Efficient Feature Selection for Detecting Botnets based on Network Traffic and Behavior Analysis

Imtiyaz Hossain, Sakib Eshrak, Minhaj Jami Auvik,
Syed Faiz Nasim, and Raqeebir Rab
Department of CSE, AUST. Email:{imtiyaz.aust, eshrak.sea4,
jami.minhaj,faiznasim.aust}@gmail.com, raqeebir.cse@aust.edu

Ashikur Rahman
Department of CSE,
Bangladesh Univ. of Engg. & Tech (BUET)
Dhaka, Bangladesh. Email:ashikur@cse.buet.ac.bd

***Abstract*—Ensuring integrity and security of computer networks is one of the growing concerns. The number of malware specifically designed to damage, disrupt or perform illegitimate actions on data, networks or hosts are increasing day by day. Detection of hosts infected by malware known as bots is the main focus of this paper. While Botnets are an emerging threat with hundreds of millions of computers infected, the research and solutions of it are still in their infancy stage. In this paper, at first we propose a feature selection algorithm to reduce extracted features from network flows. The selected features are lately analyzed using a supervised machine learning technique to effectively detect the presence of botnets. The experimental evaluation based on a versatile existing data set shows that the proposed model is able to effectively detect bots with more accuracy and high detection rate with moderate false alarms in the botnet's Command and Control (C&C) phase.**

I. Introduction

The botnet refers to a collection of bots (computers) controlled by a bot-master (attacker). Botnets pose major security threats such as accessing private sensitive information or denying legitimate users from accessing network resources by launching (distributed) Denial of Service (DoS) attacks. A recent study shows that (surprisingly) about 40% of computers connected to the Internet are infected by bots and controlled by bot-masters [7].

Botnet detection is a promising research topic under *Intrusion Detection* domain that has attracted many researchers from both academia and industry over the past decade. Effective botnet detection requires continuous tracking of a network and enabling machines to "read" unexpected traffic behaviour. Although each botnet exhibits specific behavior, still there exist adequate similarities in their behaviour that easily separates them from benign traffic. Therefore, scientifically botnet detection problem can be solved using an interesting blend of machine learning and networking technology. The technology part requires the use of sophisticated devices and software components to capture the network behaviour of all traffic flows passing through the network. On the other hand, the machine learning part maps the botnet detection problem into a *two-class* classification problem that successfully distinguish between botnet connections and normal connections.

The machine learning based botnet detection techniques found in the literature can be broadly classified into two groups:–(a) packet content based, and (b) network flow statistics based. Packet content based methods extract and look into the packet content at host or network level to recognize botnets in the middle of normal traffic. A major drawback of these methods is their inability to classify when packet contents are encrypted. The flow based methods on the other hand extract features from the network traces, aggregate and bind them with network flows. For example, *duration of a flow* is one of the most useful botnet detection parameters that can be derived by looking into packet traces. Usually botnets are "chatty" as their sessions consist of initial short flow due to initial connection attempts followed by a much longer communication session making flow duration unusually longer than normal sessions.

Regardless of features being used by any machine learning technique, the accurate detection of botnets greatly depends on the feature selection process. Only the correct choice of features can truly capture the right pattern within the network traces. In other words, the important features from the network packets must be selected. At the same time, redundant or irrelevant features should be excluded due to the following reasons. First of all, a large number of features slows down the classifier's performance. Secondly, un-obscured redundant features might contribute to inaccurate error-prone decisions. Thirdly, right choice of features can lead to high botnet detection rate and trigger low false alarms. Therefore, feature selection process is a very important inevitable step for any machine learning based botnet detection technique.

The contribution of this paper is a inclusion-exclusion based feature selection algorithm for botnet detection. The goal of the proposed Artificial Neural Network (ANN) model is to remove less important and redundant features, increase accuracy and detection rate. The proposed model also has a low computational complexity due to the simplified subset of features for classification. The selected features have been applied to a versatile test data sets devised by Beigi et. al. [4] The experimental results show that the built model is able to detect bots at the detection rate of 91%, with an accuracy of 87% and a moderate false alarm rate.

