Applied Intelligence
https://doi.org/10.1007/s10489-018-1141-2

Semi-supervised machine learning approach for DDoS detection

Mohamed Idhammad[1] · **Karim Afdel**[1] · **Mustapha Belouch**[2]

Abstract
Even though advanced Machine Learning (ML) techniques have been adopted for DDoS detection, the attack remains a major threat of the Internet. Most of the existing ML-based DDoS detection approaches are under two categories: supervised and unsupervised. Supervised ML approaches for DDoS detection rely on availability of labeled network traffic datasets. Whereas, unsupervised ML approaches detect attacks by analyzing the incoming network traffic. Both approaches are challenged by large amount of network traffic data, low detection accuracy and high false positive rates. In this paper we present an online sequential semi-supervised ML approach for DDoS detection based on network Entropy estimation, Co-clustering, Information Gain Ratio and Exra-Trees algorithm. The unsupervised part of the approach allows to reduce the irrelevant normal traffic data for DDoS detection which allows to reduce false positive rates and increase accuracy. Whereas, the supervised part allows to reduce the false positive rates of the unsupervised part and to accurately classify the DDoS traffic. Various experiments were performed to evaluate the proposed approach using three public datasets namely NSL-KDD, UNB ISCX 12 and UNSW-NB15. An accuracy of 98.23%, 99.88% and 93.71% is achieved for respectively NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets, with respectively the false positive rates 0.33%, 0.35% and 0.46%.

Keywords DDoS detection · Co-clustering · Entropy analysis · Information gain ratio · Feature selection · Extra-Trees

1 Introduction

Despite the important evolution of the information security technologies in recent years, the DDoS attack remains a major threat of Internet. The attack aims mainly to deprive legitimate users from Internet resources. The impact of the attack relies on the speed and the amount of the network traffic sent to the victim.

Generally, there exist two categories of the DDoS attack namely Direct DDoS attack and Reflection-based DDoS [1–4]. In the Direct DDoS attack the attacker uses the zombie hosts to flood directly the victim host with a large number of network packets. Whereas, in the Reflection-based DDoS attack the attacker uses the zombie hosts to take control over a set of compromised hosts called Reflectors. The latter are used to forward a massive amount of attack traffic to the victim host. Recently, destructive DDoS attacks have brought down more than 70 vital services of Internet including Github, Twitter, Amazon, Paypal, etc [5, 6]. Attackers have taken advantages of Cloud Computing and Internet of Things technologies to generate a huge amount of attack traffic; more than 665 Gb/s [5, 6]. Analyzing this amount of network traffic at once is inefficient, computationally costly and often leads the intrusion detection systems to fall.

Data mining techniques have been used to develop sophisticated intrusion detection systems for the last two decades. Artificial Intelligence, Machine Learning (ML), Pattern Recognition, Statistics, Information Theory are the most used data mining techniques for intrusion detection [7]. Application process of data mining techniques in general and ML techniques more specifically requires five typical steps *selection*, *preprocessing*, *transformation*,

✉ Mohamed Idhammad
idhammad.mohamed@edu.uiz.ac.ma

Karim Afdel
k.afdel@uiz.ac.ma

Mustapha Belouch
mbelouch@gmail.com

[1] LabSIV, Department of Computer Science, Faculty of Science, Ibn Zohr University, BP 8106, 80060 Agadir, Morocco

[2] LAMAI, Department of Computer Science, FSTG, Cadi Ayyad University, Marrakesh, Morocco

Published online: 13 February 2018

mining, and *interpretation* [8]. Despite that *preprocessing* and *transformation* steps may be trivial for intrusion detection applications, *selection*, *mining* and *interpretation* steps are crucial for selecting relevant data, filtering noisy data and detecting intrusions [7]. These three crucial steps are the most challenging of the existing data mining based intrusion detection approaches.

The existing Machine Learning based DDoS detection approaches can be divided into three categories. Supervised ML approaches that use generated labeled network traffic datasets to build the detection model. Two major issues are facing the supervised approaches. First, the generation of labeled network traffic datasets is costly in terms of computation and time. Without a continuous update of their detection models, the supervised machine learning approaches are unable to predict the new legitimate and attack behaviors. Second, the the presence of large amount of irrelevant normal data in the incoming network traffic is noisy and reduces the performances of supervised ML classifiers.

