Journal of King Saud University – Computer and Information Sciences xxx (xxxx) xxx

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Clustering based semi-supervised machine learning for DDoS attack classification

Muhammad Aamir *, Syed Mustafa Ali Zaidi

Shaheed Zulfikar Ali Bhutto Institute of Science & Technology, Karachi, Pakistan

ARTICLE INFO

Article history:
Received 10 October 2018
Revised 28 January 2019
Accepted 3 February 2019
Available online xxxx

Keywords:
Clustering
DDoS attacks
Machine learning
Semi-supervised

ABSTRACT

Semi-supervised machine learning can be used for obtaining subsets of unlabeled or partially labeled dataset based on the applicable metrics of dissimilarity. At later stage, the data is completely assigned the labels as per the observed differentiation. This paper provides a clustering based approach to distinguish the data representing flows of network traffic which include both normal and Distributed Denial of Service (DDoS) traffic. The features are taken for victim-end identification of attacks and the work is demonstrated with three features which can be monitored at the target machine. The clustering methods include agglomerative and K-means with feature extraction under Principal Component Analysis (PCA). A voting method is also proposed to label the data and obtain classes to distinguish attacks from normal traffic. After labeling, supervised machine learning algorithms of k-Nearest Neighbors (kNN), Support Vector Machine (SVM) and Random Forest (RF) are applied to obtain the trained models for future classification. The kNN, SVM and RF models in experimental results provide 95%, 92% and 96.66% accuracy scores respectively under optimized parameter tuning within given sets of values. In the end, the scheme is also validated using a subset of benchmark dataset with new vectors of attack.

1. Introduction

Distributed Denial of Service (DDoS) is one of the most common occurrences of web attacks where the ultimate objective of an adversary is to degrade or deny the services of a target for legitimate users (Aamir and Zaidi, 2013). One of the main reasons of its high frequency is the number of ways this attack can be created and launched. In fact, DDoS attack is possible to generate at each layer of the OSI communication model. Hence, the attacker finds a number of ways to produce successful denial of service on a targeted victim. Although this attack exists for many years, the techniques used to successfully launch the attack on targeted victims have been ever changing. Due to this fact, DDoS is still one of the major attacks to be counted for in current cyber security studies. According to the statistics from *akamai.com* for Summer-2018 (Content Delivery Network, 2018), DDoS attacks have increased by 16% as compared to the last summer of 2017. DDoS has high adaptability to adjust and generate traffic according to the nature of victims. Therefore, this attack has covered many domains including traditional networks as well as more advanced level of networking (Aamir and Zaidi, 2015).

In a typical DDoS scenario, the attacker has high-end computing power to run command & control (C&C) function which is used to pass on instructions to the next layer of machines called handlers. The handlers are used to scan vulnerable servers and hosts on the Internet, and install malware to control those vulnerable machines. The compromised machines are then called zombies and this whole network is termed as a botnet. The zombies in the botnet are used to directly attack the ultimate target and create denial of service. Zombies also collect information from the victim and pass it back to the handlers for upward communication to C&C and the attacker. Due to this distributed architecture of botnet which contains a number of compromised machines (zombies) to launch attack on the victim, this attack is called Distributed Denial of Service. A common flow of DDoS attack is shown in Fig. 1.

The research community has significantly contributed towards proposing solutions of thwarting DDoS attacks which include packet-level analyses, flow-level analyses, behavioral analyses, traffic mining and deep packet inspection to name a few. In fact, different variants of DDoS protection are available in traditional

* Corresponding author.
E-mail address: aamir.nbpit@gmail.com (M. Aamir).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2019.02.003
1319-1578/

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

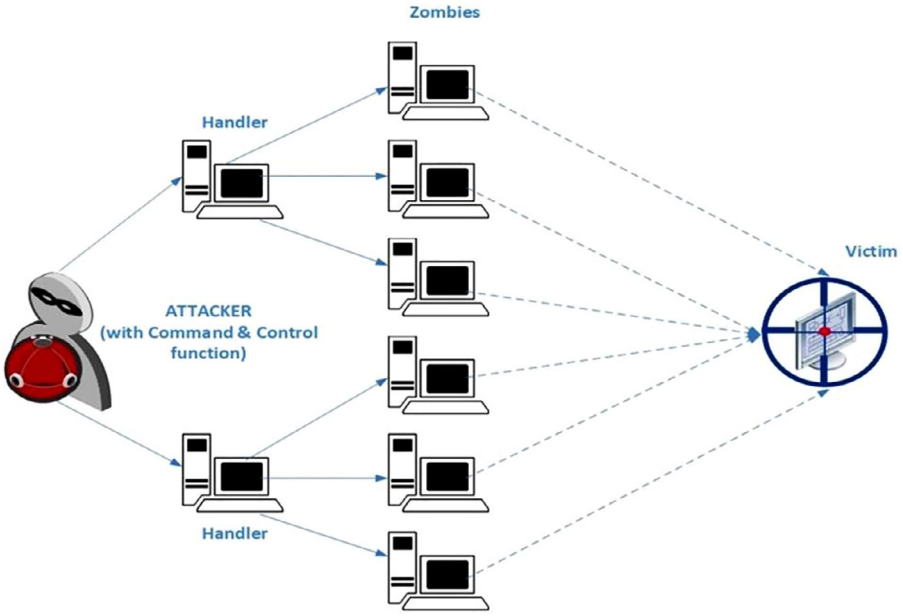

Fig. 1. Common flow of DDoS attack.

as well as advanced intrusion detection/prevention systems. Similarly, with the development of machine learning models and their successful implementations to address the critical problem areas of various fields, DDoS protection via machine learning and artificial intelligence based approaches has also been an area of interest for researchers. The need of machine learning in this area is led by the fact that DDoS attacks are now more sophisticated and deceptive than ever before. The packet specific features such as IP address and flags contain very less information to detect recent attacks as the real state of packets can easily be concealed. For example, IP address can either be spoofed or changed with high frequency within the same flow using a technique called fast flux (Hu et al., 2011). While supervised machine learning has been the most common variant to make contributions using benchmark datasets where the data is already labeled, the unsupervised and semi-supervised approaches have recently seen a boom in this domain where the first step is to label the data by observing it in a separable space driven by one or more metrics of dissimilarity. The most common approach is to apply some type of clustering algorithm. The clustered data can then be used to label the points. For semi-supervised approach, the subsequent phase of supervised machine learning is applied to classify the labels (classes) for unknown data points (Miller and Busby-Earle, 2016; Kim et al., 2018).

In this paper, we apply semi-supervised machine learning to first obtain clusters of network traffic data divided into the desired number of classes via unsupervised method, then label the points driven by a voting method to mark normal, DDoS and suspicious traffic, and finally detect the class of unknown traffic via supervised approach. During unsupervised learning phase, we also apply feature extraction technique of Principal Component Analysis (PCA) to project data points in low-dimensional space. Also during the supervised learning phase, we use optimization within a given set of values and validation techniques to find improved parameter settings of machine learning models. The other semi-supervised machine learning based research on labeling and classifying DDoS traffic has proposed some complex clustering approaches as well as supervised models with limited optimization efforts. On the other hand, the contributions of this paper are:

- Using different clustering applications on the same dataset to observe the differences and record the variations.
- Proposing a voting method to decide the labels of data points obtained in the clusters generated by more than one algorithm. It is demonstrated that we are able to assign 'k + 1' labels to the data of 'k' clusters after voting.
- Optimizing and validating the learning models i.e. finding optimal combination of parameters within a given set of values in addition to applying K-fold cross validation for improved model performance.

