Indonesian Journal of Electrical Engineering and Computer Science
Vol. 21, No. 1, January 2021, pp. 377~390
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v21.i1.pp377-390

BotDetectorFW: an optimized botnet detection framework based on five features-distance measures supported by comparisons of four machine learning classifiers using CICIDS2017 dataset

Aaya F. Jabbar, Imad J. Mohammed
Department of Computer Science, University of Baghdad, Baghdad, Iraq

Article Info

Article history:

Received Feb 29, 2020
Revised Jun 14, 2020
Accepted Aug 8, 2020

Keywords:

BotDetectorFW
Botnet detection
CICIDS2017
Distance measures
Feature selection
Machine learning
WEKA

ABSTRACT

A Botnet is one of many attacks that can execute malicious tasks and develop continuously. Therefore, current research introduces a comparison framework, called BotDetectorFW, with classification and complexity improvements for the detection of Botnet attack using CICIDS2017 dataset. It is a free online dataset consist of several attacks with high-dimensions features. The process of feature selection is a significant step to obtain the least features by eliminating irrelated features and consequently reduces the detection time. This process implemented inside BotDetectorFW using two steps; data clustering and five distance measure formulas (cosine, dice, driver & kroeber, overlap, and pearson correlation) using C#, followed by selecting the best N features used as input into four classifier algorithms evaluated using machine learning (WEKA); multilayerperceptron, JRip, IBK, and random forest. In BotDetectorFW, the thoughtful and diligent cleaning of the dataset within the preprocessing stage beside the normalization, binary clustering of its features, followed by the adapting of feature selection based on suitable feature distance techniques, and finalized by testing of selected classification algorithms. All together contributed in satisfying the high-performance metrics using fewer features number (8 features as a minimum) compared to and outperforms other methods found in the literature that adopted (10 features or higher) using the same dataset. Furthermore, the results and performance evaluation of BotDetectorFM shows a competitive impact in terms of classification accuracy (ACC), precision (Pr), recall (Rc), and f-measure (F1) metrics.

Corresponding Author:

Aaya F. Jabbar
Department of Computer Science
University of Baghdad, Baghdad, Iraq
Email: aayafadhil@gmail.com

1. INTRODUCTION

A botnet is the Internet-connected devices controlled by a Botnet owner to perform malware tasks such as to send spam, steal data, and launch DDoS attacks. There are different tools for botnet attacks like grum, windigo, storm, and ares [1]. A scenario of botnet could compose of some infected “bots” and controllers (one or more, but often just a single controller). A bot is a host that has infected that controlled remotely by a bot herder (botmaster). Meanwhile, a botmaster uses a command & control server to collect information about the bots and issue commands to one, some, or all of it simultaneously [2]. Figure 1 displays a simplified view of a botnet. In the used Botnet dataset [3], they exercise ares, which is a Pythonbased Botnet were attacker uses a Kali Linux and the victims are five different Windows OS, namely Vista, 7, 8.1 and 10. To secure devices from botnet attacks, it must develop several detection models based on previously recorded botnet data.

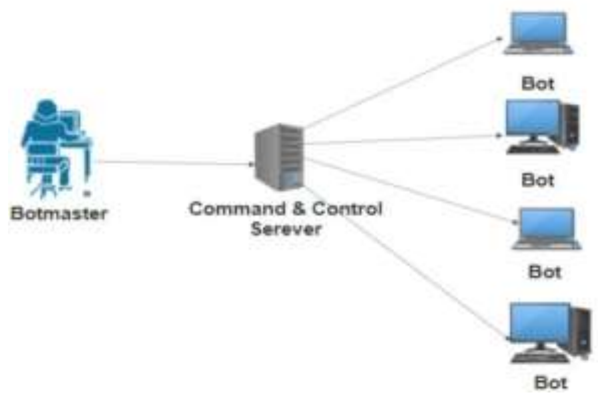


Figure 1. Botnet topology

As a paper organization, the next sections appear in a sequence to discuss; the investigation of related work, Dataset under testing, data preprocessing with cleaning and normalization processes, the proposed feature selection method (clustering and features-distance measures, classifiers and machine learning, results of comparisons with discussion, and finally conducted by a conclusion and references sections).

The authors in [4] proposed a CFS-BA algorithm to discard irrelevant features and the ensemble classifier that combined between ForestPA, RF, and C4.5 with the AOP rule used to construct the classification model. The proposed IDS evaluated using three datasets: NSL-KDD, KDDCup'99, and CICIDS2017. The Acc, Precision, AD, F-Measure, FAR, and ADR metrics used to compare the ensemble classifier and individual classifier performances. The accuracies of the proposed system reached 97% in KDDCup'99, 99% in NSL-KDD, and 96% in CICIDS2017 and the number of features using this system is reducing from 41 to 10 in NSL-KDD, from 41 to 12 in KDDCup'99, and 84 to 13 in CICIDS2017.

The authors in [5], performed their experiment based on different attacks using CICIDS2017 dataset separately. Since it contains high number of features, the dataset was cleaned up from unwanted features using a PCA method targeting the accurate selection of features. The PCA method evaluated using three well-known classifiers (KNN, C4.5, and NaiveBayes) to measure the true detection and false alarm rates. Specially, the number of best features in the botnet attack had been reduced into 23 features with the detection rate up to 98.8%. The aim from [6] was to examine incorporating auto-encoder AE and PCA (namely UBD) for dimensionality reduction and the use of classifiers (RF, NB, LDA, and QDA) towards designing an efficient intrusion detection system within CICIDS2017 dataset. The experimental analysis confirmed the significant results using UBD method and RF classifier to select the best10 features, where the accuracy rate reached 99% for multi-class dataset while the terms of true positive rate (TPR), false positive rate (FPR), Recall and Precision for the binary-class dataset (special botnet) reached 1.000.

In the paper [7], the author contributed in the improvement of AdaBoost classifier performance using CICIDS2017. It evaluated the SMOTE technique, PCA and ensemble feature selection (EFS) for features reduction. Furthermore, it improved the AdaBoost classifier, where the accuracy of 81.83%, a precision of 0.81, recall of 1, and F1 score of 0.901 with 25 features. In the study [8], the learning model developed by using deep learning-DMLP. The model applied using DDOS dataset found in CICIDS2017. The Recursive Feature elimination method and Random Forest used to get the best fewer features of the dataset. The most important features (10) tested using DMLP model and achieved accuracy up to 89%. In [9] the authors evaluated their experiment using machine learning classifiers to detect botnet traffic of CICIDS2017 dataset. The Botnet dataset divided into three training and two testing sets. In this work, five classifiers: IBk, J48, NaïveBayes, OneR, and Random Forest of WEKA tested. J48 decision tree was evaluated as the overall best when encompassing all 78 features without dimensionality reduction for both test sets. Their higher accuracy detection rate reached 98%.

In the paper [10] proposed a two-phase hybrid method based on two different feature selection techniques with recurrent neural network (RNN) and support vector machine (SVM) to get a few sets of features that improve the detection performance and reduce the computational time. The first phase combined joint mutual information maximization (JMIM) with RNN and the second phase combined correlation with SVM. The proposed system carried using two datasets: NSL-KDD dataset (consists of two training sets: KDDTrain+and KDDTrain_20%, and two testing sets: KDDTest+and KDDTest-21) and Kyoto2006+dataset. The system performance evaluated using metrics such as false alarm rate (FAR), recall, precision, detection rate (DR), F-Score and Accuracy. The results of testing the proposed system for KDDTest+, FAR of 0.0085%, recall of 97.7557%, precision of 97.2655%, f-score of 96.5025% and accuracy of 98.9256%, compared to KDDTest-21 results which are; FAR of 0.0076%, recall of 96.1749%, precision of 97.3321%, f-score of 97.0041% and accuracy of 98.9749%, and for Kyoto2006+; FAR of 0.0068%, recall of 99.2199%, precision of 95.5998%, f-score of 96.97.7879% and Accuracy of 97.9443%.

