International Journal of Applied Engineering Research ISSN 0973-4562 Volume 11, Number 6 (2016) pp 4161-4166
© Research India Publications. http://www.ripublication.com

An Ensemble Data Preprocessing Approach for Intrusion Detection System Using variant Firefly and Bk-NN Techniques

Mrs. D. Shona*
Research Scholar, Bharathiar University,
Assistant Professor, Department of Computer Science,
Sri Krishna Arts and Science, Coimbatore, India.

Dr. M. Senthilkumar
Research Supervisor, Bharathiar University
Assistant Professor, Department of Computer Science,
Government Arts College, Ooty, India.

Abstract
The Hasty intensification of Internet communication and accessibility of systems to infringe the network, network security has become requisite. This paper focuses on development of efficient IDS in MANET. The KDD cup 99 dataset is considered for this proposal. Before performing any detection mechanism the dataset has to be improvised to overcome two important factors namely, redundant removal and handling missing values. Redundancy in dataset causes learning algorithms to be biased towards frequent records and unbiased towards infrequent records. Using dataset with incomplete data leads to false classification due to poor input inference. In this proposed work weighted minkowski based Firefly algorithm is applied to eliminate redundant record set and enhanced KNN based imputation method with the help of bagging technique to handle missing value is introduced. The experimental results shows that after preprocessing there is more improvement in the accuracy of learning algorithm during classification of normal and abnormal packets.

Keywords: MANET, KDD, redundancy, missing value, weighted minkowski, Firefly, k-NN, bagging

Introduction

Intrusion Detection System (IDS) is the science of finding of malevolent action on a computer network. Intrusions are defined as attempts to compromise the confidentiality, integrity or availability of computer or network. Due to the massive amount of dataset already existing and newly appearing network data is on progress the need for Data mining in Intrusion Detection System becomes essential. Often network data suffers from missing values. The presence of missing values is due to various reasons, such as manual data entry procedures, equipment errors and incorrect measurements. It is usual to find missing data in most of the information sources used. Missing values usually appears as “NULL” values in database or as empty cells in spreadsheet table. However missing values can also appears as outliers or wrong data. These data must be removed before intended analysis, and are much harder to find out.

Several machine learning techniques are used in acquiring information about the intrusion detection. This section discuss about recent existing works like ANN, k-NN, Fuzzy logic, Neutrosophic Logic, negative selection and support vector machines which helps in detecting intrusions.

Tsai C et al. [1] had proposed a hybrid learning model based on The Triangle Area Based Nearest Neighbors (TANN) in order to detect attacks. Result shows that TANN can efficiently notice intrusion attacks. Compared to three base-line models based on support vector machines, k-NN, and the hybrid centroid based classification this proposed work produces more accuracy.

Kavitha et al. [2] had anticipated an intrusion detection system using Neutrosophic Logic classifier which is an extension/combination of the fuzzy logic, intuitionistic logic, paraconsistent logic, and the three-valued logics that use an indeterminate value. The false alarm rate and the undetected attack rates are the two factors that define the cost function of proposed intrusion detection system. Proposed method gave the best result on KDD Cup data set.

An efficient negative selection algorithm with further training for anomaly detection was deployed by Gong et al. [3]. The experimental comparison among the proposed algorithm, the self-detector classification, and the Vdetector on seven artificial and real-world data sets show that the proposed algorithm can get the highest detection rate and the lowest false alarm rate in most cases.

Pastrana S et al. [4] had presented a comparison of the effectiveness of six different classifiers to detect malicious activities in MANETs. Results show that genetic programming and support vector machines may help in detecting malicious activities in MANETs.

Pereira C R et al. [5] had introduced an Optimum Path Forest framework for intrusion detection in computer network. The experiments have been carried out on three datasets aiming to compare OPF against Support Vector Machines, Self Organizing Maps and a Bayesian classifier. Results show that the OPF is the fastest classifier and always with better results.

Decision tree based light weight intrusion detection using a wrapper approach for anomalies detection in network was presented by Sindhu et al. [6]. This proposed method has evaluated using detection percentage and error percentage.

