SCIDS: A Soft Computing Intrusion Detection System

Ajith Abraham[1], Ravi Jain[2], Sugata Sanyal[3], and Sang Yong Han[1]

[1] School of Computer Science and Engineering, Chung-Ang University, Korea
ajith.abraham@ieee.org, hansy@cau.ac.kr
[2] Land Operations Division, Defence Science & Technology Organisation (DSTO), Australia
[3] School of Technology and Computer Science,
Tata Institute of Fundamental Research, India
sanyal@tifr.res.in

Abstract. An Intrusion Detection System (IDS) is a program that analyzes what happens or has happened during an execution and tries to find indications that the computer has been misused. This paper evaluates three fuzzy rule based classifiers for IDS and the performance is compared with decision trees, support vector machines and linear genetic programming. Further, Soft Computing (SC) based IDS (SCIDS) is modeled as an ensemble of different classifiers to build light weight and more accurate (heavy weight) IDS. Empirical results clearly show that SC approach could play a major role for intrusion detection.

1 Introduction

Intrusion detection is classified into two types: misuse intrusion and anomaly intrusion detection [3][7]. Data mining approaches for IDS were first implemented in mining audit data for automated models [2]. Several data mining algorithms are applied to audit data to compute models that accurately capture the actual behavior of intrusions. This paper introduces three fuzzy rule based classifiers and compares the performance with Linear Genetic Programming (LGP) [1], Support Vector Machines (SVM) [8] and Decision Trees (DT) [5]. Further, we used Soft Computing (SC) [9] based IDS (SCIDS) as a combination of different classifiers to model light weight and more accurate (heavy weight) IDS. Rest of the paper is organized as follows. Section 2 provides the technical details of the three fuzzy rule based systems. Experiment results are presented in Section 3 and some conclusions are also provided towards the end.

2 Fuzzy Rule Based Systems

Let us assume that we have a n dimensional c-class pattern classification problem whose pattern space is an n-dimensional unit cube $[0, 1]^n$. We also assume that m patterns $x_p = (x_{p1},\ldots,x_{pn})$, $p = 1,2,\ldots,m,$ are given for generating fuzzy *if-then* rules where $x_p \in [0,1]$ for $p = 1,2,..., m.$

A. Sen et al. (Eds.): IWDC 2004, LNCS 3326, pp. 252–257, 2004.

2.1 Rule Generation Based on the Histogram of Attribute Values (FR_1)

In this method, use of histogram itself is an antecedent membership function. Each attribute is partitioned into 20 membership functions $f_h(.)$, h=1,2,...,20. The smoothed histogram $m_i^k(x_i)$ of class k patterns for the i^{th} attribute is calculated using the 20 membership functions $f_h(.)$ as follows:

$$m_i^k(x_i) = \frac{1}{m^k} \sum_{x_p \in Class\ k} f_h(x_{pi}) \quad for\ \beta_{h-1} \le x_i \le \beta_h,\ h=1,2,...,20 \tag{1}$$

where m_k is the number of Class k patterns, $[\beta_{h-1},\beta_h]$ is the h^{th} crisp interval corresponding to the 0.5-level set of the membership function $f_h(.)$

$$\beta_1=0,\ \beta_{20}=1,\ \beta_h=\frac{1}{20-1}\left(h-\frac{1}{2}\right) \text{ for } h=1,2,...,19 \tag{2}$$

The smoothed histogram in (1) is normalized so that its maximum value is 1. A single fuzzy *if-then* rule is generated for each class. The fuzzy *if-then* rule for the k^{th} class can be written as *If* x_1 is A_1^k and ... and x_n is A_1^k *then* class k (3)

where A_i^k is an antecedent fuzzy set for the i^{th} attribute. The membership function of A_i^k is specified as

$$A_i^k(x_i)=\exp\left(-\frac{(x_i-\mu_i^k)^2}{2(\sigma_i^k)^2}\right) \tag{4}$$

where μ_i^k is the mean of the i^{th} attribute values x_{pi} of class k patterns, and σ_i^k is the standard deviation. Fuzzy *if-then* rules for the two-dimensional two class pattern classification problem are written as follows:

If x_3 is A_3^1 and x_4 is A_4^1 *then* class 2 or *If* x_3 is A_3^2 and x_4 is A_4^2 *then* class 3.