The study in [11] proposed a Restricted Growing SOM method with clustering reference vector (RGSOM-CRV) and Parallel RGSOM-CRV to improve the efficiency of attack classification in KDD Cup 1999 dataset and evaluated using metrics: accuracy (ACC), false alarm rate (FAR), detection rate (DTR) or recall, and precision. Parallel RGSOM-CRV method outperformed the regular GSOM, as it reached up to 91.86% ACC, 20.58% FAR, 95.32% DTR or Recall, and Precision up to 94.35%. The authors of paper [12] proposed a genetic algorithm (GA) as a tool that able to identify harmful types of connections in a computer network, as it analyzed features of connection data and types of connection in the network to generate a set of classification rules. The proposed method uses the combination of genetic operators which are cloning, crossover, and mutation processes to generate new chromosomes. Fitness value indicated the quality of a chromosome (candidate solution) that can detect a set of predetermined attack connection of data during the training process. The proposed method applied using KDD Cup 99 dataset. From the results obtained, it indicated that the average of the success rate and the probability value were directly proportionate, as the high rate for an average of the success was 99.9825% for 0.5 probability value.

Paper [13] proposed a method to analyze and identify the characteristic of a botnet traffic behavior in P2P environment based on the UDP protocol using network traffic analysis tools such as the botnet detection strategy based on the signature, DNS anomaly approach. In signature approach, it showed one of the well-known botnet name conficker that classified in the network-based as NetBIOS attack. DNS anomaly approach used to analyze the behavior of the DNS to define the characteristic of the network botnet. The identified anomalies are DNS packet request, anomalous DNS MX query, the NetBIOS attack, UDP flood attack and DNS amplification attack. It used DNS packets in anomalous network traffic as an indicator for the presence of the botnet, then compared the normal and anomalous traffic to analyze the DNS protocol for identification of the DNS amplification attack, UDP flood attack and spambot activities (which was defined when there is an anomalous DNS MX query in the network).

2. THE PROPOSED FEATURE SELECTION METHODS OF BOTDERTECTORFM

Extra features can cause an increase in the computation time; also can add negative impact to an accuracy of the detection system. Therefore; it is advisable to reduce the number of the features in a dataset using a suitable feature selection technique that searches for the best set of features that could optimize a classification of data [14]. In literature, feature selection techniques are classed into filter [15], wrapper [16], and embedded techniques [17]. Filter methods evaluate the effectiveness of selected features, separately from learning methods, while wrapper methods require learning methods to evaluate the quality of detection system.

Comparatively, embedded methods execute feature selection as one of the components during the process of model construction [4, 18]. The feature selection technique of our proposed framework, BotDetectorFM, uses the embedded type. CICIDS2017 contains features that have been recorded while acquiring data flow, those features are related to a specific network and don't have any impact on model results. One of the dataset preprocessing approaches is data dimension reduction adopted inside BotDetectorFM by removing all those meaningless features. Among useless features, we validate four nominal ones: Flow ID, source IP, destination IP, and timestamp by removing them as done in [5]. Figure 2 depicts the main components and inner flow relationships of BotDetectorFM.

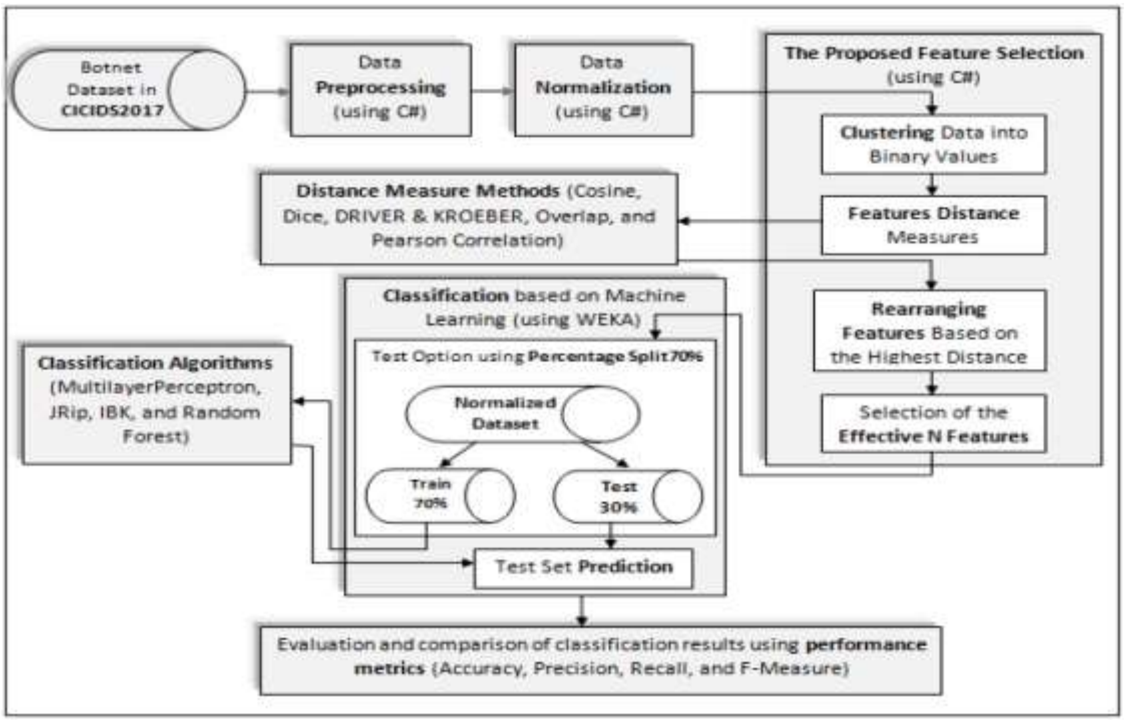

Botnet Dataset in CICIDS2017
Data Preprocessing (using C#)
Data Normalization (using C#)
The Proposed Feature Selection (using C#)
Clustering Data into Binary Values
Features Distance Measures
Distance Measure Methods (Cosine, Dice, DRIVER & KROEBER, Overlap, and Pearson Correlation)
Rearranging Features Based on the Highest Distance
Selection of the Effective N Features
Classification based on Machine Learning (using WEKA)
Test Option using Percentage Split 70%
Normalized Dataset
Train 70%
Test 30%
Test Set Prediction
Classification Algorithms (MultilayerPerceptron, JRip, IBK, and Random Forest)
Evaluation and comparison of classification results using performance metrics (Accuracy, Precision, Recall, and F-Measure)

Figure 2. The proposed Botnet detection framework (BotDetectorFM using CICIDS2017 Dataset)

3. RESEARCH METHOD

3.1. Dataset

CICIDS2017 dataset, the current paper focus, includes benign and attacks traffic [19]. It contains the analysis results of network traffic using CICFlowMeter and labelled flow based on timestamp, IP with port for source, IP with Port for destination, protocols, and attack. The data capturing period began from Monday, July_3_2017 to Friday July_7_2017, for a total of five days. The attacks in this dataset include Heartbleed, DoS, infiltration, brute force SSH, brute force FTP, DDoS, web attack, and Botnet. Other datasets of IDS separated the training from the testing dataset, but CICIDS2017 gathered all records of each attack to CSV file [3]. This dataset contains 85 network flow features. The definition of extracted features is available in Table 2. In current work, the used dataset from CICIDS2017 which is related to the botnet dataset traffic that describes in Table 1.