Kang I et al. [7] had devised a new one class classification method with differentiated anomalies to enhance intrusion detection performance for harmful attacks. They also proposed new extracted features for host based intrusion detection based on three viewpoints of system activity such as dimension, structure, and contents.
Mabu S et al. [8] implemented an intrusion detection model based on Fuzzy Class association rule mining using genetic network programming. Experimental results on the proposed method using KDD Cup and DARPA98 databases shows that it provides competitively high detection rates compared with other machine learning techniques.
Hussein S M et al. [9] had introduced hybrid IDS by integrated signature based (Snort) with anomaly based (Naive Bayes) to enhance system security to detect attacks. Accuracy, detection rate, time to build model and false alarm rate were used as parameters to evaluate performance between hybrid Snort with Naïve Bayes, Snort with J48graft and Snort with Bayes Net.
The literature study reveals that there is a very few works are done on both redundancy elimination and missing value handling in KDD cup 99 dataset hence this paper focuses on both these problems effectively.
This proposal aims to filter out redundant information and handling missing value which will significantly reduce number of computer resources, both memory and CPU time required to detect attack.

Related Work

Dataset Description

The KDD 99 intrusion detection datasets are based on the 1998 DARPA [15] initiative, which provides designers of intrusion detection systems (IDS) with a benchmark on which to evaluate different methodologies [MIT. L. L 98]. To do so, a simulation is made of a factitious military network consisting of three ‘target’ machines running various operating systems and services. Additional three machines are then used to spoof different IP addresses to generate traffic. Finally, there is a sniffer that records all network traffic using the TCP dump format. The total simulated period is seven weeks. Normal connections are created to profile that expected in a military network and attacks fall into one of four categories:

- Denial of Service (DoS): Attacker tries to prevent legitimate users from using a service.
- Remote to Local (R2L): Attacker does not have an account on the victim machine, hence tries to gain access.
- User to Root (U2R): Attacker has local access to the victim machine and tries to gain super user privileges.
- Probe: Attacker tries to gain information about the target host.

Table 1 shows various attack types in our experimental dataset which falls under the four categories.

Table 1: Various Attack types

Categories	Attack Types
DOS	Apache2, Back, Land, Mail bomb, Neptune, Pod, process Table, Smurf, Tear drop, Udpstrom.
PROBE	IPsweep, Mscan, nMap, Portsweep, Saint, Satan.
U2R	Buffer Overflow, http tunnel, load module, perl, root kit, ps, sqlattack, xterm
R2L	Ftpwrite, guesspasswd, imap, multihop, named, phf, send mail, snmp getattack, snmpguess, warezmaster, worm, xlock, xsnoop.

The KDDCup’99 Intrusion Detection benchmark consists of 3 components which are detailed in Table 2. Only “10% KDD” dataset is employed for the purpose of training. This dataset contains 22 attack types and is a more concise version of the “Whole KDD” dataset. Because of their nature, denial of service attacks account for the majority of the dataset.

Table 2: Intrusion Detection benchmark

Data set	DoS	Probe	U2r	R2l	Normal
“10% KDD”	391458	4107	52	1126	97277
“KDD Corrected”	229853	4166	70	16347	60593
“Whole KDD”	3883370	41102	52	1126	972780

On the other hand the “Corrected KDD” dataset provides a dataset with different statistical distributions than either “10% KDD” or “Whole KDD” and contains 14 additional attacks.

Firefly Algorithm

- The Firefly Algorithm (FA) is a Meta heuristic algorithm, inspired by the flashing behavior of fireflies [10]. The brightness of a firefly is affected or determined by the landscape of the objective function to be optimized [11], [12]. The primary purpose for a firefly's flash is to act as a signal system to attract other fireflies and find the duplicate records based on the flashing behavior of the each firefly. Xin-She Yang[10] formulated this firefly algorithm by assuming:
- All fireflies are unisexual, so that one firefly will be attracted to all other fireflies;
- Attractiveness is proportional to their brightness, and for any two fireflies, the less bright one will be attracted by (and thus move to) the brighter one; however, the brightness can decrease as their distance increases;
- If there are no fireflies brighter than a given firefly it will move randomly