II. Methodology

The proposed botnet detection process is divided into five major steps:–(i) Passive monitoring and pre-processing of network traffic, (ii) Network traffic reduction, (iii) Feature Extraction, (iv) Feature Selection, and (v) Classification. Fig. 1 shows all steps in sequence. Each step is described next.

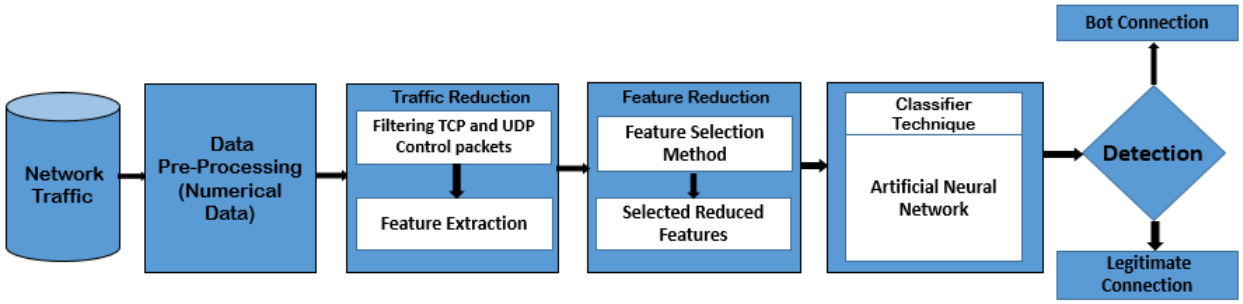

Fig. 1: Implemented approach to detect Botnet based on network traffic classification.

A. *Passive monitoring and pre-processing of network traffic*

A network monitor examines each packet passing through it to understand the network behavior. Packet level information is then mapped into flow level features. A flow can be identified by a combination of five tuples namely *Source IP Address*, *Destination IP Address*, *Source Port*, *Destination Port* and *Protocol Identifier*. Such information can easily be extracted from the header of each packet.

B. *Network traffic reduction*

A network usually exhibits a large amount of traffic. Traffic reduction is very important for managing this large traffics as resources are constrained at the monitoring device. The main goal of this work is to detect a botnet by analyzing as small network flows as possible. To obtain this goal, the monitoring device captures only the TCP and UDP traffic because these protocols are widely considered as the main carriers of Botnet Command and Control (C&C) and attack traffic [9]. All other layers' traffics are (safely) ignored. This filtering step drastically reduces the volume of network traffic being examined as well as increases the performance of the network monitor by reducing the processing overhead.

C. *Feature Extraction*

In this phase, the features that are important in detecting the bot's malicious behavior are extracted. We used both *flow based* and *conversation based* features. A flow is identified by its *signature* which defined as the combination of five attributes namely $\langle source\ \ IP, destination\ \ IP, sourceport, destinationport\rangle$-tuple. Two different packets having the same signature belong to the same flow. In order to identify which packet belongs to which flow only the header of the packets need to be examined. Note that a group of packets are usually exchanged between the same pair of hosts.

In a network connection usually there exists a pair of flows, one for the up-link and the other for the down-link. The combination of a pair of up and down link flows is together called a *conversation* in this paper.

Based on the network traffic characteristics we have carefully extracted eight flow-based features and four conversation-based features. A short description of these 12 features and the rationale behind their selection is given below:

