A study on fuzzy intrusion detection

J.T. Yao   S.L. Zhao   L. V. Saxton

Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
E-mail: [jtyao,zhao200s,saxton]@cs.uregina.ca

ABSTRACT

Current intrusion detection techniques mainly focus on discovering abnormal system events in computer networks and distributed communication systems. Clustering techniques are normally utilized to determine a possible attack. Due to the uncertainty nature of intrusions, fuzzy sets play an important role in recognizing dangerous events and reducing false alarms level. This paper proposes a dynamic approach that tries to discover known or unknown intrusion patterns. A dynamic fuzzy boundary is developed from labelled data for different levels of security needs. Using a set of experiment, we show the applicability of the approach.

1. INTRODUCTION

Due to the popularization of the Internet and local networks, intrusion events to computer systems are growing. Intrusion detection systems are becoming increasingly important in maintaining proper network security.[2, 7] Most intrusion detection systems are softwares.[3] People use these softwares to monitor the events occurring in a computer system or network, analyze the system events, detect suspected intrusion, and then raise an alarm.

A typical intrusion detection system consists of three functional components[3]: an information source, an analysis engine and a decision maker. The information source provides a stream of event records. This component can also be considered as an event generator. It monitors different data sources and generates data that are well formatted and suitable for analysis. The data sources can be divided into three categories: first, data sources related to operating systems, such as system calls and system logs; second, network traffic monitors which generate raw network packets[13]; third, data collectors of different applications.

The analysis engine finds signs of intrusions. There are two basic approaches used to detect intrusions.[3] The first approach is called misuse detection. A system using this approach detects intrusion events which follow well-known patterns. The patterns may describe a suspect set of sequences of actions or takes other forms. The primary limitation of this approach is that it cannot detect novel intrusions, i.e., events that are never happened in the system. The second approach is called anomaly detection. An anomaly detection based system analyzes the event data of a training system, recognizes patterns of activities that appear to be normal. If a testing event lies outside of the patterns, it is reported as a possible intrusion. In this paper we mainly focus on anomaly intrusion detection.

A decision maker applies some rules on the outcomes of the analysis engine, and decides what reactions should be done based on the outcomes of the analysis engine. The major function of the decision maker is to increase the usability of an intrusion detection system.

Many artificial intelligence techniques have been applied to the intrusion detection area.[10] By the approaches used on analysis engine, the intrusion detection systems can be categorized into misuse detection systems or anomaly detection systems. For a misuse detection system, an expert system can be used to store a set of rules designed to detect the known intrusion activities.[1] The knowledge of past intrusions can be encoded by human experts into expert system rules. Kumar *et al.*[11] proposed a misuse detection system based on pattern matching. In their model, the known intrusion signatures are encoded as patterns, and then matched against the audit data introduced by the analysis component.

From the viewpoint of classification, the main work of building an anomaly intrusion detection system is to build a classifier which can classify normal event data and intrusion event data from an original data set. Anomaly intrusion detection mainly consists of two processes, which are training the parameters of a classifier from a training data set and using this classifier to classify a test data set. Here we just mention some popular approaches to develop classifiers. Qiao *et al.*[18] presented an anomaly detection method by using a Hidden Markov Model to analyze the trace of system calls coming from a UNIX system. Lee *et al.*[14] established an anomaly detection model that integrated the association rules and frequency episodes with fuzzy logic to produce patterns for intrusion detection. Mohajeran *et al.*[16] developed an anomaly intrusion detection system combining neural networks and fuzzy logic. Wang *et al.*[21] applied genetic algorithms to optimize the membership function for mining fuzzy association rules.

The above researchers have made various contributions on using artificial intelligence techniques on anomaly intrusion detection, but all their methods are using a static classifier or a static decision boundary to classify data, then detect possible intrusions. However, the security needs may differ for various applications. It would be good if a dynamic decision boundary can be set for different applications. This research is aim to develop a dynamic decision boundary for different levels of security needs. Our research is also within the anomaly detection category. It is natural to think that there are some connections between the detection accuracy of a decision boundary and the computation complexity of classification using this boundary. With a dynamic boundary adjustable to users' requests, the users may obtain a suitable detection rate according to their own needs along with a acceptable computation time.

The organization of this paper is as follows. In the next section, we propose a fuzzy intrusion detection approach with dynamic boundary. Experiments by using KDD Cup 1999 data are presented in Section 3. Preliminary results and their analysis are discussed in Sections 4. A conclusion section is followed.

