World Applied Sciences Journal 7 (Special Issue of Computer & IT): 80-86, 2009
ISSN 1818-4952
© IDOSI Publications, 2009

Combination of Elman Neural Network and Classification-Based Predictive Association Rules to Improve Computer Networks' Security

[1]*Mansour Sheikhan and* [2]*Davood Gharavian*

[1]Electrical Engineering Department, Islamic Azad University, South Tehran Branch, Tehran, Iran
[2]Electrical Engineering Department, Shahid AbbaspourUniversity, Tehran, Iran

Abstract: To have a higher correct classification rate in pattern recognition problems, several machine learning-based schemes have been proposed. In this way, an approach called associative classification was proposed in the last decade to integrate association rule mining and classification. Currently, constructing new hybrid intelligent knowledge processing systems is a hot research topic. So, in this paper a new algorithm is reported that combines a classification-based association rule approach with an Elman neural network, as a hybrid attack recognizer system for improving the security of computer networks. This combination provides improved classification results as compared to each of its constituent algorithms and also many of the previous systems. In addition, a feature relevance analysis is used to decrease the number of applied features to association rule section of the system. Experimental results show that the proposed hybrid model improves classification results, especially for remote-to-local (R2L) and user-to-root (U2R) attack categories. The proposed method is effective in terms of detection rate (DR) and false alarm rate (FAR) as compared to other machine learning methods, as well.

Key words: Elman neural network · Classification · Predictive association rules · Network security

INTRODUCTION

With the aim of simulating the operation of the human brain, neural networks (NNs) have been adopted in pattern recognition problems such as classification, mainly because of their flexibility and adaptability to environmental changes. However, the defects of NNs are that its computational load is very heavy and it is difficult to interpret the relationship between inputs and outputs.

To have a higher correct classification rate (CCR), several other machine learning-based schemes have been proposed, e.g. Bayesian networks [1], Markov models [2, 3], fuzzy logic techniques [4] and genetic algorithms [5].

On the other hand, an approach called associative classification was proposed in the last decade [6] to integrate association rule mining [7] and classification. It used association rule mining algorithm, such as Apriori [7] or PFgrowth [8] to generate the complete set of association rules.

Currently, constructing new hybrid intelligent knowledge processing systems is a hot research topic. So, in this paper a new algorithm is reported that combines a classification based on predictive association rule (CPAR) approach with an Elman NN [9], as a hybrid attack recognizer system for improving the security of computer networks. This combination provides improved classification results as compared to each of its constituent algorithms and also many of the previous systems. In addition, a feature relevance analysis is used to decrease the number of applied features to association rule section of the system.

The remainder of this paper is organized as follows. Section 2 reviews the architecture of Elman NN. The foundation of CPAR approach is described in Section 3. The details of training and test data, which are used in developing attack recognizer, are presented in Section 4. The feature selection and preprocessing procedures are reported in Section 5. The architecture of the hybrid attack recognizer system is proposed in Section 6. Experimental results and conclusions are also drawn in Sections 7 and 8, respectively.

Corresponding Author: Dr. Mansour Sheikhan, P.O. Box 11365/4435, Post-Graduate Center, South Tehran Branch, Islamic Azad University, Iran

ELMAN NEURAL NETWORK

Unlike feedforward NNs, the recurrent architecture permits to include context information about past data in form of previous hidden layer states in the recurrent connections [9]. Thus, feature trajectories can be modeled in a better way. In Figure 1, the architecture of an Elman NN is shown in which $S_i(t)$ indicates the states of the input layer, $S_h(t)$ the states of the hidden layer and $S_o(t)$ the states of the output layer.

CLASSIFICATION BASED ON PREDICTIVE ASSOCIATION RULES

In this paper, an approach called CPAR [10] is used as the rule-mining part of the proposed hybrid system. CPAR inherits the basic idea of first order inductive learner (FOIL) [11] in rule generation and integrates the features of associative classification in predictive rule analysis.

