ANOMALY DETECTION OF INTRUSION BASED ON INTEGRATION OF ROUGH SETS AND FUZZY *c*-means

Witcha Chimphlee[1], Mohd Noor Md Sap[2], Abdul Hanan Abdullah[3], Siriporn Chimphlee[4]

Faculty of Computer Science and Information Systems
University Technology of Malaysia,
81310 Skudai, Johor, Malaysia
Tel: (607)-5532070, Fax: (607) 5565044
[1]witcha_chi@dusit.ac.th, [2]mohdnoor@fsksm.utm.my, [3]hanan@fsksm.utm.my,[4]siriporn_chi@dusit.ac.th

Abstract: *As malicious intrusions are a growing problem, we need a solution to detect the intrusions accurately. Network administrators are continuously looking for new ways to protect their resources from harm, both internally and externally. Intrusion detection systems look for unusual or suspicious activity, such as patterns of network traffic that are likely indicators of unauthorized activity. New intrusion types, of which detection systems are unaware, are the most difficult to detect. The amount of available network audit data instances is usually large; human labeling is tedious, time-consuming, and expensive. The objective of this paper is to describe a rough sets and fuzzy c-means algorithms and discuss its usage to detect intrusion in a computer network. Fuzzy systems have demonstrated their ability to solve different kinds of problems in various applications domains. We are using a Rough Sets to select a subset of input features for clustering with a goal of increasing the detection rate and decreasing the false alarm rate in network intrusion detection. Fuzzy c-Means allow objects to belong to several clusters simultaneously, with different degrees of membership. Experiments were performed with DARPA data sets, which have information on computer networks, during normal behavior and intrusive behavior.*

Keywords: Anomaly detection, Unsupervised clustering, Rough Set, Fuzzy *c*-means, Clustering

1. Introduction

As defined in [1], intrusion detection is "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. It is also defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network". Anomaly Intrusion Detection Systems (IDSs) aim at distinguishing an abnormal activity from an ordinary one. IDS have become a major focus of computer scientists and practitioners as computer attacks have become an increasing threat to commercial business as well as our daily lives.

Intrusion detection model is a composition model that needs various theories and techniques as show in fig 1. Identifying of suspicious activities before they have an impact to perform situational assessment to respond in a more timely and effective manner. Events that may not be actual security violations but those that do not fit in the normal usage profile of a user may be termed as suspicious events. Monitoring suspicious activities may help in finding a possible intrusion.

There are two main intrusion detection systems. *Anomaly intrusion detection system* is based on the profiles of normal behaviors of users or applications and checks whether the system is being used in a different manner [2]. The second one is called *misuse intrusion detection system* which collects attack signatures, compares a behavior with these attack signatures, and signals intrusion when there is a match.

Generally, there are four categories of attacks [3]. They are:

- DoS (denial-of-service), for example ping-of-death, teardrop, smurf, SYN flood, and the like.
- R2L : unauthorized access from a remote machine, for example guessing password,
- U2R : unauthorized access to local super user (root) privileges, for example, various "buffer overflow" attacks,
- PROBING: surveillance and other probing, for example, port-scan, ping-sweep, etc.

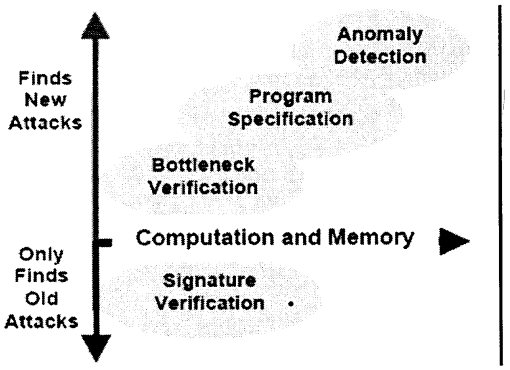

Fig. 1. Approaches to Intrusion Detection [4]

Some of the attacks (such as DoS, and PROBING) may use hundreds of network packets or connections, while on the other hand attacks like U2R and R2L typically use only one or a few connections [5]. Without any prior knowledge of attacks classification, if we attempt to divide this set of data into similar groupings, it would not be clear how many groups should be created [6]. IDS can be classified based on the functional characteristics of detection methods as knowledge based intrusion detection and behavior based intrusion detection [7]. The task of an intrusion detection system is to protect a computer system by detecting and diagnosing attempted breaches of the integrity of the system. To monitor activities on individual workstations, activities of users and network traffic for suspected intrusive behavior and pass information about the suspicious activities to the decision engine.

