Journal of Computational Methods in Sciences and Engineering 10 (2010) S51–S60
DOI 10.3233/JCM-2010-0267
IOS Press

An effective intrusion detection method using optimal hybrid model of classifiers

Sultan Aljahdali
Computer Science Department, College of Computers and Information Systems, Taif University, Taif, Saudi Arabia
E-mail: aljahdali@tu.edu.sa

Abstract. With increasing connectivity between computers, the security of computer networks plays a strategic role in modern computer systems. In order to enforce high protection levels against threats, a number of software systems have been currently developed. Intrusion detection systems (IDS) have become an essential component at detecting intruders. In this paper, an ensemble approach to network intrusion detection based on the fusion of multiple classifiers is proposed. A computational machine is built to derive optimal parsimonious hybrid model of classifiers in intrusion detection based on the following classification methods, Naïve Bayes, Support Vector Machine, K-nearest neighbor, and Neural networks. The weighted voting fusion strategy for intrusion detection is assessed by experiments and its performances compared. The potentialities of classifiers fusion for the development of effective intrusion detection systems are evaluated and discussed. The experimental results indicate that hybrid approach effectively generates a more accurate intrusion detection model on detecting both normal usages and malicious activities. In this paper, we aim to build a robust classifier combination system given a classifier set.

Keywords: Intrusion detection, optimal hybrid model, network security, weighted voting fusion strategy, Naïve Bayes, Support Vector Machine, K-nearest neighbor, Neural Networks

1. Introduction

With the development of the internet, the information security threat is becoming one of the most crucial problems. Intrusion detection is the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource. Despite the effort devoted to carefully designing intrusion detection systems, network security is very difficult to guarantee, since attacks exploit unknown weaknesses or bugs, which are always contained in system and application software [21]. Intrusion Detection Systems (IDS) placed inside a protected network, looking for known or potential threats in network traffic and/or audit data recorded by hosts.

There are generally two distinct approaches in the field of intrusion detection: misuse (signature-based) detection and anomaly detection [32]. Misuse detection utilizes attack signatures, usually taking the form of rules, to detect intrusion. It gains a high detection rate for those well-known intrusions, but often fails to detect novel intrusions. Anomaly detection, however, tries to build up normal profiles, the patterns of normal behaviors. Any deviant from the normal profiles is considered as anomalies [8,30]. Because it is difficult to precisely establish the normal profiles, anomaly detection usually suffers from a higher false positive rate, the possibility that a normal behavior is mistakenly classified as an attack instance.

There have been plenty of methods in intrusion detection. A statistical method is proposed in [5], where several "*metrics*" are paid attention to and their statistical normal profiles are constructed. Enlightened

by that, many researchers try to build statistical models of a host system from various aspects [11]. Data mining is also widely studied and used in intrusion detection [9,33]. It focuses on extracting so-called "association rules" and "frequent episodes" from voluminous data, which are a specific kind of rules to describe the network activities. The trade-off between the ability to detect new attacks and the ability to generate a low rate of false alarms is the key point to develop effective IDS. Therefore, the misuse (signature-based) detection model is currently the most widely used due to its ability to produce very low false alarm rates at the price of a very limited ability to detect new attacks.

Recently, it is particularly popular to utilize the methods in machine learning to detect intrusions. The main motivation in using pattern recognition approaches to develop advanced IDSs is their generalization ability, which may support the classification of previously unseen intrusions that have no previously described patterns. In particular, machine learning approaches should allow the detection of the so-called attack "*variants*". The machine learning algorithms are primarily driven by the statistics that can be derived from the feature vectors [26,28]. One of the most used methods is the Bayesian classification; it attempts to calculate the probability that an event is an intrusion based upon previous feature frequencies in attack/non-attack event. Other famous learning algorithms used in intrusion detection systems are support vector machines, neural networks [10,34], k-nearest neighbor [34] and hidden Markov modeling [20, 30].