Advantages

- It is easy to implement and there are few parameters to adjust.
- Compared with GA, all the fireflies tend to converge to the best solution quickly even in the local version in most cases

Pseudo Code for Firefly

```
1) Objective function: f(x), x= ($x_1, x_2 \ldots x_d$);
2) Generate an initial population of fireflies $x_i$ ( i = 1, 2, …, n)
3) Formulate light intensity I so that it is associated with f(x)
(for example, for maximization problems, I α f(x) or simply I= f(x)
4) Define absorption coefficient $\gamma$
While (t < MaxGeneration)
For i = 1 : n (all n fireflies)
For j = 1 : n (n fireflies)
If ($I_j > I_i$)
Move firefly i towards j
Vary attractiveness with distance
r via exp($-\gamma$ r )
Evaluate new solutions and update light
Intensity
End if
End for j
End for i
Rank fireflies and find the current best;
End while
Post-processing the results and visualization
End
```

Here r is the distance between two fireflies. The distance between any two fireflies *i* and *j* at x_i and y_i is a Euclidean distance measure as shown in equation 1.

$$r = d(x,y) = \sqrt{\sum_{i}^{n} (x_i - y_i)^2} \qquad (1)$$

Bagging

The expansion of Bagging is "bootstrap aggregating" it a unique method for integration decision tree or other classifiers [13]. It makes the base learning algorithm to run iteratively in successive rounds. During each round with the help of bootstrap replicate the base learner gets trained from the original training set. If the training set consists of *n* examples. Then the bootstrap replicate is a new training set that also consists of n examples, and which is formed by repeatedly selecting uniformly at random and with replacement of *n* examples from the original training set. It means that the same example may appear multiple times in the bootstrap replicate, or it may appear not at all. Thus, on each of *M* rounds of bagging, a bootstrap replicate is created from the original training set. A base classifier is then trained on this replicate, and the process continues. After *M* rounds, a final combined classifier is formed which simply predicts with the majority choice of all of the base classifiers.

Bagging (prediction algorithm A, dataset D, iterations T)

1) Model generation

```
For i = 1 to T
Generate a bootstrap sample D(i) from D
Let M(i) be result of training A on D(i)
```

2) Prediction for a given test instance x

```
For i = 1 to T
Let C(i) = output of M(i) on x
Return class that appears most often among C(1). . C(T)
```

k-Nearest Neighbour

In the k-Nearest Neighbour based imputation an attribute-att with missing value is imputed by finding its k-Nearest Neighbour and assigning its value to the attribute att.

Algorithm

```
Input :
Split the input Dataset DS into two:
$D_m$-dataset containing the instances in which at least one of the attribute value is missing
$D_c$-dataset containing complete attribute information

Output: Dataset DS containing no missing values

Method
For each vector x in $D_m$:
Divide the instance vector into observed and missing parts as x = [xo; xm].
Calculate the distance between the xo and all the instance vectors from the set Dc.
Use only those features in the instance vectors from the complete set Dc,
which are observed in the vector x.
Use the K closest instances vectors (k-Nearest Neighbors) and perform a majority voting estimate of the missing values for categorical attributes.
For continuous attributes replace the missing value using the mean value of the attribute in the k nearest neighborhood.
```

Pseudo-code for the basic k-NN classifier

```
Input: Dataset, D = {(x1, c1), (xN, cN)},
Input query t = (x1, . . . , xn), k-number of neighbour

Output: Class ‘c’ to be identified for new instance of dataset ‘t’
Begin
For each labeled instance (xi, ci)
Calculate d (xi, x)
Order d (xi, x) from lowest to highest, (i = 1, . . . , N) $D^K_x$
Select ‘K’ nearest instances to x: Assign to x the most frequent class in End $D^K_x$
End
```

Proposed Work of enhanced preprocessing in IDS

The dataset for preprocessing is collected from KDD Cup’99 dataset. Before performing any attack detection the duplication in dataset is removed using weighted minkowski based firefly algorithm and in the second stage the presence of missing value is handled using three techniques namely replace using mean value, replace using k-NN imputation and replace using proposed bagging k-NN. The performance of each technique is validated using Rule Induction Classifier. From the result obtained the complete dataset generated by proposed bagging k-NN shows a better result than the two former techniques and the complete dataset will be used for feature extraction. The detailed explanation of these two phases is shown in the following section.