The goal for handle intrusion detection problem is to classify patterns of the system behavior in two categories (normal and abnormal), using patterns of known attacks, which belong to the abnormal class, and patterns of the normal behavior. In general, the input data to classifiers is in a high dimension feature space, but not all of features are relevant to the classes to be classified. A key problem is how to choose the features (attributes) of the input training data on which learning will take place. Since not every feature of the training data may be relevant to the detection task and, in the worse case, irrelevant features may introduce noise and redundancy into the design of classifiers, choosing a good subset of features will critical to improve the performance of classifiers [8].

This paper is organized as follows: in section 2, we briefly the related works; introduced rough set in section 3; fuzzy *c*-means is presented in section 4; Experimental setup and results are illustrated in section 5. We are reported experiments results and conclusion are drawn in section 6.

2. Related works

Most intrusion occurs via network using the network protocols to attack their targets. Twycross [9] proposed a new paradigm in immunology, Danger Theory, to be applied in developing an intrusion detection system. Intrusion detection is a critical component of secure information systems. Many approaches have been proposed which include statistical [6], machine learning [10], data mining [11] and immunological inspired techniques [12]. Alves *et al* [5] presents a classification-rule discovery algorithm integrating artificial immune systems (AIS) and fuzzy systems. For example, during a certain intrusion, a hacker follows fixed steps to achieve his intention, first sets up a connection between a source IP address to a target IP, and sends data to attack the target [2]. Punch *et al.* [13], Pei *et al.* [8] and Huang *et al.* [14] address the problem of feature selection and extraction by using genetic algorithm to find an optimal (or nearly optimal) weighting of features for k-Nearest Neighbor classifier. Huang *et al.* [14] apply a GA to find optimal subset of features for a Bayes classifier and a linear regression classifier.

Attack connection and normal connections have their special feature values and flags in the connection head, and package contents can be used as signatures for normal determination and intrusion detection. Intrusions belonging to the same intrusion category have identical or similar attack principles and intrusion techniques. Therefore they have identical or similar attack connections and are significantly different from normal connections [2]. Machine learning can be applied to either signature or anomaly detection. If we are given training instances labeled both normal and hostile (or labeled with the type of attack), then we are using signature detection. Such data is difficult to obtain.

3. Rough Sets

Rough set theory (RST) has been used successfully as a selection tool to discover data dependencies and reduce the number of attributes contained in a dataset by purely structural methods [15, 16]. Rough sets remove superfluous information by examining attribute dependencies. It deals with inconsistencies, uncertainty and incompleteness by imposing an upper and a lower approximation to set membership. Given a dataset with discretized attribute values, by the use of rough sets it is possible to find a subset (termed a reduct) of the original attributes using rough sets that are the most informative; all other attributes can be removed from the dataset with minimal information loss [17].

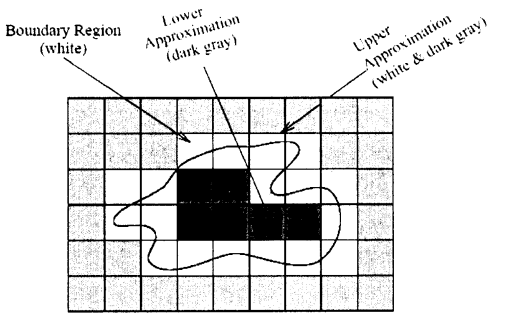

Fig. 2: Rough Representation of a Set with Upper and Lower Approximations

The rough sets theory has been developed for knowledge discovery in databases and experimental data sets. An attribute-oriented rough sets technique reduces the computational complexity of learning processes and eliminates the unimportant or irrelevant attributes so that the knowledge discovery in database or in experimental data sets can be efficiently learned.