In this paper, an approach to intrusion detection in computer networks based on the optimal fusion of multiple classifiers is proposed. Each member of the classifier ensemble is trained on a distinct feature representation of patterns, and then the individual results are combined using the weighted voting fusion strategy. The proposed ensemble machine for intrusion detection presented in this paper groups three statistical methods and one computational model to improve the accuracy of our intrusion detection machine, this will be accomplished by combining different classifiers to achieve the best possible detection performance:

1. Naïve Bayes, NB
2. Support vector machine, SVM
3. K-nearest neighbor, K-nn
4. Neural networks, NN

These methods are combined with different technique of feature selection: chi-square χ^2, entropy and mutual information (MI). We have considered three linear fusion methods (voting, averaging and recursive least square) to combine the statistical methods.

The rest of the paper is organized in the following manner. Section 2 provides a survey of related works. Section 3 describes the classifications methods used in our computational machine. Section 4 describes the computational machine implemented in this paper. Section 5 describes the results and finally, the conclusions and future research needed to improve both the model and the machine described in Section 6.

2. Related works

The problem of huge network traffic data size and the invisibility of intrusive patterns which normally are hidden among the irrelevant and redundant features have posed a great challenge in the domain of intrusion detection [1]. One way to address this issue is to reduce these input features in order to disclose the hidden significant features. Thus, an accurate classification can be achieved, besides identifying significant features that can represent intrusive patterns; the choice of classifier can also

influence the accuracy and classification of an attack. The literature suggests that hybrid or assembling multiple classifiers can improve the accuracy of detection [22,34]. Classifier ensembles also known as committees are aggregations of several classifiers whose individual predictions are combined in some manner (e.g., averaging or voting) to form a final prediction [6,18]. An important advantage for combining redundant and complementary classifiers is to increase robustness, accuracy and better overall generalization in most applications [18,27]. Mukkamala et al. [23] demonstrated the use of ensemble classifiers gave the best accuracy for each category of attack patterns. Ensemble methods aim at improving the predictive performance of a given statistical learning or model fitting technique. The general principle of ensemble methods is to construct a linear combination of some model fitting method, instead of using a single fit of the method. In designing a classifier, the first step is to carefully construct different connectional models to achieve best generalization performance for classifiers. Chebrolu et al. [22] proposed CART-BN approach, where CART performed best for *Normal*, *Probe* and $U2R$ and the ensemble approach worked best for $R2L$ and *DoS*. Meanwhile, Abraham and Jain. [2] illustrated that ensemble Decision Tree was suitable for *Normal*, LGP for *Probe*, *DoS* and $R2L$ and Fuzzy classifier was for $R2L$. In their later work, Abraham et al. [3] also demonstrated the ability of their proposed ensemble structure in modeling light-weight distributed IDS. Meanwhile, Mukkamala et al. [18] proposed three variants of Neural Networks, SVM and MARS as components in their IDS. This combining approach has demonstrated better performance when compared to single classifier approach. Giorgio et al. [7] took a slightly different approach. Their anomaly IDS was based on modular multiple classifier system where each module was designed for each group of protocols and services. Each module might contain either individual or combination of different classifiers. The modular architecture would allow putting a rejection threshold of each module as to optimize the overall attack detection rate given a desired total false alarm rate for the ensemble. They reported that there was an improvement on attack detection rate and significant reduction on false alarm.

3. Statistical and computational models overview

3.1. Naïve Bayes classifier

Naïve Bayes classifier uses a probabilistic approach based on applying Bayes' theorem with strong (naive) independence assumptions for estimating probabilities of individual feature values, given a class, from training data and to then allow the use of these probabilities to classify new records. In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even though these features depend on the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple. Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods [19].

3.2. Support vector machine

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression. A support vector machine constructs a hyperplane or set of hyperplanes in a high or

infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training datapoints of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. Support vector machine constructs a two class classifier function that divides the feature space into two subspaces, one for each class. Using training set, SVM specifies in advance which data should cluster together [4].

3.3. K-nearest neighbors

The k-nearest neighbors' algorithm (k-NN) is a method for classifying objects based on closest training examples in the feature space. k-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. The k-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of its nearest neighbor.

3.4. Neural networks

The neural network algorithm that we have considered in this paper is back propagation. Back-propagation is the best known training algorithm for neural networks and the most useful. Back-propagation, or propagation of error, is a common method of teaching artificial neural networks how to perform a given task. It is a supervised learning method, and is an implementation of the Delta rule. It requires a teacher that knows, or can calculate, the desired output for any given input. It is most useful for feed-forward networks. It has lower memory requirements than most algorithms, and usually reaches an acceptable error level pretty quickly [4,23].