The core idea of this paper is that the clustering approach inherently leads to high false positives (Berkhin, 2006). Therefore, applying different schemes of clustering and then analyzing them through voting can bring confidence for a data point to be in a specific class. If different clustering algorithms vote for a data instance to place in the same class, it can induce an element of trust while reducing the inherent false positive problem of the single clustering method. On the other hand, if different clustering algorithms do not agree for a data point to be in the same class, there is an element of uncertainty which brings the data instance to a suspicious category. Ultimately, the system tends to reduce false positives whereas the suspicious data points are brought to the new class for further inspection and categorization accordingly. It is effectively an ensemble method of clustering with bagging approach where voting in one clustering method is independent from the other, while the collective voting results decide for a data point to be in the specific class.

The rest of this paper is organized as follows: Section 2 provides the background and related work in this research area. Section 3 explains different steps of our proposed semi-supervised machine learning approach. Section 4 demonstrates the experimental analysis in detail. Finally, Section 5 concludes the paper.

2. Background and related work

Semi-supervised machine learning is an approach which incorporates both unsupervised and supervised machine learning (Fitriani et al., 2016). When the labels of target class are not

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

available in data, the points can be grouped together under different clusters. For classifications of future data instances at later stage, supervised learning methods are taken into account to obtain the trained models. The clustering technique of unsupervised machine learning is used for segmentation of data into unlabeled but separable points. The similar instances of data are grouped into same cluster whereas the clusters are separated by an applied metric of dissimilarity. In semi-supervised mode, clusters are obtained to label the data in subsequent phase so that detections are possible for future instances. The semi-supervised methods have different types to process data for learning (Zhu, 2006), such as *Generative Model* is for identifying samples of data for labeling from a mixture of distribution such as Gaussian, *Graph-based Model* is to identify the nodes for labeled and unlabeled data, *Self-training* or *Bootstrapping* uses both labeled and unlabeled data points to learn from own detections and improve learning for subsequent iterations. Hence it is an iterative process of self-learning. Finally, *Co-training* is a method of learning that uses multi-view algorithm for data point arrangements into subsets to train with different classifiers. The types of clustering approach (Xiang and Min, 2010) under semi-supervised learning include *Constraint-based* for a direction to assign clusters to data points according to the pair-wise (must-link or cannot-link) constraint. If the constraint is must-link between two samples, they are assigned the same cluster. If it is cannot-link constraint, the samples are placed in different clusters. Also we have *Distance-based* which follows only pair-wise constraint to form clusters based on distance metric. The distance measurements are repeatedly evaluated to improve clustering. Finally we have *Constraint and Distance based*, which is the combination of above two methods.

In this research, we work on a dataset 'D' of 'm' rows (data instances) and 'n' columns (features). The dataset is composed of network flows which carry a mix of normal and DDoS traffic. Our aim is to apply semi-supervised machine learning in a systematic manner so that DDoS attacks may be identified in the mixed traffic. The clustering is *distance-based* and the most common metric of *Euclidean distance* (Davies and Bouldin, 1979) given in Eq. (1) is used. Here 'd(a,b)' is Euclidean (straight line) distance between 'a' and 'b' where (a_1,a_2) and (b_1,b_2) are the points in two dimensional space.

$$d(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2} \quad (1)$$

Gu et al. (2017) propose multiple features based constraint K-means (MF-CKM) for DDoS attack detection. Their method produces the feature vector to detect attacks. The base features used to obtain the vector include source and destination IPs (packet-level features). However, source IPs are usually spoofed due to which the feature vector may get infected in performance. Further, they initially need some labeled data to guide their unsupervised technique for clustering the unlabeled data. An hybrid technique using the combination of K-means and random forest (kM-RF) is proposed by Soheily-Khah et al. (2018) for classifying the intrusion detection in network systems. Their analysis is made on ISCX benchmark dataset. The work contains a major part of data cleansing where a number of calculated normal instances are removed initially using K-means clustering as a representation of controlling the amount of big data for mining and analytics. For detection phase, they use RF (Random Forest) supervised learning. Their accuracies and detections are not made on the whole dataset for collective picture but the flows are classified on the basis of application layer protocols (HTTP, DNS and SSH etc.). By experimental analysis, they claim that kM-RF is better in accuracy and detection than traditional machine learning algorithms (Naïve Bayes, Neural Network and Decision Tree etc.). Each flow in their experiment is represented by 50 features. However, IP address is not used in the phase of supervised learning via RF. Idhammad et al. (2018) propose a semi-supervised method where the initial co-clustering, information gain ratio and entropy analyses are used to reduce the amount of normal traffic in an effort to improve DDoS detection by eliminating the noisy component of data. The clusters formed under time-based sliding-window entropy variations provide differentiated traffic flows. The later part of supervised learning is based on extra-trees (ET) algorithm to detect DDoS attacks. They claim that their analysis over proposed time-windows mechanism provides more accuracy than direct application of ET algorithm on some benchmark datasets as well as two other proposed approaches taken from the literature survey and applied on NSL-KDD benchmark dataset. However, their analysis is limited to ET algorithm in the supervised phase of learning. Kato and Klyuev (2014) analyze patterns of DDoS attacks for each IP address. They also use other features for bps (bytes per second) calculations in the pattern analysis. For detection phase, they use SVM supervised learning. Later in Kato and Klyuev (2017), they use Apache Spark and Hadoop platforms for real time intrusion detection including unsupervised and supervised methods. Boroujerdi and Ayat (2013) propose an ensemble of adaptive neuro-fuzzy classifiers with a booting technique called Marliboost. They find a good detection accuracy using NSL-KDD dataset.

As the benchmark datasets are available with labeled classes, we have a more realistic approach of generating unclassified traffic, keeping the most common flow-level features under monitoring, proposing a voting method for labeling of 'k + 1' types of traffic out of 'k' clusters, and applying three different algorithms for classification of DDoS attacks. It is observed that the unsupervised and semi-supervised approaches, where the first step is to distinguish the data by observing it in a separable space, are less available in literature than supervised approaches having labeled datasets. It complements the fact that benchmark datasets are available with labeled classes on which most of the contributions have been presented by the research community. Using data of unsupervised nature initially in our proposed scheme, inherent high probabilities of false positive rate are reduced by applying different forms of clustering and then observing the dominant result through a voting method.

3. The proposed scheme

3.1. Agglomerative clustering

We start with applying agglomerative clustering (AC) on 'D'. AC is a variant of hierarchical clustering (Kaufman and Rousseeuw, 2009) which forms the clusters of data by considering each point as a cluster itself in the beginning (the bottom-up approach). So initially we have 'm' clusters in the data space. In the next step, the two closest clusters are joined to form one cluster. So we now have 'm−1' clusters. The process continues until we have one big cluster covering all the data points. The importance of AC lies in the memory which is built during the whole process. So in the end, we have a dendrogram to observe the process and find an optimal number of clusters possible from the dataset. We use the metric of Euclidean distance to find the closest clusters during this process. Euclidean distance is the straight line distance between two points.