Table 1. The details of botnet dataset

Dataset Name	CICIDS2017
CSV File Used	Friday-WorkingHours-Morning.pcap_ISCX
Year Of Release	2017
Total Number Of Instances	191033
Number Of Attributes Used in This Paper	85
Number Of Class	2 (BENIGN And BOT)

Table 2. The explanation names of features in CICIDS2017 Dataset

No.	Feature Name	No.	Feature Name	No.	Feature Name
1	Flow ID	29	Fwd IAT Std	57	ECE Flag Count
2	Source IP	30	Fwd IAT Max	58	Down/Up Ratio
3	Source Port	31	Fwd IAT Min	59	Average Packet Size
4	Destination IP	32	Bwd IAT Total	60	AvgFwd Segment Size
5	Destination Port	33	Bwd IAT Mean	61	AvgBwd Segment Size
6	Protocol	34	Bwd IAT Std	62	FwdAvg Bytes/Bulk
7	Time stamp	35	Bwd IAT Max	63	FwdAvg Packets/Bulk
8	Flow Duration	36	Bwd IAT Min	64	FwdAvg Bulk Rate
9	Total Fwd Packets	37	Fwd PSH Flags	65	BwdAvg Bytes/Bulk
10	Total Backward Packets	38	Bwd PSH Flags	66	BwdAvg Packets/Bulk
11	Total Length of FwdPck	39	Fwd URG Flags	67	BwdAvg Bulk Rate

No.	Feature Name	No.	Feature Name	No.	Feature Name
12	Total Length of BwdPck	40	Bwd URG Flags	68	SubflowFwd Packets
13	Fwd Packet Length Max	41	Fwd Header Length	69	SubflowFwd Bytes
14	Fwd Packet Length Min	42	Bwd Header Length	70	SubflowBwd Packets
15	FwdPck Length Mean	43	Fwd Packets/s	71	SubflowBwd Bytes
16	Fwd Packet Length Std	44	Bwd Packets/s	72	Init_Win_bytes_fwd
17	Bwd Packet Length Max	45	Min Packet Length	73	Act_data_pkt_fwd
18	Bwd Packet Length Min	46	Max Packet Length	74	Min_seg_size_fwd
19	Bwd Packet Length Mean	47	Packet Length Mean	75	Active Mean
20	Bwd Packet Length Std	48	Packet Length Std	76	Active Std
21	Flow Bytes/s	49	Packet Len. Variance	77	Active Max
22	Flow Packets/s	50	FIN Flag Count	78	Active Min
23	Flow IAT Mean	51	SYN Flag Count	79	Idle Mean
24	Flow IAT Std	52	RST Flag Count	80	Idle Packet
25	Flow IAT Max	53	PSH Flag Count	81	Idle Std
26	Flow IAT Min	54	ACK Flag Count	82	Idle Max
27	Fwd IAT Total	55	URG Flag Count	83	Idle Min
28	Fwd IAT Mean	56	CWE Flag Count	84	Label

3.2. Data preprocessing

Realistic data typically takes from heterogeneous platforms and may be redundant, incomplete, and inconsistent [20]. Thus, it requires a preprocessing step that converts data into a suitable format for analysis and discovery [21]. In this work, the preprocessing step includes cleaning data from outliers, redundant, and data transforming [4]. Also, before the using of a dataset for detection model evaluation, it has been necessary to clean up the dataset from errors that could occur while flow data are acquiring [5]. In general, preprocessing consumes necessary time and an essential step for the detection system.

3.2.1. Removing redundant attributes

In the first of preprocessing, the CICIDS2017dataset contains 85 attributes but must check if there are redundant attributes [22]. Therefore any redundant attribute must be removed to as a requirement of the accurate model analysis (ex. 'Fwd Header Length' that appeared twice in the list of attributes in the number of an attribute (41) and (62), removing one of them). The number of features after that become 84 [5].

3.2.2. Transforming of missing and infinity values

This raw data of CICIDS2017 contains anomalous instances, which may influence the performance of the detection system taking into consideration that some detection methods do not accept these types of values [23]. Thus, replacing them by other values could be a solution such as missing values that can be replaced by minimum and infinite values by a maximum of their attribute values. For example, the feature 'Flow Packets/s' in the Botnet dataset includes abnormal values as 'Infinity' and 'NaN' [4].

3.2.3. Transforming data

CICIDS2017 contains nominal attributes. As many classifiers do not accept nominal values, the transforming process is vital and has an impact on detection system accuracy [24]. Thus, it is necessary to replace every single value in a nominal attribute with an integer to handle the symbolic values. For example, transforming the IP source and destination, flow ID, and timestamp into an integer representation is adopted. Besides, the label attribute contains two nominal values; bot and benign that can transforme into binary numeric values such as 0 instead of benign value and 1 instead of Bot value. Also, normalized methods used to transform all attributes value into the same range [25]. Minmax is one of the normalization strategies which transform dataset values from range to another in each attribute [6].

$$Y=\frac{X-Xmin}{Xmax-Xmin} \tag{1}$$

Where X is the set of dataset values of x, Xmin and Xmax are minimum value and maximum value of x attribute values. The new range of data is 0–1 range.

3.3. Clustering data

The clustering data process converts or normalizes the values of the features into 0 to 1 range based on means (M) and standard deviation (STD) values per feature. Algorithm 1 outlines the steps of the clustering process.

Clustering data algorithm of BotDetectorFM:

```
Input: DS (Dataset matrix), R (No. of instances), F (Index of feature)
Output: DS_Clustering (Binary Dataset matrix that consist of 0 and 1 values)
-----------------------------------------------------------------------------------------------------
 1:   // Mean and STD calculation
2:  Calculate Mean (M) for each feature (F)
3:  Calculate Standard Deviation (STD) for each feature (F)
4:
5:   Calculate cluster1=M+STD//addition of mean and standard deviation
6:   Calculate cluster2=M-STD//difference between mean and standard deviation
7:
8:   //Clustering per Feature values Loop
9:   j=0        //j is a counter for No. of instances (R) in each feature (F)
10:  While j<R (stopping condition)
11:     Calculate value1=DS [F, j]-cluster1
12:    Calculate value2=DS [F, j]-cluster2
13:  if value1<value2
14:     DS_Clustering [F, j]=1
15:  else
16:    DS_Clustering [F, j]=0
17:  j++
18: Repeat step (10) until the stopping condition is reached
```

3.4. Distance measure

It takes the results of binary clustering, the normalized data, as input and evaluates the distance between the feature and label class. In general, the domain of distance measures depicts different formulas for measuring the distance between numerical vectors of a similar length. BotDetectorFM tries to open or head the investigation of five formulas on CICIDS2017, experiences them on the produced normalized CICIDS2017 that may optimize the performances of the classifiers consequently the overall botnet detection process. Let us assume there are two real-valued X and Y vectors such as X=[x1, x2… xn] and Y=[y1, y2… yn], where X is any feature values, while Y is a label class values. Also, the distance measure as a real number D is assumed. Therefore, it is common for D values to be within 1<=D<=0. The value of distance increases as D approaches 1, and decreases as D approaches 0 [26, 27]. The proposed distance measures for CICIDS2017 evaluated according to the following formulae:

– Cosine Measure [28]:

$$\text{DCosine (X, Y)}=\frac{\sum_{i=0}^{n} Xi*Yi}{\sqrt{\sum_{i=0}^{n}(Xi)^2} * \sqrt{\sum_{i=0}^{n}(Yi)^2}} \tag{2}$$

