Development of an Optimized Botnet Detection Framework based on Filters of Features and Machine Learning Classifiers using CICIDS2017 Dataset

Aaya F. Jabbar 1, Imad J. Mohammed 2

1, 2 Department of Computer Science, University of Baghdad, Baghdad, Iraq
aayafadhil@gmail.com1, Dr.Imadjm@scbaghdad.edu.iq2

Abstract

Botnet is a malicious activity that tries to disrupt traffic of service in a server or network and causes great harm to the network. In modern years, Botnets became one of the threads that constantly evolving. IDS (intrusion detection system) is one type of solutions used to detect anomalies of networks and played an increasing role in the computer security and information systems. It follows different events in computer to decide to occur an intrusion or not, and it used to build a strategic decision for security purposes. The current paper suggests a hybrid detection Botnet model using machine learning approach, performed and analyzed to detect Botnet attacks using CICIDS2017 dataset. The proposed model designed based on two types of filters to the botnet features; Correlation Attribute Eval and Principal Component deployed to reduce the dataset dimensions and to decrease the time complexity of the botnet detection process. The detection enhancement achieved by reducing the features of the dataset from 85 to 9. The training stage of classifiers is developed and compared based on six classifiers called (Random Forest, IBK, JRip, Multilayer Perceptron, Naive Bayes and OneR) evaluated to accomplish an optimized detection model. The performance and results of the proposed framework are validated using well-known metrics such as Accuracy (ACC), Precision (Pr), Recall (Rc) and F-Measure (F1). The consequence is that the combination of Correlation Attribute Eval (filter) with JRip (classifier) together can satisfy significant improvement in the Botnet detection process using CICIDS2017 dataset.

Keyword: Botnet Detection, Machine Learning, WEKA, Feature Selection, CICIDS2017 dataset.

1-Introduction

Despite the many good features of Internet technology, but some participants use it for harmful purposes such as illegal access to confidential information to either steal or destroy or may be competing with the business using various attack tools [1]. Along the time, protection systems against various attacks have been established and developed but almost suffers from restrictions because of every day created a various type of attack. One of those attacks to the Internet called Botnets are a group of the internet-connected devices that are remotely managed by an attacker to carry out malicious tasks. The attacker who is accountable for such an attack called botmaster. The botmaster host controls and sends commands to the infected devices (called bots or zombie computers) for performing unwanted tasks such as unauthorized access for monitoring personal data or stealing it. Furthermore, the botmaster can send spam, DDoS attacks ..., etc. Figure 1 displays a simplified view of Botnet components [2, 3].
2-Related Works
In paper [4], a PCA-RNN framework developed to combine the Principal Component Analysis and Recurrent Neural Network to detect DDoS attacks using KDD dataset. PCA applied to reduce the dataset dimensions and consequently diminishes the detection time. RNN developed as training process to formulate a detection model. Accuracy, sensitivity, precision, and F-score used for results evaluation. The main outcome of PCA-RNN was in the improvement of accuracy by up to 98.7%.

The authors of [5] proposed a Botnet detection system based on the machine learning techniques using DNA (Domain Name Service) query data. It deals with machine learning classifiers such as Naive Bayes, k-Nearest Neighbor, Decision Tree and Random Forest to assess the effectiveness of detection model supported by some measures of classifiers such as Positive Predictive Value, False Positive Rate, True Positive Rate, Accuracy and F1-Measure. The best detection accuracy reached 90.80% using Random Forest algorithm.

The paper [6] discusses the detection system of a DDoS attack within CICIDS2017 by evaluating and applying some machine learning methods. It uses common filter model methods of feature selection called information gain, which measures the importance of each attribute based on the higher entropy concerning the class. It selects the best ten important attributes from 80 features of CICIDS2017 dataset. Then experimented four machine learning models using C5.0, Naive Bayes, SVM and Random Forests algorithms, and compared based on Accuracy, Recall, Precision, Detection Rate and False alarm to choose the best model. The Random Forest (RF) and C5.0 classifiers surpass the others by obtaining the highest results with Accuracy (86.80% and 86.45% respectively), Recall (0.86290 and 0.85925 respectively), Precision is about 99%, Detection Rate is about 81% and False Alarm (0.05072 and 0.04637 respectively).