2. DEVELOPMENT OF A DYNAMIC FUZZY BOUNDARY

A hybrid system intrusion detection approach is proposed. The method and rational of choosing Fuzzy logic and Support Vector Machine is disused in this section.

2.1. Overview of Fuzzy logic and Support Vector Machine

Fuzzy logic was introduced by Dr. Lotfi Zadeh of UC/Berkeley in the 1960's as a means to model the uncertainty of natural language.[22] For several reasons, fuzzy logic is very appropriate for using on intrusion detection.[9] One reason is that usually there is no clear boundary between normal and anomaly events. The use of fuzziness of fuzzy logic helps to smooth the abrupt separation of normality and abnormality. Another reason is that when to raise an alarm is fuzzy. There would be too many alarms if we raise an alarm every time when we find an intrusion event. At what degree of intrusion we should raise an alarm is often depends on different situation.

Support Vector Machine or SVM in short is a machine learning method based on statistical learning theory.[20] It relies on preprocessing the data to represent patterns in a high dimension which is typically higher than the original feature space. SVM classifies data by determining a set of support vectors, which are members of a set of training inputs. These support vectors outline a hyper plane (decision boundary) in the feature space.[20] SVM has two unique features.[5, 20] Based on Structural Risk Minimization principal, SVM minimizes the generalization error. Therefore the first feature of SVM is its good generalization ability of the learning model, which means even from a relatively small training data set, we can still have a good accuracy when we use SVM from classification. The high generalization ability of SVM is very useful when we apply SVM on an intrusion detection system. In an anomaly intrusion detection system, we need to form a training data set and attach each record in the set with a label. This label identifies if the record is an intrusion or not. The whole process of forming a training data set costs a lot of time. The SVM's feature of using relatively small training set saves time of forming the training data set in a intrusion detection system.

The second feature of SVM is the ability to overcome the curse of dimensionality. SVM constructs the classifier by evaluating an kernel function between two vectors of the training data instead of explicitly mapping the training data into the high dimensional feature space. Therefore SVM is capable of handling a large number of features. On an intrusion detection system, we need to collect event data for representing different aspects of a system. The more data features we have, the more possible we can detect an intrusion. A good capability to handle high dimensional data is very useful to a intrusion detection system.

The nonlinear discriminant function of SVM is

$$f(\vec{x}) = sgn(\sum_{i=1}^{l} \alpha_i y_i K(\vec{x_i}, \vec{x_i}) + b)$$

where l is the number of training records, $y_i \in \{-1, +1\}$ is the label associated with the training data, $0 \leq \alpha_i \leq C,\ C > 0$ is a constant and $K(\vec{x_i}, \vec{x_i})$ is the kernel function. Users can choose a suitable function for themselves.

The (Radial Bias Function(RBF)[8] $e^{-||\vec{x_i}-\vec{x}||^2 \cdot \gamma}$ is used as our kernel function in this study. The final discriminant function we are using is

$$f(\vec{x}) = sgn(\sum_{i=1}^{l} \alpha_i y_i e^{-||\vec{x_i}-\vec{x}||^2 \cdot \gamma} + b)$$

where the value of variance parameter γ is set in advance.

2.2. A method to develop a dynamic fuzzy boundary

Fuzzy logic and SVM have their own unique features appropriate for intrusion detection systems. SVM is a classification method which has shown good performances on the accuracy and efficiency, specially in high input dimensions. Mukkamala *et al.*[17] compared the performance of intrusion detection systems using SVM and Neural Networks, and concluded that SVM outperforms Neural Networks. The uncertainty of fuzzy logic is also very suitable for intrusion detection systems. Therefor, we use a hybrid method consisting of SVM and fuzzy logic techniques to develop a dynamic and fuzzy decision boundary. We use this boundary to discriminate normal and abnormal events and detect the possible intrusions. The dynamic decision boundary is based on a set of support vectors generated by SVM and fuzzed with fuzzy logic technique. We hope the decision boundary generated by our method obtains the high generalization from SVM and flexibility from fuzzy logic.

In this paper we focus on developing an anomaly intrusion detection system with a dynamic decision boundary. The basic thought of our method is extracting a fuzzy rule set from support vectors which are the training result of a SVM. Then we apply some fuzzy functions on the rule set to develop a fuzzy classifier or a fuzzy decision boundary .[6] To make the decision boundary dynamic, we train a SVM several times using different values of parameters, obtain several sets of support vectors, extract different fuzzy rule sets from different sets of support vectors, and at last build a dynamic decision boundary according to the fuzzy rule sets. The different positions of the boundary correspond to different fuzzy rule set.

