Analysis of Data Pre-processing Influence on Intrusion Detection using NSL-KDD Dataset

Nerijus Paulauskas
Vilnius Gediminas Technical University
Department of Computer Engineering
Vilnius, Lithuania
nerijus.paulauskas@vgtu.lt

Juozas Auskalnis
Vilnius Gediminas Technical University
Department of Computer Engineering
Vilnius, Lithuania
j.auskalnis@gmail.com

***Abstract*— Data pre-processing for machine learning methods is key step for knowledge discovery process. Depending on nature of the data, pre-processing might take the majority time of data analysis. Correctly prepared data for processing guarantees precise and reliable results of data analysis. This paper analyses initial data pre-processing influence to attack detection accuracy by using Decision Trees, Naïve Bayes and Rule-Based classifiers with NSL-KDD dataset. In addition, the results of detected attacks accuracy dependency by selecting different attacks grouping options and using ensembles of various classifiers are presented.**

Keywords— pre-processing; data mining; classifiers; intrusion detection

I. Introduction

Computer system attacks, data leakage, distribution of malicious software, denial of service attacks, data manipulation and other malicious activities disrupts the normal information systems activity. Due to these malicious activities companies losing their rivalry, client trust, profit and financial stability. In order to minimize the risk for information systems and possibly negative outcomes, it is essential to identify attacks in the initial stage, before they compromise the system. One way to deal with attacks is by using intrusion detection systems (IDS). Currently widely used intrusion detection and prevention systems are working like antivirus systems – known attacks patterns are used for intrusion detection as well as prevention. By using templates known attacks can be detected efficiently, however new attacks are created everyday and in order to work efficiently, systems needs to be updated with new signatures constantly and this approach will not detect zero day exploits because there will be no signature for the new attack type yet.

Every attack can be seen in data flow as network anomaly, hence these new attack types can be detected by using anomaly detection methods. By applying these methods, data can be classified to normal and potential attack. Anomalous data in the data set contains only the small fraction of all data compared to normal data, hence this data can be classified by applying intelligent algorithms and creating a model which later can be used to detect intrusions. Various papers select different anomaly detection approaches. Data can be classified only to normal or anomaly, however not every anomaly is the attack, so either all attacks must be used separately, or attacks must be differentiated to attack types. Network attacks can be grouped into four attack types [1]:

- Denial of Service attack type where the intruder makes a computer resources too full so the legitimate network requests can not be served.
- Probing attack in which the intruder scans a machine or a network device in order to determine vulnerabilities that may later be exploited.
- Remote to User attack where a user sends malicious packets to a remote machine in order to expose the machines vulnerabilities and gain privileges of a local user.
- User to Root attacks are exploitations in which the intruder with a normal user account attempts to gain super user privileges.

In this paper we grouped all attacks into four different groups as this paper does not analyze the individual attack and rather concentrates on attacks in general and their detection.

II. Related Work

Various different anomaly detection techniques and data set analyses are discussed in this section. A. J. Ghazali et al. [2] analyses five classification techniques: BFTree, SimpleCart, Ridor, PART and Naïve Bayes for anomalies detection in computer networks. Data preprocessing with Chi-Merge discretization and data normalization is used in this paper. All data were divided into 10 subsections. 9 sub parts used as training and 1 as testing set. Highest prediction accuracy rate was obtained by using PART classifier of 96.7% which is low keeping in mind that only known attacks were used. A. H. Bhat et al. [3] proposed the virtual machine monitor anomaly detection on cloud virtual machines using machine learning approach. Best accuracy was obtained with Naïve Bayes classifier and was 99.6%. The experiment was done while using 10% of training and testing NSL KDD data set data, so it is unclear if the result would be the same when using all data set. M. Tavallaee et al. [4] analyzed the initial KDD'99 data set which is based on data captured in DARPA'98 [5]. In their work they found a lot of redundant records in KDD'99 data set which led to incorrectly learned