In this way, let T be a set of tuples. Each tuple in T follows the scheme $(A_1, A_2, \ldots, A_k)$, where $A_1, A_2, \ldots, A_k$ are k attributes. Each continuous attribute is first turned discrete into a categorical attribute.

A literal p is an attribute-value pair, taking the form of (A_i, v), in which A_i is an attribute and v a value. A tuple t satisfies a literal $p=(A_i, v)$ if and only if $t_i=v$, where t_i is the value of the i^{th} attribute of t.

A rule, which takes the form of "$p_1 \wedge p_2 \wedge \ldots \wedge p_l \rightarrow c$", consists of a conjunction of literals $p_1, p_2, \ldots, p_l$, associated with a class label c. A tuple t satisfies rule r's body if and only if it satisfies every literal in the rule. If t satisfies r's body, r predicts that t is of class c.

A *PNArray* stores the following information corresponding to rule r, in which P and N are the numbers of positive and negative examples satisfying r's body. $P(p)$ and $N(p)$ for each possible literal p, are the numbers of positive and negative examples satisfying the body of rule r', the rule constructed by applying p to r, as well. After literal p is added to r, there are P^* positive and N^* negative examples satisfying the new rule's body.

By using *PNArray*, the predictive rule mining (PRM) algorithm (Figure 2) selects only the best literal and ignores all the others. But, CPAR builds rules by adding literals one by one, which is similar to PRM. However, instead of ignoring all literals except the best one, CPAR keeps all close-to-the-best literals during the rule building process. So, at a certain step in the process of building a rule, after finding the best literal p, another literal q that has similar gain (Equation 1) as p may be found:

$$gain(p) = |P^*| \left(\log \frac{|P^*|}{|P^*| + |N^*|} - \log \frac{|P|}{|P| + |N|} \right) \quad (1)$$

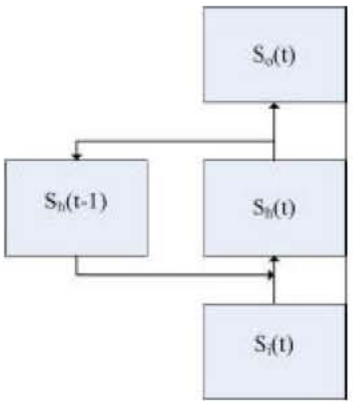

Fig. 1: Architecture of an Elman neural network

```
Input: Training set $D = P \cup N$.
Output: A set of rules for predicting class labels for examples.
Procedure Predictive Rule Mining
  Set the weight of every example to 1
  Rule set $R \leftarrow \Phi$
  $total\ Weight \leftarrow TotalWeight(P)$
  $A \leftarrow$ Compute PNArray from D
  While $TotalWeight(P) > \delta.totalWeight$
    $N' \leftarrow N$, $P' \leftarrow P$, $A' \leftarrow A$
    Rule $r \leftarrow emptyrule$
    While true
      find best literal $p$ according to $A'$
      if gain($p$) < $min\_gain$ then break
      append $p$ to $r$
      for each example $t$ in $P' \cup N'$ not satisfying $r$'s body
                remove t from $P'$ or $N'$
                change $A'$ according to the removal of $t$
      end
    end
    $R \leftarrow R \cup \{r\}$
    for each example $t$ in $P$ satisfying $r$'s body
      $t.weight \leftarrow a\ .\ t.weight$
      change $A$ according to the weight decreased
    end
  end
  return $R$
```

Fig. 2: Predictive rule mining algorithm [10]

Besides continuing building the rule by appending p to r, q is also appended to the current rule r to create a new rule r', which is pushed into the queue [10].

To evaluate the prediction power of rule r (r="$p_1 \wedge p_2 \wedge \ldots \wedge p_l \rightarrow c$"), the Laplace expected error estimate is used to estimate the accuracy of rules [10]:

$$LaplaceAccuracy = (n_c+1)/(n_{tot}+k) \quad (2)$$

where k is the number of classes, n_{tot} is the total number of examples satisfying the rule's body, among which n_c examples belong to c, the predicted class of the rule.