We use RBF(Radial Bias Function) kernel function $e^{-||\vec{x_i}-\vec{x_j}||^2 \cdot \gamma}$ when we train the SVM. Suppose we have totally l records in the training data set. The training result is a set of support vectors $\{\vec{z_1}y_1, ..., \vec{z_m}y_m\}$ and the bias b, where $m < l, y_i \in \{-1, 1\}$. From the subsection above we know that there are two parameters C and γ needed to be set in advance for training a SVM with RBF kernel. To let the final decision boundary generated be dynamic, we fix the value of C and let γ be equal to several different values. Each value of γ corresponds to a different set of support vectors. From the different support vectors we build different decision boundaries.

After we have several sets of support vectors, we can develop several fuzzy rule sets from the sets of support vectors. Each fuzzy rule set corresponds to a set of support vectors. The fuzzy rule set would be like[6]:

Rule 0: IF A_0^1 AND A_0^2 AND ... A_0^n THEN b_0
Rule 1: IF A_1^1 AND A_1^2 AND ... A_1^n THEN b_1
...
Rule m: IF A_m^1 AND A_m^2 AND ... A_m^n THEN b_m

where $b_0 = b, A_0^k = a^k(0), \quad b_i = \alpha_i y_i, A_i^k = a^k(z_i^k), \quad k = 1, ..., n,\ \ i = 1, ..., m.$ m is the number of support vectors we found, n is the number of features in the data space and $a^k(x)$ is the fuzzy function.

From the fuzzy rule set, the binary fuzzy discriminant function can be written as the following form[6]:

$$f(\vec{x}) = sgn(\frac{(b_0 + t) + \sum_{i=1}^{m}(b_i + t)\prod_{k=1}^{n} a_i^k(x_k - z_i^k)}{1 + \sum_{i=1}^{m}(b_i + t)\prod_{k=1}^{n} a_i^k(x_k - z_i^k)})$$

where t is a threshold. $\vec{x} = x_k, k = 1...n$.

3. EXPERIMENT

Using the method introduced above, we build a dynamic decision boundary based on KDD (Knowledge Discovery in Databases) Cup 1999 data set.[13] The performance of the dynamic decision boundary on different positions with different training parameters γ is evaluated. A discussion about the dynamic decision boundary is provided. To prove the good generalization ability of SVM, a trimmed training set and a full-size training set are used in the experiment. Comparisons with results from different training sets prove the good generalization ability of SVM.