Membership function of each antecedent fuzzy set is specified by the mean and the standard deviation of attribute values. For a new pattern $x_p = (x_{p3},x_{p4})$, the winner rule is determined as follows:

$$A_3^*(x_{p3}).A_2^*(x_{p4})=\max\left\{A_1^k(x_{p3}).A_2^k(x_{p4})\,|\,k=1,2\right\} \tag{5}$$

2.2 Rule Generation Based on Partition of Overlapping Areas (FR_2)

In this method m - dimensional pattern space is partitioned and a single fuzzy *if-then* rule is generated for each fuzzy subspace. Because the specification of each membership function does not depend on any information about training patterns, this approach uses fuzzy *if-then* rules with certainty grades. The local information about

training patterns in the corresponding fuzzy subspace is used for determining the consequent class and the grade of certainty. In this approach, fuzzy *if-then* rules of the following type are used.

If x_1 is A_{j1} and .. and x_n is A_{jn} *then* class C_j, with $CF = CF_j,\ j = 1, 2, .., N$ (6)

where j indexes the number of rules, N is the total number of rules, A_{ji} is the antecedent fuzzy set of the i^{th} rule for the i^{th} attribute, C_j; is the consequent class, and CFj is the grade of certainty. The consequent class and the grade of certainty of each rule are determined by the following simple heuristic procedure:

Step 1: Calculate the compatibility of each training pattern $x_p = (x_{p1}, x_{p2}, \ldots, x_{pn})$ with the j^{th} fuzzy *if-then* rule by the following product operation:

$$\pi_j\left(x_p\right) = A_{j1}\left(x_{p1}\right) \times \ldots \times A_{jn}\left(x_{pn}\right),\ p = 1, 2, \ldots, m. \tag{7}$$

Step 2: For each class, calculate the sum of the compatibility grades of the training patterns with the j^{th} fuzzy *if-then* rule R_j:

$$\beta_{class\ k}\left(R_j\right) = \sum_{x_p \in class\ k}^{n} \pi\left(x_p\right),\ \mathrm{k}=1,2,\ldots,\mathrm{c} \tag{8}$$

where $\beta_{class\ k}\left(R_j\right)$ the sum of the compatibility grades of the training patterns in class k with the j^{th} fuzzy if-then rule R_j.

Step 3: Find Class A_j^* that has the maximum value $\beta_{class\ k}\left(R_j\right)$: $\beta_{class\ \mathrm{k}_j^*} = Max\{\beta_{class\ 1}(R_j), \ldots, \beta_{class\ \mathrm{c}}(R_j)\}$.

If two or more classes take the maximum value or no training pattern compatible with the j^{th} fuzzy *if-then* rule (i.e., if $\beta_{Class\ k}(\mathrm{R_j})=0$ for $k = 1,2,\ldots, c$), the consequent class C_i can not be determined uniquely. In this case, let C_i be ϕ.

Step 4: If the consequent class C_i is 0, let the grade of certainty CF_j be $CF_j = 0$. Otherwise the grade of certainty CF_j *is* determined as follows:

$$CF_j = \frac{\left(\beta_{class\ \mathrm{k}_j^*}(R_j) - \bar{\beta}\right)}{\sum_{k=1}^{c} \beta_{class\ \mathrm{k}}\left(\mathrm{R_j}\right)}, \text{ where } \bar{\beta} = \sum_{\substack{k=1 \\ k \neq k_j^*}} \frac{\beta_{Class\ \mathrm{k}}(R_j)}{(c-1)} \tag{9}$$

2.3 Neural Learning of Fuzzy Rules (FR_3)

In a fused neuro-fuzzy architecture, neural network learning algorithms are used to determine the parameters of fuzzy inference system (membership functions and number of rules). We made use of the Evolving Fuzzy Neural Network (EFuNN) to implement a Mamdani type FIS [10].