– Dice Measure [29]:

$$\text{DDice (X, Y)}=\frac{2*\sum_{i=0}^{n} Xi*Yj}{\sum_{i=0}^{n}(Xi)^2 + \sum_{i=0}^{n}(Yj)^2} \tag{3}$$

– Driver & Kroeber measurer [11]:

$$\text{DDRIVER \& KROEBER (X, Y)}=\frac{\sum_{i=0}^{n} Xi*Yi}{2*\sqrt{\sum_{i=0}^{n}(Xi)^2}} + \frac{\sum_{i=0}^{n} Xi*Yi}{2*\sqrt{\sum_{i=0}^{n}(Yi)^2}} \tag{4}$$

– Overlap measure [30]:

$$\text{DOverlap (X, Y)}=\frac{\sum_{i=0}^{n} Xi*Yi}{\min(\sum_{i=0}^{n}(Xi)^2 ,\sum_{i=0}^{n}(Yi)^2)} \tag{5}$$

– Pearson correlation measure [31]:

$$\text{DPearson\_correlastion (X, Y)}=\frac{\left(N*\sum_{i=0}^{n} Xi*Yi\right)-(\sum_{i=0}^{n} Xi,j*\sum_{i=0}^{n} Yi)}{\sqrt{\left[\left(N*\sum_{j=0}^{n}(Xi)^2\right)-(\sum_{j=0}^{n} Xi)^2\right]*[(N*\sum_{i=0}^{n}(Yi)^2)-(\sum_{j=0}^{n} Yi)^2]}} \tag{6}$$

Where i-item is a counter for the number of features (N). After calculating the distance for each feature, the features rearranged relative to the highest distance value. Thus, the important features advance at the highestranking and the unrelated features at the least-ranking. Then, retrieving the original values for features (the original values are the values before applying feature selection steps on it). The best distance measure is equivalent to the highest performance measures characterized by the least selected features.

3.5. Machine learning

It is techniques or tools that are gaining popularity not only in terms of unknown and known malware detection but also learning from the environment, which may detect attacks [32]. WEKA is one of these tools written in Java. It is software freely accessible. It contains tools for several tasks such as clustering, classification, association rules ...etc, in addition; tools for analyzing the learning results [33]. The current paper uses WEKA 3.9 version as the testing environment of classifiers. Test option used in all techniques such that the percentage split equal to 70%, this option divides the dataset into a train set 70% and test set 30%. The classification module of BotDetectorFM investigates five classifiers known as random forest within the decision trees algorithms [34, 35], IBK within lazy algorithms [35], JRIP within rules algorithms [34, 36], and MultilayerPerceptron within functions algorithms [33, 37].

3.6. Performance evaluation metrics

Different metrics designed to measure the efficiency of the Detection system. Typically, these metrics measured from a confusion matrix perspective. In BotDetectorFM, a confusion matrix used to represent the results of the detection model. It handled as an analysis tool to measure whether the classifier is good in recognizing the instances of different classes [38]. Table 3 identifies the confusion matrix.

Table 3. Confusion matrix

Actual	Perdicted	
	Attack	Normal
Attack	TP	FP
Normal	FN	TN

The following are basic terms to classify events that depicted in Table 3 [39]:
- TP (True Positive): The value obtained from intersecting positive actual and positive predictive values.
- FP (False Positive): The value obtained from intersecting negative actual and positive predictive values.
- FN (False Negative): The value obtained from intersecting actual positive and negative predictive values.
- TN (True Negative): value obtained from intersecting negative actual and predictive negative values.

The values of TP and TN provide information when the classifier of the data is true, while FP and FN provide information when the classifier is wrong in classifying the data [40].

3.7. Metrics from the confusion matrix

Many performance metrics defined by confusion matrix variables. Thus, these metrics produce numeric values that make simply comparable. This study uses some performance metrics to appraise the performance of the detection system, including precision (PR), recall (Rc), f-measure (F1), and accuracy (ACC). The definitions of these metrics are provided below [39]:
- Accuracy (ACC): It defined as the ratios measure of the correctly classified an object as either normal or attack. Accuracy calculates using (7).

$$\text{ACC}=\frac{TP+TN}{TP+FP+TN+FN} \tag{7}$$

- Precision (PR): It is the fragment of data instances predicted as positive that are positive precision represents the ratio between positive predictions and all number of positive values. Precision can be defined using (8)

$$\text{Pr}=\frac{TP}{TP+FP} \tag{8}$$

- Recall (Rc) or sensitivity: is the system's ability to detect all existing attacks. Recall can calculate from the number of detected intrusions using the system on all of the actual intrusions. This metric is equivalent to the detection rate. Recall can be defined using (9).

$$\text{Rc}=\frac{TP}{TP+FN} \tag{9}$$

– F-Measure (F1): It is a harmonic combination of precision (Pr) and recall (Rc) into a single measure. F-Measure calculated using (10).

$$F1=\frac{2}{\frac{1}{Pr}+\frac{1}{Rc}} \tag{10}$$

4. RESULTS AND DISCUSSION

As mentioned earlier, the objective is to reduce the data dimension of CICIDS2017 that expected to impact the botnet detection process positively. The selected features are the highest ten features determined from each feature selection method considered as a set of optimal features Table 4.

Table 4. Show higher ten features order from each distance measure

Distance Measure	Highest 10 Selected Features
Cosine	Destination Port, PSH Flag Count, Init_Win_bytes_forward, URG Flag Count, Fwd Packet, Length Std, Fwd Packet Length Max, Down/Up Ratio, ACK Flag Count, Bwd Packets/s, Flow Packets/s
Dice	Destination Port, URG Flag Count, PSH Flag Count, Init_Win_bytes_forward, Flow Packets/s, Fwd Packet Length Std, Fwd Packet Length Max, ACK Flag Count, Fwd Packets/s, Bwd Packets/s
DRIVER&KROEBER	**Destination Port, Down/Up Ratio, Init_Win_bytes_forward, PSH Flag Count, URG Flag Count, Source Port, Fwd Packet Length Std, Fwd Packet Length Max**, ACK Flag Count, Bwd Packets/s
Overlap	**Destination Port, Down/Up Ratio, Init_Win_bytes_forward, PSH Flag Count, Source Port, ACK Flag Count, Fwd Packet Length Std, Fwd Packet Length Max**, Bwd Packets/s, URG Flag Count
Pearson Correlation	Destination Port, PSH Flag Count, URG Flag Count, Init_Win_bytes_forward, Fwd Packet Length Std, Flow Packets/s, Fwd Packet Length Max, ACK Flag Count, Bwd Packets/s, Fwd Packets/s

As noted in the table above, most of the ten selected features are the same in all distance measures, but they differ in a few. In addition, the order of features is different in all measures. This also may affect the results when applying some classification algorithms. Then testing the optimal features displayed in Table 1 using classification algorithms. The implementation results of classification algorithms are displayed in Tables 5, 6, and 7, which consider a total of 10, 9, and 8 features respectively. The results are obtained using a test set. Bold values exhibit the highest result values obtained by applying some classifier models. Figures 3, 4, and 5 exhibit performance comparisons of classifiers. The displayed bars in blue, red, green and purple colours represent the accuracy of MLP, IBk, JRip, and random forest classifiers respectively.