3.5. Combining classifiers

The mantra goal of an optimal hybrid model of classifiers is to determine the best achievable performance for attacks detection within the available classifiers. It has been observed that different classifier designs offer complementary information about the type of attacks to be detected, which could be combined to improve the performance of detecting different types of intrusion. A large number of combination methods have been proposed in the literature [6,32]. Actually, there is an extend recognition of advantages of combining multiple classifiers over the traditional monolithic approach to classifier design. For instance, Tumer and Ghosh [15,16,24] used simple average of classifier outputs to analyze the performance improvement of combined classifiers. Combining classifiers using the majority voting rule was provided by Lam and Suen [18]. Kuncheva [16] compared the classification error at a given point in the feature space, for majority voting, simple average, and order statistics rules. Kittler and Alkoot [10] compared the sum and majority vote rules both by experiments and theoretically.

In this paper, the focus on linear combiners and only the weighted voting combination technique for combining multiple classifiers is addressed. The weighted voting combination method operates as follows: we obtain classification from different classifiers, but instead of just choosing the most existing class (majority voting); we assign a weight to each and choose the highest. The weights are used to adjust the relative importance of each classifier. Let $C = \{C_1, C_2, ..., C_k\}$ be a set of k trained classifiers, and $\Omega = \{w_1, w_2, ..., w_m\}$ be a set of m class labels. Each classifier gets as its input a

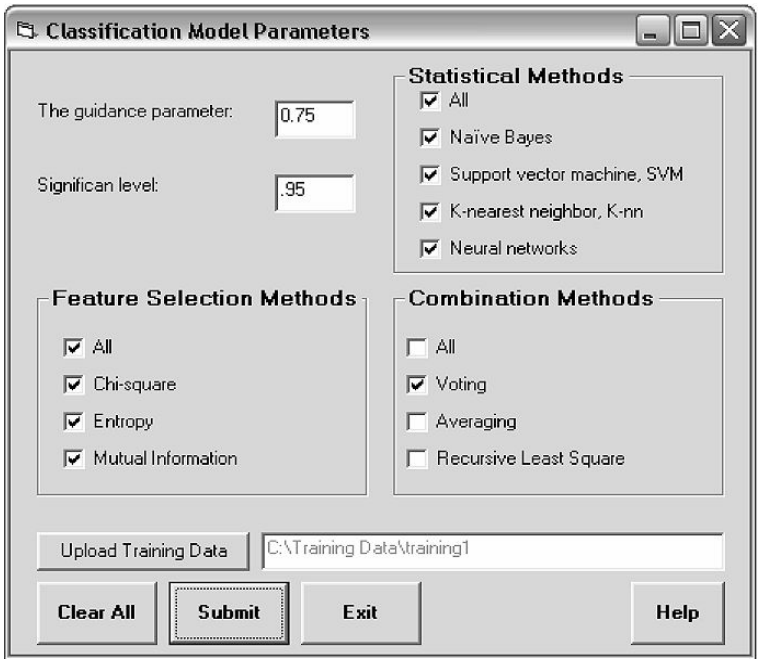

Fig. 1. Parameters interface.