3.2. Principal component analysis with K-Means clustering

Principal Component Analysis (PCA) (Jolliffe, 2011) is the algorithm to reduce dimensions (feature space) of a dataset from 'n' to 'p' where p < n. It does not reduce the exact features, but reduce dimensions via feature extraction. The extraction is performed by

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

the algorithm to keep the most variance of feature values. It obtains eigenvectors and eigenvalues from covariance matrix to choose 'p' eigenvectors corresponding to the 'p' largest eigenvalues, where 'p' is the number of dimensions in the new feature space. As PCA does not need labeled classes for feature extraction, it is quite applicable on unsupervised data. Another method, called K-means clustering, is one of the most common methods to form clusters by observing distance of data points from initially selected centroids. Throughout this process, the centroids get updated with relocation to the new centers as the clusters are reshaped by the algorithm. In our work, K-means clustering is applied on principal components obtained via PCA. The distance used in calculations is the Euclidean distance given in Eq. (1). We choose a clustering method different from AC which we applied initially on 'D' to bring diversity and unbiasedness in the analysis so that the variations in clustering outputs may be observed and recorded for next steps. Another reason to apply different clustering is the computational cost. The computational complexity (worst case analysis) of AC is $O(m^2)$ which is second degree complex irrespective of the number of clusters. Here 'm' represents the number of data points (Koga et al., 2007). On the other hand, the complexity of K-means is O (mnk) which is linear in 'm' as m >> n & k in our case. Here 'm' is number of instances, 'n' is number of dimensions (features) and 'k' is number of clusters (Xu and Wunsch, 2005). The low complexity of K-means clustering is therefore reasonable to use with PCA which already carries its own complexity of $O(m^3 + m^2n)$. It contains the components of covariance matrix computation ($O(m^2n)$) as well as eigenvalue decomposition ($O(m^3)$). Here 'm' is number of data points and 'n' is number of dimensions in the original dataset (Du and Fowler, 2008).

The metric called Within Cluster Sum of Squares (WCSS) can be used to find optimal number of clusters possible using K-means. However, as we already have the number of clusters for our problem area (i.e. 2 clusters), we can alternatively use the same measure to find the spread of dissimilarity metric (distance in this case) covered with the increase of number of clusters using Eq. (2), where 'dist' is the distance metric, sum of squares is taken for each *i*th point in the cluster, and all clusters from 1 to 'j' are taken into consideration.

$$WCSS = \sum_i dist(P_i, C_1)^2 + \cdots + \sum_i dist(P_i, C_j)^2 \quad (2)$$

3.3. Voting method

The voting method is introduced to deal with clustering results which appear to be opposite when we use different clustering algorithms. For example, a data instance may be clustered as a normal flow by one algorithm and as DDoS by the other. Therefore, we argue that there may be different approaches of voting strategy to deal with it such that (1) the final class should be the one which is, if exists, clustered in the same segment by all algorithms. If at least one algorithm produces a different result (or a threshold may be set for the number of differing algorithms), the data instance may be marked as an additional class, or (2) the final class should be the one which is, if exists, clustered in a segment by majority of the algorithms. If there are more than one majority values, the data instance may be marked as an additional class.

Both of the above mentioned criteria apply when we choose k = 2 as the number of clusters. As we use both AC and K-means with PCA, we have two outputs from two different kinds of clustering approaches. If both approaches put a data instance into the same segment i.e. normal or DDoS, we mark the appropriate class for that data point. However, if one of the algorithms puts a data point in one segment while the other puts it in the opposite one, we introduce an additional class, labeled as 'Suspicious', with an argument that such kind of traffic flows may be subject to further analysis by DDoS protection systems such as deep packet inspection or flow analysis. Hence, it is shown that we are able to assign 'k + 1' labels to the data of 'k' clusters using this voting method. The Algorithm 1 specifies this approach of voting.

Algorithm 1: Voting method to label classes

For each data point: **Do**
  Observe segments where data point is placed by each clustering algorithm:
  If all segments are the same:
    Assign corresponding class to the data point (Normal OR DDoS)
  Else:
    Assign newly introduced class to the data point (Suspicious)
Finish

3.4. Classification of attack

After assignment of labels to the data points, supervised learning methods are applied for training the models and detecting classes of unknown instances. In this analysis, we use k-Nearest Neighbors (kNN), Support Vector Machine (SVM) and Random Forest (RF) algorithms. kNN model detects the class for an unknown data point by measuring its distance from existing (trained) points (Larose and Larose, 2014). The majority of the closest points of same class among k-closest points is the decisive factor to which class the data point belongs. The distance metric used in our case is Euclidean distance mentioned in Eq. (1). SVM model detects the class of a new data instance by creating optimal hyperplanes for each feature of the dataset during training. It separates each dimension of the data in such a way that the margin of the closest points across hyperplanes is maximized. The kernel trick can be used for optimization in SVM models where the Radial Basis Function (RBF) is commonly deployed to obtain the classifications in nonlinear data sets. SVM is popular for handling high-dimensional and complex data. However, it is still sensitive towards noise and overfitting (Suthaharan, 2016). RF model is based on classification votes by individual decision trees working behind the algorithm. The results of decision trees are processed through specifically configured ensemble method such as bagging or boosting. In the RF model's configuration in this paper, we use bagged trees approach which is essentially a method in which a decision tree classifies the data independently from other trees (Breiman, 2001). In the end, the result is compiled by the RF algorithm with majority voting. Fig. 2 shows the flow chart of proposed semi-supervised machine learning scheme for DDoS attack classification.

4. Experimental analysis and results

4.1. Simulation and clustering

The dataset used in this research is obtained with the network traffic generated in OPNET Modeler 14.5 simulator (SteelCentral Riverbed Modeler, xxxx) (now called "Riverbed Modeler") using the simulation setup given in Fig. 3. The 'Botnet' is included to generate DDoS traffic, without which we have the normal scenario with a web server, 10 local users on a Local Area Network (LAN) and 15 public users accessing the server. However, once we add the botnet of 125 remotely compromised nodes (100 users on a remote LAN and 25 individual users with high-end machines) to

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

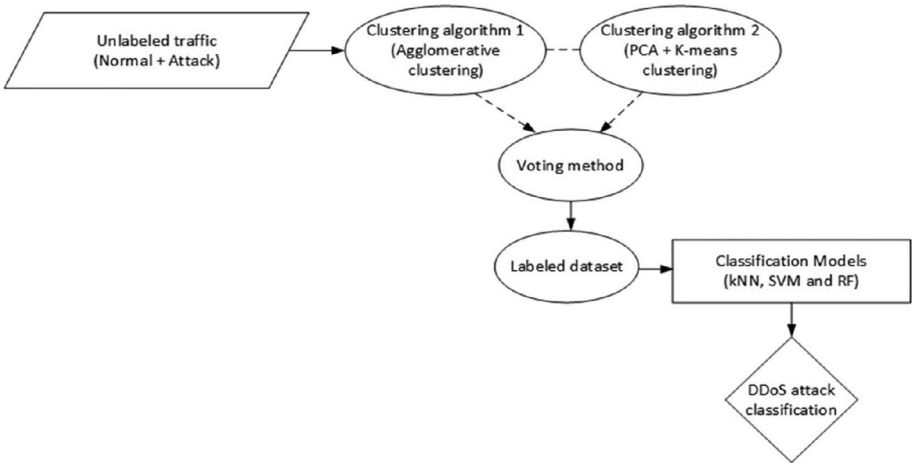

Fig. 2. Flow chart of semi-supervised machine learning for DDoS attack classification.