3.1. The experimental data set

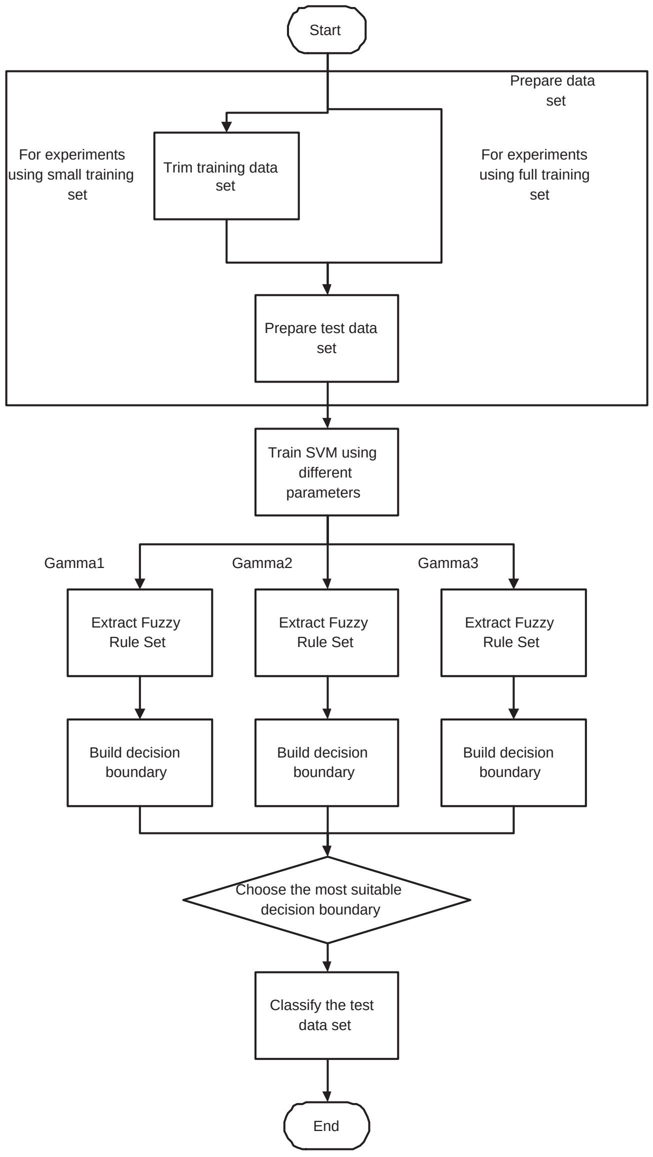

Figure 1. Experiment Procedures

In the three categories of data we have mentioned in the introduction section, the operating systems level data depend on different operating systems, and the data from running applications depend on an application. To maximize the generalization of our discussion, in this paper, we use KDD'99 data set which consists of raw network packets and is independent of a operating system or an application.

The KDD'99 data set has four intrusion categories and one normal category. Twenty-four types of attacks fall into four main categories which are denial-of-service (DOS), e.g., SYN flood, unauthorized access from a remote machine (R2L), e.g., guessing password, unauthorized access to local superuser (root) privileges (U2R), e.g., various "buffer overflow" attacks, and surveillance and other probing, e.g., port scanning.

The whole data set is collected TCP/IP dump records from hosts located on a simulated military network. The original data set contains 744 MB data with 4,940,000 connection records. It has 41 features for each record. 10% of the original data are training data with a label which identify to which category the record belongs. In this paper we only discuss binary classification. Therefore, instead of four intrusion categories and one normal category, we consider all intrusion records in one category and all normal records in another category.

3.2. Experiment procedures

There are four steps in the experiments as shown in Figure 1: data preparation, SVM training, rule extraction and dynamic boundary building.

The first step is to prepare the data sets. On this step we prepare both the training and test data sets. Since we do binary classification, we attach a class label with value +1 or -1 to each data example of all data including training and test data. The label value +1 means the example is a normal record and -1 means the example is an intrusion record.

For the training process, we prepare two data sets. A small data set has 50,000 records randomly selected from the KDD'99 training set. A large data set has all the 494,022 records from the KDD'99 training set. The volume of KDD'99 data set is large. Because of the good generalization ability of SVM mentioned above, we do not use all the training examples of KDD'99. Instead, we can use the small data set as our training set. With this way, the number of generated support vectors are decreased, the computational complexity of classification is lowered, and the performance is still on the same level as we show below.

For the test process, to gain a better generalization performance of experiments, we randomly select 50,000 records from KDD'99 test data, and split these records into five test sets with each set has 10,000 examples.

The second step is to generate the support vectors by training the SVM. We use a modification of the SVM implementation developed by Joachims,[12] set $C = 1$ and change the value of γ to be $10^{-2}, 10^{-6}$ and 10^{-8} in different training processes. Three different sets of support vectors are generated from training.

The third step is to extract fuzzy rule sets from support vectors. Each rule set corresponds to one set of support vectors.

The last step is to build a dynamic decision boundary from the three different fuzzy rules. To simplify our experiment, for the parameters of the fuzzy classifier, we set all $a_i^k(x) = \gamma \cdot e^{-||x_k - z_i^k||^2}$, threshold $t = 0$, and the γ has the same value as it is on the training phase. In our experiments we compare the detection abilities of all the decision boundaries we generated. For the users in the real world, they can choose the most suitable boundary according to their security needs and use only this boundary to detect intrusions.

4. RESULTS AND ANALYSIS

Table 1 shows the training results from the SVM using different values of γ. All the three experiments use a same training set which consists of 50,000 randomly chosen data records with 41 features from the original training set which has 494,020 records. The table shows that the different sets of support vectors generated by using different values of γ. It is observed that the larger value of γ results a larger number of support vectors generated and the smaller value of γ results small number of support vectors.

Table 1: Training results with different value of gamma.