Duration: Duration is the duration time of a flow. This is important as botnets usually have much longer communication sessions compared to normal sessions. For example, alevo botnet, IRC botnets etc. are knwon to be *chatty* [4].
AvgDur: This is the average duration per conversation, i.e., average of up and down link flows' duration of a conversation.
PBS: Payload Bytes Sent per flow. Botnet traffics is likely to be more uniform as bots execute mostly predefined actions. User generated traffics are more diverse and random.
AvgPBS: Average Payload Bytes Sent per conversation, i.e., average payload lengths of up and down link flows of a conversation.
TBS: Total Bytes Sent per flow. As bots execute fixed length commands, this feature can be used to extract similarity in botnet traffic.
PBR: Payload Bytes Received per flow.
AvgPBR: Average Payload Bytes Received per conversation.
TBR: Total Bytes Received per flow. This feature is used to extract similarity in Botnet traffic.
MissedBytes: Size of missed bytes per flow.
PktSent: Number of packets sent per flow. Bots usually try to keep their connections alive. In order to do so they send a large number of packets. Thus number of packets sent helps to identify this behavior.
PktRcvd: Number of packets received per flow. As bots receive a large number of packets to keep their connection alive this feature helps to locate this behavior.
SRPR: Ratio of the number of packets received to the number of packets sent. Some studies show that usually there exist even distribution between inbound and outbound traffic [6] on the Internet for many protocols. On the other hand, distribution of botnet traffic showed this equality does not hold [3]. Thus, this is a very important parameter in botnet detection.

D. *Feature Selection*

Feature selection is the process by which we can reduce the number of features to improve the performance of the detection model by eliminating less importance features during classification process. It enables the machine learning algorithm to train faster, reduces the complexity of the model and makes it easier to interpret. In order to reduce the over-fitting problem, it is important to choose the right subset of features.

The most popular feature selection method that selects the most optimal features for a specific algorithm is called

the *Wrapper Method*. This method is greedy in nature as it evaluates all possible combinations of the features and selects the combination that produces the best result. A drawback of this approach is testing all possible combinations of the features can be computationally expensive, particularly if the feature set is very large. For example, if there are n features then the total number of combinations is 2^n. Thus, in our case with 12 features the total possible combinations is 4096. Such a large number will overburden the model with significant computational cost especially when dealing with large datasets. To reduce this number, we propose a heuristic algorithm 1 which works with feature exclusion principle.

Algorithm 1 starts by calculating the accuracy using all features and considering it as the baseline. At each iteration it excludes every feature one by one and calculate the accuracy with the remaining features. The set of remaining features providing the highest accuracy becomes the baseline for the next iteration. This process continues until the accuracy of all remaining features are less than the previous iteration.

E. Classification

For classifying malicious and non-malicious traffic we have used Artificial Neural Network (ANN) which is a supervised multilayer feed-forward neural network with an adaptive learning rate. ANN needs to be trained using supervised learning. Once trained, ANN generates a model that maps inputs to expected output and can be used to predict output for some future unseen inputs. Three hidden layers were used in the ANN. Studying the effect of increasing the number of layers was ruled out considering the involved time and space complexity. It was observed that accuracy increased steadily as the number of nodes in each hidden layer was increased up to 20. After that, the accuracy oscillated around a certain mean. Three activation functions were studied:–Tanh, ReLu, Sigmoid or logistic. It was observed that the Tanh function showed the best accuracy, hence it was used as the activation function in the proposed ANN model.

Algorithm 1 Find Best Features (Features)

Input : Features
Output: Final Features Set

initial accuracy calculated by all features
$FinalFeaturesSet \leftarrow$ all features
while *accuracy is increasing* **do**
  for each feature f_i in $FinalFeaturesSet\{f_1, f_2, ...f_n\}$ **do**
    exclude feature f_i calculate accuracy a_i using all remaining features
  end
  find maximum accuracy a_m update accuracy a_i with new maximum accuracy a_m if $a_m > a_i$ update $FinalFeaturesSet$ with those features for which accuracy is maximum
end
return $FinalFeaturesSet$

TABLE I: Distribution of botnet types in training and test data set [1].