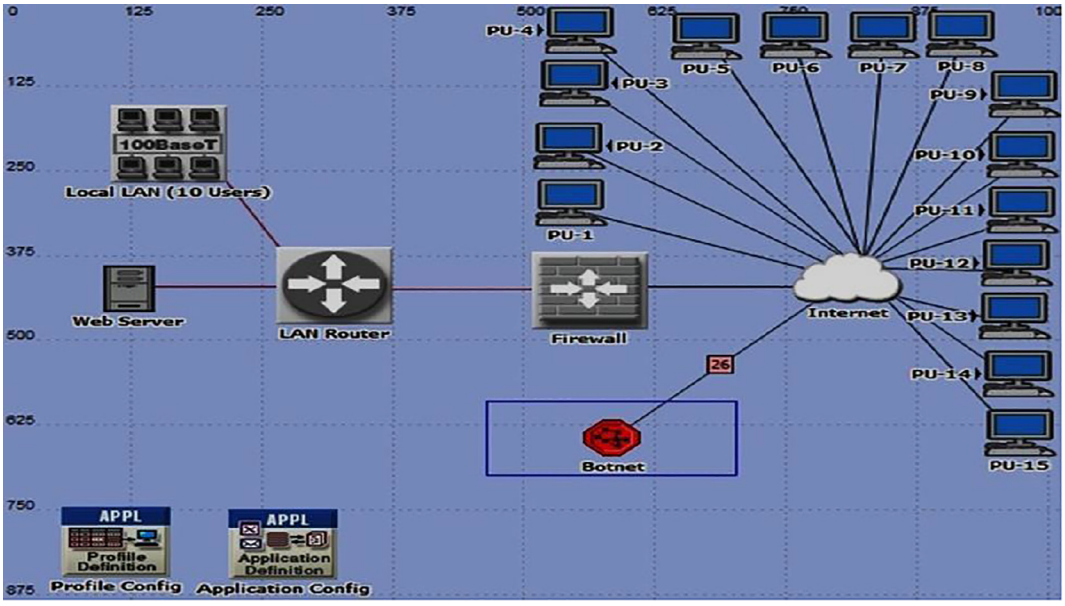

Fig. 3. Network setup in OPNET.

create an attack scenario, massive traffic is generated to produce an effect of DDoS attack on the web server (victim in this case). Nonetheless, the overall traffic remains a mix of normal and DDoS flows. In Figs. 4 and 5, comparisons are provided for traffic received by the server with normal and attack scenarios, and the server's CPU utilization with both scenarios respectively. A large increase is observed in the received traffic while CPU utilization also reaches 100% particularly in the beginning and at the end when more traffic is accumulated at server side.

OPNET is an event-based simulator. The traffic which we collected in this experiment is configured to record 100 events per minute. Hence for a period of 10 min of the simulation run, 1000 records are collected to form the dataset. In order to mimic the real-world network traffic, a significant portion of only normal flows in simulated traffic takes about 60–70% utilization of the web server with underlying transport layer protocol of TCP. Traffic distribution is also *Poisson* to reflect the real-world scenario of randomization in traffic. However, this simulated scenario can further be analyzed with diversified traffic such as using UDP protocol for delay sensitive applications. The dataset is converted to normalized state where the purpose of normalization is to bring the scattered values of different features on same scale to represent them fairly in calculations performed by machine learning algorithms under the hood. We have chosen min-max-normalized approach in the range of {0,1} (both inclusive) which is very common. All features of the dataset are normalized while the dataset is kept to small number of instances to reduce the performance overhead during machine learning phase and emphasize on presenting the proposed scheme of voted weightage assigned to clustering outputs. The features which we recorded for the dataset are following:

- *Traffic Rate:* The rate of arrival of network traffic at web server in bytes per second.
- *Processing Delay:* The delay of processing of network packets at web server in seconds.
- *Utilization:* The percentage of utilization (current load to the maximum load that CPU can handle) of web server's CPU at the time when event was recorded.

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

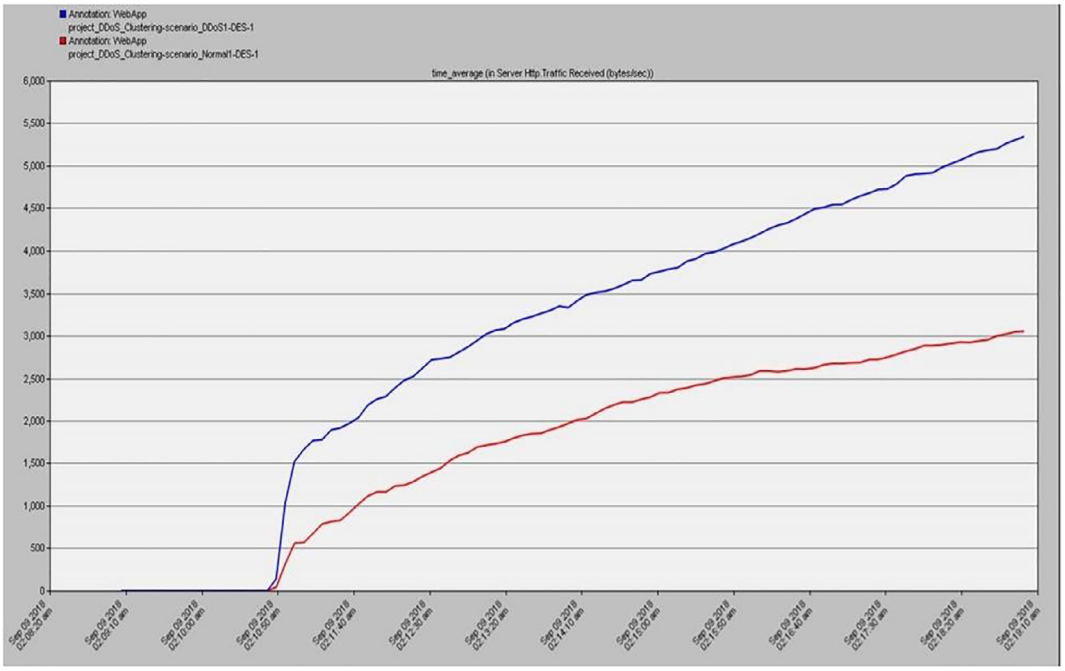

Fig. 4. Traffic received by web server (victim) in normal and attack scenarios.

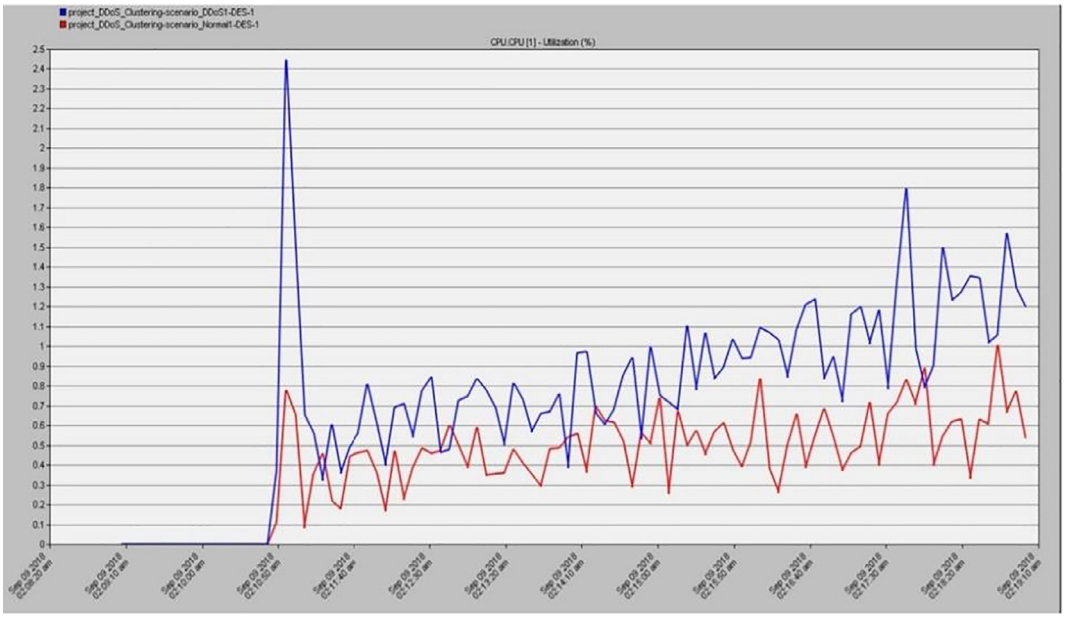

Fig. 5. CPU utilization of web server (victim) in normal and attack scenarios.