Training Result	Exp1	Exp2	Exp3
# of training records	50,000	50,000	50,000
# of features	41	41	41
kernel	RBF	RBF	RBF
Value of γ	10^{-3}	10^{-6}	10^{-8}
# of generated SVs	6,498	1,868	1,057

Table 2 shows three different test results for different rule sets and different values of γ. The three fuzzy rules are extracted from the Support Vector Sets showed in table 1. All the tests are conducted on a same computer with Pentium 4 2.66 GHz CPU and 512 M RAM. There are two measures that evaluate the performance of a intrusion detection system: False Positive and False Negative Rate. A False Positive occurs when an actual intrusive action has occurred but the system allows it to pass as non-intrusive behavior. A False Negative occurs when the intrusion detection system classifies an action as an intrusion when it is a legitimate action.

From these results we conclude that larger number of rules results in higher detection accuracy and higher computation costs. In experiment 1, the accuracy is the highest. Experiments 2 and 3 have similar accuracies and testing time. This result is quite reasonable. From the form of fuzzy classifier we know the computational complexity of testing process is $O(mn)$ with m is the number of rules, n is the number of features in the data space. Therefor, the testing time is increased along with the increasing of the number of rules. Another phenomenon we noticed is that while experiment 1 has much larger number of rules much longer testing time than experiment 2 and 3. The accuracy of these three experiments are on the same level.

Table 2: Test results with different values of gamma.

Test Result	Exp1	Exp2	Exp3
# of test records	10,000	10,000	10,000
# of features	41	41	41
# of rules	6,498	1,868	1,057
Value of $gamma$	10^{-3}	10^{-6}	10^{-8}
# of misclassifications	44	63	211
Accuracy	99.56%	99.37%	97.89%
# of False Positives	37	52	176
# of False Negative	7	11	35
CPU-second	49.53	11.34	8.32

Table 3 shows the test results of 5 different test data sets. To prove that the decision boundary obtained is consistent across the whole original data set, we generate 5 test sets. Each set has 10,000 data records randomly chosen from the original test set which has over four million records. The results show little differences on accuracy, false positive and false negative number, and testing time. It is suggested that the decision boundary we generated is consistent for whole data set.

Table 3: Test results with different data sets.

Test Result	Exp1	Exp2	Exp3	Exp4	Exp5
# of test records	10,000	10,000	10,000	10,000	10,000
# of features	41	41	41	41	41
# of rules	1,868	1,868	1,868	1,868	1,868
Value of $gamma$	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}
# of misclassifications	67	63	62	66	69
Accuracy	99.33%	99.37%	99.38%	99.34%	99.31%
# of False Positives	54	52	56	58	55
# of False Negative	13	11	6	8	14
CPU-second	11.07	11.87	11.34	11.36	11.11

Table 4 shows the test results on the small and full size of training data sets. To prove the generalization ability of SVM, we compare the performances of two decision boundaries trained from two different sizes of training sets. Every decision boundary is tested on two test sets. The results show that accuracy of the test is not decreased when we use a smaller training set. The volume of training sets does not effect the performance of trained decision boundaries.

Table 4: Test results with different volume of training sets.

Test Result	Exp1	Exp2	Exp3	Exp4
# of training records	494,022	494,022	50,000	50,000
# of test records	10,000	10,000	10,000	10,000
# of features	41	41	41	41
# of rules	9,460	9,460	1,868	1,868
Value of γ	10^{-6}	10^{-6}	10^{-6}	10^{-6}
# of misclassifications	63	67	62	66
Accuracy	99.37%	99.33%	99.38%	99.34%
# of False Positives	52	54	56	58
# of False Negative	11	13	6	8
CPU-second	60.85	65.28	11.34	11.36

From the all the experiment results we have three analysis conclusions.

- Using the proposed method, the decision boundary can be adjusted easily and the computing costs corresponding to different decision boundaries are different. When the value of C is fixed, by choosing different values of γ of the SVM kernel function, users can easily adjust the generated decision boundary. The larger value of γ results a decision boundary with a larger number of support vectors and a large number of rules in the fuzzy set which means possible higher detection rate but also high computation costs. Conversely, a small value of γ results a decision boundary with lower detection rate and lower computation costs. The different users focusing more on detection rate or computing time may prefer different decision boundaries.
- The adjusting of the decision boundary must be within a range using our method, when the accuracy of detection intrusion is above some level (over 99% in our case), increasing the accuracy becomes more difficult. A small accuracy increase costs a lot computation. Therefore adjusting the decision boundary is only efficient within a certain range. On the other hand, this means when using the dynamic decision boundary, users may decrease the computation cost with only a small accuracy sacrifice.
- The SVM has good generalization ability. The results of our experiments show that difference of detection accuracy between using a full training set and using a small training set is very small. This may be viewed as another experimental proof on the good generalization ability of SVM.