Botnet name	Type	Portion of flows in training data	Portion of flows in test dataset
Neris	IRC	21159 (12%)	25967 (5.67%)
Rbot	IRC	39316 (22%)	83 (0.018%)
Menti	IRC	-	2878(0.62%)
Sogou	HTTP	-	89 (0.019%)
Murlo	IRC	-	4881 (1.06%)
Virut	HTTP	1638 (0.94%)	58576 (12.80%)
NSIS	P2P	4336 (2.48%)	757 (0.165%)
Zeus	P2P	31 (0.01%)	502 (0.109%)
SMTP Spam	P2P	11296 (6.48%)	21633 (4.72%)
UDP Storm	P2P	-	44062 (9.63%)
Tbot	IRC	-	1296 (0.283%)
Zero Access	P2P	-	1011 (0.221%)
Weasel	P2P	-	42313 (9.25%)
Smoke Bot	P2P	-	78 (0.017%)
Zeus Control(C&C)	P2P	20 (0.01%)	31 (0.006%)
ISCX IRC bot	P2P	-	1816 (0.387%)

	Time	Source	SrcPort	Destination	DestPort	Protocol	Length
1	0.000000	88.162.105.165	26985	172.16.2.12	30319	UDP	87
2	0.000075	88.162.105.165	26985	172.16.2.12	30319	UDP	76
3	0.010209	172.16.2.12	30319	128.61.74.153	6882	UDP	96
4	0.053728	89.132.1.151	5511	172.16.2.12	1680	TCP	96
5	0.053982	77.200.208.2	62459	172.16.2.12	1748	TCP	96
6	0.062097	90.14.88.143	49645	172.16.2.12	1712	TCP	96
7	0.064591	172.16.2.12	1680	89.132.1.151	5511	TCP	60
8	0.068269	90.14.88.143	49645	172.16.2.12	1712	TCP	96
9	0.070244	172.16.2.12	1712	90.14.88.143	49645	TCP	60
10	0.073670	209.34.83.157	20001	172.16.2.12	1827	TCP	96

Fig. 2: A snapshot of ISCX 2014 Dataset.

III. Dataset used for training and testing

The proposed ANN model needs to be trained and tested. For that purpose, we use ISCX-Bot-2014 dataset [1]. This data is in PacketCapture (PCAP) format. The details of this dataset can be found in [4]. A snapshot of the ISCX 2014 Dataset is shown in Fig. 2. This dataset is divided into training and testing datasets that included 7 and 16 types of Botnets respectively. The training dataset is 5.3 GB in size consisting of about 43.92% of malicious traffic and the remaining are non-malicious. The testing dataset is 8.5 GB in size of which the distribution of malicious and non-malicious traffic is 44.97% and 55.03% respectively. In both training and test dataset, there exist different types of Botnet e.g. IRC, P2P, HTTP. The detail distribution of training and test dataset is given in Table I.

A. Data preparation

At first we create our own dataset from the raw dataset obtained from [1]. The raw data was in PCAP format, we used an online PCAP analyzer PacketTotal [2] for extracting necessary information from each traffic flow. This tool gives us the data in CSV format with 19 informative features which are very essential to detect a bot. A snapshot of the CSV file is shown in Fig. 3. We labeled each flow as malicious or non-malicious traffic based on previously detected malicious IP addresses as provided in the botnet dataset [1]. Thus the dataset became ready for supervised learning for the ANN feed-forward machine.