Here we argue that even with such a smaller set of features, we can effectively classify DDoS attacks with strategic flow of clustering, labeling and supervised learning, provided the relevant set of features is carefully chosen to prioritize flow-level features over packet-level features (Miller and Busby-Earle, 2016). The flow-level features cover the flow statistics such as delay, rate and utilization features. On the other hand, the packet-level features deal with the characteristics involved with individual packets such as IP addressing, TCP flag status and payload length etc. Although almost all benchmark datasets include both types of features and the classifications are made with considering both of these feature types, it has been identified that flow-level features are better classifiers and smaller datasets consisting of only flow-level features can efficiently detect DDoS attacks (Kirubavathi and Anitha, 2016; Gao et al., 2016). According to the study conducted in Jonker et al. (2017) for recent years, web servers are the most common targets of DDoS attacks. Also in the victim-end defense, legitimate and attack traffic can be distinguished more clearly as compared to source-end and core-end defenses (Beitollahi and Deconinck, 2012). Therefore, the applied scenarios are quite relevant to observe the presence of DDoS attacks.

We use *scikit-learn* machine learning library in Python 3 to demonstrate the proposed semi-supervised approach. For agglomerative clustering, we obtain the dendrogram mentioned in Fig. 6. Our required number of clusters is 2, and the dendrogram also provides a good split with 2 clusters if we extend a virtual horizontal

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

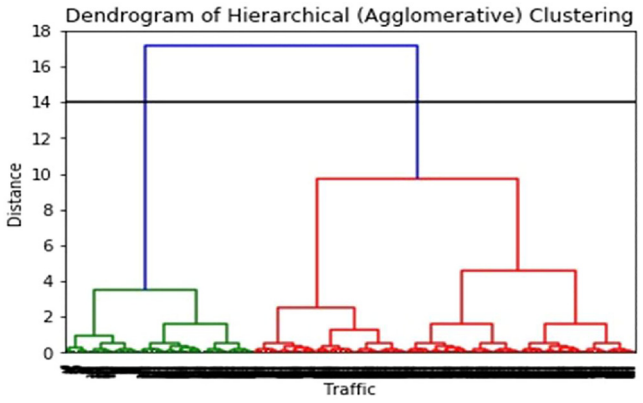

Fig. 6. Dendrogram with agglomerative clustering.

line cutting the longest vertical arm of the dendrogram, as it cuts through two vertical lines (one of the techniques used in the interpretation of dendrogram (Data Mining, xxxx). Similarly with K-means over PCA, we obtain two dimensional clusters where WCSS analysis using Eq. (2) reveal that nearly 80% of the dissimilarity is covered simply with two clusters as shown in Fig. 7. The classes are initially assigned considering the fact that normal traffic has comparatively less entropy (disorder) in traffic distribution which is to be reflected by the features in the dataset. In other words, DDoS attacks introduce more disturbances in network flow than normal traffic as zombies send large number of randomized packets towards the victim. So we calculate the entropy of each feature within a cluster, and the cluster with more cumulative entropy is labeled as DDoS. Eq. (3) represents entropy 'H(d)', where 'p_i' is the probability of information component in the vector 'd' and 'N' is total number of information values that the vector 'd' contains. The entropy values of DDoS and normal clusters, as observed in both forms of clustering algorithms, are provided in Table 1.

$$H(d) = -\sum_{i=1}^{N} p_i \log_2 p_i \tag{3}$$

The voting method described in Algorithm 1 is applied to obtain the labeled dataset with 3 classes. Hence we introduce another class called 'Suspicious' to label the data points which are placed in opposite clusters during unsupervised learning. With clustering outputs, we get 545 and 530 data instances clustered as DDoS by

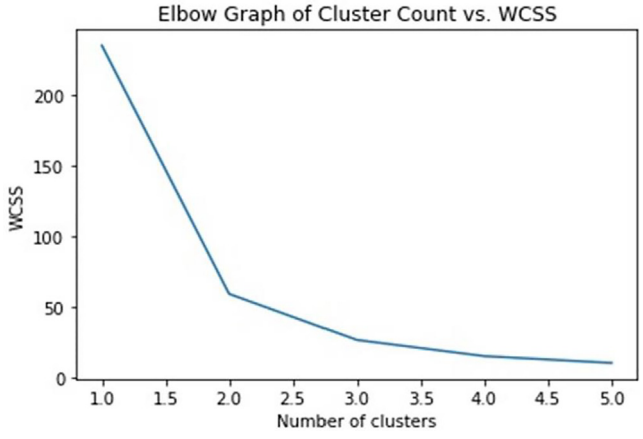

Fig. 7. WCSS analysis of K-means driven clusters.

Table 1
Cumulative entropy values of normal and DDoS clusters.

Clustering approach	Entropy of the cluster labeled as Normal	Entropy of the cluster labeled as DDoS
Agglomerative	18.09	18.75
PCA + K-means	18.01	19.42

Table 2
Clustered data points before and after voting.

Status	Classes	Clustered Data Points	
		Agglomerative Clustering	K-means with PCA
Before voting	Normal	455	470
	DDoS	545	530
After Voting	Normal	455	
	DDoS	470	
	Suspicious	75	

AC and K-means over PCA algorithms respectively. After applying the voting method, we get final labels i.e. 470 DDoS, 455 normal and 75 suspicious instances. This is shown in Table 2.

With the kNN, SVM and RF algorithms, we initially run the models with default settings in scikit-learn library, and then with optimized configurations within given sets of values. In this paper, more than one classification models are used to demonstrate that the assigned labels contain valuable information and this can be used to create the models based on different algorithms for classification of DDoS attacks. The optimized parameters within given sets of values are obtained with *Elbow method* for kNN and *GridSearchCV* function of scikit-learn for SVM and RF models respectively. It is noted that the models provide competitive accuracies, where RF model classifies more accurately with better performance. The train-test split of the data is divided with 70–30 percentage. The test dataset contains 300 instances (143 DDoS, 137 normal and 20 suspicious).

4.2. Classification models

4.2.1. kNN model

Our experiment with kNN model involves optimizing the k-value within a given set of values via elbow method. With the default value of 5 neighbors (k = 5), the accuracy score of the labeled dataset is 91.66% with 25 incorrect detections. However, the default value of 5 neighbors is not an optimal value unless tested. The dataset is studied for different k-values ranging from 1 to 25 against the average error rate. The variations of 'k' parameter are analyzed against the error via elbow method as shown in Fig. 8. It is observed that the optimal k-value of the model within a given range is 11 for which the error rate is 0.049.

The accuracy is considered as the main metric of a machine learning model's effectiveness. We calculate the accuracy by obtaining the ratio of correct detections in the test dataset to total instances in the dataset. In Algorithm 2, Python code of elbow method is provided to obtain the optimal k-value within a given range of values. Error rates are stored in *err_rate* array. In the end, the index of array holding the least error value provides the optimal k-value within the given range. The feature training dataset, output training dataset, feature test dataset and output test dataset are represented by variables *X_train*, *y_train*, *X_test* and *y_test* respectively.