3 Experiment Setup and Results

Certain features may contain false correlations, which hinder the process of detecting intrusions. Extra features can increase computation time, and can impact the accuracy of IDS. Feature selection is done based on the contribution the input variables made to the construction of the decision tree. Variable importance, for a particular variable is the sum across all nodes in the tree of the improvement scores that the predictor has when it acts as a primary or surrogate (but not competitor) splitter [11].

The data for our experiments was prepared by the 1998 DARPA intrusion detection evaluation program by MIT Lincoln Labs [6]. The data set has 41 attributes for each connection record plus one class label and they are named as *A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, AA, AB, AC, AD, AE, AF, AG, AH, AI, AJ, AK, AL, AM, AN*, *AO* and *AP* respectively. The data set contains 24 attack types that could be classified into four main categories *DoS*: Denial of Service, *R2L*: Unauthorized Access from a Remote Machine, *U2Su*: Unauthorized Access to Local Super User (root) and *Probing*. Our experiments have three phases namely input feature reduction, training phase and testing phase. In the data reduction phase, important variables for real-time intrusion detection are selected by feature selection. In the training phase, the different soft computing models were constructed using the training data to give maximum generalization accuracy on the unseen data. The test data is then passed through the saved trained model to detect intrusions in the testing phase. The training and test comprises of 5,092 and 6,890 records respectively [4]. All the training data were scaled to (0-1). The decision tree approach helped us to reduce the number of variables to 12 variables. The list of reduced variables is *C, E, F, L, W, X, Y, AB, AE, AF, AG* and *AI*. Using the original and reduced data sets, we performed a 5-class classification. We examined the performance of all three fuzzy rule based approaches (FR_1, FR_2 and FR_3) mentioned in Section 2. When an attack is correctly classified the grade of certainty is increased and when an attack is misclassified the grade of certainty is decreased. A learning procedure is used to determine the grade of certainty. Triangular membership functions were used for all the fuzzy rule based classifiers. For FR_1 and FR_2 two triangular membership functions were assigned and 2 and 2^{12} rules were learned respectively for the reduced data set. For FR_3, 4 triangular membership functions were used for each input variable. A sensitivity threshold *Sthr* = 0.95 and error threshold *Errthr* = 0.05 was used for all the classes and 89 rule nodes were developed. For comparison purposes various other results were adapted from [1][7][11].

A number of observations and conclusions are drawn from the results illustrated in Table 1. Using 41 attributes, the FR_2 method gave 100% accuracy for all the 5 classes, showing the importance of fuzzy inference systems. Using 12 attributes most of the classifiers performed very well except the fuzzy classifier (FR_2). For detecting U2R attacks FR_2 gave the best accuracy. However, due to the tremendous reduction in the number of attributes (about 70% less), we are able to design a computational efficient (light weight) IDS. Since a particular classifier could not provide accurate results for all the classes, we propose to use a combination of different classifiers to detect different attacks.

Table 1. Performance comparison using full/reduced data set

Attack type	Classification accuracy on test data set (%)					
	FR_1	FR_2	FR_3	DT	SVM	LGP
Full Dataset						
Normal	40.44	**100.00**	98.26	99.64	99.64	99.73
Probe	53.06	**100.00**	99.21	99.86	98.57	99.89
DOS	60.99	**100.00**	98.18	96.83	99.92	99.95
U2R	66.75	**100.00**	61.58	68.00	40.00	64.00
R2L	61.10	**100.00**	95.46	84.19	33.92	99.47
Reduced Dataset						
Normal	74.82	79.68	99.56	**100.00**	99.75	99.97
Probe	45.36	89.84	99.88	97.71	98.20	**99.93**
DOS	60.99	60.99	98.99	85.34	98.89	**99.96**
U2R	**94.11**	**99.64**	65.00	64.00	59.00	68.26
R2L	91.83	91.83	97.26	95.56	56.00	**99.98**