A rough set is an approximation of a vague concept by a pair of precise concepts, called lower and upper approximations (which are a classification of the domain of interest into disjoint categories).The classification formally represents knowledge about the problem domain. Objects belonging to the same category characterized by the same attributes (or features) are not distinguishable [18]. Let I = (U, A) be an information system, where U is a non-empty set of finite of objects (the universe). A is a non-empty finite set of attributes such that $a : U \rightarrow V_a$

For every $a \in A$; V_a is the value set for attribute a. In a decision system, $A = \{C \cup D\}$ where C is the set of conditional attributes and D is the set of decision attributes. With any $P \subseteq A$ there is an associated equivalence relation IND (P):

Table 1: Decision table for rough set

Instance	Attributes			Decision field
	Service	Count	Srv_count	
1	http	1	4	Yes
2	ftp_data	2	3	Yes
3	Private	1	5	No
4	http	1	1	Yes
5	Domain_u	2	3	No
6	http	0	2	No

$$IND(P) = \{(x, y) \in U^2 \mid \forall_a \in P_a(x) = a(y)\} \quad (1)$$

If $(x, y) \in IND(P)$, then x and y are indiscernible by attributes from P. The partition of U, generated by IND (P) is denoted U/P and can be calculated as follows:

$$U / P = \otimes\{a \in P : U / IND(\{a\})\}, \text{ where} \quad (2)$$

$$A \otimes B = \{X \cap Y : \forall X \in A, \forall Y \in B, X \cap Y \neq \varnothing\} \quad (3)$$

To illustrate the operation of Rough Set Attribute Reduction (RSAR), an example dataset is presented as in Table 1, instance 1, 2, and 4 are attack data from properties in that table.

4. Fuzzy *c*-means (FCM) clustering

Clustering is an unsupervised learning technique of data mining that takes unlabeled data points and tries to group them according to their similarity: points assigned to the same cluster have high similarity, while the similarity between points assigned to different clusters is low [19]. Fuzzy *c*-means (FCM) algorithm, also known as fuzzy ISODATA, was introduced by Bezdek [1981] as extension to Dunn's [20] algorithm to generate fuzzy sets for every observed feature. The Fuzzy *c*-means clustering algorithm is based on the minimization of an objective function called *c-means functional.*

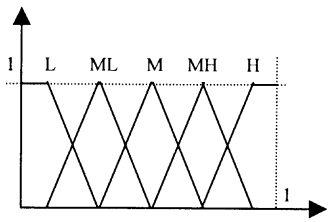

Fig. 3: A fuzzy space of five membership function

The fuzzy membership functions corresponding to the informative regions are stored as cases. A collection of fuzzy sets, called fuzzy space, defines the fuzzy linguistic values or fuzzy classes. A sample fuzzy space of five membership function is shown in Fig. 3. The fuzzy *c*-mean (FCM) algorithm provides fairly good results when applied to the analysis of multi-spectra thematic maps. Fuzzy clustering methods allow for uncertainty in the cluster assignments. Rather that partitioning the data into a collection of distinct sets (where each data point is assigned to exactly one set), fuzzy clustering creates a fuzzy pseudo partition, which consists of a collection of fuzzy sets. Fuzzy sets differ from traditional sets in that membership in the set is allowed to be uncertain. A fuzzy set is formalized by the following definitions.

It partitions a set of N patterns $\{X_k\}$ into c clusters by minimizing the objective function

$$J = \sum_{k=1}^{N} \sum_{i=1}^{c} \left(\mu_{ik}\right)^{m'} || X_k - m_i ||^2 \qquad (4)$$

Where $1 \le m' < \infty$ is the fuzzifier, m_i is the i^{th} cluster center, $\mu_{ik} \in [0,1]$ is the membership of the k^{th} pattern to it, and $||\cdot||$ is the distance norm, such that

$$m_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^{m'} X_k}{\sum_{k=1}^{N} (\mu_{ik})^{m'}} \qquad (5)$$

and

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik}}{d_{jk}} \right)^{\frac{2}{m'-1}}} \tag{6}$$

$\forall i$, with $d_{ik} = \| X_k - m_i \|^2$, subject to $\sum_{i=1}^{c} \mu_{ik} = 1, \quad \forall i$, and $0 < \sum_{k=1}^{N} \mu_{ik} < N, \forall i.$

The algorithm proceeds as follows [16].