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

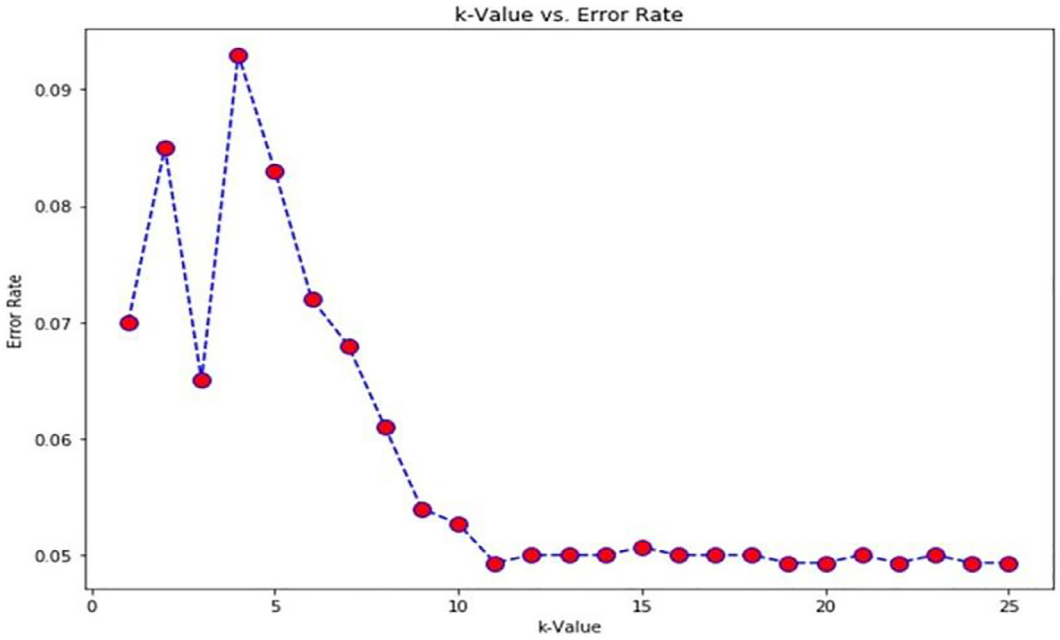

Fig. 8. k-value vs. error rate using elbow method.

Algorithm 2: Python code of elbow method to find optimal k-value in kNN model

```
From sklearn.neighbors import KNeighborsClassifier
err_rate = [ ]
For k in range(1, 26):
  knn = KNeighborsClassifier(n_neighbors = k)
  knn.fit(X_train, y_train)
  k_opt = knn.predict(X_test)
  err_rate.append(np.mean(k_opt != y_test))
```

4.2.2. SVM model

The experiment with SVM model involves testing different combinations of SVM-related parameters to find the optimal response within a given set of values. Under the most used kernel function for nonlinear data i.e. 'Radial Basis Function (RBF)', four different values of each of the 'C' parameter (SVM's penalty parameter) and 'gamma' (the kernel coefficient) are analyzed. The values of 'C' and 'gamma' parameters examined are mentioned below:

- C = 1, 10, 100, 1000
- gamma = 1, 0.1, 0.01, 0.001

To automate the process of obtaining the best combination of these parameters, 'GridSearchCV' class of 'sklearn.model_selection' package in Python is used. In Algorithm 3, Python code of GridSearchCV is provided to obtain the optimal values of 'C' and 'gamma' parameters within the given set of values. The *param_grid* dictionary holds the values of parameters to be tested. In the end, the *grid_svm* array holds the optimal parameter values.

Algorithm 3: Python code to find optimal parameter values in SVM model

```
From sklearn.model_selection import GridSearchCV
From sklearn.svm import SVC
param_grid = {'C': [1, 10, 100, 1000], 'gamma': [1, 0.1, 0.01,
  0.001], 'kernel': ['rbf']}
grid_svm = GridSearchCV(SVC(), param_grid, refit = True)
```

4.2.3. RF model

The experiment with RF model involves using different numbers of bagged decision trees to find the optimal response within a given set of values. Under the default decision criterion of 'Gini Impurity', four different values of 'n_estimators' parameter (no. of decision trees) are analyzed. The values of 'n_estimators' parameter examined are mentioned below:

- n_estimators = 10, 100, 500, 1000

To automate the process of obtaining the optimal quantity of bagged decision trees within a given set of values for the problem under consideration, 'GridSearchCV' class of 'sklearn.model_selection' package in Python is used. In Algorithm 4, Python code of GridSearchCV is provided to obtain the optimal value of 'n_estimators' parameter within the given set of values. The *param_grid* dictionary holds the numbers of trees to be tested. In the end, the *grid_rf* array holds the optimal response.

Algorithm 4: Python code to find optimal number of decision trees in RF model

```
From sklearn.model_selection import GridSearchCV
From sklearn.ensemble import RandomForestClassifier
param_grid = {'n_estimators': [10,100,500,1000], 'criterion':
  ['gini']}
grid_rf = GridSearchCV(RandomForestClassifier(), param_grid,
  refit = True)
```

4.2.4. Results

The accuracy scores in our experiments are also validated through K-fold cross validation. In this technique, the train-test split is randomly applied with 'K' number of splits of the whole dataset under 'K' total rounds of model fitting. There are 'K−1' splits of data for training and one split is reserved for testing. With each round, the testing split is changed and an accuracy score is measured. It can validate that the applied configurations do not lead to overfitting problem if the accuracies are only marginally

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

Table 3
Classification results (default).

Model	Decisive component	Default parameters	Default accuracy	Average precision	No. of incorrect detections
kNN	Euclidean distance	k = 5	91.66%	92%	25
SVM	Radial Basis Function (RBF) kernel	C = 1.0, gamma = 'auto'	88.66%	89%	34
RF	Bagged decision trees	n_est = 10	96.00%	96%	12

Table 4
Classification results (optimized).

Model	Optimized parameters	Optimized accuracy	Cross-validation average accuracy	Average precision	No. of incorrect detections
kNN	k = 11	95.00%	94.70%	96%	15
SVM	C = 100.0, gamma = 1.0	92.00%	91.56%	92%	24
RF	n_est = 100	96.66%	96.14%	97%	10

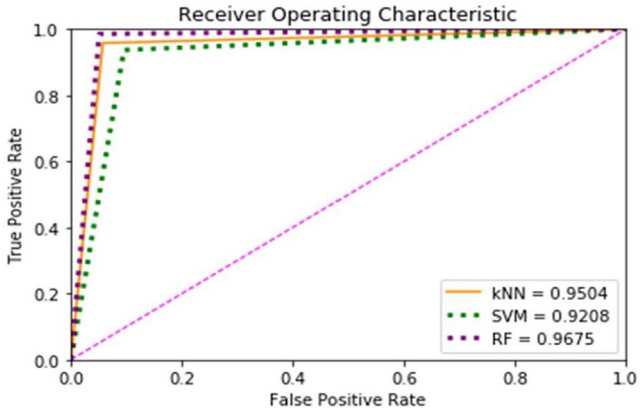

Fig. 9. Area under curve analyses of optimized classification models.

changed within a short range of values. For example, the accuracy of kNN model under optimal parameter tunings is 95%. With K-fold cross validation of K = 10, the average accuracy is found to be 94.7%. As this accuracy is close to the optimal accuracy that we obtained initially, it is validated that the optimal accuracy is true accuracy without overfitting. In Tables 3 and 4, the results of classification models are provided in default and optimized states respectively. Here, 'k' is no. of nearest neighbors, 'C' is SVM's penalty parameter, 'gamma' is the kernel coefficient, and 'n_est' is the number of decision trees. The optimized results are also validated with K-fold cross validation (where K = 10) to confirm that the average accuracy is close to the optimized one. For SVM, the value of gamma = 'auto' is the default state of SVC object of 'sklearn.svm' class in Python. The term 'auto' represents a value equal to **1/n** where 'n' is the number of input features in a given dataset.