Unlike the first category, in the unsupervised approaches no labeled dataset is needed to built the detection model. The DDoS and the normal traffics are distinguished based on the analysis of their underlying distribution characteristics. However, the main drawback of the unsupervised approaches is the high false positive rates. In the high dimensional network traffic data the distance between points becomes meaningless and tends to homogenize. This problem, known as 'the curse of dimensionality', prevents unsupervised approaches to accurately detect attacks [9]. The semi-supervised ML approaches are taking advantages of both supervised and unsupervised approaches by the ability to work on labeled and unlabeled datasets. Also, the combination of supervised and unsupervised approaches allows to increase accuracy and decreases the false positive rates. However, semi-supervised approaches are also challenged by the drawbacks of both approaches. Hence, the semi-supervised approaches require a sophisticated implementation of its components in order to overcome the drawbacks of supervised and unsupervised approaches.

In this paper we present an online sequential semi-supervised ML approach for DDoS detection. A time based sliding window algorithm is used to estimate the entropy of the network header features of the incoming network traffic. When the entropy exceeds its normal range, the unsupervised co-clustering algorithm splits the incoming network traffic into three clusters. Then, an information gain ratio [10] is computed based on the average entropy of the network header features between the network traffic subset of the current time window and each one of the obtained clusters. The network traffic data clusters that produce high information gain ratio are considered as anomalous and they are selected for preprocessing and classification using an ensemble classifiers based on the Extra-Trees algorithm [11].

Our approach constitutes of two main parts unsupervised and supervised. The unsupervised part includes entropy estimation, co-clustering and information gain ratio. The supervised part is the Extra-Trees ensemble classifiers. The unsupervised part of our approach allows to reduce the irrelevant and noisy normal traffic data, hence reducing false positive rates and increasing accuracy of the supervised part. Whereas, the supervised part is used to reduce the false positive rates of the unsupervised part and to accurately classify the DDoS traffic. To better evaluate the performance of the proposed approach three public network traffic datasets are used in the experiment, namely the NSL-KDD [12], the UNB ISCX IDS 2012 dataset [13] and the UNSW-NB15 [14, 15]. The experimental results are satisfactory when compared with the state-of-the-art DDoS detection methods.

The main contributions of this paper can be summarized as follows:

- Presenting an unsupervised and time sliding window algorithm for detecting anomalous traffic data based on co-clustering, entropy estimation and information gain ratio. This algorithm allows to reduce drastically the amount of network traffic to preprocess and to classify, resulting in a significant improvement of the performance of the proposed approach.
- Adopting a supervised ensemble ML classifiers based on the Extra-Trees algorithm to accurately classify the anomalous traffic and to reduce the false positive rates.
- Combining both previous algorithms in a sophisticated semi-supervised approach for DDoS detection. This allows to achieve good DDoS detection performance compared to the state-of-the-art DDoS detection methods.
- The unsupervised part of our approach allows to reduce the irrelevant and noisy normal traffic data, hence reducing false positive rates and increasing accuracy of the supervised part. Whereas, the supervised part allows to reduce the false positive rates of the unsupervised part and to accurately classify the DDoS traffic.

This paper is organized as follows. Section 2 highlights state-of-the-art DDoS detection methods. Section 3 presents a brief of the benchmark datasets used in this paper. The proposed approach is detailed in Section 4. Section 5 describes the conducted experiments and the performance metrics used to evaluate the proposed approach. The obtained results, the results discussion and the conducted comparisons are given in Section 6. Finally, Section 7 draws the conclusion and outlines future works.

2 Related works

Several approaches have been proposed for detecting DDoS attack. Information theory and machine learning are the

most common techniques used in the literature. This section summarizes some of the recent works in DDoS detection.