978-1-5386-3998-6/17/$31.00 ©2017 IEEE

model creation mainly because learning algorithms were biased towards frequent data and not infrequent data such as U2R or R2L attacks. The redundant data was removed and only selected data was kept in the proposed data set. Chidananda Murthy P. et al. [6] builds and tests the model by using the same train data set and dividing the data into 80% of data set for training and 20% for testing. Because all attacks known by trained model, accuracy results of proposed methods are in 99% range. NSL_KDDTest data set contains additional 14 attacks so the accuracy then using different data set is expected to be lower. Yogita B. Bhavsar et al. [7] used support vector machine classifier for intrusion detection. NSL_KDD dataset was used in this paper as well. The drawback with using support vector machine classifier was long training time. To reduce the time, radial basis function (RBF) was applied. Herve Nkiama et al. [8] proposed a method which consists of feature selection and recursive feature elimination using decision tree classifier. Presented results also falls in 99% range, however features which were selected then accuracy of 99.9% was obtained are not presented in the paper, hence it is difficult to reproduce proposed method.

Most of the observed works uses the same data set for training and testing by dividing it into two parts. Our experiment will use all NSL KDD training data set for training and all testing data set for testing mainly because testing data set contains additional attacks which are not in training data set and mimics new attack type in the network. Also more then one classifier will be used for model in order to detect if it will have impact on the accuracy.

III. Data Set

In order to create the accurate model, data set for training and testing must have various attack types and distribution of those attacks must reflect the real world scenario. One of the first data sets for network anomaly detection was created in 1998 by Lincoln Laboratory in Massachusetts Institute of technology called DARPA'99 which one year later was improved by introducing KDD'99 data set [4].

For our experiment we use NSL-KDD data set because it does not include redundant records, there is no duplicate records and due to this the performance of the learners are not biased towards more frequent records. This data set is a successor of KDD'99 data set – an improved DARPA'98 data set which was criticized by McHugh [5].

Data set for method analyses was selected NSL-KDD. In this data set, the amount of repetitive data is reduced in comparison with KDD 99 data set. This way the data set is closer to the real world scenario. Data set consists of 125973 network flow data, 41 feature and one class which is marked as normal or attack. In the train data set there are 24 different attack types. Data set for testing contains additional new 14 attacks. In this way, testing is done with unknown attacks as well. Advantages of using NSL-KDD data set:

1. Redundant records are removed to enable the classifiers to produce an unbiased result.

2. Sufficient number of records is available in the train and test data sets, which is reasonably rational and enables to execute experiments on the complete set.

3. The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD data set.

Full each feature description are well explained in paper written by Y. Wahba et al. [12]. The distribution of data types in NSL_KDDTrain+ and NSL_KDDTest+ data sets are presented in the Table 1.

TABLE I. Data Types Distribution in Data Sets

	DOS	NORMAL	PROBE	R2L	U2R
Train Data set	45927	67343	11656	995	52
Test Data set	7460	9710	2421	2885	67

Table 1 shows that most of the records in Train data set are DOS and NORMAL types and while Test data set is distributed more equally across the data set. U2R type takes only small fraction of the whole data sets.

IV. Methodology

Previous works on anomaly detection and analysis on NSL-KDD data set showed that this data set is widely used for benchmark of classification models. Various papers uses popular feature extraction and classification algorithms to build the anomaly detection models. Most of them use following steps to build the model:

- Data pre-processing and feature extraction.
- Model creation and evaluation.
- Model improvement.

In data pre-processing step, the data set is loaded and prepared for feature extraction. If necessary, data can be normalized, standardized, unnecessary fields removed.

Depending on methods used, after pre-processing is done, features can be extracted using any of machine learning algorithms. Popular approaches are using Filter or Wrapper methods [13]. For wrapper method subset evaluator is used to create all possible subsets from feature vector. Popular subset evaluators are CfsSubsetEval and ConsistencySubsetEval. For Filter method instead of giving the selected attributes, all attributes are used in a ranking order. The attribute with last order has least priority.

Processed data is passed to classifiers and the prediction model is created. For wrapper method, classification is performed by using Best First, Greedy Stepwise and Genetic Search algorithms. For Filter method ranker algorithms are used to put features in order from highest priority to lowest. Next the data is loaded to classifier and prediction model is created. The most popular classifiers are:

- C4.5 and C5.0 decision treles.
- Naïve Bayes.

- Rule-Based classifiers (PART, OneR etc.).

Testing is done by loading test data into prediction model.

The relationship between the positive class and negative class predictions can be described as a confusion matrix that shows whether predictions fall into one of the four categories:

- True Positive (*TP*): Correctly classified as the class of interest.
- True Negative (*TN*): Correctly classified as not the class of interest.
- False Positive (*FP*): Incorrectly classified as the class of interest.
- False Negative (*FN*): Incorrectly classified as not the class of interest.