(i) Pick the initial means m_i, i=1,...,c. Choose values for fuzzifier m' and threshold ε. Set the iteration counter t=1.

(ii) Repeat Steps (iii)-(iv), by incrementing t, until $| \mu_{ik}(t) - \mu_{ik}(t-1) | > \varepsilon$.

(iii) Compute μ_{ik} by Eq. (6) for c clusters and N data objects.

(iv) Update means m_i by Eq. (5).

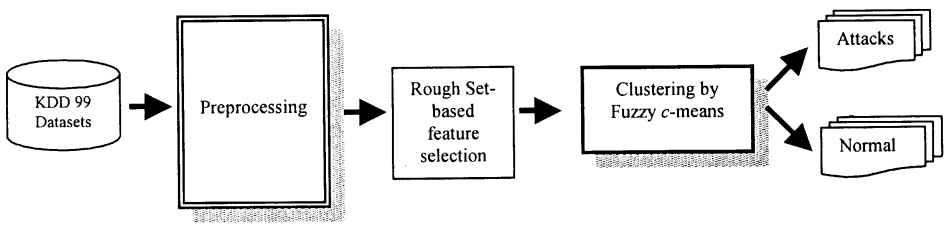

Fig 4. Overall Structure of Proposed Method

Note that for $\mu_{ik} \in [0,1]$ the objective function of Eq. (4) boils down to the hard c-means case, whereby a *winner-take-all* strategy is applied in place of membership values in Eq (5).

A detailed description of this method is shown in Fig 4. In first phase, we processed about preprocessing also handle missing and incomplete data. In second phase, feature selection using genetic algorithm and optimization fuzzy membership functions by fuzzy c-means for detection group of data. In addition to this process, we manipulated the KDD'99 data set with importance attribute for processing.

The preprocessing module performs the following tasks:

1. Identifies the attributes and their value.
2. Convert categorical to numerical data.
3. Data Normalization
4. Performs redundancy check and handle about null value.

5. Experimental setup and results

In this experiment, we use a standard dataset the raw data used by the KDD Cup 1999 intrusion detection contest [20]. This database includes a wide variety of intrusions simulated in a military network environment that is a common benchmark for evaluation of intrusion detection techniques. Test data use filename *"corrected.gz"* contains a total of 38 training attack types. It consists of approximately 300,000 data instances, each of which is a vector of extracted feature values from a connection record obtained from the raw network data gathered during the simulated intrusion and is labeled normal or a certain attack type. The 41 features can be divided into three groups; the first group is the basic feature of individual TCP connections, the second group is the content feature within a connection suggested by domain knowledge, and the third group is the traffic feature computed using a two-second time window. The distribution of attacks in the KDD Cup dataset is extremely unbalanced. Some attacks are represented with only a few examples, e.g. the *phf* and *ftp_write* attacks, whereas the *smurf* and *neptune* attacks cover millions of records. In general, the distribution of attacks is dominated by probes and denial-of-service attacks; the most interesting – and dangerous – attacks, such as compromises, are grossly under-represented [21].

The data set has 41 attributes (show in table 3) for each connection record plus one class label. There are 24 attack types, but we treat all of them as an attack group. A data set of size N is processed. The nominal attributes are converted into linear discrete values (integers). After eliminating labels, the data set is described as a matrix X, which has N rows and m=41 columns (attributes). There are m_d=8 discrete-value attributes and m_c = 33 continuous-value attributes. We ran our experiments on a system with a 1.5 GHz Pentium IV processor and 512 MB DDR RAM running Windows XP. All the preprocessing was done using MATLAB®. MATLAB's Fuzzy Logic Toolbox [22] was used for Fuzzy c-means clustering. In practice, the number of classes is not always known beforehand. There is no general theoretical solution to finding the optimal number of clusters for any given data set. We choose k = 5 for the study. We will compare five classifiers which have been also used in detecting these four types of attacks.