The fitness of classification models is further validated with Area Under Curve (AUC) analyses by obtaining Receiver Operating Characteristic (ROC) shown in Fig. 9. The confusion matrix can manipulate the facts of model's learning and provide values of true and false classifications at a single operating point. This condition can lead to a paradox where the given accuracy value may not be valid for other operating points or changes in the model's performance. This is known as accuracy paradox. To avoid this paradox, the Receiver Operating Characteristic (ROC) graph is plotted between true positive and false positive rates. The Area Under Curve (AUC) statistic provides the real accuracy of a model's classification for varying true positive and false positive rates. Here it is shown in Fig. 9 that kNN, SVM and RF models have effectively learnt from data and thus the area under curve values under optimal performance are close to the calculated accuracies obtained from confusion matrices in Python. The RF model classifies the data more accurately than kNN and SVM.

4.3. Scheme validation with benchmark dataset

In order to test the effectiveness of our scheme with new vectors of attack (features) explained by benchmark datasets, we use a subset of CICIDS2017 which is a benchmark intrusion detection dataset of 2017 published by CIC (Canadian Institute for Cybersecurity) in 2018 (Sharafaldin et al., 2018). The data is used from DDoS attacks Friday – working hours' scenario and a subset of 10,000 instances are taken for analysis (10 times of our simulated dataset). The four most important features of classification among 78 total variables, as mentioned by the dataset creators in (Sharafaldin et al., 2018) through preliminary examination, are taken into consideration i.e. 'Bwd Packet Length Std', 'Average Packet Size', 'Flow Duration', and 'Flow IAT Std'. Although they are different variables from our dataset features, the scheme can be validated with the new variables of attack if reasonably accurate labels are obtained with proposed clustering and voting approach. The DDoS attacks Friday – working hours' scenario of CICIDS2017 originally contains 225,745 instances out of which 97,718 are labeled as 'Benign' whereas the remaining 128,027 are 'DDoS'. For the sake of reducing the overhead on experimenting computer[1], we extract 10,000 data instances with shuffling and randomization of data in an independent manner through *shuffle* class of *sklearn.utils* package in Python. This subset of 10,000 data points carries similar proportion of traffic as in full dataset i.e. 4355 benign and 5645 DDoS instances. However, this subset is also reduced in the feature space with four most important variables as described before. Our aim is to demonstrate that an acceptable level of accuracy may be obtained while labeling the dataset with proposed scheme. It is hence tested against a representative benchmark dataset with recent vectors of attack (only four most important flow-level features). The AC and WCSS behavior are shown in Figs. 10 and 11 respectively. It can be noticed that three clusters represent better differentiation of the given data subset. However, as we are interested in two clusters with normal and DDoS labels, we extract three clusters as suggested by the clustering algorithms[2] but place two clusters in DDoS category which contain higher scores of cumulative entropy as compared to the remaining cluster. The third cluster with

[1] The experiments are conducted on Intel® Core™ i7, 7500U CPU @2.70 GHz with 4 cores and 8GB of primary storage (RAM).
[2] Dendrogram for AC, and WCSS for K-means.

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

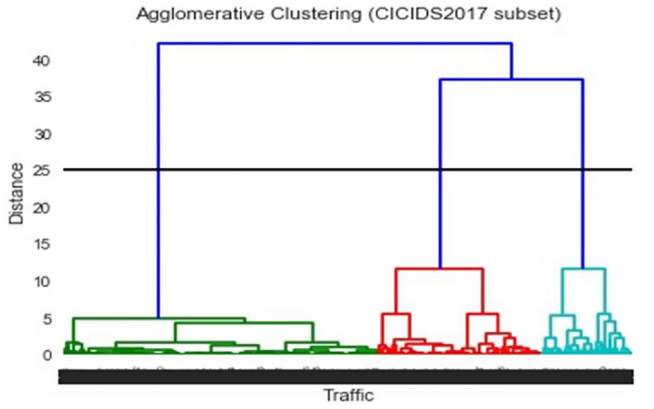

Fig. 10. Dendrogram of CICIDS2017 subset.

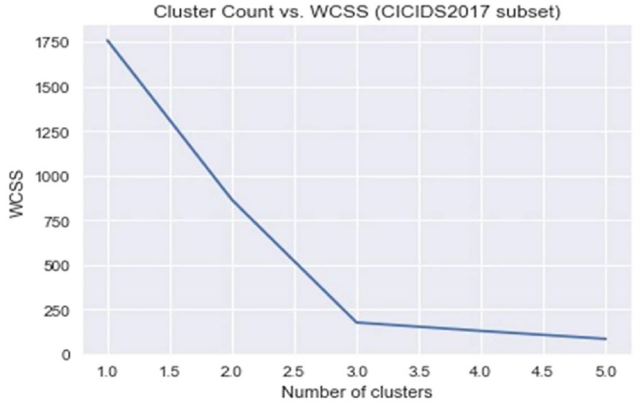

Fig. 11. WCSS analysis of CICIDS2017 subset.

Table 5
Cumulative entropies of clusters in CICIDS2017 subset.

Clustering algorithm	Cumulative entropy of DDoS clusters	Cumulative entropy of normal cluster
Agglomerative	60.471	29.319
PCA + K-means	60.531	29.080

the lowest cumulative entropy is ultimately placed in the normal category. The entropy scores are shown in Table 5. It is also observed from WCSS analysis that more than 88% of the dissimilarity is covered with suggested three clusters.

After we apply the voting method as explained in Section 3.3, we obtain 3851 benign, 5444 DDoS and 705 suspicious labels. While comparing this with the original labels of CICIDS2017 data subset, we calculate the precision scores as 78.53% for benign and 84.85% for DDoS labels. The overall accuracy score appears to be 82.10% as shown in Table 6. It is observed that the proposed scheme is promising as we obtain more than 82% in labeling accuracy. It can further be improved and refined by introducing additional methods of unsupervised learning such as Self Organizing Maps (SOM). However, the elements of neural computing are not considered in this paper and we leave it for the future research. Also for classification, we can train neural network based models such as multi-layer perceptron.

4.4. Comparison with related work approaches

In Table 7, a comparison of this work's approach is provided with other related works mentioned in Section 2. It is observed that the average detection accuracy of this work is competitive to other approaches. In addition to this, our work offers more diversity by having multiple classification algorithms and multiple clustering for unlabeled data to reduce false positives. Hence, analyzing the data through proposed voting method after multiple layers of clustering brings a confidence for a particular data instance to be in a certain class. If different clustering algorithms vote a data point for the same class, it can induce an element of trust and reduce the inherent false positive problem of using a single clustering method (the distinguished approach adopted in this paper). On the other hand, if different clustering algorithms do not agree for a data instance to place in the same class, there is an element of uncertainty which brings the data instance to a suspicious category.

5. Conclusion

In this paper, we apply a semi-supervised machine learning approach to classify DDoS attacks. It starts with unlabeled traffic statistics obtained against three features for victim-end defense i.e. the web server. The features include traffic rate, processing delay and CPU utilization. The unlabeled data is clustered by two different clustering algorithms and a voting method decides the final labeling of traffic flows. The instances falling in opposite clusters are labeled with an additional class called 'Suspicious'. The supervised learning algorithms of kNN, SVM and RF are applied on labeled data to classify DDoS attacks. The experimental results of 95%, 92% and 96.66% accuracy scores are obtained with kNN, SVM and RF models respectively under optimized parameter tunings within given sets of values. The scheme is also validated for the accuracy of label assignments using a representative subset of the benchmark CICIDS2017 dataset with new vectors of attack. It seems promising as we obtain more than 82% accurate labels. In future, we plan to seek improved methods of voting for the labeling of data and include more machine learning algorithms during clustering and classification.