From the confusion matrix we can check the precision and accuracy of the model. In this paper we are interested in accuracy because the set can be precise if the values are close to the average value, while for set to be accurate the values must be close to the true value of the quantity being measured.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \quad (1)$$

In this formula, the terms *TP*, *TN*, *FP* and *FN* refer to the number of times the model's predictions fell into each of these categories. The accuracy is a proportion that represents the number of true positives and true negatives, divided by the total number of predictions.

Another approach used is called ensemble. All the ensemble methods are based on the idea that by combining multiple weaker learners, a stronger learner is created. Advantages of using ensemble versus a single classifier:

- Over fitting occurrence is reduced because the situation when one classifier is dominating is omitted.
- Models that divide the task into smaller portions are likely to more accurately capture patterns that a single global model might miss.

Model used in experiment is presented in Fig. 1.

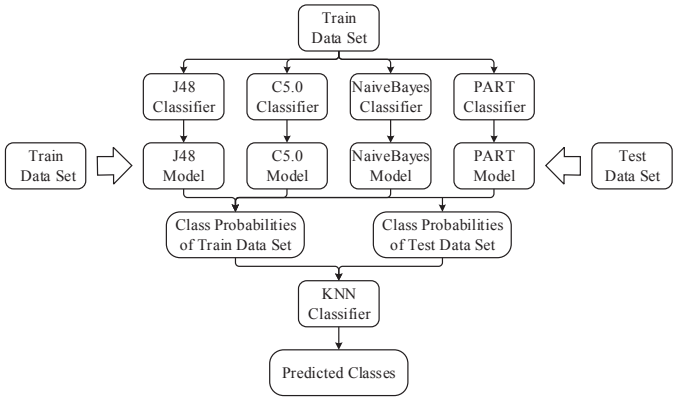

Fig. 1. Ensamble model with four classifiers.

Model used in the experiment does not contain pre-processing and features extraction. Raw data is passed to the classifiers. Classifiers creates a prediction models and train data as well as test data is passed again to the created models in order to get class probabilities. Once probabilities are calculated, both train and test data probabilities are passed to KNN classifier to find the closes match.

V. Experiment and Results

Experiments consisted of several different approaches:

- With and without preprocessing using only train data set and dividing it into 70% for training and 30% for testing. Attacks were grouped into 1 (BAD) or 4 (DOS, PROB, R2L, U2R) classes;
- With and without preprocessing using training data set and testing data set; Attacks were grouped into 1 or 4 classes;
- Use of ensemble model of four different classifiers: J48, C5.0, Naïve Bayes and PART.

For the first experiment we take train data set NSL_KDDTrain+ and group all attacks to four groups: "DOS", PROBE", "R2L" and "U2R". For this experiment we will not apply pre-processing. Grouped data is then divided into two parts: 70% for training and remaining 30% for testing. Results are presented in Fig. 2a. As it can be seen from the figure, when pre-processing is not applied and the same data set is used for training and testing results are highly accurate. Three out of four classifiers show accuracy higher than 99%. When using Naive Bayes classifier, accuracy was only 67%. If we group all attacks into only one "BAD" group and repeat the experiment, accuracy of Naive Bayes is increased and is more than 89% (Fig. 2b) while this modification had no impact to other classifiers.

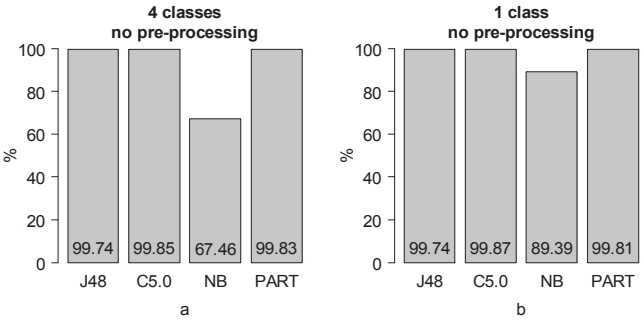

Fig. 2. Accuracy results using only train dataset splitted into 70% for training and 30% for testing. Attacks were grouped into 4 classes (a) and 1 class (b).