5.1. Data preprocessing

Before preparing data sets, we give two assumptions. The first assumption is that the number of normal instances far exceeds the number of intrusive instances. The second assumption is that the intrusive instances are qualitatively different from the normal instances. A considerable amount of data-preprocessing had to be undertaken before we could do any of our modeling experiments. It was necessary to ensure though, that the reduced dataset was as representative of the original set as possible. The test dataset began with more than 300,000 records was reduced to approximately 111,000 records. Table 2 shows the dataset after balanced among category for attack distribution over modified the normal and other attack categories. Preprocessing consisted of two steps. The first step involved mapping symbolic-valued attributes to numeric-valued attributes and the second step implemented non-zero numerical features.

Table 2. Dataset for attack distribution

Attack Category	% Occurrence	Number of records
normal	54.52	60593
Probe	3.75	4166
DoS	26.96	29970
U2R	0.06	70
R2L	14.71	16347

5.2 Feature extraction system

A feature extraction algorithm has been developed for the purposes of improving classification accuracy. The algorithm uses a genetic algorithm to generate a set of features, which may be of reduced dimensionality compared with the original set. The genetic algorithm includes the operators: crossover, mutation, and deletion/reactivation – the last of these effects dimensionality reduction. A feature selection method is proposed to select a subset of variables in data that preserves as much information present in the complete data as possible.

The benefits and affects of feature subset selection include [23]:

- Feature subset selection affects the accuracy of a learning algorithm.
- Feature subset selection reduces the computational effort required by a learning algorithm. Reducing the feature set to exclude irrelevant features reduces the size of the search space and then reduces the learning effort.
- The number of examples required to learn a classification function depends on the number of features.
- Feature subset selection can also result in lower cost of classification.

Let A be a set of given instances, which are characterized by d features $X = \{X_j : j = 1,...,d\}$. Each feature is either a nominal or a linear attribute. An attribute is linear attribute. An attribute is linear if the evaluation of the difference between two of its values has sense (being discrete or continuous); otherwise it is nominal. Furthermore, each instance has a label that indicates the class to which it belongs. In order to carry out the task of classifying by means of supervised learning, we consider the subset of instances T $\subset$ A in which labels are known and can be used as training examples, and the subset V = A\T of instances to be classified (validation instances). The labels of V will only be used to measure the performance of the classifier. In the feature subset selection problem, the set of features with the best performance must be obtained. The accuracy percentage is often used to measure the performance of a classifier. Then, the optimization problem associated consists of finding the subset S $\subseteq \{Xj : j = 1,. . . ,d\}$ with higher accuracy percentage. However, this percentage can only be estimated using the validation instances, since V is only a subset of the set of instances to be classified.

5.3 Intrusion Detection with Clustering

Based on the practical assumption that normal instances dominate attack instances, our simple self-labeling heuristic for unsupervised intrusion detection consists of the following steps [24]:

1. Find the largest cluster and label it *normal;*
2. Sort the remaining clusters in ascending order of their distances to the largest cluster;
3. Select the first *K1* clusters so that the number of data instances in these clusters sum up to $\approx \eta N$, and label them as *normal,* where η is the percentage of normal instances (this number needs to be estimated if not known *a priori*);
4. Label all the other clusters as *attacks.*

Unsupervised intrusion detection is more appropriate for anomaly detection in a dynamic intrusion detection environment for accommodating the changes in the characteristics of attacks.

5.4 Performance measure

Standard measures for evaluating IDSs include *detection rate, false alarm rate, trade-off between detection rate and false alarm rate* [23], *performance* (Processing speed + propagation + reaction), and *Fault Tolerance* (resistance to attacks, recovery, and subversion). 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 numbers of normal connections that are incorrectly misclassified as attacks [25]. These are good indicators of performance, since they measure what percentage of intrusions the system is able to detect and how many incorrect classifications are made in the process.