Akilandeswari V. et al. [16] have used a Probabilistic Neural Network to discriminate flash crowd events from DDoS attacks. The method achieves high DDoS detection accuracy with lowe false positives rates. Similarly, Ali S. B. et al. [17] have proposed an innovative ensemble of Sugeno type adaptive neuro-fuzzy classifiers for DDoS detection using an effective boosting technique named Marliboost. The proposed technique was tested on the NSL-KDD dataset and have achieved good performance. Mohiuddin A. and Abdun Naser M. [18] have proposed an unsupervised approach for DDoS detection based on the co-clustering algorithm. The authors have extended the co-clustering algorithm to handle categorical attributes. The approach was tested on the KDD cup 99 dataset and achieved good performance. Alan S. et al. [19] have proposed a DDoS Detection Mechanism based on ANN (DDMA). The authors used three different topologies of the MLP for detecting three types of DDoS attacks based on the background protocol used to perform each attack namely TCP, UDP and ICMP. The mechanism detects accurately known and unknown, zero day, DDoS attacks. Similarly, Boro D. et al. [20] have presented a defense system referred to as DyProSD that combines both the merits of feature-based and statistical approach to handle DDoS flooding attack. The statistical module marks the suspicious traffic and forwards to an ensemble of classifiers for ascertaining the traffic as malicious or normal. Recently, Van Loi C. [21] proposed a novel one-class learning approach for network anomaly detection based on combining auto-encoders and density estimation. Authors have tested their method on the NSL-KDD dataset, and obtained satisfactory results. Mohamed I. et al. [22] have proposed a supervised DoS detection method based on a feed-forward neural network. This method consists of three major steps: (1) Collection of the incoming network traffic, (2) selection of relevant features for DoS detection using an unsupervised Correlation-based Feature Selection (CFS) method, (3) classification of the incoming network traffic into DoS traffic or normal traffic. The approach achieves good performances on the UNSW-NB15 and NSL-KDD datasets. Mustapha B. et al. [23] have presented a two-stage classifier based on RepTree algorithm and protocols subset for network intrusion detection system. The first phase of their approach consists of dividing the incoming network traffic into three type of protocols TCP, UDP or Other. Then classifying it into normal or anomaly traffic. In the second stage a multi-class algorithm classify the anomaly detected in the first phase to identify the attacks class in order to choose the appropriate intervention. Two public datasets are used for experiments in this paper namely the UNSW-NB15 and the NSL-KDD.

The performances of network intrusion detection approaches, in general, rely on the distribution characteristics of the underlaying network traffic data used for assessment. The DDoS detection approaches in the literature are under two main categories unsupervised approaches and supervised approaches. Depending on the benchmark datasets used, unsupervised approaches often suffer from high false positive rate and supervised approach cannot handle large amount of network traffic data and their performances are often limited by noisy and irrelevant network data. Therefore, the need of combining both, supervised and unsupervised approaches arises to overcome DDoS detection issues.

3 Network traffic datasets

In this paper three datasets are used namely the NSL-KDD [12], the UNB ISCX IDS 2012 [13] and the UNSW-NB15 [14, 15].

The NSL-KDD dataset contains four types of attacks namely DoS, Probe, R2L and U2R. It has 41 features divided to three groups: Basic features, Traffic features and Content features. This dataset contains a total number of 148,517 records in both training and testing sets. We selected this dataset for three main reasons. First, it is widely used for IDSs benchmarking in the literature. Second, it contains an important percentage of attack traffic with 34.62% of the dataset corresponds to DoS attacks. Also, it overcomes some of the inherent problems of its predecessors KDD Cup'99 and DARPA'98 [12], such as records redundancy and duplication.

The UNB ISCX IDS 2012 dataset consists of labeled network traces, including full packet payloads in pcap format, which along with the relevant profiles. The dataset is built on profiles that contain detailed descriptions of intrusions and abstract distribution models for applications, protocols, or lower level network entities. A testbed network of 21 interconnected workstations is used to build 7 parts of the dataset during a week, one is dedicated to DDoS attacks only. This is the main reason why we used this dataset in this paper. The labeled ISCX-Dataset-15-June has 19 features and contains a total of 196,032 records of DDoS and normal traffic. We notice that 19.11% of the dataset correspond to DDoS traffic which makes it a very important benchmark for DDoS detection systems.

The UNSW-NB15 dataset contains nine types of modern attacks and new patterns of normal traffic. It has 49 features split into five groups namely Flow features, Basic features, Content features, Time features and Additional generated features. This dataset contains a total number of 257,705 records labeled whether by an attack type label or a normal label. A number of 16,353 records correspond to the DDoS

attack which represents 6.34% of the dataset. Three major reasons motivated us to use the UNSW-NB15 dataset. The dataset contains modern normal and attack traffic, it is well structured and comprehensible and it is more complex than other previous datasets which makes it a good benchmark to evaluate our approach.

The datasets above are split into train subsets and test subsets using a configuration of 60% and 40% respectively. The train subsets are used to fit the Extra-Trees ensemble classifiers and the test subsets are used to test the entire proposed approach. Before fitting the classifiers the train subsets are normalized using the $MinMax$ method presented in Section 4.3.2. Also, we notice that other attacks traffic are filtered from the datasets as this paper addresses only the detection of DDoS attacks.