Fig. 3 shows results obtained when attacks were grouped into 4 groups and MinMax normalization as well as Z-Score standardization was applied in pre-processing stage. After MinMax normalization, numeric values are changed to values in the range between 0 and 1, and when Z-Score standardization is applied, mean value of feature becomes equal to 0 and standard deviation equals to 1. Results presented in Fig. 3a and 3c are obtained for all numeric values

when MinMax and Z-Score processing is applied and all data is divided into 70% and 30% parts afterwards.

Results in Fig. 3b and 3d are obtained when data was divided into 70% and 30% at first and only after the data is divided, MinMax and Z-Score processing is applied. Then only MinMax is applied, results are nearly the same, however when Z-Score is applied, accuracy is lower by approximately 25%. MinMax normalization did not have noticeable impact, because 20 out of 41 train data set features had max value of 1 even before MinMax was applied. In this case, when using continuous variables together with categorical values, MinMax normalization method is more suitable, because Z-scores fall in an unbound range of negative and positive numbers and, unlike the normalized values, they have no predefined minimum and maximum, therefore extreme values are not being fetched towards the center, and hence are making bigger impact.

Fig. 2 shows that high accuracy is obtained when the same data set is used for classifiers testing and training. Results changes significantly when train data set is used for training and test data set for testing. Accuracy results are presented in Fig. 4. If we group all attacks in 4 classes, highest accuracy is obtained with PART classifier and is equal to 78% (Fig. 4a). If we group all attacks to only one "BAD" class, detection accuracy is increased by 4% and is equal to 82%.

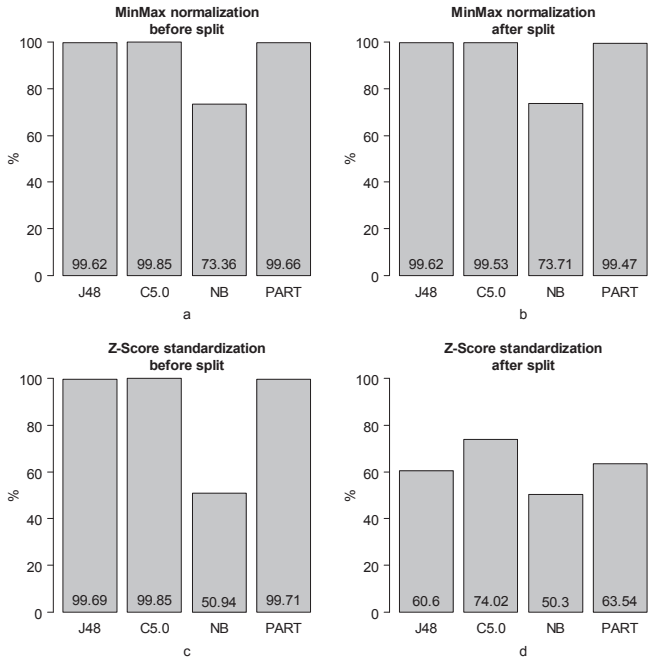

Fig. 3. Accuracy results using only train dataset with applied MinMax normalization and Z-Score standardization.

Next we analyzed how accuracy values changed when testing data was normalized with respect to training data. If model was trained with normalized data then new record also must be normalized with respect to train data numeric values, e.g., train data set feature V1 max value is equal to 42910 and test data set V1 feature max value is 57720 (in total test data set has four V1 feature values higher then 42910), hence after we add train and test data sets and apply the MinMax or Z-Score, all values will change, because with MinMax we change max value and with Z-score we change the mean value and standard deviation. When processing train and test data sets separately and paying attention to train data numeric values, after MinMax normalization four V1 feature values will be greater then 1. This is correct because in computer networks, sessions differs significantly one from another (e.g. amount of transferred data, session time etc.). As it can be seen from the results in Fig. 5, when applying MinMax normalization to test data and train data separately, we get lower accuracy rate. Especially it can be seen while using Naïve Bayes classifier. Even bigger difference is observed when Z-Score is applied. With Z-Score standardization, accuracy rate is lower by 7% (Fig. 6).