Standard metrics that were developed for evaluating network intrusions usually correspond to detection rate as well as false alarm rate.

1. True Positives (TP), the number of malicious executables correctly classified as malicious;
2. True Negatives (TN), the number of benign programs correctly classified as benign;
3. False Positives (FP), the number of benign programs falsely classified as malicious,
4. False Negative (FN), the number of malicious executables falsely classified as benign.

May be defines as follows:

$$\text{Detection Rate (DTR)} = \frac{TP}{(TP + FN)} \tag{7}$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{(TN + FP)} \tag{8}$$

$$Overall\ Accuracy\ (OA) = \frac{TP + TN}{TP + TN + FP + FN} \quad (9)$$

Another valuable tool for evaluating an anomaly detection scheme is the receiver operating characteristic (ROC) curve, which is the plot of the detection rate against the false alarm rate. The nearer the ROC curve of a scheme is to the upper-left corner, the better the performance of the scheme is. Anomaly detection amounts to training models for normal traffic behavior and then classifying as intrusions any network behavior that significantly deviates from the known normal patterns and to construct a set of clusters based on training data to classify test data instances.

5.5 Experimental Results

In our experiments, we consider a subset of the KDD data, which consists of 18217 instances. Except for a few seed points, the labeling information is not utilized in the experiments. Our results are summarized in the Table 4 is comparison of different 5 classes. Define membership function equal than 0.5. It is, however, used after our experiments are performed, to evaluate the results and calculate the Detection Rate and False Alarm Rate. As shown in Table 4, the normal activities had detection rate is high as 99.76% and DoS type had low detection rate as 82.05%. However, detection rates about Normal, Probe, U2R, and R2L are comparatively high, but detection rates about DoS are so low. The primarily results show that the performance of a proposed approach based on fuzzy c-means after using fuzzy c-means is good.

6. Conclusions

Rapid expansion of computer network throughout the world has made security a crucial issue in a computing environment. Anomaly-based network intrusion detection is a complex process. In this paper, rough sets are used to select the best feature subset and also to optimize fuzzy membership functions in order to improve the performance of Network Intrusion Detection System. Genetic feature subset selection significantly reduces the number of features used without adversely affecting the accuracy of the predictions. The advantage of using fuzzy logic is that it allows one to represent concepts that could be considered to be in more than one category (or from another point of view – it allows representation of overlapping categories). Feature subset selection reduced the number of features in the data, which should result in less data required for training due to smaller search space. Feature selection also gave equivalent accuracy with a smaller set of features. These results are very promising since detection accuracy at low false-positive rates is extremely important in IDS. One or two models can hardly offer satisfying results. We plan to apply other theories and techniques in intrusion detection in our future work.

7. References

[1] R. Bace and P. Mell, "Intrusion Detection Systems", *NIST Special Publications SP 800-31* November 2001.

[2] H. Jin, J. Sun, H. Chen, and Z. Han, "A Fuzzy Data Mining Based Intrusion Detection System", *Proceedings of 10th International Workshop on future Trends in Distributed Computing Systems (FTDCS04) IEEE Computer Society*, Suzhou, China, May 26-28, 2004, pp. 191-197.

[3] W. Lee, S. Stolfo, and K.Mok, "A Data Mining Framework for Building Intrusion Detection Models", *Proceedings of the 1999 IEEE Symposium on Security and Privacy,* May 1999, pp. 120-132.

[4] R. K. Cunnigham, "Detecting and Displaying Novel Computer Attacks with Macroscope", *Proceeding of 2000 IEEE*, West Point, NY, 6-7 June, 2000.

[5] R.T. Alves, M.R.B.S. Delgado, H.S. Lopes, A.A. Freitas, "An Artificial Immune System for Fuzzy-rule Induction in Data Mining", *Lecture Notes in Computer Science*, Berlin: Springer-Verlag, v. 3242, 2004, pp. 1011-1020.