In [7], the authors proposed a semi-supervised machine learning model called RPCA-MD, which combines Robust Principal Component Analysis and Mahalanobis Distance applied to CTU-13 Botnet dataset. The metrics used to evaluate the proposed model are Precision, Recall and F-Score. Since CTU-13 dataset divided into scenarios, therefore each scenario applied separately. The highest results achieved in scenario 7, where Precision reached 0.959, Recall to 0.95 and F-Score to 0.955.

The authors in [8] proposed a detection model that combines PSO and CFS algorithms to botnet features of a dataset. The proposed model applied on different data sets (KDD Cup99, UNSWNB15 and Kyoto 2006). They tested three classifiers; Naive Bayes, K-NN, and SVM and evaluated the results using three metrics; Accuracy, TP Rate, and FN Rate. The model achieved best results using SVM, where the accuracy reached 99.9%, TPR 99.9% and FPR 0 in the KDD Cup99 dataset.

The methodology in [9] developed to detect the Botnet attack of KDD99 and UNBS-NB 15 datasets using some machine learning techniques such Support Vector Machine (SVM), Naive Bayes (NB), Artificial Neural Network (ANN), Decision Tree (DT), and Unsupervised Learning (USML). All the techniques evaluated according to the measure of Accuracy, Specificity, False Alarm Rate, Sensitivity,
False positive rate and AUC of data sets. The show results concluded that the Performance of KDD99 dataset can perform better than the Performance of UNBS-NB 15 such as the highest Accuracy value for KDD99 reached 98.08%. Also, the results of used classifiers demonstrated that USML (unsupervised learning) was the best for Botnet detection based on the performance metrics.

In project[10], authors tested a solution to classify malware traffic in a network using Machine Learning by analyzing the CTU 13 Botnet dataset, which contains 13 scenarios, to extract 22 features. They compared the feature selection processes including PCA and t-SNE to reduce the dataset dimensionality. Their experiment composed of five machine learning classifiers tested on Botnet dataset; Logistic Regression, Gradient Boosting, Random Forest, a Dense Neural Network and Support Vector Machine. Also, the results discussed based on Recall, Precision and F-Score metrics. The results show that the Random Forest Classifier can detect more than 90% of the Botnet traffics for 8 out of 13 scenarios.

3- Milestones of the proposed methodology
Current research introduces an optimized detection framework for Botnet attack. The workflow of the proposed framework (Figure 1) divided into the following main components as described in Table 1.

| No. | Framework Component | Function/Description |
|-----|---------------------|---------------------|
| 1   | Botnet dataset      | Selecting a CSV file that contains a Botnet attack from CICIDS2017 dataset, which is a real dataset previously created. |
| 2   | Data preprocessing  | Cleaning data from unwanted value and Redundant attributes to obtain the most accurate data, implemented using C# language. |
| 3   | Data Normalization  | Standardizing a range to all attributes in the dataset using minmax normalization to make them appear in a single range. This step performed using C# language too. |
| 4   | Machine learning    | Executing Feature selection and classification processes by Attribute Selected Classifier method with cross-validation (10 folds) using WEKA machine learning. Feature selection for determining the features of Botnet detection in CICIDS2017 dataset using two filter methods; Correlation Attributes Eval and Principal Component separately, then using classification algorithms (IBK, JRip, Multilayer perceptron, Naïve Bayes, OneR and Random Forest) for testing selected features. |
| 5   | Evaluation          | Comparisons of the produced results from the tested machine learning algorithms based on the performance measures such as Accuracy ACC, Precision Pr, Recall Rc, and F-Measure F1. Consequently, the best detection algorithm promoted as a solution for realism environment. |
3.1- CICIDS2017 Dataset
CICIDS2017 is a known Intrusion Detection System (IDS) dataset adopted for the construction of the proposed framework, which is a public dataset and can be obtained free from (http://www.unb.ca/cic/datasets/IDS2017.html) [11]. It contains real data that were recorded depending on the behavior of a network of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols. CICIDS2017 created in July 2017, in the form of separate files on several days. The capturing period began at 9 A.M Monday and ended at 5 P.M Friday. One of its advantages in providing the data as separate files, each file introduces a specific type of attack. It contains benign and various malicious attack traffic such as Web Attack, Infiltration, Heart bleed, Botnet..., etc. CICIDS2017 dataset recorded depending on important fields which are timestamp, IP and port of source, IP and port of destination IPs, protocol and attack that were processed and analyzed using CICFlowMeter with labeled flows to extract 80 important features for various types of attacks. The other advantage of this dataset is to provide data using labeled CSV format which is acceptable in many applications [12]. One of the CICIDS2017 dataset files related to the Botnet attack (named Friday-WorkingHours-Morning.pcap_ISCX) selected and tested in our proposed framework. The mentioned file contains 1966 Botnet traffic and 191033 benign traffic.