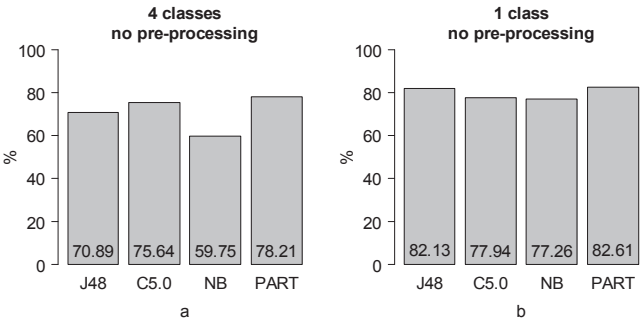

Fig. 4. Accuracy results using train dataset for training and test dataset for testing.

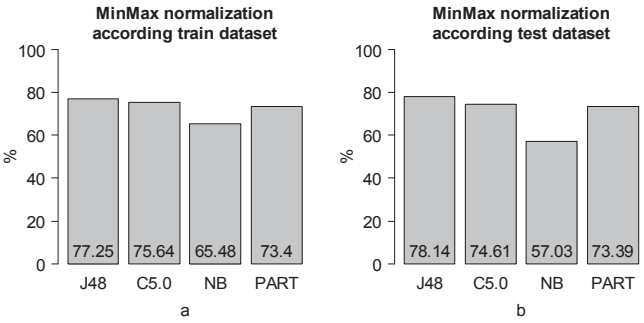

Fig. 5. Accuracy results when Test data normalized according minimal and maximum values of train dataset. Train dataset used for training and test dataset used for testing. Attacks were grouped into 4 classes.

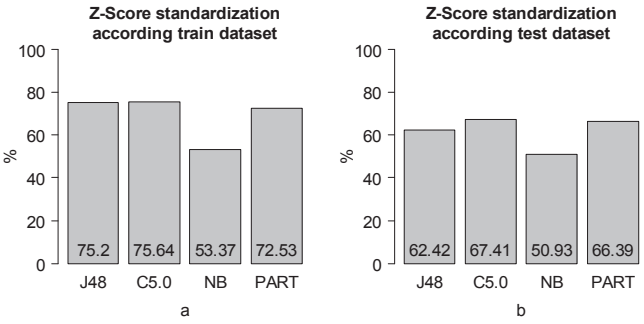

Fig. 6. Accuracy results when Test data scaled according mean and standard deviation values of train dataset. Train dataset used for training and test dataset used for testing. Attacks were grouped into 4 classes.

Final experiment goal was to examine how attack detection accuracy will change when J48, C5.0, Naive Bayes and PART classifiers are used to form the ensemble (Fig. 1).

For this experiment NSL_KDDTrain+ data was used for training and NSL_KDDTest+ data for testing. All attacks were grouped into "BAD" type. No pre-processing was done before classification. Data was trained with J48, C5.0, NB and PART classifiers. Output model was then again fed with train data to get the probabilities for BAD and NORMAL types. In order to use these probabilities for classification, test dataset needs to be presented as probabilities as well. To get the test data as probabilities, trained model was fed with test dataset. For testing only BAD type probabilities were used because, NORMAL probabilities are reciprocal of BAD probability values.

KNN (k nearest neighbor) algorithm was used to tie train data classified with "J48", "C5.0", "NB", "PART" classifiers and test data classified with the same classifiers. K value is set to 1 so that the most accurate first neighbor could be find. Also, the accuracy was checked using different possible "J48", "C5.0", "NB", "PART" classifiers combinations. In total 11 combinations are possible when using only 2, 3 and all 4 classifiers. In Fig. 7 the combinations of classifiers are labeled according first letters of each classifier. Then comparing Fig. 7 with Fig. 4b, we see that classifier ensemble obtained better attack detection accuracy. Best accuracy was obtained while using (J48 and PART) classifiers and (J48, Naïve Bayes and PART) classifiers and is equal to more then 84%. After repeating the experiment with different k values (k=3; 5; 11; 15 and 25) most stable results was observed with ensemble of J48, Naïve Bayes and PART classifiers.

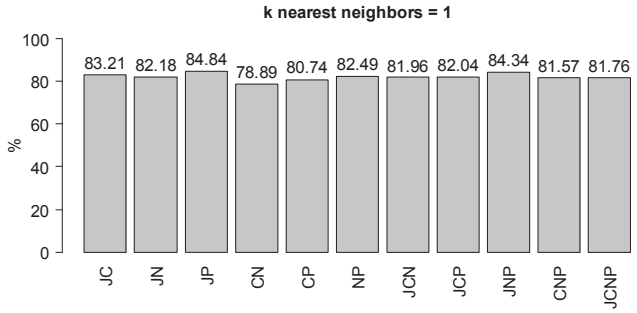

Fig. 7. Results of accuracy using different combination of classifiers and k=1.

