99 Classifier learning contest, http://www-cse.ucsd.edu/users/elkan/clresults.html.
[23] F. Provost, and T. Fawcett, "Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions", Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 1997.
[24] F. Provost, and T. Fawcett, "The case against accuracy estimation for comparing induction algorithms", Proceedings of the Fifteenth International Conference on Machine Learning, 1998.

ISBN 555555555/$10.00