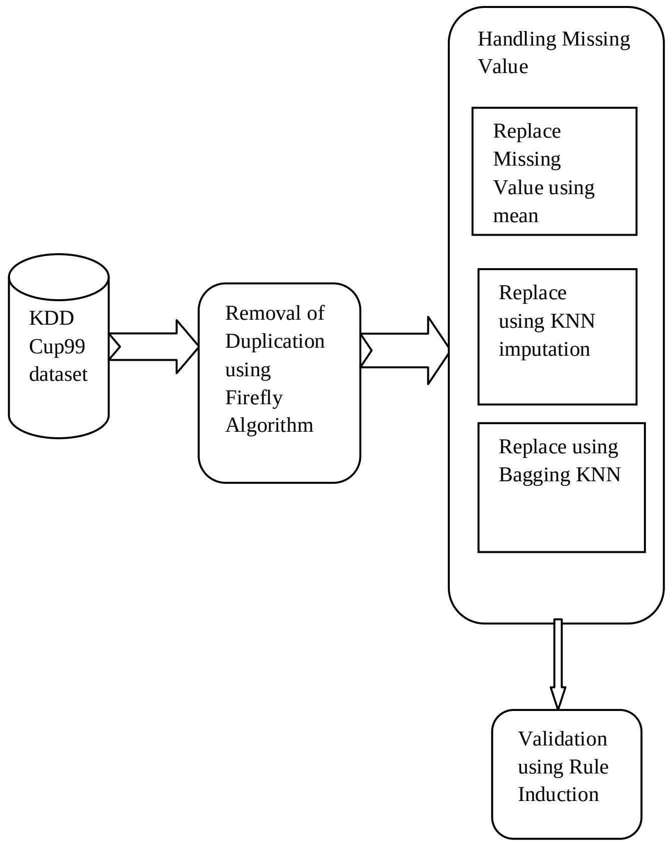
KDD Cup99 dataset
Removal of Duplication using Firefly Algorithm
Handling Missing Value
Replace Missing Value using mean
Replace using KNN imputation
Replace using Bagging KNN
Validation using Rule Induction

Figure 1: Proposed Architecture

Pseudo Code for Proposed Work Improvised Preprocessing in Intrusion detection system

Procedure:

Phase 1: Data Deduplication

Input : KDD Cup 99 Dataset

Output: Deduplicated Dataset

```
1.  Objective function objf(z), z=(z1, z2, …, zd)$^T$
2.  Initialize a population of fireflies $z_i$(i = 1, 2, . . , n)
3.  Define light absorption coefficient $\gamma$
4.  While (t<MaximumGeneration)
5.  For i=1:n (all n fireflies)
6.  For j=1:i
7.  Light intensity is determined by objf(zi)
8.  If (Ii > Ij)
9.  Move firefly i towards j in all d dimensions
10. Else
11. Move firefly i towards best solution in that iteration
12. End if
13. Attractiveness varies with distance via exp ($-\gamma^{r2}$))
14. End for j
15. End for i
16. Rank the fireflies and find the current best
17. Define normal distribution
18. For k = 1: n all n fireflies
19. $x_i = x_i + a * ( 1-p) * U(x, y)$
20. Evaluate new solution(new_cost(k))
21. If((new_cost(k)<cost(i))&&(new_cost(k)<
    last_cost_iteration(k)))
22. Move firefly i towards current best
23. End if
24. End for k
25. End while
26. End
```

Phase 2: Handling Missing Value

Input : Unique Dataset of KDDCUP obtained using the resultant of Phase 1.