Table 5. The results of classification algorithms for test set based on the highest ten (10) selected features

Distance Measure	Classification Algorithm	ACC	Pr	Rc	F1
Cosine	MultilayerPerceptron	99.5899 %	0.978	0.614	0.754
	JRip	99.9564 %	0.972	0.986	0.979
	IBK	99.8238 %	0.893	0.940	0.916
	Random Forest	99.9651 %	0.988	0.978	0.983
Dice	MultilayerPerceptron	99.5934 %	0.971	0.622	0.759
	JRip	99.9634 %	0.981	0.983	0.982
	IBK	99.8168 %	0.888	0.940	0.913
	Random Forest	99.9651 %	0.990	0.976	0.983
DRIVER & KROEBER	MultilayerPerceptron	99.8377 %	0.996	0.845	0.917
	JRip	**100 %**	**1.000**	**1.000**	**1.000**
	IBK	99.9948 %	0.998	0.997	0.997
	Random Forest	**100 %**	**1.000**	**1.000**	**1.000**
Overlap	MultilayerPerceptron	99.8447 %	0.994	0.854	0.919
	JRip	**100 %**	**1.000**	**1.000**	**1.000**
	IBK	99.9948 %	0.998	0.997	0.997
	Random Forest	**100 %**	**1.000**	**1.000**	**1.000**
Pearson Correlation	MultilayerPerceptron	99.6039 %	0.997	0.616	0.761
	JRip	99.9616 %	0.978	0.985	0.81
	IBK	99.8168 %	0.888	0.940	0.913
	Random Forest	99.9668 %	0.990	0.978	0.984

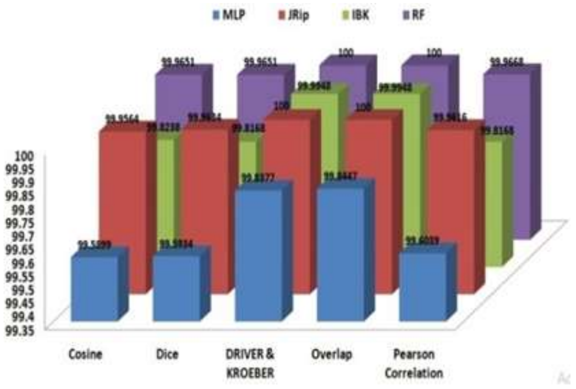
Table 6. The results of classification algorithms for test set based on the highest nine (9) selected features

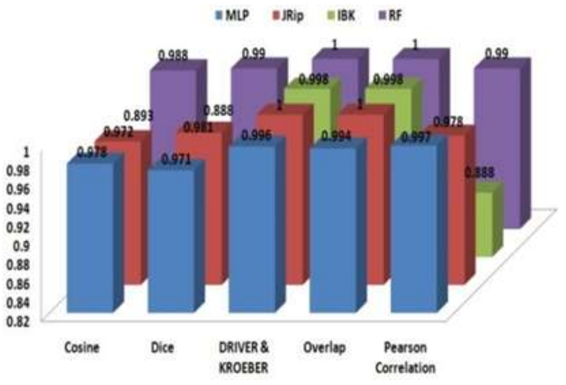
Distance Measure	Classification Algorithm	ACC	Pr	Rc	F1
Cosine	MultilayerPerceptron	99.6039 %	0.997	0.616	0.761
	JRip	99.9651 %	0.980	0.986	0.983
	IBK	99.8342 %	0.903	0.939	0.921
	Random Forest	99.9791 %	0.991	0.988	0.990
Dice	MultilayerPerceptron	99.6022 %	0.995	0.616	0.761
	JRip	99.9581 %	0.975	0.985	0.980
	IBK	99.829 %	0.896	0.942	0.919
	Random Forest	99.9721 %	0.988	0.985	0.986
DRIVER & KROEBER	MultilayerPerceptron	99.7313 %	0.989	0.747	0.851
	JRip	**100%**	**1.000**	**1.000**	**1.000**
	IBK	99.9965 %	0.998	0.998	0.998
	Random Forest	**100%**	**1.000**	**1.000**	**1.000**
Overlap	MultilayerPerceptron	99.836 %	0.996	0.844	0.913
	JRip	**100%**	**1.000**	**1.000**	**1.000**
	IBK	99.9916 %	0.995	0.997	0.996
	Random Forest	**100%**	**1.000**	**1.000**	**1.000**
Pearson Correlation	MultilayerPerceptron	99.6022 %	0.995	0.616	0.761
	JRip	99.9511 %	0.971	0.981	0.976
	IBK	99.8098 %	0.883	0.939	0.910
	Random Forest	99.9721 %	0.990	0.983	0.986

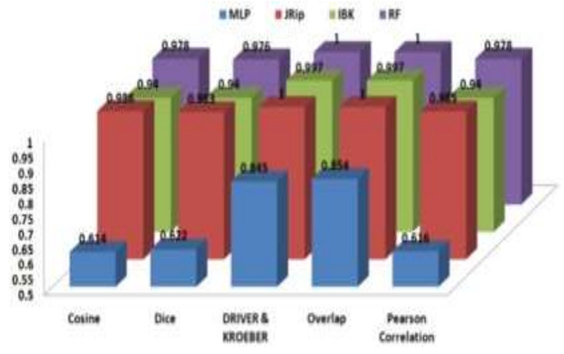
Table 7. The results of classification algorithms for test set based on the highest eight (8) selected features

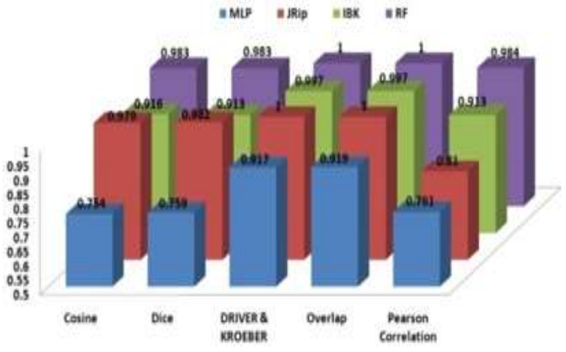
Distance Measure	Classification Algorithm	ACC	Pr	Rc	F1
Cosine	MultilayerPerceptron	99.6039 %	0.997	0.616	0.761
	JRip	99.9686 %	0.975	0.995	0.985
	IBK	99.9424 %	0.966	0.978	0.972
	Random Forest	99.9686 %	0.985	0.985	0.985
Dice	MultilayerPerceptron	99.6022 %	0.995	0.616	0.761
	JRip	99.9651 %	0.981	0.985	0.983
	IBK	99.815 %	0.891	0.934	0.912
	Random Forest	99.9756 %	0.988	0.988	0.988
DRIVER & KROEBER	MultilayerPerceptron	99.8447 %	0.994	0.854	0.919
	JRip	**100%**	**1.000**	**1.000**	**1.000**
	IBK	99.9965 %	0.998	0.998	0.998
	Random Forest	**100%**	**1.000**	**1.000**	**1.000**
Overlap	MultilayerPerceptron	99.5376 %	0.903	0.616	0.732
	JRip	**100%**	**1.000**	**1.000**	**1.000**
	IBK	99.9965 %	0.998	0.998	0.998
	Random Forest	**100%**	**1.000**	**1.000**	**1.000**
Pearson Correlation	MultilayerPerceptron	99.6039 %	0.997	0.616	0.761
	JRip	99.9564 %	0.973	0.985	0.979
	IBK	99.815 %	0.891	0.934	0.912
	Random Forest	99.9756 %	0.990	0.986	0.988

It noticed that the best distance measures (results) gained from BotDetectorFM when overlap and Driver&Kroeber measures implemented and proved by the performance of classifiers. Both give significant results based on the performance metrics in all classification methods. Especially, random forest and JRip algorithms. As a result and distinction, BotDetectorFM succeeded in reducing the number of best features of CICIDS2017 from ten to eight (8) providing fewer complexities in terms of time and space processing as a botnet detection system compared to the explored previous works. Also, it is critical to point out that the number of significant features cannot be reduced to less than 8 because it may negatively affect the detection performance and its accuracy. Moving further in the discussion of classification results using the performance metrics may show more deep perspectives or indications. First, the accuracy can see in the label (a) on Figure 3, Figure 4, and Figure 5, the highest accuracy (ACC) for all distance measurements is in the RF algorithm followed by the JRip, IBK, and MLP algorithms respectively.