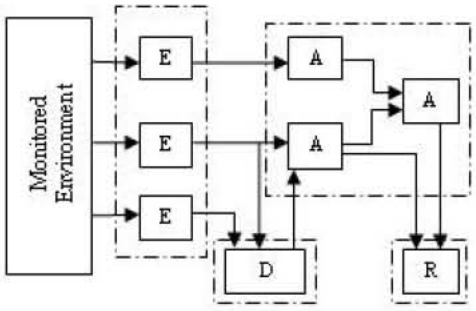

Fig. 3: General architecture for intrusion detection systems

Given a rule set containing rules for each class, the best K rules of each class is used and by comparing the *LaplaceAccuracy* of the best K rules of each class, the class with the highest accuracy is chosen as the predicted class.

TRAINING AND TEST DATASETS

Intrusion detection system (IDS) plays a strategic role in security of computer networks. So, to evaluate the performance of the proposed hybrid knowledge processing system, IDS is selected as the case study in this paper. Intrusion detection working group (IDWG) has defined a general IDS architecture (Figure 3). This architecture is based on four functional modules: event (E), database (D), analysis (A) and response (R) [12].

The event blocks acquire information events to be analyzed by other blocks. The database blocks store information from E blocks for subsequent processing by A and R boxes. The analysis blocks are processing modules for analyzing events and detecting potential hostile behavior. The response block executes a response, if any intrusion occurs.

In 1999, recorded network traffic from the DARPA 98 Lincoln lab dataset [13] was summarized into network connections with 41 features per connection. This formed the benchmark provided by the international knowledge discovery and data mining group (KDD) [14]. The descriptions of 11 sample features are reported in Table 1.

There are four main categories of attacks given in the KDD. They are denial-of-service (DoS), probe, user-to-root (U2R) and remote-to-local (R2L) [15]. Each of these categories of attacks represents generalizations of specific attack types.

The KDD intrusion detection benchmark consists of three components: "Whole KDD", "Corrected KDD" and "10% KDD". Table 2 lists the attack categories along with the attack types and number of samples in the "10% KDD" dataset. As indicated in Table 2, "10% KDD" dataset contains 22 attack types. "Corrected KDD" dataset provides a dataset with different statistical distributions than either "10% KDD" or "Whole KDD" and contains 14 additional attacks.

Table 1: Description of 11 sample features in KDD dataset

Feature	Description
duration	Duration of the connection (in seconds)
protocol_type	Type of the connection protocol
src_bytes	Number of bytes sent from source to destination
dst_bytes	Number of bytes sent from destination to source
land	1 if connection is from/to the same host/port; 0 otherwise
num_root	Number of "root" accesses
num_file_creations	Number of file creation operations
num_access_files	Number of operations on access control files
same_srv_rate	Percent of connections to the same service
dst_host_count	Number of connections having the same destination host
dst_host_diff_srv_rate	Percent of different services on the current host

Table 2: Attack types and number of their samples in 10% KDD dataset

Category	Type (Number of samples)
DoS	smurf (280790), neptune (107201), back (2203), teardrop (979), pod (264), land (21)
Probe	satan (1589), ipsweep (1247), portsweep (1040), nmap (231)
R2L	warezclient (1020), guess_passwd (53), warezmaster (20), imap (12), ftp_write (8), multihop (7), phf (4), spy (2)
U2R	buffer_overflow (30), rootkit (10), loadmodule (9), perl (3)

Table 3: Distribution of training and test samples

Class	Number of training samples	Number of test samples
DoS	15900	11075
Probe	3277	201
R2L	1124	780
U2R	47	21
Normal	3867	2923

39215 records from KDD 99 dataset were chosen for our experiments. 24215 of them from "10% KDD" built the training set and the rest (15000 samples from "Corrected KDD") built the test set (Table 3).