3.2- Data Preprocessing
In literature and designing of security systems, a pre-processing step for data considered as one of the important and essential stages in intrusion detection systems. The presence of dataset in different representation and sizes usually affect the computational performance and the accuracy of the detection system, specifically when using a dataset of high dimensions and many redundant besides related features. To surmount this hurdle, the dataset must be processed before the learning phase to reduce or eliminate the undesirable characteristics of the data. Also, the attributes contain different types of values, including symbolic and numerical values, thus data may include NAN and infinite or fixed values that
need certain manipulation. The implemented steps of Botnet dataset preprocessing considered the preprocessing in [13, 14] as discussed in the subsequent subsections:

3.2.1- Dataset Cleaning

The presence of repeated and unwanted values in the dataset is a common problem that affects the performance of a system because repeated values mostly cause time-consuming without interest; therefore, it is necessary to delete them. For example, NAN and Infinity values must be deleted or replaced with other values with the existence of detection algorithms that do not deal with these values.

In the current research, a repeated attribute such as 'Fwd_Header_Length' that appeared twice in Botnet dataset is deleted. Also, some attributes contain NAN and Infinity values (such as 'Flow Bytes/s', 'Flow Packet/s') are replaced by other values. For example, while NAN value replaced with a minimum value, an Infinity value replaced with a maximum as a solution for such type of attributes [13].

3.2.2- Removing Zero-Attributes

Zero-attribute is an attribute that has a single value equal to zero for all records. It can be found by applying several methods such as calculating the summation or the minimax values that give zero. In our proposed work, the dataset attribute is considered as zero-attributes type when its minimum and maximum values of its values is zero. Thus, it is expected that removing the zero-attributes increases our model accuracy. The analysis of CICIDS2017 dataset shows that it contains ten zero-attributes with the same value for all records namely; 38-Bwd PSH Flags, 39-Fwd URG Flags, 40-Bwd URG Flags, 59-CWE Flags Count, 63-Fwd Avg Bytes/Bulk, 64-Fwd Avg Packets/Bulk, 65-Fwd Avg Bulk Rate, 66-Bwd Avg Bytes/Bulk, 67-Bwd Avg Packets/Bulk and 68-Bwd Avg Bulk Rate.

3.2.3- Removing relevant network flow attributes

CICIDS2017 contains attributes that were registered while getting data flow. These attributes do not affect model results. Thus, removed to reduce the data dimension consequently affecting the expected performance positively. For example, the four network flow attributes; source IP address, destination IP address, and timestamp considered as none relevant network flow attributed. By deleting them, the processing overhead of the nominal attributes disappears because some detection models require numeric values instead of the nominal values.

3.3- Data Normalization

The attributes values in CICIDS2017 dataset show a diverse range of values. Thus, data normalization required to support both the accuracy and performance of the developed Botnet detection framework. In current work, Minmax normalization proposed as one of the normalization strategies that transform the dataset values into a certain suitable processing range per attribute, as formulated in the following equation (1).

\[ Y = \frac{X_i - X_{min}}{X_{max} - X_{min}} \]

Where \( i \) is the counter of values \( X_i \) of an attribute \( X \), \( X_{min} \) and \( X_{max} \) are minimum and maximum values of an attribute \( X \). The new range of data would be within a 0–1 range [14].

3.4- Machine learning

Machine learning algorithms usually used to carry out many tasks as well as to facilitate and speed up many domains that need these tasks. In literature, different machine learning tools used to test the detection models, one of which is Weka [15]. Weka is a machine learning tool developed by Waikato University using Java language. This tool supports and performs various tasks such as data preprocessing, classification, clustering, selection of attributes, association rules, visualization, etc. Also, it supports a predictive model and analyzing of data [16].