(a) accuracy


(b) precision


(c) recall


(d) f-measure

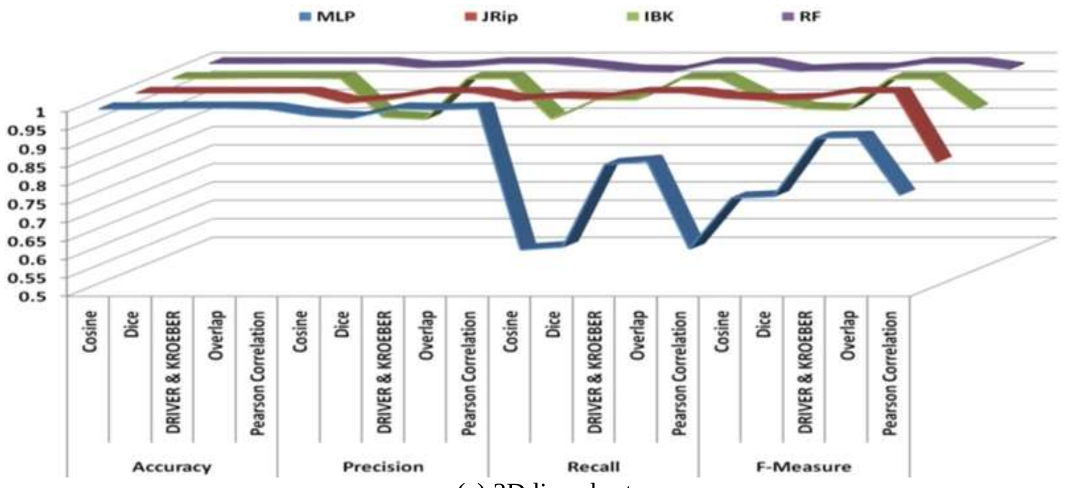
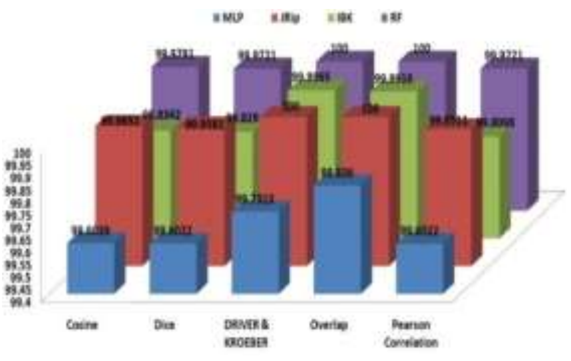
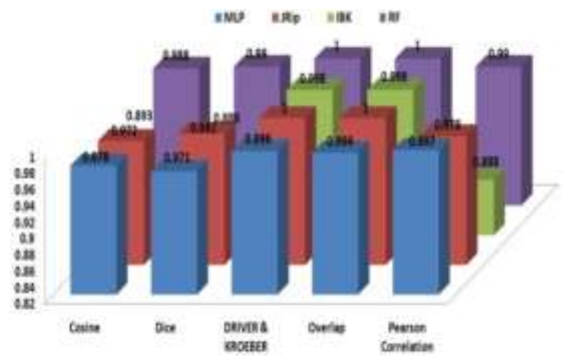

(e) 3D line chart

Figure 3. Comparison of performance metrics with feature selection methods based on the highest Ten selected features

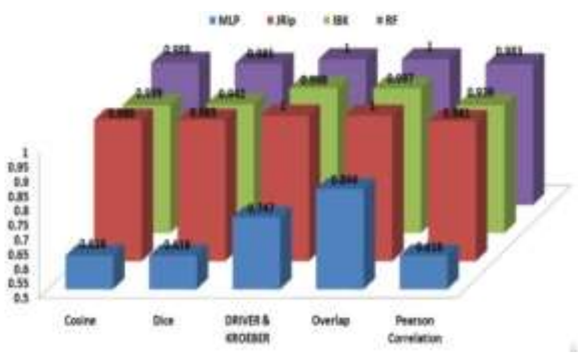
The accuracy rate in overlap and Driver&Kroeber measures reached to 100%. second, precision (Pr) measure can notice in the label (b) on Figure 3, Figure 4, and Figure 5), the highest precision value is in the RF and JRip algorithms for the overlap and Driver&Kroeber distance methods reached to 1 (highest value), but the highest precision value for other distance methods is in the MLP algorithm. This difference in the values of Pr due to its dependency on the TP for botnet instances and FN for benign instances, and since the benign instances exceed the instances of botnet very much, it is the reason for this difference as well as this is one of the problems of this dataset. Thirdly, recall (Rc) metric is shown in the label (c) on Figure 3, Figure 4, and Figure 5. It is the most important measure for botnet detection because it depends only on the instances of a botnet. The highest values for this metric found between the RF and JRip algorithms. The high recall value is in JRip and RF classifiers for overlap and Driver&Kroeber measures is the same value reached to 1 (highest value), but the high recall value for cosine measure is in JRip classifier while for other used distance measures is in RF classifiers. Finally, the f-measure (F1) metric shown in the label (d) of Figure 3, Figure 4, and Figure 5, which is related to Pr and Rc metrics. The highest value of this metric result from two algorithms JRip and RF reached to 1 using overlap and Driver&Kroeber measures. While the highest f-measure value for other distance methods got in the RF algorithm.