PREPROCESSING AND SELECTION OF FEATURES

Preprocessing: Features in the KDD datasets have different forms: discrete, continuous and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence, preprocessing is required. First, we describe the preprocessing of 41 input features to Elman NN. Then, the preprocessing details of selected features, input to CPAR, are reported.

Symbolic–valued features, such as *protocol_type* (3 different symbols), *service* (70 different symbols) and *flag* (11 different symbols) are mapped to integer values ranging from *0* to *S-1*, where *S* is the number of symbols. Continuous features having smaller integer value ranges like *wrong_fragment:* [0,3], *urgent:* [0,14], *hot:* [0,101], *num_failed_logins:* [0,5], *num_compromised:* [0,9], *num_root:* [0,7468], *num_file_creations:* [0,100], *num_shells:* [0,5], *num_access files:* [0,9], *count:* [0,511], *srv_count:* [0,511], *dst_host_count:* [0,255], *dst_host_srv_count:* [0,255] are also scaled linearly to the range [0,1].

Logarithmic scaling (base 10) is applied to three features spanned over a very large integer range, namely *duration:* [0,58329], *src_bytes:* [0,1.3billion] and *dst_bytes:* [0,1.3billion], to reduce the ranges to [0,4.77] and [0,9.11], respectively. Other features are either Boolean, like *logged_in*, having binary values, or continuous, like *diff_srv_rate*, in the range of [0,1] and no scaling is needed for these features. So, each of the mapped features are linearly scaled to the range [0,1].

As seen in CPAR foundations, the usage of association rules requires a discrete set of items. These items are literal values of the dataset attributes. In order to use association rules on continuous attributes, the attributes must first be turned discrete. So, for the selected features based on the procedure that is subscribed in the next subsection, the continuous ones are partitioned into equal-sized partitions by utilizing equal frequency intervals [16]. In equal frequency intervals method, the feature space is partitioned into an arbitrary number of partitions where each partition contains the same number of data points. In other words, the range of each partition is adjusted to contain *I* dataset instances. If a value occurs more than *I* times in a feature space, it is assigned a partition of its own. In "10% KDD" dataset, certain classes such as DoS attacks and normal connections occur in order of hundreds or thousands, whereas other classes such as U2R and R2L attacks occur in order of tens or hundreds. Therefore, to provide sufficient resolution for the minor classes, it is assumed that *I*=10.

Feature Selection: Feature ranking and selection is an important issue in intrusion detection. Elimination of less significant features lowers the load of CPAR, enhancing the attack recognition rate and speeding up the computation, thus improving the overall performance of IDS.

Table 4: Chi-square values of input features with respect to the attack class

	Attack type			
Feature	DoS	Probe	R2L	U2R
dst_host_diff_srv_rate	1334.82	3686.28	1114.09	2531.96
rerror_rate	1016.26	2734.53	1016.54	613.39
dst_host_srv_rerror_rate	967.87	2707.66	586.24	301.06
srv_rerror_rate	805.55	2515.68	583.34	244.92
dst_ host_rerror_rate	732.80	2251.96	560.59	207.83
diff_srv_rate	551.75	1228.26	350.12	39.88
dst_host_same_srv_rate	449.23	793.25	311.15	39.16
service	438.75	588.68	249.51	36.74
dst_host_srv_count	433.03	546.12	23916	32.61
logged_in	363.64	427.17	141.75	25.15
dst_host_srv_diff_host_rate	353.49	422.29	141.31	25.01
srv_count	344.88	123.40	141.22	15.46
same_srv_rate	336.85	91.77	126.07	15.33
protocol_type	328.69	84.58	125.02	10.68
num_compromised	308.37	70.40	116.02	10.26
wrong_fragment	275.58	68.62	99.83	6.35

Logistic regression has been used to rank the features based on the Chi-square values for different subsets selected using best subset selection model [17]. The higher the Chi-square value, the higher is the ranking. In Table 4 the ranking results of the Chi-square test on KDD dataset are listed for the 16 most significant features which are used as the inputs to CPAR algorithm, after preprocessing.