3.4.1- Feature selection

It is a dominated process used to eliminate the unrelated features from a dataset without much loss of the information. Feature selection is known or represented as a “select attributes option” in WEKA. Filter approach is a type of feature selection class besides another two approaches called wrapper and embedded [17]. Since CICIDS2017 dataset used in this research contains various features that are specific for all attacks, therefore select attributes step is used to identify the attributes that are related to the Botnet attack. In the current paper, two filter ranking algorithms (Correlation Attribute Eval and Principal Components) formulated and proposed to support the attributes selection during the Botnet detection process. The following subsections introduce the description and formulation of the two
A) Correlation Attribute Eval
Correlation Attribute Eval is Pearson’s correlation filter method used to calculate the correlation among each attribute to the target class. The filter used with ranking search to evaluate the specific attributes of the target class, and to arrange features in ascending order based on the highest rank. Followed by selecting the best (or highest) specific number among these features that can accomplish better performance results [18] as formulated in equation (2).

\[
r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}
\]

(2)
where X denotes an attribute vector, Y represents a target class vector and i used as a counter for values of the attribute vector [19].

B) Principal Component Analysis
It is also a filter method deployed for dimensionality reduction of attributes [20], which differs in its work from the other filters reliant on ranked features to choose the best features. PCA method depends on both the eigenvalues and eigenvectors in its arrangement and evaluation process of features as follows [21]:
1- Create a data matrix from the dataset.
2- Subtract the mean value from each data value.
3- Find a covariance matrix by calculating the covariance per dimension of the dataset [22].

\[
\text{Cov}(X, Y) = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{(n-1)}
\]

(3)
4- Calculate eigenvectors and eigenvalues based on a covariance matrix.
5- Choose the most impact of features based on a covariance matrix.

3.4.2- Classification
Typically, most dataset classification processes by machine learning paradigm used to categorize each item of the dataset to one of the predefined classes [16]. In the current paper, some classification algorithms are tested on CICIDS2017 dataset to verify the accuracy and the capability of these algorithms on detecting the botnet attack. Six classification algorithms are implemented for achieving the objective of current research called; IBK (Instance Based Learner), JRip, Multilayer perceptron (MLP), Naïve Bayes (NB), OneR and Random Forest (RF). The following subsections explain these six algorithms briefly.

A) IBK Classifier
IBK (Instance-Based Learner) is one of laze classifier supported in Weka that works based on nearest neighbors. IBK is easy to execute and yet performs quite complex classification tasks. Furthermore, IBK classifier supports clustering and regression processes. It uses distance rules to get the nearest neighbors in selecting a target class of nearest neighbors for the classification of a new item [23].

B) JRip Classifier
It is a rule-based classifier. It generates a rule for one class as the initial step using incrementally reduced error JRip (RIPPER) and handles the optimization by selecting the best rule. Then, it proceeds to duplicate the same process for the next classes [24, 25].

C) Multilayer Perceptron Classifier (MLP)
It is a neural network classifier that consumes little time to test the data even though it takes a very long time during the training stage. It used for several tasks like classification, regression and prediction. MLP works based on three layers: classes number (output), dataset (input) and hidden layers. Also uses weights for every node at the neural network. It promotes the effective attributes that get large weights [26].

D) Naïve Bayes classifier (NB)
It is a simple Bayes classifier that built with strong independence probabilities. It assumes the features are symmetrically independent [21].

E) OneR Classifier
Considered as simple rule classifier. Typically, OneR can deduce simple rules from the set of instances by creating a rule for each attribute, followed by selecting the rule that produces a minimum error rate. Also, it can handle missing values and considered as flexible algorithm despite simplicity [21].

F) Random Forest Classifier (RF)
It combines characteristics of various tree classifiers to predict new unlabeled data; therefore, considered as an ensemble algorithm of classification and prediction processes. The prediction classes represent the
number of forests. Each forest contains several trees. In RF, attributes can be selected randomly. Though, the classified trees that produced from various elements of the dataset used to vote during the testing of new objects. For example, when the new object seeks to implement, each tree needs to vote for a decision to select the forest that registers the highest votes number [22].

3.5- Performance Metric
Performance metrics are measures used to compare and evaluate the results of learning methods applied to the data [27]. Well-known performance metrics depend on a confusion matrix. Table 2 exhibits the elements of a 2 * 2 confusion matrix in which the main diagonal represents the correct predictions that contain:
- True-Positive: TP that means data are positive and classified as positive.
- True-Negative: TN activated when data are negative and classified as negative.
While the secondary diagonal of the matrix indicates incorrect predictions that contain:
- False-Positive: FP points out that data are negative and classified as positive.
- False-Negative: FN justifies data are positive and classified as negative.

| Table 2. Confusion matrix |
|----------------------------|
| Actual | Predicted |
|        | Attack    | Normal     |
| Attack | True Positive TP | False Positive FP |
| Normal | False Negative FN | True Negative TN |