Timestamp	Connection ID	Sender IP	Sender Port	Target IP	Target Port	Transport	Service	Duration	Payload B	Total Byte	Payload B	Total Byte	Missed By	Packets Se	Packets Rec	Originatec	Tunnel Par	History
2007-10-08 1	CTL3JqeaFHCFi	172.16.2.12	1475	80.99.152.	5173	tcp	null	71.24	0	11812	15488	60	0	254	1	null	(empty)	Ad
2007-10-08 1	CanIeSqYx2zNi	172.16.2.12	1712	90.14.88.1	49645	tcp	null	71.55	416538	18021	873936	9060	0	416	200	null	(empty)	AaD
2007-10-08 1	CvPpEU7hgXKt	172.16.2.12	1680	89.132.1.1	5511	tcp	null	71.09	0	27136	0	0	0	567	0	null	(empty)	A
2007-10-08 1	CAqPYa5aI1Go	172.16.2.12	1748	77.200.208	62459	tcp	null	71.3	1396	20176	623628	813	0	470	15	null	(empty)	Ada
2007-10-08 1	CjSoRX6KQ3HE	88.162.105.	26985	172.16.2.1	30319	udp	null	71.1	3150	5530	3319	5867	0	85	91	null	(empty)	Dd
2007-10-08 1	CAHoIe8vKUFc	172.16.2.12	1669	81.76.16.2	61746	tcp	null	71.34	371	8900	125532	766	0	221	14	null	(empty)	Ada
2007-10-08 1	CHOuoI9yDd2g	172.16.2.12	1442	87.229.102	1991	tcp	null	69.73	88	2000	5936	153	0	50	3	null	(empty)	Aad
2007-10-08 1	CG2KmMxM9H	172.16.2.12	30319	136.160.25	25401	udp	null	null	null	54	null	0	0	1	0	null	(empty)	D
2007-10-08 1	CSWF812wHLV	172.16.2.12	30319	213.186.60	4200	udp	null	65.67	504	840	0	0	0	12	0	null	(empty)	D
2007-10-08 1	C91ybA1UHTj9	172.16.2.12	30319	84.75.48.2	13395	udp	null	65.66	546	910	0	0	0	13	0	null	(empty)	D
2007-10-08 1	CYFH8k1j3C3jp	172.16.2.12	1826	81.19.18.1	10010	tcp	null	9.42	595	220	888	168	1483	5	4	null	(empty)	ShAaFf
2007-10-08 1	Cjqyg32P5abZ	85.224.25.2	65206	172.16.2.1	30319	udp	null	null	null	56	null	0	0	1	0	null	(empty)	D

Fig. 3: A snapshot of CSV file from PacketTotal

Duration	AvgDuration	PBS	AvgPBS	Total Bytes Sent	PBR	AvgPBR	Total Bytes Received	Missed Bytes	Packets Sent	Packets Received	SRPR	class
4.28	6.039027778	1174	856.833333	1894	11862	27450.72	12462	0	18	15	0.8333333	0
3	1.5	0	0	192	0	0	0	0	4	0	0	1
0	1.5	0	0	96	0	0	0	0	2	0	0	1
2.96	2.021923077	0	0	96	0	0	0	0	2	0	0	1
0.15	0.159373278	0	0	96	0	0	40	0	2	1	0.5	1
0.58	7	0	148.699248	88	0	19616.8	44	0	2	1	0.5	0
1.01	2.276900826	0	81.1239669	240	11247	15930.23	11647	0	6	10	1.6666667	0
0.36	0.413057065	206	192.76087	474	1317	1185.293	1585	1523	5	5	1	1
3.77	31.41504798	0	356.458733	120	0	28422.39	0	0	3	0	0	0
0	0	0	0	96	0	0	0	0	2	0	0	1
0.15	0.159373278	0	0	96	0	0	40	0	2	1	0.5	1

Fig. 4: A snapshot of preprocessed dataset

B. Data pre-Processing

Data pre-processing is an integral step in machine learning as the quality of data and the useful information that can be derived from it directly affects the learning ability of the prediction model. In the dataset, there have different types of instances. For achieving better results we only take the numerical instances. We also ignore source port and destination port which is nothing but an address. We drop out all null and empty values from the dataset as no model can handle NULL or NaN values. We replace all missing data with the mean value. A snapshot of the the data set after pre-processing is shown in Fig. 4. This data was fed to the ANN model for learning and testing.

IV. Performance Analysis

In this section we present experimental results.

A. Performance metrics

We define the following five performance metrics that will be used mainly for two purposes:–(i) to assist feature selection process of the ANN model, and (ii) to determine the botnet detection ability of the trained model.

Precision: Precision is an indicator of consistency. High precision indicates an algorithm has returned far more important tests than non-related ones. Precision is defined as:

$$Precision = \frac{TP}{(TP + FP)} \tag{1}$$

Recall (Detection Rate): Recall is defined as:

$$Detection\ rate = \frac{TP}{(TP + FN)} \tag{2}$$

Recall is also known as Detection rate.