HYBRID STRUCTURE FOR ATTACK RECOGNITION

Benefits of the CPAR algorithm can be combined with those of the Elman NN. The key idea is to take advantage of different abilities of neural networks and knowledge-based algorithms [18] for intrusion detection. Performance evaluation of different neural networks to classify the attacks in the KDD test set [19] shows that the correct classification rate (CCR) of Elman NN is comparable with the best neural model (differs by at most 0.8%), but consumes the lower learning time in comparison to other NNs.

So, the Elman NN is selected as NN module of the proposed IDS in this paper. The block diagram of proposed IDS is shown in Figure 4. The more detailed structure of the Elman NN/CPAR hybrid is also depicted in Figure 5.

EMPIRICAL RESULTS

In this study, three experiments were performed based on the structure shown in Figure 4: using only the upper branch (Elman NN classifier), using only the

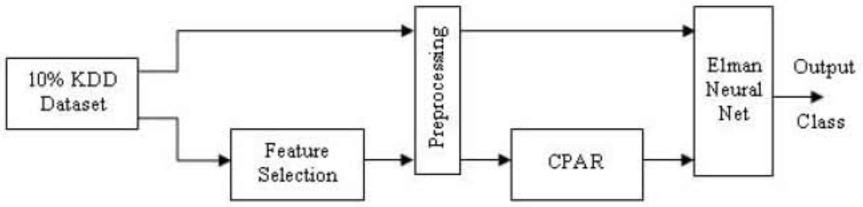

Fig. 4: Block diagram of the proposed hybrid model for IDS

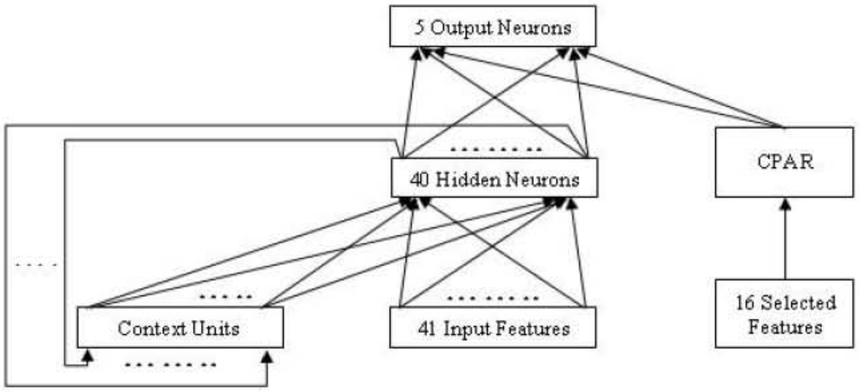

Fig. 5: Structure of the hybrid Elman NN/CPAR classifier

lower branch (CPAR with selected input features) and using the full structure (hybrid Elman NN/CPAR classifier).

Before discussing about the results of experiments, it seems necessary to mention the standard metrics that have been developed for evaluating IDS. Detection rate (DR) and false alarm rate (FAR) are the two most common metrics for attack recognition in IDS. DR is computed as the ratio between the number of correctly detected attacks and the total number of attacks, while FAR is computed as the ratio between the number of normal connections that is incorrectly misclassified as attacks and the total number of normal connections.

In the first experiment, an Elman NN with a hidden layer of 40 neurons with tangent-sigmoid activation function, five linear output neurons (representing 4 attack categories and 1 normal category) and 41 external input nodes, is used. Levenberg-Marquardt (LM) is used as the training function in this experiment, utilizing MATLAB NN Toolbox. The confusion matrix of this classification method is shown in Table 5.

In the second experiment, CPAR is used for classification with selected features as input. The parameters of CPAR in rule generation algorithm are set as: δ=0.05, *min_gain*=0.7 and α=2/3. The best four rules are used in prediction, too. The confusion matrix of CPAR classification is reported in Table 6.