There are many performance metrics defined by confusion matrix elements. In this study, certain performance metrics used to appraise the performance of the detection system, including Precision (PR), Recall (Rc), F-Measure (F1), and Accuracy (ACC). The definition and formulation of these metrics provided below:

- Accuracy (ACC): denotes a metric to calculate the accuracy of the detection system as it calculates the ratio of the elements that were correctly detected.

\[
ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{4}
\]

- Precision (Pr): a metric that is used to calculate the number of attacks detected by the detection system depending on TP and FN.

\[
Pr = \frac{TP}{TP + FP} \tag{5}
\]

- Recall (Rc): signifies a metric that is used to count the number of detected attacks from a total number of attacks based on TP and FP.

\[
Rc = \frac{TP}{TP + FP} \tag{6}
\]

- F-Measure (F1): characterizes the rate between precision and recall.

\[
F1 = \frac{2}{\frac{1}{Pr} + \frac{1}{Rc}} \tag{7}
\]

4-Results and Discussion
This section traces and discusses the implementation stages of the proposed framework besides the experimental results. The tested Botnet dataset CICIDS2017 includes 85 attributes and can be classified as related, unrelated and noise attributes. Table 3 shows the preprocessing results of CICIDS2017 that frees the defects and consequently promotes 71 (out of 85) attributes passed to the next processing stage.

| Table 3. The preprocessed results of CICIDS2017 Dataset produced 71 attributes (out of 85). |
| No. | Feature Name              | No. | Feature Name              | No. | Feature Name              |
|-----|---------------------------|-----|---------------------------|-----|---------------------------|
| 1   | Source Port               | 25  | Fwd IAT Std               | 49  | ECE Flag Count            |
| 2   | Destination Port          | 26  | Fwd IAT Max               | 50  | Down/Up Ratio             |
| 3   | Protocol                  | 27  | Fwd IAT Min               | 51  | Average Packet Size       |
| 4   | Flow Duration             | 28  | Bwd IAT Total             | 52  | Avg Fwd Segment Size      |
| 5   | Total Fwd Packets         | 29  | Bwd IAT Mean              | 53  | Avg Bwd Segment Size      |
| 6   | Total Backward Packets    | 30  | Bwd IAT Std               | 54  | FWD Header Length1        |
Component method).

The number of attributes in Correlation Attributes Eval method and the number of eigenvalues in Principal Component methods with six classification algorithms to generate three sets of significant attributes or three cases (three times repeated to select 10, 9 and 8 attributes separately).

Each case aims to determine the lowest number of attributes with the best results in Correlation Attribute Eval (as shown in Table 4) and the lowest number of eigenvalues with the best results in Principal Component (as shown in Table 5). Figures (3, 4) illustrate the results labeled a, b, c, and d, for singular attributes.

For example, label (a) represents Accuracy, label (b) denotes Precision, label (c) signifies Recall, label (d) characterizes F-Measure. Labels (e and f) used to present the four metrics in a 3D-Columns chart and using a 3D-Line chart respectively. Each label shows six classifiers (colors represent the number of classifiers) and for each classifier have three testings. In other words, testings represent the number of attributes in Correlation Attributes Eval method and the number of eigenvalues in Principal Component method).

It is noticed that the highest values of performance metrics produced from JRip classifier where Accuracy (ACC) reached to 99.995%, Precision (Pr) up to 1, Recall (Rc) around 0.99 and F-Measure (F1) up to 1. It is essential to mention that the most important performance metric of Botnet attack detection is pointed in Rc metric that reached 0.99, which considered as a very good percentage. Also, JRip classifier outperforms the rest tested algorithms which are Random Forest, IBK, Multilayer Perceptron, OneR and Naïve Bayes; ordered in ascending according to their ranks as results. Comparatively, the worst result found at the implementation of Naïve Bayes classifier where ACC reached to 81.32%, Pr to 0.039, Rc to 0.732, and F1 to 0775. Since the results are inspiring, we tried to reduce the dataset features to enhance the performance time besides the getting of the best features that support the proposed Botnet detection framework.

Tables (4, 5) illustrate the results of applying the performance metrics using the two: Correlation Attribute Eval and Principal Component methods with six classification algorithms to generate three sets of significant attributes or three cases (three times repeated to select 10, 9 and 8 attributes separately).