Declaration of interest

None.

Table 6
Labeling comparison with CICIDS2017 subset.

Original label	Original number of instances	Correct number of labels after voting method	Number of labels classified as opposite category	Number of labels classified as suspicious category
Benign	4355	3420	654	281
DDoS	5645	4790	431	424
Total	10,000	8210	1085	705

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003

Table 7
Comparison with research approaches of related works.

Research	Traffic Analysis	Classification Analysis	Average Detection Accuracy (%)	Strengths	Limitations
Soheily-Khah et al. (2018)	K-means	kM-RF, Naïve Bayes, neural network, decision tree, 1-NN, SVM, RF	99.97 (kM-RF)	Simplicity, Multiple classification algorithms	Classification on the basis of application layer protocols only
Idhammad et al. (2018)	Time-based sliding-window entropy variations	Extra Trees	99.88	Entropy-variations, Noise filtering	Complex analysis of unlabeled data, Single classification algorithm
This work	Multiple Clustering (AC, K-means over PCA)	kNN, SVM, RF	96.66 (RF)	Diversified approach, Multiple classification algorithms and multiple clustering to reduce false positives, Voting method for final labeling	Low number of features (however the labeling scheme is validated with a subset of recent benchmark dataset CICIDS2017 having high number of features originally and showing good labeling results)
Kato and Klyuev (2017)	K-means	OCSVM	94.3	Real time intrusion detection system supporting big data platforms, Scalability	Efficiency and detection issues for unseen data, High rate of false positives
Boroujerdi and Ayat (2013)	ANFIS	Random tree, J48, Naïve Bayes, SVM, RF, NB tree, MLP, Proposed ensemble with boosting (Marliboost)	96.38 (Marliboost)	Multiple classification algorithms, Ensemble method of detection	Low number of features, Complex analysis

References

Aamir, M., Zaidi, M.A., 2013. A survey on DDoS attack and defense strategies: from traditional schemes to current techniques. Interdisciplinary Inf. Sci. 19 (2), 173–200.

Aamir, M., Zaidi, S.M.A., 2015. Denial-of-service in content centric (named data) networking: a tutorial and state-of-the-art survey. Security Commun. Networks 8 (11), 2037–2059.

Beitollahi, H., Deconinck, G., 2012. Analyzing well-known countermeasures against distributed denial of service attacks. Comput. Commun. 35 (11), 1312–1332.

Berkhin, P., 2006. "A survey of clustering data mining techniques". In: Grouping Multidimensional Data. Springer, pp. 25–71.

Boroujerdi, A.S., Ayat, S., 2013. "A, robust ensemble of neuro-fuzzy classifiers for DDoS attack detection", in. In: Computer Science and Network Technology (ICCSNT), 2013 3rd International Conference on, pp. 484–487.

Breiman, L., 2001. Random forests. Machine Learn. 45 (1), 5–32.

"Content Delivery Network (CDN) & Cloud Computing Services | Akamai." [Online]. Available: https://content.akamai.com/us-en-PG11224-summer-2018-soti-web-attack-report.html.

"Data Mining." [Online]. Available: http://www.stat.cmu.edu/~ryantibs/datamining/.

Davies, D.L., Bouldin, D.W., 1979. A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. 2, 224–227.

Du, Q., Fowler, J.E., 2008. Low-complexity principal component analysis for hyperspectral image compression. Int. J. High Perf. Comput. Appl. 22 (4), 438–448.

Fitriani S., Mandala S., Murti M.A. "Review of semi-supervised method for Intrusion Detection System" in Multimedia and Broadcasting (APMediaCast) Asia Pacific Conference on, 2016, 2016, 36-41.

Gao Y., Feng Y., Kawamoto J., and Sakurai K., "A machine learning based approach for detecting DRDoS attacks and its performance evaluation," in Information Security (AsiaJCIS), 2016 11th Asia Joint Conference on, 2016, pp. 80–86.

Gu, Y., Wang, Y., Yang, Z., Xiong, F., Gao, Y., 2017. Multiple-features-based semisupervised clustering DDoS detection method. Math. Problems Eng. 2017.

Hu, X., Knysz, M., Shin, K.G., 2011. "Measurement and analysis of global IP-usage patterns of fast-flux botnets", in *INFOCOM*. Proc. IEEE 2011, 2633–2641.

Idhammad, M., Afdel, K., Belouch, M., 2018. "Semi-supervised machine learning approach for DDoS detection,". Appl. Intell., 1–16

Jolliffe, I., 2011. "Principal component analysis". In: International Encyclopedia of Statistical Science. Springer, pp. 1094–1096.

Jonker, M., King, A., Krupp, J., Rossow, C., Sperotto, A., Dainotti, A., 2017. "Millions of targets under attack: a macroscopic characterization of the DoS ecosystem". Proceedings of the 2017 Internet Measurement Conference, pp. 100–113.

Kato, K., Klyuev, V., 2014. An intelligent DDoS attack detection system using packet analysis and Support Vector Machine. Int. J. Intell. Comput. Res. IJICR 14 (5), 3.

Kato, K., Klyuev, V., 2017. "Development of a network intrusion detection system using Apache Hadoop and Spark", in *Dependable and Secure Computing*. IEEE Conference on 2017, 416–423.

Kaufman, L., Rousseeuw, P.J., 2009. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons.

Kim, J., Sim, A., Tierney, B., Suh, S., Kim, I., 2018. "Multivariate network traffic analysis using clustered patterns,". Computing, 1–23.

Kirubavathi, G., Anitha, R., 2016. Botnet detection via mining of traffic flow characteristics. Comput. Electr. Eng. 50, 91–101.

Koga, H., Ishibashi, T., Watanabe, T., 2007. Fast agglomerative hierarchical clustering algorithm using Locality-Sensitive Hashing. Knowl. Inf. Syst. 12 (1), 25–53.

Larose, D.T., Larose, C.D., 2014. Discovering Knowledge in Data: An Introduction to Data Mining. John Wiley & Sons.

Miller S. and Busby-Earle C., "The role of machine learning in botnet detection," in Internet Technology and Secured Transactions (ICITST), 2016 11th International Conference for, 2016, pp. 359–364.

Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., 2018. "Toward generating a new intrusion detection dataset and intrusion traffic characterization". ICISSP, 108–116.

Soheily-Khah S., Marteau P.-F., Béchet N., "Intrusion Detection in Network Systems Through Hybrid Supervised and Unsupervised Machine Learning Process: A Case Study on the ISCX Dataset", in Data Intelligence and Security (ICDIS), 2018 1st International Conference on, 2018, 219-226.

"SteelCentral Riverbed Modeler," Riverbed. [Online]. Available: https://www.riverbed.com/sg/products/steelcentral/steelcentral-riverbed-modeler.html.

Suthaharan, S., 2016. "Support vector machine". In: Machine Learning Models and Algorithms for Big Data Classification. Springer, pp. 207–235.

G. Xiang and W. Min, "Applying Semi-supervised cluster algorithm for anomaly detection," in Information Processing (ISIP), 2010 Third International Symposium on, 2010, pp. 43–45.

Xu, R., Wunsch, D., 2005. Survey of clustering algorithms. IEEE Trans. Neural Networks 16 (3), 645–678.

Zhu, X., 2006. Semi-supervised learning literature survey. Computer Sci., Univ. Wisconsin-Madison 2 (3), 4.

Please cite this article as: M. Aamir and S. M. A. Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.02.003