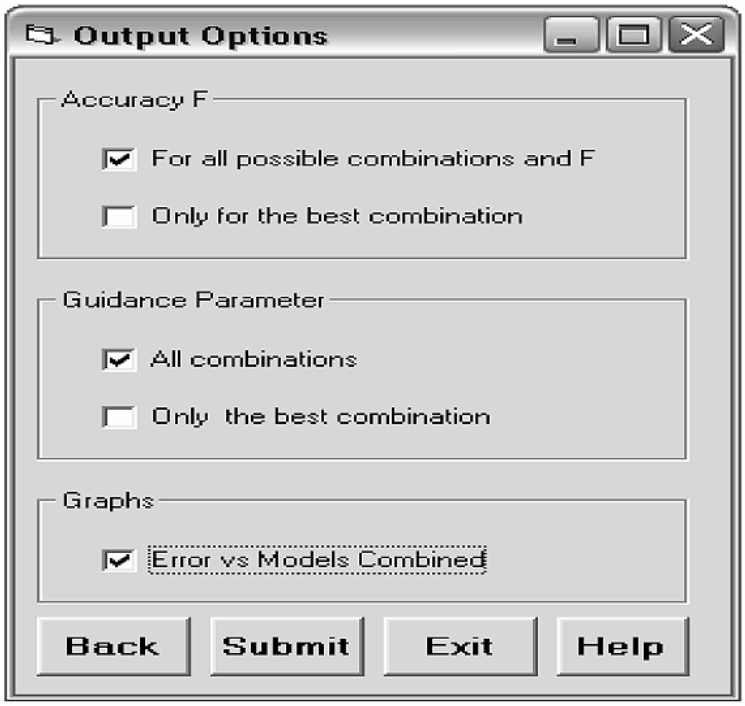

Fig. 2. Output options interface.

feature vector $f \in \Re^n$ and assigns it to a class label from Ω, i.e., $C_i : \Re^n \rightarrow \Omega$ or equivalently, $C_i(f) \in \Omega, i = 1...k$. The majority vote assigns f to the class label most represented among the classifier outputs. Let assume that the label outputs of the classifiers are given as m-dimensional binary vectors $[\Delta_{i,1}, \Delta_{i,2}, ..., \Delta_{i,m}] \in \{0,1\}, i = 1, ..., k$ where $\Delta_{i,j} = 1$ if C_i labels f in w_j, and 0, otherwise, $\sum_{j=1}^{m} \Delta_{i,j} = 1$. The plurality vote will pick class w_c if

$$\sum_{i=1}^{k} \Delta_{i,c} = \max_{j=1}^{m} \sum_{j=1}^{k} \Delta_{i,j} \tag{1}$$

If the classifiers perform differently on each class of the data, it is reasonable to try to provide the more competent classifiers in defining specific class of the data with more power in making the final decision for this specific class using the weighted majority vote.

We introduce the weights $(\alpha_{ij})_{i:=1..k,j:=1..m}$, where α_{ij} is the weight corresponding to the classifier C_i and the class j, and rewrite (1) as:
Choose class labelw_c if

$$\sum_{i=1}^{k} \alpha_{ic} \times \Delta_{i,c} = \max_{j=1}^{m} \sum_{j=1}^{k} \alpha'_{ij} \times \Delta_{i,j} \tag{2}$$

In our machine, we have formalized the weights as following: if we consider $(E_{ij})_{i:=1..k,j:=1..m}$ where E_{ij} is the harmony error corresponding to the classifier C_i and the class j, α'_{ij} are the normalized weights α_{ij} defined as:

$$\alpha_{ij} = -\log\left(\frac{E_{ij}}{1 - E_{ij}}\right) \tag{3}$$

4. Computational machine description

The main aim of this computational machine is to create an optimal hybrid model of classifiers for intrusion detection to segregate between attack and non-attack events. The optimal model for classification is data dependent [24]. The users have to specify the parameters for the computational machine; these parameters include the guidance parameter λ, the significance level α, the combination method and the number of classifiers to be considered. The optimality criteria considered is based on the harmonic error $E_\lambda = 1–F_\lambda$, where F_λ is Van Rijbergen's F-measure of accuracy [6], defined as a combination of both recall (R) and precision (P):

$$F_\lambda = \frac{1}{\lambda R^{-1} + (1 - \lambda) P^{-1}} \tag{4}$$

The user then uploads the training data and submits his preferences to obtain the optimal parsimonious hybrid model for his data. Figure 1 shows the parameter interface where the user enters all the required information. After then, the user uploads the data to be detected, at that time an output interface appears so the user can customize the output file. Figure 2 shows all the possible options.

During the whole process, the computational machine interacts with the user through user interfaces that have been meticulously designed and compounded with a detailed help explaining all steps and terms