Comparatively, the worst result found at the implementation of Naïve Bayes classifier where ACC reached to 99.995%, Precision (Pr) up to 1, Recall (Rc) around 0.99 and F-Measure (F1) up to 1. It is essential to mention that the most important performance metric of Botnet attack detection is pointed in Rc metric that reached 0.99, which considered as a very good percentage. Also, JRip classifier outperforms the rest tested algorithms which are Random Forest, IBK, Multilayer Perceptron, OneR and Naïve Bayes; ordered in ascending according to their ranks as results. Comparatively, the worst result found at the implementation of Naïve Bayes classifier where ACC reached to 81.32%, Pr to 0.039, Rc to 0.732, and F1 to 0775. Since the results are inspiring, we tried to reduce the dataset features to enhance the performance time besides the getting of the best features that support the proposed Botnet detection framework.

Table 4. The results of Correlation Attributes Eval algorithm (sets: 10, 9 and 8 attributes)

| No. of Features | Classification Algorithm       | ACC     | Pr  | Rc  | F1  |
|-----------------|--------------------------------|---------|-----|-----|-----|
| 10              | Multilayer Perceptron          | 99.8398 | 0.986 | 0.857 | 0.917 |
|                 | JRip                           | 99.9995% | 1.000 | 0.999 | 1.000 |
|                 | IBK                            | 99.9932% | 0.996 | 0.997 | 0.997 |
|                 | Random Forest                  | 99.9974% | 0.998 | 0.999 | 0.999 |
|                 | Naïve Bayes                    | 81.32%  | 0.039 | 0.732 | 0.075 |
|                 | OneR                           | 99.6032% | 0.911 | 0.681 | 0.779 |
| 9               | Multilayer Perceptron          | 99.8424% | 0.989 | 0.857 | 0.918 |
|                 | JRip                           | 100%    | 1.000 | 1.000 | 1.000 |
|                 | IBK                            | 99.9932% | 0.996 | 0.997 | 0.997 |
|                 | Random Forest                  | 99.9974% | 0.999 | 0.998 | 0.999 |
|                 | Naïve Bayes                    | 78.0483% | 0.036 | 0.779 | 0.068 |
|                 | OneR                           | 99.6032% | 0.911 | 0.681 | 0.779 |
| 8               | Multilayer Perceptron          | 99.8189% | 0.990 | 0.832 | 0.904 |
|                 | JRip                           | 99.9618% | 0.995 | 0.967 | 0.981 |
|                 | IBK                            | 99.9524% | 0.981 | 0.973 | 0.977 |
Also, when features reduced to 9, it shows better results in all used classifiers except Naïve Bayes that demonstrates a reduction in results comparatively. The highest accuracy noticed at JRip that reached 100%, which was the highest level and 1 for the metrics Pr, Rc and F1. Furthermore, when we tried to decrease the number of features to 8, the results show fewer advantages, and low accuracy compared to the previous two attempts (10 or 9 features) such as ACC to 99.9618%, Pr to 0.099, Rc 0.967, and F1 0.981. Thus, it must stop at this attempt and roll back to 9 features that shows the most valuable results in detecting the botnet using Correlation Attribute Eval.

Table 5. The results of Principal Component algorithm

| No. of Eigenvalues | Classification Algorithm | ACC   | Pr   | Rc   | F1   |
|--------------------|--------------------------|-------|------|------|------|
| 10                 | Multilayer Perceptron    | 99.0007% | 0.577 | 0.109 | 0.183 |
|                    | JRip                     | 99.8864% | 0.959 | 0.929 | 0.944 |
|                    | IBK                      | 99.9482% | 0.973 | 0.977 | 0.975 |
|                    | Random Forest            | 99.9607% | 0.992 | 0.969 | 0.981 |
|                    | Naïve Bayes              | 97.1659% | 0.023 | 0.043 | 0.030 |
|                    | OneR                     | 99.3404 % | 0.742 | 0.551 | 0.632 |
| 9                  | Multilayer Perceptron    | 98.9824% | 0.696 | 0.020 | 0.039 |
|                    | JRip                     | 99.8843% | 0.955 | 0.932 | 0.943 |
|                    | IBK                      | 99.956% | 0.978 | 0.980 | 0.979 |
|                    | Random Forest            | 99.9607% | 0.992 | 0.970 | 0.981 |
|                    | Naïve Bayes              | 99.9607% | 0.992 | 0.970 | 0.981 |
|                    | OneR                     | 99.3404 % | 0.742 | 0.551 | 0.632 |
| 8                  | Multilayer Perceptron    | 98.9709% | 0.500 | 0.020 | 0.038 |
|                    | JRip                     | 99.8786% | 0.958 | 0.922 | 0.940 |
|                    | IBK                      | 99.956% | 0.978 | 0.980 | 0.979 |
|                    | Random Forest            | 99.9581% | 0.992 | 0.967 | 0.979 |
|                    | Naïve Bayes              | 97.8684% | 0.024 | 0.027 | 0.026 |
|                    | OneR                     | 99.3404% | 0.742 | 0.551 | 0.632 |