Accuracy: Informally, accuracy is the percentage of observations that resulted in correct classification. It is defined as:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{3}$$

False alarm rate: Known as false positive rate is defined as:

$$False\ alarm\ rate = \frac{FP}{(FP + TN)} \tag{4}$$

F-measure: The F-score is defined as the weighted harmonic mean of precision and recall. Formally F-score is:

$$f = \frac{(2 * Precision * Recall)}{(Precision + Recall)} \tag{5}$$

In the above equations,

- **True positive (TP)** - is the number of malicious behaviors correctly detected as malicious.
- **True negative (TN)** - is the number of non-malicious behaviors correctly detected as non-malicious activities.
- **False positive (FP)** - the number of non-malicious behaviors detected as malicious activities.
- **False negative (FN)** - the number of malicious behaviors detected as non-malicious activities.

B. Feature selection results

Fig. 5 shows accuracy achieved in each iteration of the feature selection Algorithm 1. Two peaks shown are the peak accuracy of first two iteration. On the first iteration the highest accuracy of 86.964% was achieved by excluding the *MissingBytes* feature from the feature set. Then in the second iteration further excluding *AvgPBS* resulted in highest accuracy of 87.18%. The third iteration could not improve the accuracy further. Hence the final feature set includes ten

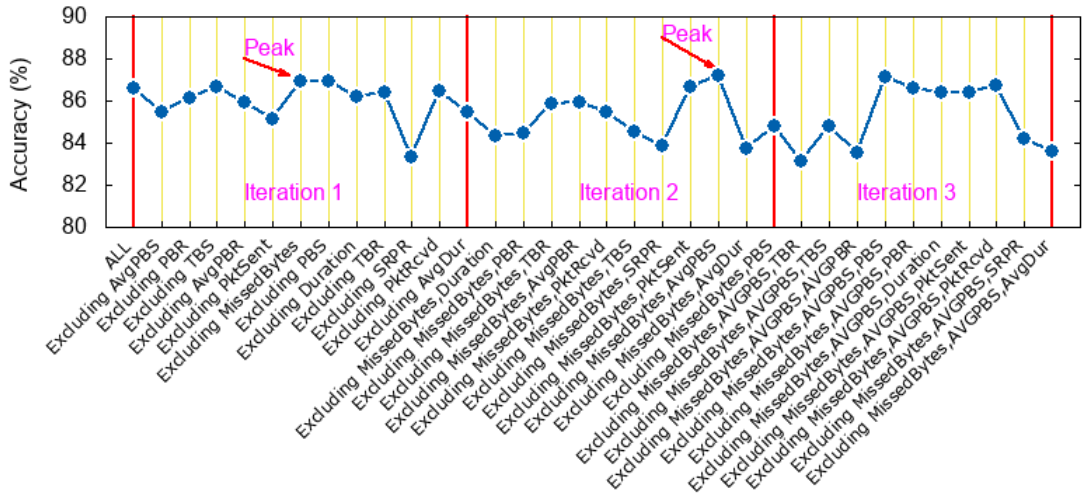

Fig. 5: Selecting features set based on accuracy

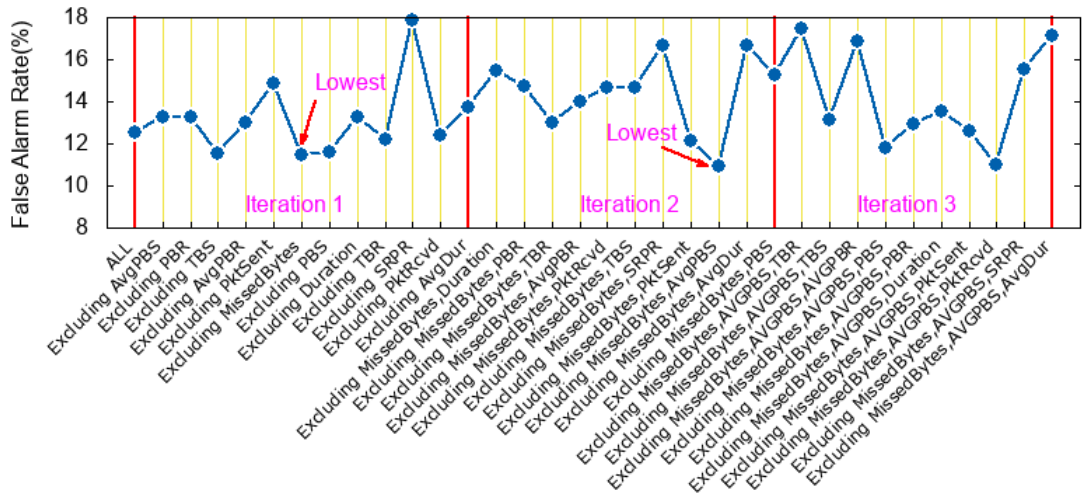

Fig. 6: False alarm rate in different iterations of the feature selection algorithm

features namely 'Duration', 'AvgDur', 'PBS', 'TBS', 'PBR', 'AvgPBR', 'TBR', 'PktSent', 'PktRcvd', and 'SRPR'.

False positive rates of the experiments are shown in Fig. 6. On the first iteration excluding *MissingBytes* again resulted in lowest false alarm rate of 11.44%. On the second iteration it got improved to 10.9% by eliminating *AvgPBS* attribute further. The third iteration again failed to improve any further.

C. Test results

Finally, the trained ANN model was fed with the test data set. The performance of the built detection model is shown in Fig. 7 in the form of confusion matrix and in Table II. As shown in Fig. 7, there are 56565 TP, 29846 TN, 7682 FP, 5907 FN flows. That boils down to 86.41% accuracy, a detection rate of 91%, precision rate of 88%, a F-measure of 89% and a false alarm rate of about 18% as shown in Table II.