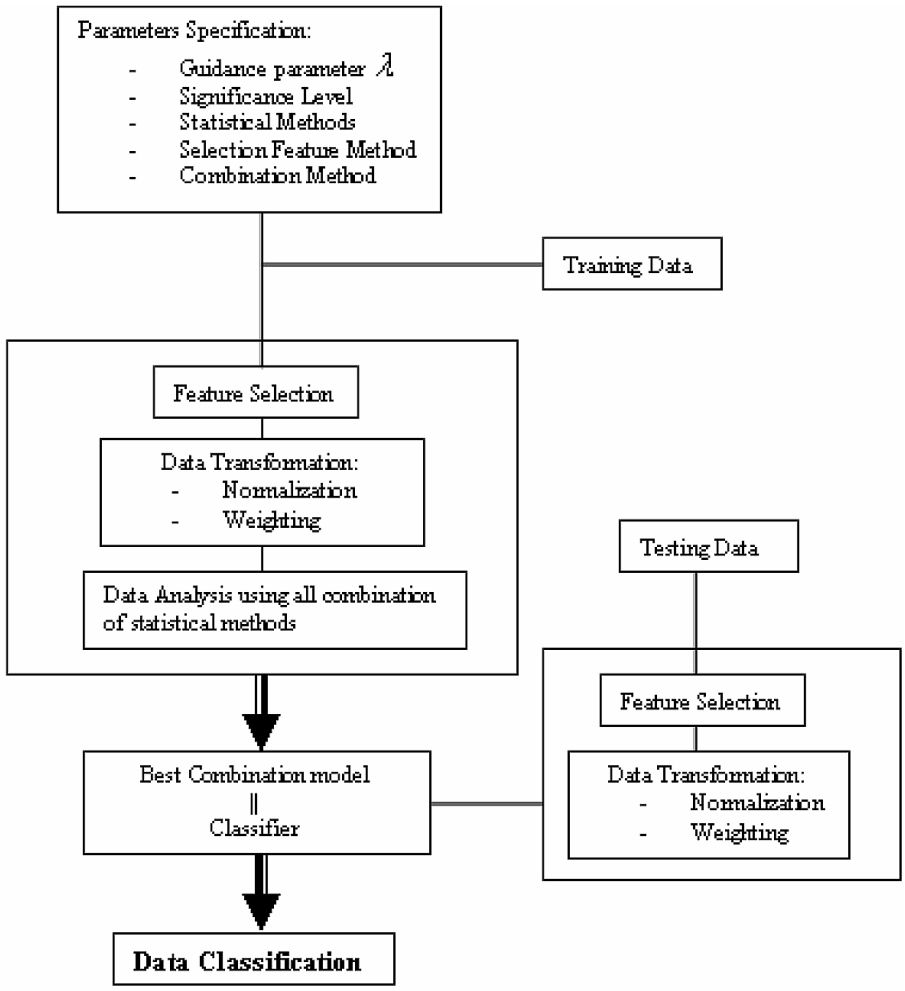

Fig. 3. Intrusion detection computational model.

in the computational machine. The statistical analysis is done using S-Plus and the output is generated as a text file [24].

The network intrusion detection computational machine can be formulated as shown in Fig. 3. It includes the different steps for training data: features selection, data transformation and finally data analysis in S-Plus software. The output is the best combination model that will be used to classify new data.

5. Experiments

We considered a set of KDD 1999 cup intrusion detection data [14] consisting of four classes: three major categories of attack: Denial-of-Service attacks (DOS: deny legitimate requests to the system), user-to-root attacks (U2R: unauthorized access to local administrator or root), and Probe, attacks generated by gathering information, and the normal class. We have trained three classifiers: Naïve Bayes (NB),

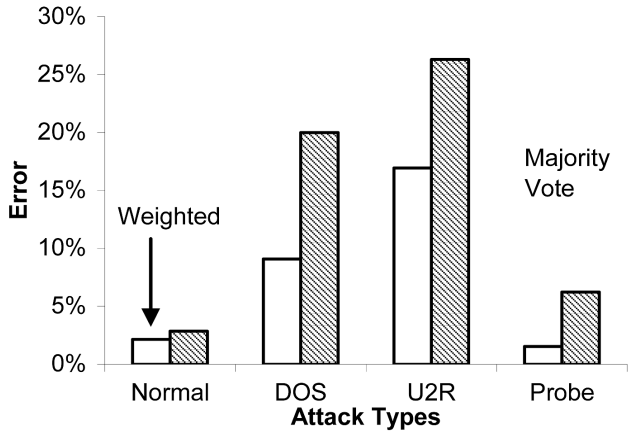

Fig. 4. Intrusion detection error for majority vote and weighted vote fusion methods.