Output : Complete Dataset

```
27. Given training data (x1, y1), . . . , (xn, yn)
28. For m=1, . . . , M
29. Form bootstrap replicate dataset BSt by selecting n
    random examples from the training set with
    replacement
30. let km be the result of k-NN algorithm on BSt
31. output combined classifier:
32. K(x) = majority(k1(x), . . . , kM(x))
```

In the proposed work the firefly algorithm a variant of the Minkowski function, the weighted Minkowski distance function, has also been applied to measure similarity in the dataset. The basic idea is to introduce weighting to identify important features. Assigning each feature a weighting coefficient w_i (i = 1. . . p), the weighted Minkowski distance function is defined in equation 2.

$$r = (\left(\sum_{i=1}^{p} w_i \mid x_i - y_i \mid^n \right)^{\frac{1}{n}} \quad (2)$$

By applying a static weighting vector for measuring similarity, the weighted Minkowski distance function assumes that similar records will resemble same record in the same features.

The original data set is first portioned in to groups. The records having missing values in their attributes are in one set and the records without any missing values are placed in a separate group. The k-NN classifier is trained with the complete data sets, and afterward the imperfect data is agreed to the bagging model for predicting the missing feature values. The scheme is recurrent for the whole set of attributes that have missing values. At the last part of training, this training dataset and absent value imputed datasets are combined to formulate the absolute data. The concluding dataset is then fed to the chosen k-NN classifier for classification.

Rule Induction Classifier

Rule induction [14] is one of the most important techniques of machine learning. Since regularities hidden in data are frequently expressed in terms of rules, rule induction is one of the fundamental tools of data mining at the same time. Usually rules are expressions of the form if (attribute − 1, value − 1) and (attribute − 2, value − 2) and ⋯ and (attribute − n, value − n) then (decision, value). Some rule induction systems induce more complex rules, in which values of attributes may be expressed by negation of some values or by a value subset of the attribute domain. Data from which rules are induced are usually presented in a form similar to a table in which cases (or examples) are labels (or names) for rows and variables are labeled as attributes and a decision.

Experimental Setup

In our work we have selected 65000 records as sample dataset out of 3, 11, 029 Corrected KDD dataset connections. However, because the sample number of Probe, U2R, and R2L is less, the number of records of above attack types will be constant in any sample rate. The remaining records out of 65, 000 are 44, 417 which are resulted by excluding the Probe, U2R and R2L types of records. Out of 44417, 20% of Normal connection is selected, and the Dos accounts remaining 80% of the dataset. The data sampling number and ratio are shown in Table 3.

Table 3: Amount and ratio of sampling

Category	**Corrected Dataset**		**Randomly Selected Sampled Records for Proposed Work**	
Normal	60593	19. 48%	8883	13. 67%
Probe	4166	1. 34%	4166	6. 4%
DOS	229853	73. 9%	35534	54. 67%
U2R	158	. 02%	70	. 11%
R2L	16347	5. 26%	16347	25. 15%
Total	3, 11, 029	100%	65000	100%

The proposed work consist of two stages in the first stage redundancy is removed and in the second stage incomplete dataset is converted to complete dataset by using missing value imputation. The implementation of first phase Data deduplication is done using the MATLAB and for the second phase handling missing values Rapid Miner is used.

Evaluation Metrices

Precision: Precision is a measure of the accuracy provided that a specific class has been predicted. It is defined by:

Precision = tp/(tp + fp) (3)

Where tp and fp are the numbers of true positive and false positive predictions for the considered class

Recall: Recall is a measure of the ability of a prediction model to select instances of a certain class from a data set. It is commonly also called sensitivity, and corresponds to the true positive rate. It is defined by the formula:

Recall = Sensitivity = tp/(tp+fn) (4)

where tp and fn are the numbers of true positive and false negative predictions for the considered class. tp + fn is the total number of test examples of the considered class [16].

Accuracy: The accuracy of a measurement describes how close it is to the 'real' value. This real value need not be very precise; it just needs to be the 'accepted correct value'. The accuracy is the proportion of true results (both true positives and true negatives) in the population. It is a parameter of the test. [16]

Accuracy = (tp+tn)/(tp+fp+fn+tn) (5)

The accuracy of rule Induction classifier before and after redundancy removal using Firefly algorithm is shown in the table 4 as follows:

Table 4: The accuracy of rule Induction classifier before and after redundancy removal.