4 The proposed approach

This section presents the details of the proposed approach and the methodology followed for detecting the DDoS attack. The proposed approach consists of five major steps: Datasets preprocesing, estimation of network traffic Entropy, online co-clustering, information gain ratio computation and network traffic classification. Our methodology to detect DDoS attack is illustrated in Fig. 1.

4.1 Network traffic entropy estimation

A time based sliding window algorithm is used to estimate the entropy of a set of network flow-header features. By definition the entropy is a measure of the diversity or the randomness of a distribution, e.g. network flow data [24]. The analysis of the network flow entropy over time windows allows to reduce high dimensionality of the network traffic distribution to a single metric describing its dispersion [25, 26]. The flow size distribution (FSD) features, the source/destination packets count and the source/destination bytes count, are used to estimate the entropy. The reason for using the FSD features is that during a DDoS attack the zombie hosts send a large number of packets to the victim which generates a large amount of network flow data. Hence, estimating entropy of the FSD features allows to detect sensitively the abrupt changes in the network flow caused by DDoS attack. The Shannon formula is used to estimate the FSD entropy which is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \cdot \log(p(x_i)) \tag{1}$$

Where X represents a FSD feature, $x_i (i = 1, \cdots, n)$ are the frequencies of items of X received during the current time window and n is the total number of items of X. To better interpret the estimated entropy values, the FSD entropy values are normalized using the formula: $H_0(X) = \frac{H(X)}{\log n}$.

The pseudo-code of the Entropy estimation time-based sliding window algorithm is given in Algorithm 1. The Algorithm 1 gives as output the average of the computed entropy of the FSD features set S. This allows us to use only two thresholds, lower (δ_l) and upper (δ_u), for each S in order to detect anomaly within the incoming network traffic.

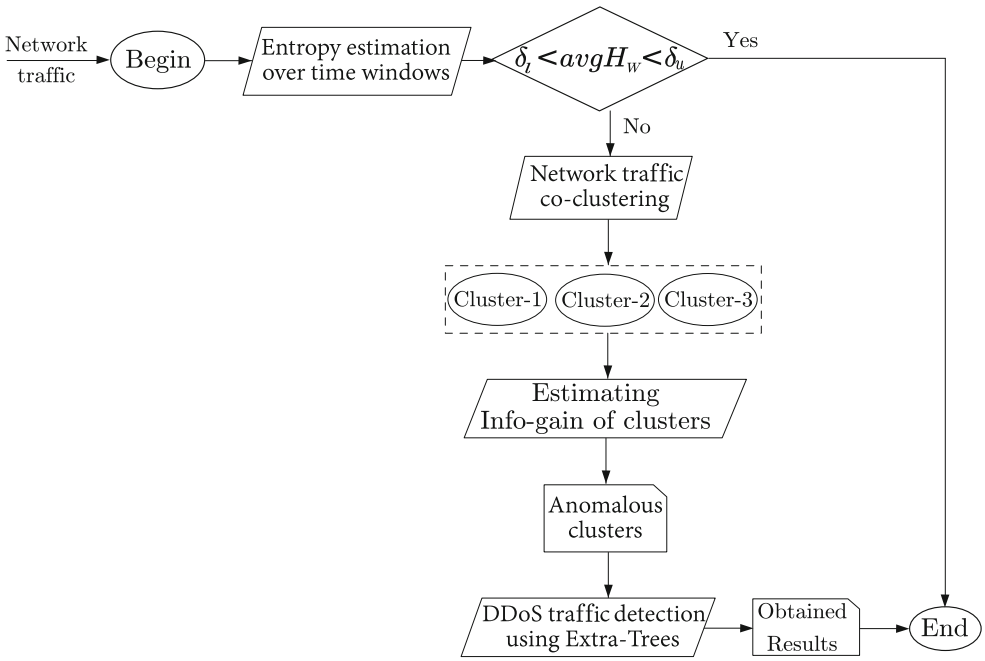

Fig. 1 Flowchart of the proposed approach

Algorithm 1 Estimating the entropy of the network FSD features

avgEntropy $(data, S)$
Input : Network traffic data $data$, Features set S
Output : $AverageEntropy$
$H_S \leftarrow \{\}$
foreach X **in** S **do**
  $p(x_i) \leftarrow \frac{\sum_j^N x_i}{N}$ //Estimate the probability mass function
  $H(X) \leftarrow -\sum_i^n p(x_i) \cdot \log p(x_i)$ //Compute the Shannon entropy
  $H_0(X) \leftarrow \frac{H(X)}{\log n}$ //Normalize the entropy
  $H_S \leftarrow H_S \cup H_0(X)$
end
$AverageEntropy \leftarrow \frac{sum(H_S)}{4}$ //Return average of the FSD entropy time series