5. CONCLUSION

In this paper, a method to develop a dynamic decision boundary based on SVM and fuzzy logic has been introduced. The experiment results show that users can adjust dynamic boundary easily for different requests of accuracy and computation complexity. Further more, because of using SVM, this method can handle a large number of features efficiently. It is also possible to build a dynamic decision boundary using other popular artificial intelligence techniques such as neural networks, decision tree and Bayesian Networks. A further work is to develop methods that build dynamic decision boundary using other techniques and comparing the results.

REFERENCES

1. D. Anderson, T. Frivold, and A. Valdes, *Next-generation Intrusion Detection Expert System: A Summary*, SRI International Computer Science Laboratory Technical Report SRI-CSL-95-07, 1995.
2. J. Allen, A. Christie, and W. Fithen, *State of the Practice of Intrusion Detection Technologies*, Technical Report, CMU/SEI-99-TR-028, 2000.
3. R.G. Bace, *Intrusion Detection*, Macmillan Technical Publishing, Indianapolis, USA, 2000.
4. S.M. Bridges and R.B. Vaughn, Intrusion Detection via Fuzzy Data Mining, 12th Annual Canadian Information Technology Security Symposium, June 19-23, 2000, pp109-122.
5. C. Burge, A Tutorial on Support Vector Machines for Pattern Recognition, *Data mining and knowledge discovery journal*, **2**(2), 121-167, 1998.

6. Y. Chen and J.Z. Wang, Support Vector Learning for Fuzzy Rule-Based Classification Systems, *IEEE Transactions on Fuzzy Systems*, **11**(6), 716-728, 2003.
7. B.V. Dasarathy, Intrusion detection, *Information Fusion*, **4**(4), 243-245, 2003.
8. D. Lowe, Characterising complexity by the degrees of freedom in a radial basis function network, *Neurocomputing*, **19**(1-3), 199-209, 1998.
9. J.E. Dickerson, J. Juslin, O. Loulousoula, and J. A. Dickerson, Fuzzy Intrusion Detection, *IFSA World Congress and 20th North American Fuzzy Information Processing Society (NAFIPS) International Conference*, 2001, pp1506-1510.
10. K. Julish, Data Mining for Intrusion Detection, a Critical Review, in Applications of Data Mining in Computer Security, D. Barbara and S. Jajodia (Eds.), Kluwer Academic Publisher, 2002.
11. S. Kumar and E. Spafford, A software Architecture to Support Misuse Intrusion Detection, *18th National Information Security Conference*, 1995, pp194-204.
12. T. Joachims, Making large-Scale SVM Learning Practical, in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges and A. Smola (Eds.), MIT-Press, 1999.
13. KDD Cup 1999: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
14. W. Lee and S.J. Stolfo, Data Mining Approaches for Intrusion Detection, *7th USENIX Security Symposium*, 1998, pp. 79-94.
15. J. Luo and S.M. Bridges, Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection, *International Journal of Intelligent Systems*, **15**(8), 687-704, 2000.
16. M. Mohajerani, A. Moeini and M. Kianie, NFIDS: A Neuro-fuzzy Intrusion Detection System, *Proceedings of the 10th IEEE International Conference on Electronics*, Circuits and Systems, 2003, pp348-351.
17. S. Mukkamala and A.H. Sung, Artificial Intelligent Techniques for Intrusion Detection, *IEEE International Conference on Systems, Man and Cybernetics*, 2003, vol.2, 1266-1271.
18. Y. Qiao, X.W. Xin, Y. Bin and S. Ge, Anomaly Intrusion Detection Method Based on HMM, *Electronics Letters*, **38**(13), 663-664, 2002.
19. H. Shah, J. Undercoffer and A. Joshi, Fuzzy Clustering for Intrusion Detection, *The 12th IEEE International Conference on Fuzzy Systems*, May 2003, vol. 2, pp1274-1278.
20. V.N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995.
21. W.D. Wang and S. Bridges, Genetic Algorithm Optimization of Membership Functions for Mining Fuzzy Association Rules, *Proceedings of the 7th International Conference on Fuzzy Theory & Technology*, Atlantic City, NJ, 2000, pp131-134.
22. L.A. Zadeh, Fuzzy Sets, *Information and Control*, **8**(3), 338-353, 1965.