	Accuracy
With Redundancy	78%
Removal of Redundancy using weighted minkowski based firefly algorithm	88%

The table 4 shows that after removing the duplication using weighted minkowski based firefly algorithm increased the accuracy of the dataset with 88 percentages because it gives a narrowed observation of pattern identification in IDS. The rule classifier produces its best result only after the redundancy is removed from the given input dataset.

After performing removal of redundancy next missing value is imputed. In this paper three techniques are compared based on the precision, recall and Accuracy. The existing techniques for comparison are Mean Value, Imputation using-KNN, Proposed Bagging k-NN.

Table 5: Comparison of Mean value, KNN Imputation and Proposed Bagging KNN

	Replace with Mean Value		**Imputation Method**		**Proposed Bagging KNN**	
	class precision	**class recall**	**class precision**	**class recall**	**class precision**	**class recall**
pred. neptune	92.31%	97.30%	94.77%	97.97%	98.65%	98.65%
pred. normal	94.74%	94.74%	95.13%	96.38%	97.99%	96.05%
pred. saint	60.00%	50.00%	57.14%	66.67%	40.00%	66.67%
pred. mscan	69.70%	79.31%	71.43%	86.21%	80.77%	72.41%
pred. guess_ passwd	97.22%	97.22%	97.30%	100.00%	90.00%	100.00%
pred. smurf	100.00%	52.94%	100.00%	64.71%	100.00%	100.00%
pred. apache2	94.44%	94.44%	93.33%	77.78%	94.74%	100.00%
pred. satan	83.33%	78.95%	78.95%	78.95%	85.71%	94.74%
pred. buffer_ overflow	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
pred. back	91.67%	100.00%	84.62%	100.00%	100.00%	100.00%
pred. warez master	94.12%	80.00%	94.44%	85.00%	95.24%	100.00%

The table 5 shows that the proposed bagging k-NN produces high accuracy, recall and precision value than the two existing techniques because the k-NN is iteratively started learning about the pattern of the dataset using bagging technique which improvise the traditional k-NN which is failed on other two techniques.

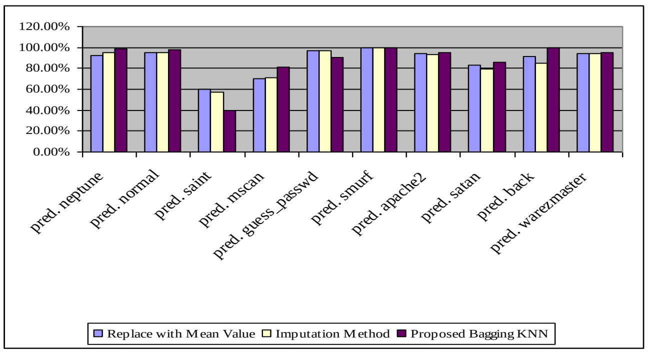

Figure 2: Performance Comparison based on Precision

The performance of mean value replacement shows better result in prediction of normal packets, saint attack, guess password attack, smurf, apache2. The K-NN imputation method produces best prediction on Neptune attack, guess password, smurf and warezmaster. But the proposed technique produces highest prediction value to most of the attacks except saint and guess pwd. Hence the work proved that the proposed work outperforms the existing techniques used in this paper.