4.2 Anomalous network traffic clustering

Here, the anomalous network traffic data is split into three clusters using a time-based sliding window co-clustering algorithm. The aim of splitting the anomalous network traffic is to reduce the amount of data to be classified by excluding the normal cluster for the classification. For DDoS detection normal traffic records are irrelevant and noisy as the normal behaviors continue to evolve. Most of the time the new unseen normal traffic instances cause the increase of the false positive rate and the decrease of the classification accuracy. Hence, excluding some noisy normal instances of the network traffic data for classification is beneficial in terms of low false positive rates and classification accuracy. Assuming that after the network traffic clustering one cluster contains only normal traffic, a second one contains only DDoS traffic and a third one contains both DDoS and normal traffic.

4.2.1 Co-clustering algorithm

Co-clustering algorithm performs a simultaneous clustering of rows and columns of a data matrix based on a specific criterion [27, 28]. It produces clusters of rows and columns which represent sub-matrices of the original data matrix with some desired properties. Clustering simultaneously rows and columns of a data matrix yields three major benefits:

1. Dimensionality reduction, as each cluster is created based on a subset of the original features.
2. More compressed data representation with preservation of information in the original data.
3. Significant reduction of the clustering computational complexity. The co-clustering computational complexity is $\mathcal{O}(mkl + nkl)$ which is much smaller than that of the traditional Kmeans algorithm $\mathcal{O}(mnk)$ [28]. Where m is the number of rows, n is the number of columns, k is the number of clusters and l is the number of column clusters.

4.3 Data preprocessing

Here, we aim to prepare data in the anomalous clusters for classification. For this purpose during each time window a set of relevant features is selected and the received network traffic data is normalized.

4.3.1 Feature selection

As mentioned in the previous section the co-clustering algorithm can be used as a dimensionality reduction technique. Each cluster produced by the co-clustering is based on a subset of the original features set. Since we aim to classify the two anomalous clusters produced by the co-clustering algorithm, we combine their corresponding feature subsets as shown in Fig. 2. This allows to preserve information of both clusters and to update the subset of relevant features. This is beneficial since the attackers are continually updating their tools and changing their behaviors, and the existing online network datasets suffer from lack of modern normal and attack traffic scenarios.

4.3.2 Data normalization

Generally, values of attributes in a network traffic dataset are not distributed uniformly. It is important to maintain a uniform distribution of each attribute values before starting the learning process. For this purpose we use the *MinMax* method. In *MinMax* the values of features are scaled to the interval [0, 1] as follows:

$$x_i^{new} = \frac{x_i - \min(X)}{\max(X) - \min(X)} \tag{2}$$

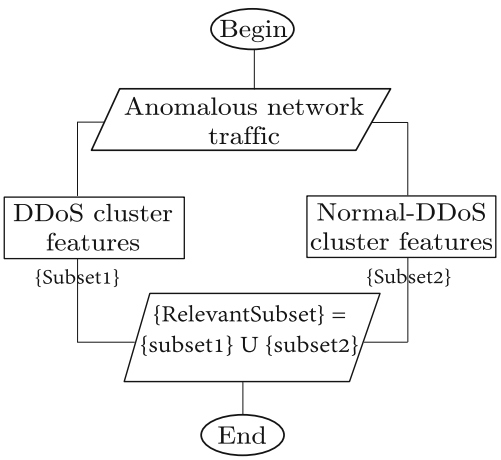

Fig. 2 Feature selection method

Where X is a relevant feature, x_i is a possible value of X within the current time window and x_i^{new} is the normalized value.

4.4 Network traffic classification

In this section we give the details of the adopted Extra-Trees ensemble classifiers for DDoS detection and the entire algorithm of the proposed method.