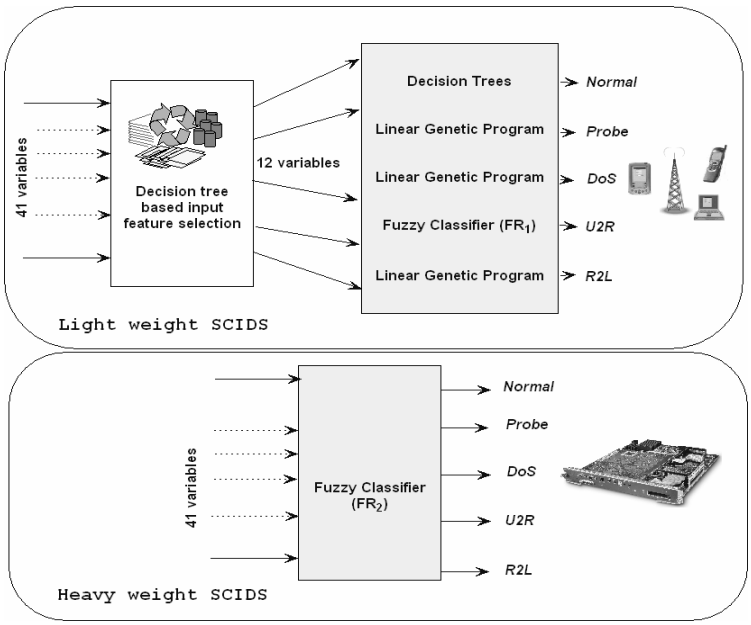

Fig. 1. Light/heavy weight SCIDS architecture

The Soft Computing Intrusion Detection System (SCIDS) using 41 attributes (heavy weight) and 12 attributes (light weight) are depicted in Figure 1. The proposed heavy weight model could detect with 100% accuracy while the light weight model could detect Normal, Probe, DOS, U2R and R2L class with 100.00%, 99.93%, 99.96%, 94.11% and 99.98% accuracies respectively.

4 Conclusions

In this paper, we have illustrated the importance of soft computing paradigms for modeling intrusion detection systems. For real time intrusion detection systems, LGP would be the ideal candidate as it can be manipulated at the machine code level. Overall, the fuzzy classifier (FR_2) gave 100% accuracy for all attack types using all the 41 attributes. The proposed hybrid combination of classifiers requires only 12 input variables. While the light weight SCIDS would be useful for MANET/distributed systems, the heavy weight SCIDS would be ideal for conventional static networks, wireless base stations etc.

References

[1] Abraham A., Evolutionary Computation in Intelligent Web Management, Evolutionary Computing in Data Mining, Ghosh A. and Jain L.C. (Eds.), Studies in Fuzziness and Soft Computing, Vol. 163, Springer Verlag Germany, 2004.

[2] Barbara D., Couto J., Jajodia S. and Wu N., ADAM: A Testbed for Exploring the Use of Data Mining in Intrusion Detection**.** SIGMOD Record, 30(4), pp. 15-24, 2001.

[3] Denning D., An Intrusion-Detection Model, IEEE Transactions on Software Engineering, Vol. SE-13, No. 2, pp.222-232, 1987.

[4] KDD Cup 1999 Intrusion detection data set: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.gz>

[5] Brieman L., Friedman J., Olshen R., and Stone C., Classification of Regression Trees. Wadsworth Inc., 1984.

[6] MIT Lincoln Laboratory. <http://www.ll.mit.edu/IST/ideval/>

[7] Peddabachigari S., Abraham A., Thomas J., Intrusion Detection Systems Using Decision Trees and Support Vector Machines, International Journal of Applied Science and Computations, USA, 2004.

[8] Vapnik V.N., The Nature of Statistical Learning Theory. Springer, 1995.

[9] Zadeh L. A., Roles of Soft Computing and Fuzzy Logic in the Conception, Design and Deployment of Information/Intelligent Systems, Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, O. Kaynak et al. (Eds.), pp 1-9, 1998.

[10] Kasabov N., Evolving Fuzzy Neural Networks-Algorithms, Applications and Biological Motivation, in Yamakawa T. et al. (Eds), Methodologies for the Conception, Design and Application of Soft Computing, World Scientific, pp. 271-274, 1998.

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