VI. Conclusions

In this paper we used various popular classifiers and combined them in one ensemble model in order to improve the results of accuracy of intrusion detection. It was noticed that when J48, C5.0, Naïve Bayes and PART classifiers are used with attacks grouped to only one "BAD" class, the accuracy of the model is increased. The highest augment was in the case with Naïve Bayes classifier.

The use of only NSL_KDDTrain+ data set for training and testing the classifiers is not recommended as even without pre-processing the accuracy is 99%. For testing, NSL_KDDTest+ data set must be used. In this case the real model performance to detect new type of attacks can be tested.

When pre-processing is used for training data set, the same pre-processing with respect to numerical characteristics of train data values must be applied for testing data. This is necessary so that test data numerical characteristics will not get fetched in respect to train data and in that case the values will fail to fit the trained model.

Using continuous variables together with categorical variables MinMax normalization method is preferred because Z-scores fall in an unbound range of negative and positive numbers and unlike the normalized values, they have no predefined minimum and maximum, therefore extreme values are not being fetched towards the center and hence are making bigger impact.

Proposed classifier ensemble model produces more accurate results. The most accurate results were obtained using ensemble of J48, Naïve Bayes and PART classifiers.

References

[1] Swati Paliwal, Ravindra Gupta, "Denial-of-Service, Probing & Remote to User (R2L) Attack Detection using Genetic Algorithm", International Journal of Computer Applications, Volume 60– No.19, 2012.

[2] Amzari Jihadi Ghazali, Waleed Al-Nuaimy, Ali Al-Atabi, Isalinda Jamaludin, "Comparison of classification models for NSL-KDD data set for network anomaly detection", Academic Journal of Science, Volume 04, 2015.

[3] Amjad Hussain Bhat, Sabyasachi Patra, Debasish Jena, "Machine Learning Approach for Intrusion Detection on Cloud Virtual Machines", International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 2, Issue 6, 2013.

[4] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set", Proceedings of the Second IEEE international conference on Computational intelligence for security and defense applications (CISDA'09), IEEE Press, Piscataway, NJ, USA, 2009, p. 53-58.

[5] M. Mahoney and P. Chan, "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection," Springer, pp. 220–238, 2003.

[6] Chidananda Murthy P., A. S. Manjunatha, Anku Jaiswal, Madhu B. R., "Building Efficient Classifiers For Intrusion Detection With Reduction of Features", International Journal of Applied Engineering Research, Volume 11, Number 6, pp. 4590-4596, 2016.

[7] Bhavsar Y. B, Waghmare K. C. "Intrusion Detection System Using Data Mining Technique: Support Vector Machine," International Journal of Emerging Technology and Advanced Engineering, Vol.3, Issue 3, pp.581-586, 2013.

[8] Herve Nkiama, Syed Zainudeen Mohd Said, Muhammad Saidu, "A Subset Feature Elimination Mechanism for Intrusion Detection System", International Journal of Advanced Computer Science and Applications, Vol. 7, No. 4, 2016.

[9] S. Revathi, A. Malathi, "A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection", International Journal of Engineering Research & Technology (IJERT), vol. 2 Issue 12, 2013.

[10] L.Dhanabal, S.P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms", International Journal of Advanced Research in Computer and Communication Engineering, vol. 4, Issue 6, 2015.

[11] Daniel T. Larose, Chantal D. Larose, Data Mining and Predictive Analytics Handbook. Wiley, 2015.

[12] Yasmen Wahba, Ehab ElSalamouny, Ghada ElTaweel, "Improving the Performance of Multi-class Intrusion Detection Systems using Feature Reduction", IJCSI International Journal of Computer Science Issues, Volume 12, Issue 3, 2015.

[13] Mohanabharathi, R., Kalaikumaran, T., Karthi, S., "Feature selection for wireless intrusion detection system using filter and wrapper model", International Journal of Modern Engineering Research (IJMER), 2(4), 1552–1556, 2012.