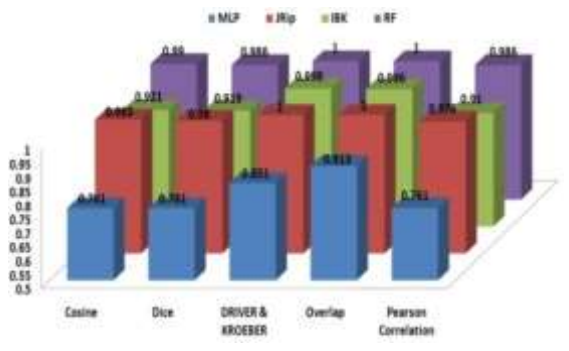
(a) accuracy



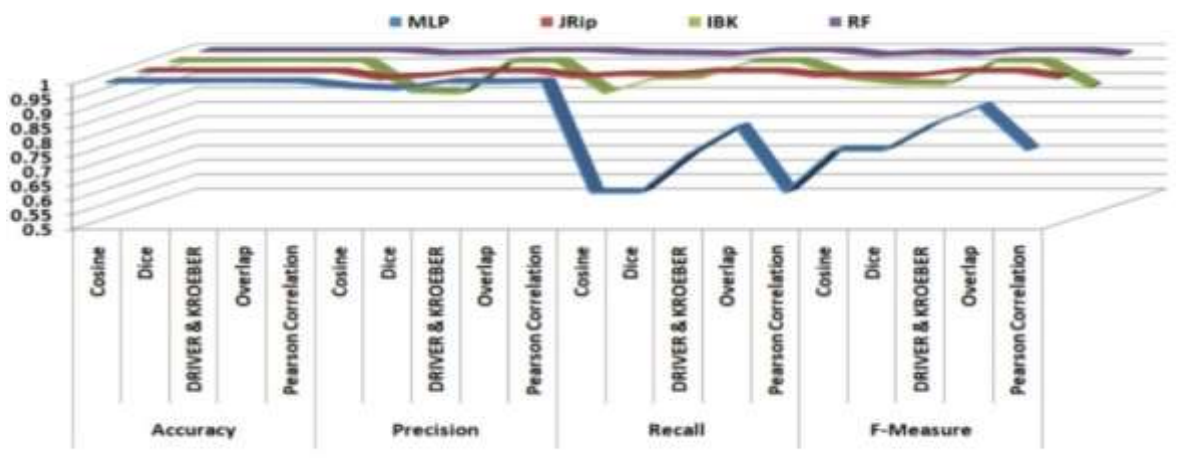
(b) precision



(c) recall



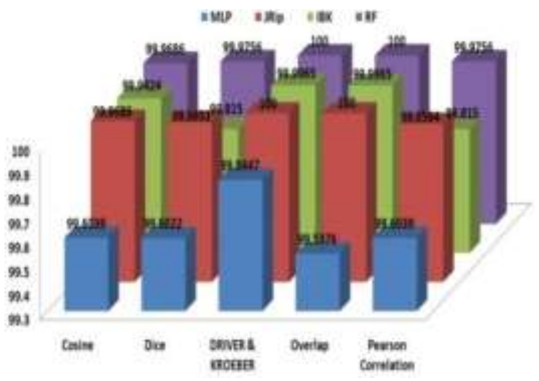
(d) f-measure



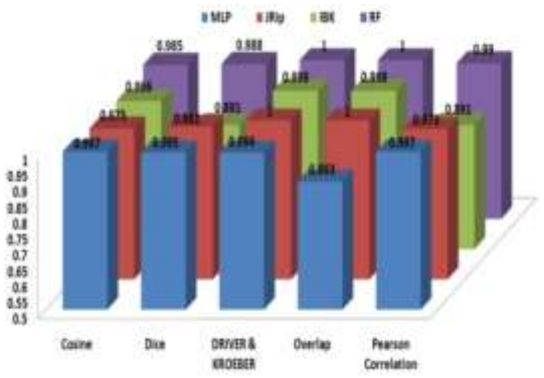
(e) 3D line chart for all metrics

Figure 4. Comparison of performance metrics with feature selection methods based on the highest Nine selected features

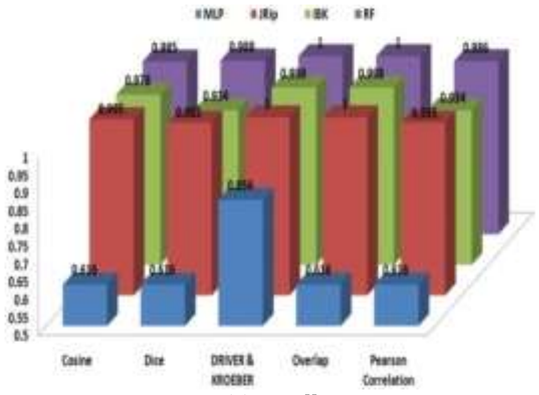
In general, the RF classifier algorithm produces high results for all performance measures, and the best distance measures that rearranged the features to select the best higher N features are overlap and Driver&Kroeber measures. The reason behind obtaining of the best features selection results by applying of overlap and Driver-Kroeber distance measures is that they work (their formulas) or behave proportionally directly with the intersection between feature and the label class. Therefore, if the intersection value becomes large this means the distance ratio is great for this feature. Thus, these methods give large values for features that have a high similarity rate compared to the label class. In terms of classifiers, the overall analysis of BotDetectorFM found that RF and JRip give the highest results because the random forest classifier works as a merge of multiple decision trees (algorithms) participate together to get more flexible, accurate and stable classification results. Also, JRip works well with a large dataset by building models interpretable easily.



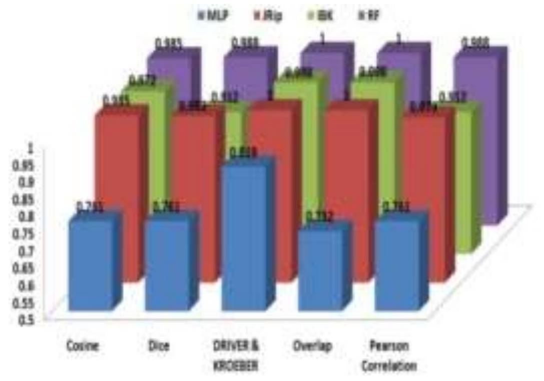
(a) accuracy



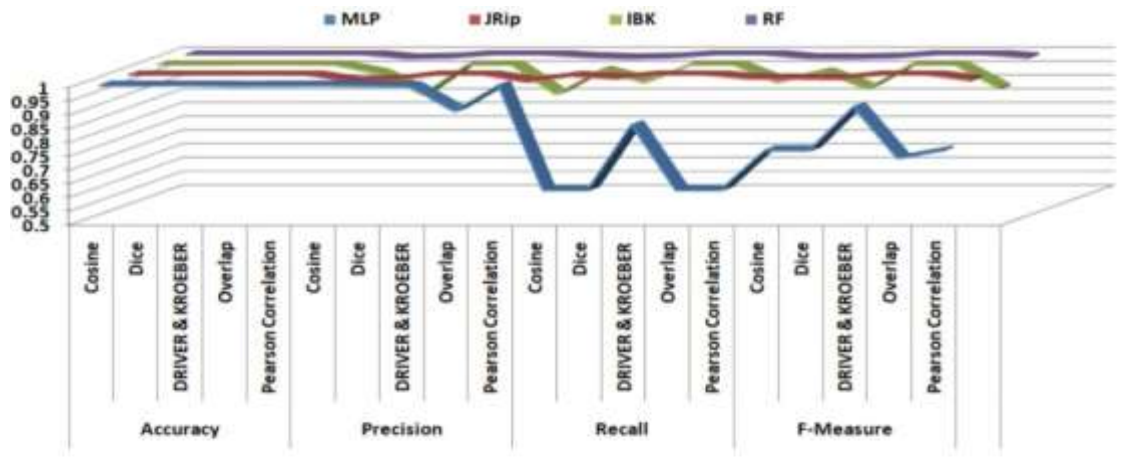
(b) precision



(c) recall



(d) f-measure



(e) 3D Line chart for all metrics

Figure 5. Comparison of performance metrics with feature selection methods based on highest eight selected features

5. CONCLUSION

In summary, by comparison with the tested methods in previous studies on the same dataset CICIDS2017, the proposed framework BotDetectorFM introduces an advance prediction with the fewer number of features (8 significant features only) that diminish the detection complexity of Botnet traffic with high detection accuracy via applying an advanced mixture of perfect features selection and suitable classification algorithms with careful preprocessing stage and advance reduction process of the dataset dimension.

REFERENCES

[1] M. Eslahi, R. Salleh, and N. B. Anuar, "Bots and botnets: An overview of characteristics, detection and challenges," in *2012 IEEE International Conference on Control System, Computing and Engineering*, pp. 349-354, 2012.

[2] G. Vormayr, T. Zseby, and J. Fabini, "Botnet communication patterns," *IEEE Communications Surveys & Tutorials*, vol. 19, pp. 2768-2796, 2017.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Intrusion Detection Evaluation Dataset (CIC-IDS2017)," 2018. Available: https://www.unb.ca/cic/datasets/ids-2017.html

[4] Y.-Y. Zhou and G. Cheng, "An Efficient Network Intrusion Detection System Based on Feature Selection and Ensemble Classifier," *arXiv preprint arXiv:1904.01352*, 2019.