[6] D. Denning, "An intrusion-detection model," *In IEEE computer society symposium on research in security and Privacy,* 1986, pp. 118-131.

[7] B. Balajinath, S.V. Raghuvan, "Intrusion Detection through Learning Behavior Model", 2001.

[8] M.Pei., E.D.Goodman, and W. F.Punch, "Feature Extraction using Genetic Algorithms", *In International Symposium on Intelligent Data Engineering and Learning'98*, pp. 371-384.

[9] J. Twycross, "Immune Systems, Danger Theory and Intrusion Detection", *presented at the AISB 2004 Symposium on Immune System and Cognition*, Leeds, U.K., March 2004.

[10] T. Lane, "*Machine Learning techniques for the Computer Security*", PhD thesis, Purdue University, 2000.

[11] W. Lee and S. Stolfo, "Data Mining Approaches for Intrusion Detection," *Proceedings of the 7th USENIX security Symposium*, 1998.

[12] D. Dagupta and F. Gonzalez, "An Immunity-based Technique to Characterize Intrusions in Computer Networks", *IEEE Transactions on Evolutionary Computation,* Vol. 6, June 2002, pp.28- 291.

[13] W. F. Punch, E.D.Goodman, M. Pei, L.Chia-Shun, P.Hovland, and R.Enbody, "Further Research on Feature Selection and Classification using Genetic Algorithms", In 5th International Conference Genetic Algorithms, 1993.

[14] Z.Huang, M.Pei, E.Goodman, Y.Huang, and G.Li, "Genetic Algorithm Optimized Feature Transformation: a Comparison with Different Classifiers, *In Proceeding of GECCO 2003*, pp. 2121-2133.

[15] A. Chouchoulas, Q. Shen, "Rough set-aided keyword reduction for text categorization", Application Artificial Intelligence, vol. 15, no.9, pp. 843-873, 2001.

[16] R. Jensen, Q. Shen, "Finding Rough Set Reducts With Ant Colony Optimization, In Proc. 2003 UK Workshop on Computational Intelligence, pp.15-22, 2003.

[17] R. Jensen and Q. Shen, "Fuzzy-Rough Data Reduction with Ant Colony Optimization", Fuzzy Sets and Systems, vol.149, no.1, pp. 5-20, 2005.

[18] R. Jensen and Q. Shen, Rough and Fuzzy Sets for Dimensionality Reduction, Proceedings of the 2001 UK Workshop on Computational Intelligence, pp. 69-74, 2001.

[19] R. Duda and P. Hart, *"Pattern Classification and Scene Analysis"*, NY: Wiley Interscience, 1973.

[20] J.C. Dunn, "A Fuzzy Relative of the ISODATA process and its Use in Detecting Compact", well Separated Clusters, *Journal of Cybernetics*, Vol. 3, No.3, 1974, pp. 32-57.

[21] P. Laskov, K. Rieck, C. Schäfer, and K.R. Müller, "Visualization of anomaly detection using prediction sensitivity", *Proceeding of Sicherheit*, April 2005, pp.197- 208.

[22] KDD data set, 1999; http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[23] W. Chimphlee, Abdul Hanan Abdullah, Mohd Noor Md Sap and S. Chimphlee, "Unsupervised Anomaly Detection with "Unlabeled Data using Clustering", *International Conference on ICT-Mercu Buana "ICT2005"*, 2005, pp. 42-49.

[24] S. Zhong, T. Khoshgoftaar, and N. Seliya, "Evaluating Clustering Techniques for Network Intrusion Detection", *10th ISSAT Int. Conf. on Reliability and Quality Design*, 2004, pp. 149-155.

[25] A. Lazarevic, A. Ozgur, L. Ertoz, J. Srivastava, and V. Kumar, "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection". *In SIAM International Conference on Data Mining*, 2003.