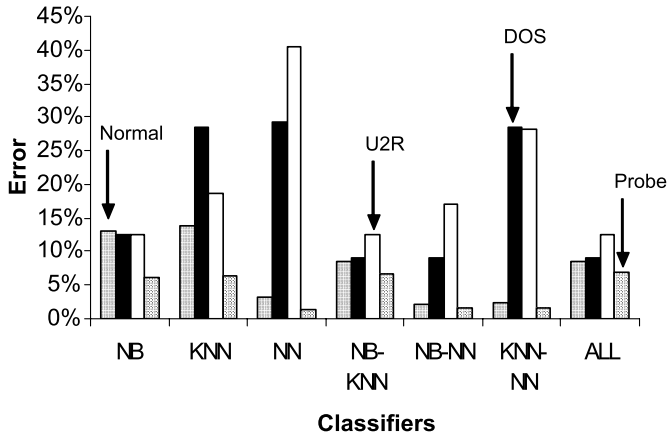

Fig. 5. Intrusion detection error for different data sets.

K-Nearest Neighbor (k-NN), and Neural Network (NN), then built a multi-classifier based on the methodology shown in Section 3.5. Figure 4 shows the performance comparison of the two methods of combining the classifiers, majority vote and weighted vote. We note that for the each class, the weighted average performed better than a simple majority vote. Figure 5 shows the performance of each classifier on each specific class. It identifies the best classifier for each attack category: combined classifier NB-NN for Normal, NB-KNN for DOS, combination of all classifiers for U2R, and NN for Probe.

6. Conclusion

In this paper, a computational machine is built to derive optimal parsimonious hybrid model of classifiers in intrusion detection based on the following classification methods, Naïve Bayes, Support Vector Machine, K-nearest neighbor, and Neural networks. The weighted voting fusion strategy for intrusion detection is assessed by experiments and its performances compared. The potentialities of classifiers fusion for the development of effective intrusion detection systems are evaluated and discussed.

The experimental results indicate that hybrid approach effectively generates a more accurate intrusion detection model on detecting both normal usages and malicious activities.

Acknowledgement

The author would like to thank Dr. Sanaa Kholfi, from School of Information Technology, George Mason University, Fairfax, VA, U.S.A, for her contribution and assistance. Furthermore the author is grateful for Dr. M. El-Telbany from Taif University, Taif University for reviewing the paper and his valuable comments.