(a) Accuracy                                                                (b) Precision
Table 5 and Figure 4 depict the implementation results of the classification algorithms based on Principal Component filter. But it differs in the parameters of work from Correlation AttributeEval in specifying the number of eigenvalues. For example, when determining the highest eigenvalues (10) and applying six classifiers, it is noted that the highest results of the metrics found at Random Forest classifier except for Rc metric, where Accuracy reached to 99.9607%, PR to 0.0992 and F1 to 0.981. Also, the highest values of Rc found in IBK classifier that reaches to 0.977 despite that the results of other metrics. IBK shows lower results compared to the Random Forest. In abstract form, the classifiers arranged (ranked) according to their Accuracy results in decreasing order; Random Forest, IBK, JRip, OneR, Multilayer Perceptron and Naive Bayes. The worst performance metrics registered at Naive Bayes classifier and reached to Accuracy 97.165%, Pr 0.023, Rc 0.043 and F1 0.030. In the state of an experiment when reducing the eigenvalues to (9), it is noted that some classifiers remain stable such as Random Forest and OneR classifiers. But the results of Random Forest show an only increase in Rc ratio. Also, got improved results using some classifiers such as in IBK and Naive Bayes.
Figure 4. Comparison of the feature selection methods using Principal Component filter based on performance metrics.

It is important to mention that the Naive Bayes classifier produced the least significant results in the case of (10) eigenvalues. Comparatively, in (9) eigenvalues, the results of Naive Bayes improved considerably with the same level of Random Forest. As for Multilayer Perceptron and JRip classifiers, most results of metrics decreased to the worst on average. For example, the implementation of Multilayer Perceptron shows an increase in the value of Rc metric, but also shows a decrease in the rest of metrics, as well as in using JRip in which the results of metrics decreased by a small percentage except in Rc metric. Furthermore, reducing eigenvalues to 8 shows worst results compared to (9
eigenvalues), where the highest Accuracy in Random Forest decreased to 99.951%, Pr to 0.9677, F1 to 0.979, but the highest value of Re 0.980 registered at IBK. However, the rest of the metrics show lower results compared to the Random Forest. Because most of the results decreased when (8) eigenvalues tested; it is time to stop at this attempt and not going beyond that. Although the decreased results in some classifiers based on 9 eigenvalues, it is found that the overall best results registered at (9), thus it can be relied upon (9) eigenvalues as suitable selection to detect the Botnet attacks.

In summary, the highest results based on performance metrics pointed at the implementation of Correlation Attribute Eval with JRip classifier in specifying (9) features. As was observed in both feature selection methods, the best results are in the second experiment (9) features in Correlation Attribute Eval and (9) eigenvalues in Principal Component.

In terms of the summary of classifiers performance testing of this study, the best results are obtained in applying Random Forest classifier based on a Principal Component filter and JRip classifier based on Correlation Attribute Eval filter.

5- Conclusion
The experiments of the current paper managed to process the CICIDS2017 dataset by comparing two feature selection methods called (Correlation Attribute Eval and Principal Component) and applied of appropriate classifiers. In each experiment, a specific number of features for Correlation Attribute Eval or eigenvalues for Principal Component identified to obtain the accurate results based on the performance metrics with the target of minimum feature selection considering the best results to detect the Botnet traffic. The best results that conclude the paper obtained during the implementation of Correlation Attribute Eval that is working on a reduction of the dataset dimension by measuring the distance between each feature and a label class to present the features that are closest to the label class.