Table 5: Confusion matrix of Elman NN classifier

	Predicted				
Actual	DoS	Probe	R2L	U2R	Normal
DoS	10628	7	4	0	436
Probe	19	152	2	0	28
R2L	0	3	91	0	686
U2R	0	1	7	0	13
Normal	37	17	16	0	2853

Table 6: Confusion matrix of CPAR classifier

	Predicted				
Actual	DoS	Probe	R2L	U2R	Normal
DoS	10666	8	47	0	354
Probe	50	105	17	0	29
R2L	1	2	34	0	743
U2R	0	4	1	2	14
Normal	9	13	1	0	2900

Table 7: Confusion matrix of Elman NN/CPAR hybrid classifier

	Predicted				
Actual	DoS	Probe	R2L	U2R	Normal
DoS	10741	14	7	42	271
Probe	10	184	1	1	5
R2L	31	7	248	5	489
U2R	4	6	2	7	2
Normal	31	18	10	18	2846

Table 8: Correct classification rate of different models for intrusion detection

	Correct classification rate (%)				
Model	DoS	Probe	R2L	U2R	Normal
Pnrule [20]	96.9	73.2	10.7	6.6	99.5
Multi-classifier [21]	97.3	88.7	9.6	29.8	NR[a]
ESC-IDS [22]	99.5	84.1	31.5	14.1	98.2
Elman NN	96.0	75.6	11.7	0.0	97.6
CPAR	96.3	52.2	4.4	9.5	99.2
Hybrid Elman NN/CPAR	97.0	91.5	31.8	33.3	97.4

[a] NR stands for "Not Reported".

Table 9: Some reported results on KDD dataset [24, 25] and the proposed model for intrusion detection

Metric	Data mining	Clustering	k-NN[a]	SVM[b]	SOM[c]	Fuzzy AR[d]	Elman /CPAR
DR (%)	80	93	91	98	91	91	92.6
FAR (%)	2	10	8	10	11	3.3	2.6

[a] k-Nearest Neighbor [b] Support Vector Machine
[c] Self Organizing Map [d] Association Rules

In the third experiment, the combined structure of Elman NN/CPAR, depicted in Figure 5, is used for classification. The same assumptions for Elman NN architecture and training function and also CPAR parameters are considered in this experiment. The confusion matrix of this hybrid classifier is shown in Table 7.

By applying 15000 test samples (Table 3), the performance of Elman NN, CPAR and combined Elman NN/CPAR classifiers has been compared with some other machine learning methods tested on the KDD dataset (Table 8).

CONCLUSIONS

In this paper, a hybrid structure was introduced for attack recognizer in computer networks that combined an Elman neural network with classification based on association rule approach. Most of the machine learning algorithms offered an acceptable level of CCR for DoS and Probe attack categories and demonstrated poor performance on the R2L and U2R categories [23]. As shown in Table 8, the proposed hybrid Elman NN/CPAR classifier demonstrates better performance in mentioned categories. This combination improved classification results, especially for R2L and U2R attack categories, as compared to each of its constituent modules (Elman NN and CPAR) and also many of previous systems.

In this way, the DR and FAR of some reported results on KDD dataset and the proposed hybrid model in this paper, are shown in Table 9, as well. Experimental results showed that the proposed method is effective in terms of DR and FAR as compared to other machine learning methods.