References

[1] A.H. Sung and S. Mukkamala, The Feature Selection and Intrusion Detection Problems, *Proceedings of Advances in Computer Science – ASIAN 2004: Higher-Level Decision Making*, 9th Asian Computing Science Conference, **Vol. 3321** (2004), 468–482. rith: Select real-world application, *Journal of Information Fusion* **9** (2007), 4–20.
[2] Abraham and R. Jain, Soft Computing Models for Network Intrusion Detection Systems. Soft Computing in Knowledge Discovery: Methods and Applications, *Springer Chap* **16** (2004), 20.
[3] Abraham, C. Grosan and C.M. Vide, Evolutionary Design of Intrusion Detection Programs, *International Journal of Network Security* **4**(3) (2007), 328–339.
[4] C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 2004.
[5] D. Denning, An intrusion-detection model, *IEEE Transactions on Software Engineering* **13**(2) (1987 February), 222–232.
[6] F. Roli and J. Kittler, (eds), Multiple Classifier Systems, *Lecture Notes in Computer Science* **2364**, Springer–Verlag, 2002.
[7] G. Giorgio, P. Roberto, R.D. Mauro and R. Fabio, Intrusion detection in computer networks by a modular ensemble of one-class classifiers, *Journal of Information Fusion* **9** (2008), 69–82.
[8] H. Javitz and A. Valdes, The SRI IDES Statistical Anomaly Detector, *Proc IEEE Symposium on Security and Privacy*, Oakland, CA, 1991.
[9] H. Jin, J. Sun, H. Chen and Z. Han, A fuzzy data mining based intrusion detection model: Distributed computing systems. In *Proceedings of loth IEEE International Workshop on Future Trends*, May 2004, pages 191–197.
[10] J. Kittler, M. Hatef, R.P.W. Duin and J. Matas, On combining classifiers, *IEEE Trans on Pattern Analysis and Machine Intelligence* **20** (1998), 226–239.
[11] J. Li and C. Manikopoulos, Novel statistical network model: the hyperbolic distribution. In *IEE Proceedings on Communications* **151** (December 2004), 539–548.
[12] J. Ryan, M. Lin and R. Miikkulainen, Intrusion detection with neural networks, *Proceedings of the 1997 conference on Advances in neural information processing systems* **10** (July 1998), 943–949, Denver, Colorado, United States.
[13] K. Tumer and J. Ghosh, Analysis of decision boundaries in linearly combined neural classifiers, *Pattern Recognition* **29** (1996), 341–348.
[14] K. Tumer and J. Ghosh, Linear and order statistics combiners for pattern classification, in: *Combining Artificial Neural Nets*, Springer, London, 1999, pp. 127–155.
[15] KDD data set, 1999, http://kdd.ics.uci.edu/databases/-kddcup99/kddcup99.html.
[16] L. Kuncheva, A theoretical study on six classifier fusion strategies, *IEEE Trans on Pattern Analysis and Machine Intelligence* **24** (2002), 281–286.
[17] L. Lam and C.Y. Suen, Application of majority voting to pattern recognition: an analysis of its behavior and performance, *IEEE Trans. on Systems, Man and Cybernetics – Part A* **27** (1997), 553–568.
[18] N. Oza and K. Tumer, Classifier ensembles: Select real-world applications, *Journal of Information Fusion* **9** (2008), 4–20.
[19] Nahla, B. Salem and E. Zied, *Naive Bayes vs Decision Trees in Intrusion Detection Systems*, 2004.
[20] O. Dirk, S. Matzner, W. Stump and B. Hopkins, Applications of Hidden Markov Models to Detecting Multi-stage Network Attacks, Proceedings of the 36th Hawaii International Conference on System Science, 2003.
[21] P. Proctor, *The Practical Intrusion Detection Handbook*, Prentice Hall, 2002.
[22] S. Chebrolu, A. Abraham and J.P. Thomas, Feature Deduction and Ensemble Design of Intrusion Detection Systems, *International Journal of Computers and Security* **24**(4) (2005), 295–307.
[23] S. Haykin, *Neural Networks and Learning Machines*, New York: Prentice Hall, 2008.

[24] S. Kholfi, M. Habib and S. Aljahdali, Best Hybrid Classifiers for Intrusion Detection, *Journal of Computational Methods in Science and Engineering* (2006), 299–307.
[25] S. Mukkamala, A.H. Hung and A. Abraham, Intrusion Detection Using an Ensemble of Intelligent Paradigms, *Journal of Network and Computer Applications* **28** (2005), 167–182.
[26] S. Mukkamala, A.H. Sung and A. Abraham, *Modeling Intrusion Detection Systems Using Linear Genetic Programming Approach*, LNCS 3029, Springer Hiedelberg, 2004, pp. 633–642.
[27] S. Peddabachigari, A. Abraham, C. Grosan and J. Thomas, Modeling intrusion detection system using hybrid intelligent systems, *Journal of Network and Computer Applications* **30** (2007), 114–132.
[28] S. Stolfo, A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. Technical report, CUCS, 2002.
[29] S.T. Sarasamma, Q.A. Zhu and J. Huff, Hierarchical Kohonenen Net for Anomaly Detection in Network Security, *IEEE Transactions on Systems, Man and Cybernetics, Part B* **35**(2) (April 2005), 302–312.
[30] V. Jecheva, *About some Applications of Hidden Markov Model in Intrusion Detection Systems, International Conference on Computer Systems and Technologies*, 2006.
[31] W. Fan and S. Stolfo, Ensemble-Based Adaptive Intrusion Detection, *Proceedings SIAM Int Conference on Data Mining*, Arlington, VA, 2002.
[32] W. Lee and S.J. Stolfo, A framework for constructing features and models for intrusion detection systems, *ACM Transactions on Information and System Security* **3**(4) (November 2000), 227–261.
[33] W. Lee, S. Stolfo, P. Chan, E. Eskin, W. Fan, M. Miller, S. Hershkop and J. Zhang, Real Time Data Mining Based Intrusion Detection, *Proceedings of DISCEX* **11**, 2001.
[34] Y. Liao arid V. Rao Vemuri, Use of K-nearest neighbor classifier for intrusion detection, *Computers & security* **21**(5) (2002), 439–448.