4.4.1 Extra-Trees ensemble classifiers

Generally, univariate decision trees are the most popular decision trees because of their low computational complexity. However, the lack of large datasets often restricts the representational power of a decision tree. Ensemble-based trees such as Random Forest and Extra-Trees are useful to overcome the representational problem of the univariate decision tree. Hence, ensemble-based trees are widely used for classification and regression problems [29].

Extremely randomized trees (Extra-Trees) is a tree-based ensemble method for supervised classification and regression problems [11]. It essentially consists of randomizing strongly both attribute and cut-point choice while splitting a tree node. The Extra-Trees algorithm builds an ensemble of unpruned decision or regression trees according to the classical top-down procedure. Its two main differences with other tree-based ensemble methods are that it splits nodes by choosing cut-points fully at random and that it uses the whole learning sample (rather than a bootstrap replica) to grow the trees.

From the bias-variance point of view, the explicit randomization of the cut-point and attribute combined with ensemble averaging reduces the variance more strongly than the randomization schemes used by other methods. Also, the use of the full original learning sample rather than bootstrap replicas allows to minimize bias. Overall, Extra-Trees achieves good bias-variance trade-off for classification problems. Moreover, the algorithm is strength in terms of accuracy and computational efficiency.

From the computational point of view, the complexity of the tree growing procedure is on the order of $\mathcal{O}(MNKlogN)$ with respect to learning sample size. Where, N is the number of samples, K is the number of variables randomly drawn from each node and M represents the number of trees in this ensemble.

4.4.2 Entire proposed approach

This sections introduces our methodology to detect the DDoS attack. The five-fold steps application process of data mining techniques in network systems discussed in [7] characterizes the followed methodology.

The main aim of combining algorithms used in the proposed approach is to reduces noisy and irrelevant network traffic data before preprocessing and classification stages for DDoS detection while maintaining high performance in terms of accuracy, false positive rate and running time, and low resources usage.

Our approach starts with estimating the entropy of the FSD features over a time-based sliding window. When the average entropy of a time window exceeds its lower or upper thresholds the co-clustering algorithm split the received network traffic into three clusters.

Entropy estimation over time sliding windows allows to detect abrupt changes in the incoming network traffic distribution which are often caused by DDoS attacks. Incoming network traffic within the time windows having abnormal entropy values is suspected to contain DDoS traffic. The focus only on the suspected time windows allows to filter important amount of network traffic data, therefore only relevant data is selected for the remaining steps of the proposed approach. Also, important resources are saved when no abnormal entropy occurs.

In order to determine the normal cluster, we estimate the information gain ratio [10] based on the average entropy of the FSD features between the received network traffic data during the current time window and each one of the obtained clusters. As discussed in the previous section during a DDoS period the generated amount of attack traffic is largely bigger than the normal traffic. Hence, estimating the information gain ratio based on the FSD features allows to identify the two cluster that preserve more information about the DDoS attack and the cluster that contains only normal traffic. Therefore, the cluster that produce lower information gain ratio is considered as normal and the remaining clusters are considered as anomalous. The information gain ratio is computed for each cluster as follows:

$$Gain_i(subset_w, C_i) = avgH(subset_w) - \frac{|C_i|}{|subset_w|} * avgH(C_i)\ (i = 1, 2, 3) \quad (3)$$

Where $subset_w$ represents the received subset of network data during the time window w, $C_i (i = 1, 2, 3)$ are the obtained clusters from $subset_w$ and $|C_i|$ is the size of the i^{th} cluster. $avgH(subset)$ is the average entropy of the FSD features of the input $subset$ and $|subset|$ represents the size of the $subset$.

The clustering of the incoming network traffic data allows to reduce important amount of normal and noisy data before the preprocessing and classification steps. More than 6% of a whole traffic dataset can be filtered at this step, for instance for the UNSW-NB15 dataset (Section 6.4.3). In the first two steps of the proposed approach entropy, information gain ratio and co-clustering algorithms are