REFERENCES

1. Heckerman, D., 1995. A Tutorial on Learning with Bayesian Networks. Microsoft Research Technical Report MSRTR-95-06.
2. Yeung, D.Y. and Y. Ding, 2003. Host-Based Intrusion Detection Using Dynamic and Static Behavioral Models. Pattern Recognition, 36: 229-243.
3. Esteves-Tapiador, J.M., P. Garcia-Teodoro and J.E. Diaz-Verdejo, 2005. Detection of Web-Based Attacks through Markovian Protocol Parsing. In the Proceedings of the IEEE Symposium on Computers and Communications, pp: 457-462.
4. Bridges, S.M. and R.B. Vaughn, 2000. Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection. In the proceedings of the National Information Systems Security Conference, pp: 13-31.
5. Lin, Y., K. Chen and X. Liao, 2004. A Genetic Clustering Method for Intrusion Detection. Pattern Recognition, 37: 924-927.
6. Liu, B., W. Hsu and Y. Ma, 1998. Integrating Classification and Association Rule Mining. In the Proceedings of KDD'98, pp: 80-86.
7. Agrawal, R. and R. Srikant, 1994. Fast Algorithms for Mining Association Rules. In the Proceedings of the International Conference on Very Large Databases, pp: 487-499.
8. Han, J., J. Pei and Y. Yin, 2000. Mining Frequent Patterns without Candidate Generation. In the Proceedings of the ACM International Conference on Management of Data, pp: 1-12.
9. Elman, J.L., 1990. Finding Structure in Time. Cognitive Science, 14: 179-211.
10. Yin, X. and J. Han, 2003. CPAR: Classification Based on Predictive Association Rules. In the Proceedings of SIAM International Conference on Data Mining, pp: 331-335.
11. Quinlan, J.R. and R.M. Cameron-Jones, 1993. FOIL: A Midterm Report. In the Proceedings of the European Conference on Machine Learning, pp: 3-20.
12. Garcia-Teodoro, P., J. Diaz-Verdejo, G. Macia-Fernandez and E. Vazquez, 2009. Anomaly-Base Network Intrusion Detection: Techniques, Systems and Challenges. Computers & Security, 28: 18-28.

13. The 1998 intrusion detection off-line evaluation plan, MIT Lincoln lab., Information Systems Technology Group, 25 Mar. 1998 (Available on http://www.11.mit.edu/IST/ideval/docs/1998/id98-eval-11.txt).
14. 1999 KDD Cup Competition (Available on http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html).
15. McHugh, J., 2001. Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. ACM Transactions on Information and System Security, 3: 262-294.
16. Wang, A.K.C. and D.K.Y. Chiu, 1987. Synthesizing Statistical Knowledge from Incomplete Mixed-Mode Data. IEEE Transactions on Pattern Analysis and Machine Intelligence, 9: 796-805.
17. Tamilarasan, A., S. Mukkamala, A.H. Sung and K. Yendrapalli, 2006. Feature Ranking and Selection for Intrusion Detection Using Artificial Neural Networks and Statistical Methods. In the Proceedings of the International Joint Conference on Neural Networks, pp: 4754-4761.
18. Sanchez, D., M.A. Vila, L. Cerda and J.M. Serrano, 2009. Association Rules Applied to Credit Card Fraud Detection. Expert Systems with Applications, 36: 3630-3640.
19. Beghdad, R., 2007. Training All the KDD Data Set to Classify and Detect Attacks. Journal of Neural Network World, 17: 81-91.
20. Agrawal, R. and M.V. Joshi, 2000. PNrule: A New Framework for Learning Classifier Models in Data Mining (A Case-Study in Network Intrusion Detection). IBM Research Division, Report No. RC-21719.
21. Sabhnani, M.R. and G. Serpen, 2003. Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context. In the Proceedings of the International Conference on Machine Learning: Models, Technologies and Applications, pp: 209-215.
22. Nadjaran Toosi, A. and M. Kahani, 2007. A Novel Soft Computing Model Using Adaptive Neuro-Fuzzy Inference System for Intrusion Detection. In the Proceedings of the IEEE International Conference on Networking, Sensing and Control, pp: 834-839.
23. Sabhnani, M. and G. Serpen, 2004. Why Machine Learning Algorithms Fail in Misuse Detection on KDD Intrusion Detection Data Set. Intelligent Data Analysis, 6: 1-13.
24. Han, S.J. and S.B. Cho, 2003. Detecting intrusion with rule-based integration of multiple models. Computers & Security, 22: 613-623.
25. Tajbakhsh, A., M. Rahmati and A. Mirzaei, 2009. Intrusion Detection Using Fuzzy Association Rules. Applied Soft Computing, 9: 462-469.