D. Comparison with other work

We have compared the proposed ANN based model's performance with the decision tree based model's performance proposed by Beigi et. al. [4]. Table III shows the result. In terms of botnet detection rate and accuracy, our proposed model outperforms the model built in [4]. The detection rate in [4] is only 75% whereas our ANN model has a high detection rate of about 91%. The reasoning of better detection rate is as follows: (i) The features used in our model are less correlated, the correlation coefficient value being less than 0.8. (ii) As a detection algorithm they employed a C4.5 version of decision tree with Reduced Error Pruning (REP) whereas we have employed an ANN model with three hidden layers each containing 20 neurons. (iii) Another major difference is in Beigi et. al. [4]'s proposed system they categorized all features into four groups namely *Byte-based*, *Time-based*, *Behavior-based* and *Packet-based*. Then they applied *group exclusion* and *feature inclusion* steps for feature selection. During group

TABLE II: Performance metrics of our system with ANN

	Precision	Recall	F-measure	Accuracy
ANN	88%	91%	89%	86.41%

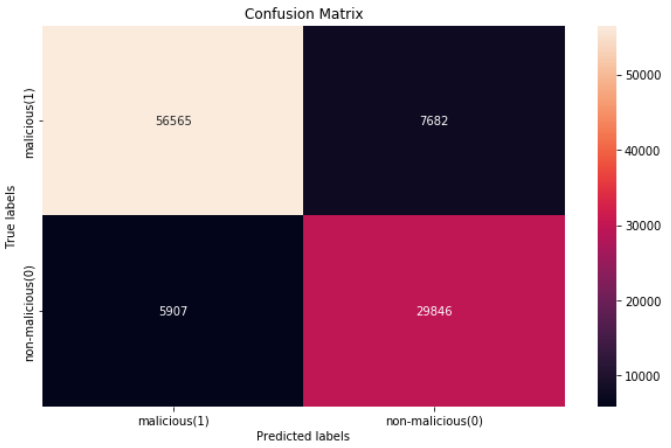

Fig. 7: Confusion matrix

TABLE III: Comparison with other work

Title	Methodology	Performance
This work	Inclusion and exclusion in Feature Selection, Multi-layer feed forward ANN	Accuracy 86.41 %, Detection rate 91 %
Beigi et. al. [4]	Group-based Feature selection, C4.5 decision tree	Accuracy 75%, Detection rate 69%.

exclusion all groups of features are evaluated and the least contributing group gets eliminated. The feature inclusion step follows by analyzing each features in the worst performing group and selecting the best one that increases the accuracy. Thus, only one feature in each group gets selected and many combination of features within a group are not even explored for possible improvement. On the other hand, we did not categorize features. Rather we treated each feature equally and at each step we eliminate the worst performing feature whose elimination increases the accuracy most. (iv) We have also made major differences in filtering network traffic. We have focused only on TCP and UDP traffics whether they chose other types of traffics e.g. HTTP, ICMP, DNS etc. as well.