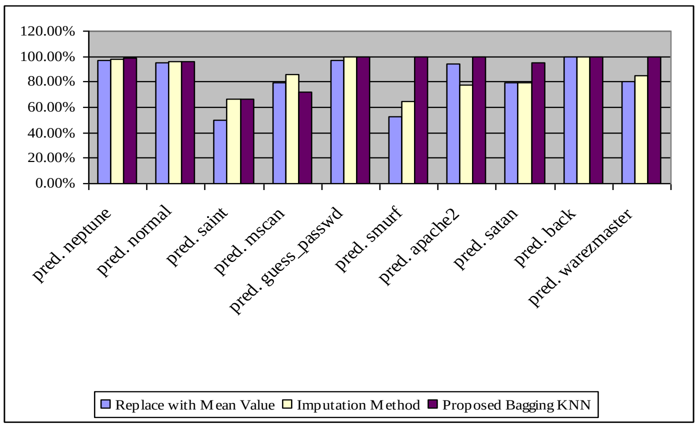

Figure 3: Performance Comparison based on Recall

Conclusion

Intrusions detection system is way of analyzing the traffic so that the unwanted packets that may contain virus or harm to the network can be detected and countermeasure. The presence of missing values in a KDD cup 99 dataset can affect the performance of a classifier constructed using that dataset as a training sample. The performance of the proposed work has significantly improved the classification accuracy and thus it reveals the importance of preprocessing in IDS.

References

[1] Tsai C. , Lin C. , A triangle area based nearest neighbors approach to intrusion detection, Pattern Recognition. 43 (2010) 222-229.

[2] Kavitha B, Karthikeyan S. , Maybell P. S. , An ensemble design of intrusion detection system for handling uncertainty using Neutrosophic Logic Classifier, Knowledge-Based Systems. 28 (2012) 88-96.

[3] Gong M. , Zhang J. , Ma J. , Jiao L. , An efficient negative selection algorithm with further training for anomaly detection, Knowledge-Based Systems. 30 (2012) 185-191.

[4] Pastrana S. , Mitrokotsa A. , Orfila A. , Peris-Lopez P. , Evaluation of classification algorithms for intrusion detection in MANETs, Knowledge Based Systems. 36 (2012) 217-225.

[5] Pereira C. R. , Nakamura R. Y. M. , Costa K. A. P. , Papa J. P. , An Optimum Path Forest framework for intrusion detection in computer networks, Engineering Applications of Artificial Intelligence. 25 (2012) 1226-1234.

[6] Sindhu S. S. S. , Geetha S. , Kannan A. , Decision tree based light weight intrusion detection using a wrapper approach, Expert Systems with Applications. 39 (2012) 129-141.

[7] Kang I. , Jeong M. K. , Kong D. , A differentiated one-class classification method with applications to intrusion detection, Expert Systems with Applications. 39 (2012) 3899-3905.

[8] Mabu S. , Chen C. , Lu N. , Shimada K. , Hirasawa K. , An Intrusion-Detection Model Based on Fuzzy Class-Association-Rule Mining Using Genetic Network Programming, IEEE Transactions on systems, MAN, and Cybernetics—Part C: Applications and Reviews. 41(2011).

[9] Hussein S. M. , Ali F. H. M. , Kasiran Z. , Evaluation Effectiveness of Hybrid IDS Using Snort with Naive Bayes to Detect Attacks, IEEE. (2012).

[10] X. S. Yang, “Firefly algorithms for multimodal optimization”, Stochastic Algorithms: Foundations and Appplications (Eds O. Watanabe and T. eugmann), SAGA 2009, Lecture Notes in Computer Science, 5792, Springer-Verlag, Berlin, pp. 169-178, 2009.

[11] X. S. Yang, (2010). “Firefly Algorithm Stochastic Test Functions and Design Optimization”. Int. J. Bio-Inspired Computation, vol. 2, No. 2, pp. 78-84, 2010

[12] X.-S. Yang, “Firefly Algorithm, Lévy Flights and Global Optimization”, Research and Development in Intelligent Systems XXVI (Eds M. Bramer, R. Ellis, M. Petridis), Springer, pp. 209-218, 2010.

[13] https://en. wikipedia. org/wiki/Bootstrap_aggregating

[14] X. S. Yang, “Engineering Optimization: An Introduction with Metaheuristic Applications”. Wiley & Sons, New Jersey, 2010

[15] http://kdd. ics. uci. edu/databases/kddcup99/ kddcup99.

[16] Powers, David M W (2011). "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation" (PDF). Journal of Machine Learning Technologies 2 (1): 37-63.