[5] A. Boukhamla and J. C. Gaviro, "CICIDS2017 dataset: performance improvements and validation as a robust intrusion detection system testbed," *International Journal of Information and Computer Security*, Sept. 2018.

[6] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection," *Electronics*, vol. 8, no. 3, pp. 322, 2019.

[7] A. Yulianto, P. Sukarno, and N. A. Suwastika, "Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset," in *Journal of Physics: Conference Series*, vol. 1192, pp. 012018, 2019.

[8] S. Ustebay, Z. Turgut, and M. A. Aydin, "Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier," in *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp. 71-76, 2018.

[9] R. McKay, B. Pendleton, J. Britt, and B. Nakhavanit, "Machine Learning Algorithms on Botnet Traffic: Ensemble and Simple Algorithms," in *Proceedings of the 2019 3rd International Conference on Compute and Data Analysis*, pp. 31-35, 2019.

[10] B. N. Kumar, M. S. B. Raju, and B. V. Vardhan, "A novel approach for selective feature mechanism for two-phase intrusion detection system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, pp. 101-112, 2019.

[11] T. Y. Christyawan, A. A. Supianto, and W. F. Mahmudy, "Anomaly-based intrusion detector system using restricted growing self organizing map," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, pp. 919-926, 2019.

[12] H. Suhaimi, S. I. Suliman, I. Musirin, A. F. Harun, and R. Mohamad, "Network intrusion detection system by using genetic algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, pp. 1593-1599, 2019.

[13] N. Z. M. Safar, N. Abdullah, H. Kamaludin, S. Abd Ishak, and M. R. M. Isa, "Characterising and detection of botnet in P2P network for UDP protocol," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, pp. 1584-1595, 2020.

[14] N. Alias, C. F. M. Foozy, N. R. Ramli, and N. Zainuddin, "Video spam comment features selection using machine learning techniques," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 15, pp. 1046-1053, 2019.

[15] O. Osanaiye, H. Cai, K.-K. R. Choo, A. Dehghantanha, Z. Xu, and M. Dlodlo, "Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 130, pp. 1-10, 2016.

[16] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," computers & security, vol. 70, pp. 255-277, 2017.

[17] Y.-D. Lan, "A Hybrid Feature Selection based on Mutual Information and Genetic Algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 7, pp. 214-225, 2017.

[18] A. F. Alharan, H. K. Fatlawi, and N. S. Ali, "A cluster-based feature selection method for image texture classification," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, pp. 1433-1442, 2019.

[19] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "A Detailed Analysis of the CICIDS2017 Data Set," in *International Conference on Information Systems Security and Privacy*, pp. 172-188, 2018.

[20] M. Jupri and R. Sarno, "Data mining, fuzzy AHP and TOPSIS for optimizing taxpayer supervision," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, pp. 75-87, 2020.

[21] T. A. Runkler, "Data Pre-processing," in *Secondary Analysis of Electronic Health Records*, ed: Springer, pp. 115-141, 2016.

[22] R. Elankavi, R. Kalaiprasath, and D. R. Udayakumar, "A fast clustering algorithm for high-dimensional data," *International Journal of Civil Engineering and Technology (Ijciet)*, vol. 8, pp. 1220-1227, 2017.

[23] A. Fatima, N. Nazir, and M. G. Khan, "Data cleaning in data warehouse: a survey of data pre-processing techniques and tools," *IJ Information Technology and Computer Science*, vol. 3, pp. 50-61, 2017.

[24] S. Samdani and S. Shukla, "A novel technique for converting nominal attributes to numeric attributes for intrusion detection," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1-5, 2017.

[25] S. Jain, S. Shukla, and R. Wadhvani, "Dynamic selection of normalization techniques using data complexity measures," *Expert Systems with Applications*, vol. 106, pp. 252-262, 2018.

[26] M. De Marsico and D. Riccio, "Weighty LBP: a new selection strategy of LBP codes depending on their information content," in *International Conference on Image Analysis and Processing*, pp. 424-434, 2017.

[27] R. Todeschini, V. Consonni, H. Xiang, J. Holliday, M. Buscema, and P. Willett, "Similarity coefficients for binary chemoinformatics data: overview and extended comparison using simulated and real data sets," *Journal of chemical information and modeling*, vol. 52, pp. 2884-2901, 2012.

[28] S. Han, C. Zhao, W. Meng, and C. Li, "Cosine similarity based fingerprinting algorithm in WLAN indoor positioning against device diversity," in *2015 IEEE International Conference on Communications (ICC)*, pp. 2710-2714, 2015.

[29] R. R. Shamir, Y. Duchin, J. Kim, G. Sapiro, and N. Harel, "Continuous dice coefficient: a method for evaluating probabilistic segmentations," *arXiv preprint arXiv:1906.11031*, 2019.

[30] F. B. Allyson, M. L. Danilo, S. M. José and B. C. Giovanni, "Sherlock N-overlap: Invasive Normalization and Overlap Coefficient for the Similarity Analysis Between Source Code," in *IEEE Transactions on Computers*, vol. 68, no. 5, pp. 740-751, 1 May 2019. doi: 10.1109/TC.2018.2881449.

[31] D. A. Walker, "JMASM 48: The Pearson Product-Moment Correlation Coefficient and Adjustment Indices: The Fisher Approximate Unbiased Estimator and the Olkin-Pratt Adjustment (SPSS)," *Journal of Modern Applied Statistical Methods*, vol. 16, pp. 29, 2017.

[32] H. R. Esmaeel, "Analysis of classification learning algorithms," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 17, pp. 1029-1039, 2020.

[33] J. Jabez, S. Gowri, S. Vigneshwari, J. A. Mayan, and S. Srinivasulu, "Anomaly Detection by Using CFS Subset and Neural Network with WEKA Tools," in *Information and Communication Technology for Intelligent Systems*, ed: Springer, pp. 675-682, 2019.

[34] E. Jedari, Z. Wu, R. Rashidzadeh, and M. Saif, "Wi-Fi based indoor location positioning employing random forest classifier," in *2015 international conference on indoor positioning and indoor navigation (IPIN)*, pp. 1-5, 2015.

[35] S. Bharati, M. A. Rahman, and P. Podder, "Breast cancer prediction applying different classification algorithm with comparative analysis using WEKA," in *2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT)*, pp. 581-584, 2018.

[36] F. MA, W. Yassin, and N. H. MS, "Scrutinized System Calls Information Using J48 And Jrip For Malware Behaviour Detection," *Journal of Engineering Science and Technology*, vol. 14, pp. 291-304, 2019.

[37] P. Tiwari, H. Dao, and G. N. Nguyen, "Performance evaluation of lazy, decision tree classifier and multilayer perceptron on traffic accident analysis," *Informatica*, vol. 41, 2017.

[38] I. Düntsch and G. Gediga, "Confusion matrices and rough set data analysis," in *Journal of Physics: Conference Series 1229*, pp. 1-6, 2019.

[39] A. Rácz, D. Bajusz, and K. Héberger, "Multi-Level Comparison of Machine Learning Classifiers and Their Performance Metrics," *Molecules*, vol. 24, p. 2811, 2019.

[40] Y. Jiao and P. Du, "Performance measures in evaluating machine learning based bioinformatics predictors for classifications," *Quantitative Biology*, vol. 4, no. 4, pp. 320-330, 2016.