V. Related works

A number of research works have been carried out to find the best possible feature set to detect botnets, although only a few works were able to achieve high accuracy and detection.

The work that closely matches with ours is the system proposed by Beigi et al. [4]. Initially they identified 16 features and group them into four groups namely *byte based*, *time based*, *behavior based*, and *packet based*. Then they devised a group based feature selection algorithm that iterative eliminates worst performing group and includes best feature of the eliminated group. Finally they select only four features from four groups. They have used decision tree for classification and detection. On the same data set their detection rate is only 69% and accuracy is 75% whereas we have both higher accuracy and detection rate with moderate increase in false alarms.

Anomaly-based intrusion detection system proposed by Ullah et al. [10] devices another filter-based feature selection algorithm. Features were evaluated based on information gain determined by four factors namely information, dependency, consistency, and distance of each feature. However the botnet detection rate was only 81% on the same ISCX dataset [1] that we have used but we got 91% detection rate.

Chen et al. [5] recently propose a P2P botnet detection technique by using neural networks on the convolutional version of flow-based features. The experimental results showed the effectiveness of the convolutional features for P2P botnet detection with high detection rate. However, detecting P2P botnets is beyond scope of this work.

Stevanovic et al. [8] investigated how quickly and accurately botnets can be detected by using flow-based network traffic and supervised machine learning. They also explored how much traffic needs to be observed per-flow to capture malicious traffic patterns. The proposed system is able to accurately detect the botnet traffic only after 10 packets and 60 seconds of monitoring per flow.

VI. Conclusion

Botnets appear to be a significant and growing threat against cybersecurity. Despite the long presence of malicious botnets, only a few formal studies have inspected the detection of the botnet with high accuracy. That being the case, we propose a machine learning-based botnet detection system that is effective in identifying botnets with high accuracy and detection rate. But a primary problem in machine learning is identifying a representative set of (good) features. In this paper we have worked out the problem of feature selection by using a feature reduction method which eliminates less important redundant features. Finally, we explore the ability of commonly used machine learning technique, Artificial Neural Network (ANN) and got a very high accuracy of 86.41%.

References

[1] "Iscx-bot-2014 dataset," https://www.unb.ca/cic/datasets/botnet.html, accessed: 2018-11-24.
[2] "Packettotal," https://packettotal.com, accessed: 2019-05-24.
[3] M. Almgren and W. John, "Tracking malicious hosts on a 10gbps backbone link," in *Infor. Sec. Tech. for App.*, 2012.
[4] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *IEEE Conf. on Comm. and Net. Sec. (CNS)*, 2014.
[5] S.-C. Chen, Y.-R. Chen, and W.-G. Tzeng, "Effective botnet detection through neural networks on convolutional features," in *12th IEEE Int. Conf. On Big Data Science And Engg. (BigDataSE)*, 2018, pp. 372–378.
[6] W. John and S. Tafvelin, "Differences between in- and outbound internet backbone traffic," *TERENA Networking Conference*, 2007.
[7] C. Li, W. Jiang, and X. Zou, "Botnet: Survey and case study," in *Fourth IEEE Int. Conf. on Innovative Com., Inf. and Control (ICICIC)*, 2009.
[8] M. Stevanovic and J. M. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in *2014 international conference on computing, networking and communications (ICNC)*. IEEE, 2014, pp. 797–801.
[9] M. Stevanovic and M. Pedersen, Jens, "An analysis of network traffic classification for botnet detection," in *IEEE CyberSA*, 2015.
[10] I. Ullah and Q. H. Mahmoud, "A filter-based feature selection model for anomaly-based intrusion detection systems," in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 2151–2159.