Table 3. Variables for intrusion detection data set

No.	Variable name	Type	Variable label
1	duration	continuous	A
2	protocol_type	discrete	B
3	service	discrete	C
4	Flag	discrete	D
5	src_bytes	continuous	E
6	dst_bytes	continuous	F
7	Land	discrete	G
8	wrong_fragment	continuous	H
9	urgent	continuous	I
10	Hot	continuous	J
11	Num_failed_logins	continuous	K
12	logged_in	discrete	L
13	Num_compromised	continuous	M
14	Root_shell	continuous	N
15	su_attempted	continuous	O
16	Num_root	continuous	P
17	Num_file_creations	continuous	Q
18	Num_shells	continuous	R
19	Num_access_files	continuous	S
20	Num_outbound_cmds	continuous	T
21	is_host_login	discrete	U
22	is_guest_login	discrete	V
23	count	continuous	W
24	srv_count	continuous	X
25	serror_rate	continuous	Y
26	srv_serror_rate	continuous	Z
27	rerror_rate	continuous	AA
28	srv_rerror_rate	continuous	AB
29	same_srv_rate	continuous	AC
30	diff_srv_rate	continuous	AD
31	srv_diff_host_rate	continuous	AE
32	dst_host_count	continuous	AF
33	dst_host_srv_count	continuous	AG
34	dst_host_same_srv_rate	continuous	AH
35	dst_host_diff_srv_rate	continuous	AI
36	dst_host_same_src_port_rate	continuous	AJ
37	dst_host_srv_diff_host_rate	continuous	AK
38	dst_host_serror_rate	continuous	AL
39	dst_host_srv_serror_rate	continuous	AM
40	dst_host_rerror_rate	continuous	AN
41	dst_host_srv_rerror_rate	continuous	AO

Table 4. Detection rates of each attack for fuzzy c-means

Attack type	Attack name	#of attack instances	Detected (detection rate)	Missed (missed rate)
DoS		3531	**2897(82.05%)**	**634(17.96%)**
1	apache2	368	215(58.42%)	153(41.58%)
2	pod	87	87(100%)	0(0%)
3	smurf	740	740(100%)	0(0%)
4	back	483	2(0.42%)	481(99.59%)
5	land	9	9(100%)	0(0%)
6	mailbomb	732	732(100%)	0(0%)
7	neptune	707	707(100%)	0(0%)
8	processtable	391	391(100%)	0(0%)
9	teardrop	12	12(100%)	0(0%)
10	udpstorm	2	2(100%)	0(0%)
Probe		2164	**2164(100%)**	**0(0%)**
11	ipsweep	306	306(100%)	0(0%)
12	portsweep	354	354(100%)	0(0%)
13	saint	674	674(100%)	0(0%)
14	mscan	501	501(100%)	0(0%)
15	nmap	84	84(100%)	0(0%)
16	satan	245	245(100%)	0(0%)
U2R		70	**67(95.72%)**	**3(4.29%)**
17	buffer_overflow	22	21(95.46%)	1(4.55%)
18	loadmodule	2	2(100%)	0(0%)
19	perl	2	2(100%)	0(0%)
20	ps	16	16(100%)	0(0%)
21	rootkit	13	11(84.62%)	2(15.39%)
22	sqlattack	2	2(100%)	0(0%)
23	xterm	13	13(100%)	0(0%)
R2L		6689	**6145(91.87%)**	**558(8.34%)**
24	guess_passwd	110	110(100%)	0(0%)
25	multihop	18	18(100%)	0(0%)
26	named	17	13(76.47%)	4(23.53%)
27	phf	2	2(100%)	0(0%)
28	sendmail	17	17(100%)	0(0%)
29	snmpgetattack	1975	1975(100%)	0(0%)
30	xlock	9	4(44.44%)	5(55.56%)
31	xsnoop	4	4(100%)	0(0%)
32	ftp_write	3	3(100%)	0(0%)
33	httptunnel	158	157(98.37%)	1(0.63%)
34	imap	1	1(100%)	0(0%)
35	snmpguess	2771	2771(100%)	0(0%)
36	warezmaster	1602	1068(66.67%)	534(33.33%)
37	worm	2	2(100%)	0(0%)
Normal	Normal	5763	**5749(99.76%)**	**14(0.24%)**
Total		18217	**17022(93.44%)**	**1195(6.56%)**