References
1. Hoque, N., D.K. Bhattacharyya, and J.K. Kalita, Botnet in DDoS attacks: trends and challenges. IEEE Communications Surveys & Tutorials, 2015. 17(4): p. 2242-2270.
2. Alomari, E., et al., Botnet-based distributed denial of service (DDoS) attacks on web servers: classification and art. arXiv preprint arXiv:1208.0403, 2012.
3. Wainwright, P. and H. Kettani, An Analysis of Botnet Models. in Proceedings of the 2019 3rd International Conference on Compute and Data Analysis. 2019.
4. Li, Q., et al. DDoS Attacks Detection using Machine Learning Algorithms. in International Forum on Digital TV and Wireless Multimedia Communications. 2018. Springer.
5. Hoang, X.D. and Q.C. Nguyen, Botnet detection based on machine learning techniques using DNS query data. Future Internet, 2018. 10(5): p. 43.
6. Abdulrahman, A.A. and M.K. Ibrahim, Evaluation of DDoS attacks Detection in a New Intrusion Dataset Based on Classification Algorithms. Iraqi Journal of Information & Communications Technology, 2018. 13(3): p. 49-55.
7. Vilaça, E.S., et al. Botnet traffic detection using RPCA and Mahalanobis Distance. in 2019 Workshop on Communication Networks and Power Systems (WCNPS). 2019. IEEE.
8. Ahmad, T. and M.N. Aziz, Data Preprocessing and Feature Selection for Machine Learning Intrusion Detection Systems. ICIC Express Letter, 2019. 13(2): p. 93-101.
9. Tuan, T.A., et al., Performance evaluation of Botnet DDoS attack detection using machine learning. Evolutionary Intelligence, 2019: p. 1-12.
10. Delplace, A., S. Hermoso, and K. Anandita, Cyber Attack Detection thanks to Machine Learning Algorithms. arXiv preprint arXiv:2001.06309, 2020.
11. IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB, 2020; Available from: https://www.unb.ca/cic/datasets/ids-2017.html.
12. Sharafaldin, I., A.H. Lashkari, and A.A. Ghorbani. A Detailed Analysis of the CICIDS2017 Data Set. in International Conference on Information Systems Security and Privacy, 2018. Springer.
13. Al-Utaibi, K.A. and E.-S.M. El-Alfy, Intrusion detection taxonomy and data preprocessing mechanisms. Journal of Intelligent & Fuzzy Systems, 2018. 34(3): p. 1369-1383.
14. Xiao, Y. and X. Xiao, An Intrusion Detection System Based on a Simplified Residual Network. Information, 2019. 10(11): p. 356.
15. Pandey, A.K. and D.S. Rajpoot, A comparative study of classification techniques by utilizing weka. in 2016 International Conference on Signal Processing and Communication (ICSC). 2016. IEEE.
16. Alam, F.R. and S. Pachauri. Comparative Study of J 48 , Naive Bayes and OneR Classification Technique for Credit Card Fraud Detection using WEKA. 2017.
17. K.Pavya , D.B.S., Feature Selection Techniques in Data Mining: A Study. International Journal of Scientific Development and Research IJSDR, 2017.
18. Gnanambal, S., et al., Classification Algorithms with Attribute Selection: an evaluation study using WEKA. International Journal of Advanced Networking and Applications, 2018. 9(6): p. 3640-3644.
19. Blessie, E.C. and E. Karihikeyan, Sigmis: A feature selection algorithm using correlation based method. Journal of Algorithms & Computational Technology, 2012. 6(3): p. 385-394.
20. Zhang, J., *Machine Learning With Feature Selection Using Principal Component Analysis for Malware Detection: A Case Study*, arXiv preprint arXiv:1902.03639, 2019.

21. Parveen, A.N., H.H. Inbarani, and E.S. Kumar, *Performance analysis of unsupervised feature selection methods*, in 2012 International Conference on Computing, Communication and Applications. 2012, IEEE.

22. Smith, L.I., *A tutorial on principal components analysis*, 2002.

23. Siang, T.C., et al., *A review of cancer classification software for gene expression data*, International Journal of Bio-Science and Bio-Technology, 2015, 7(4): p. 89-108.

24. Parsania, V., N. Bhalodiya, and N. Jani, *Applying Naïve bayes, BayesNet, PART, JRip and OneR Algorithms on Hypothyroid Database for Comparative Analysis*, 2014.

25. Gupta, A., et al., *A comparative study of classification algorithms using data mining: crime and accidents in Denver City the USA*, Education, 2016, 7(7): p. 374-381.

26. Obeidat, I., et al., *Intensive pre-processing of kdd cup 99 for network intrusion classification using machine learning techniques*, 2019.

27. Tharwat, A., *Classification assessment methods*. Applied Computing and Informatics, 2018.