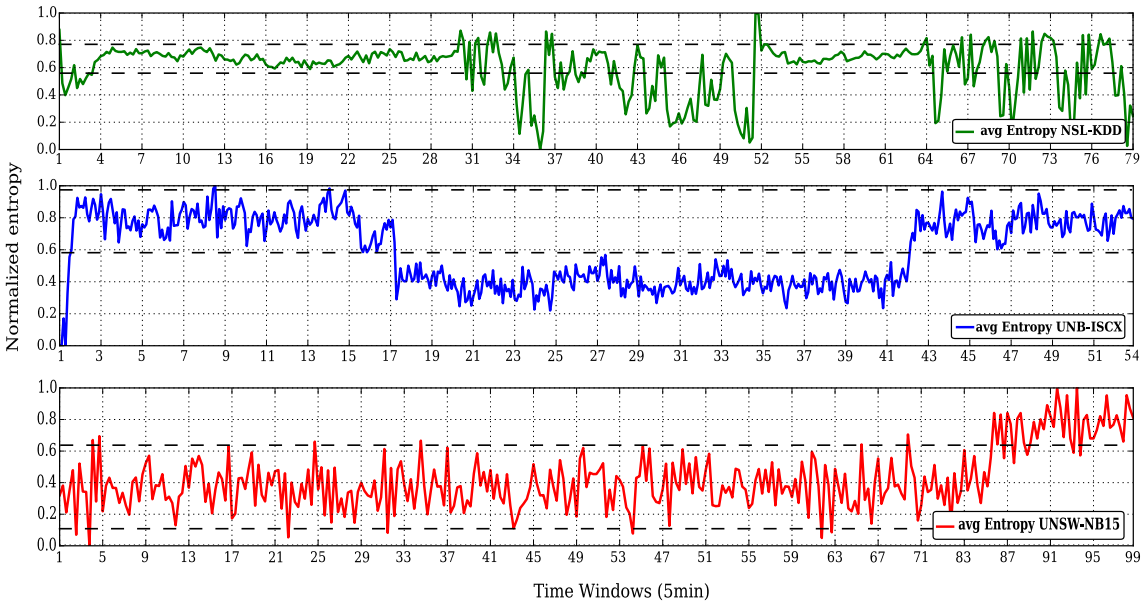

Fig. 3 Estimated average Entropy of network FSD features on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets

used to filter noisy normal traffic data and select relevant suspicious traffic data for DDoS detection. Furthermore, this unsupervised part of our method transform network traffic data from flow format or time series format into time window format. This allows to preprocess and classify the network traffic data of each time window separately. The classification results of each time window are interpreted isolatedly which allows to early take DDoS mitigation actions.

The anomalous clusters are selected for pre-processing and classification using the Extra-Trees algorithm, as illustrated in Algorithm 2. Generally, in network flow data the amount normal flows is far bigger than that of the attack flows. Otherwise Internet services would not work normally. Except during a DDoS, the amount of attack flows get bigger than the normal one at the victims' sides. This general characteristic of the network flow data is noticed in the three datasets used in this paper as shown in Section 3. This effect is reflected in the models learned from these datasets which are often highly biased causing low detection accuracy and high false positive rates. As discussed in Section 4.4.1 Extra-Tree algorithm is able to achieve good bias-variance trade-off despite the underlying learning samples. This is a major advantage of the Extra-Trees algorithm over other ensemble learning methods such as Random Forest [30], Bagging [31], etc. The reasons above are our motivation to use Extra-Trees in the proposed DDoS detection approach.

Algorithm 2 Semi-supervised DDoS detection approach

Semi-DDA $(data, w)$ //Semi-supervised DDoS Detection Approach
Input : network traffic data, time window size w
Output: Classification Results
$S \leftarrow \{src\_pkts,\ dst\_pkts,\ src\_bytes,\ dst\_bytes\}$ //FSD features set
$W \leftarrow w$
while *incoming network traffic* **do**
  foreach W **do**
    $data_W \leftarrow \{incoming\_network\_traffic\_data\}$
    $avgH^W \leftarrow avgEntropy(data_W, S)$
    if $avgH^W \notin]\delta_u, \delta_l[$ **then**
      $clusters \leftarrow Co - Clustering(data_W)$
      for cl **IN** $clusters$ **do**
        $Gain(data_W, cl) \leftarrow avgEntropy(data_W, S) - avgEntropy(cl, S) * \frac{size\ of\ cl}{size\ of\ data_W}$
      end
      $AnomalousClusters \leftarrow \{clusters \setminus \min(Gain(data_W, cl))\}$
      $Preprocessing(AnomalousClusters)$
      $Results \leftarrow ExtraTrees(AnomalousClusters)$
    end
  end
end

The pseudocode of the entire proposed approach is given in Algorithm 2. The Algorithm 2 starts with setting the time window size. To efficiently maintaining the trade-off between DDoS detection performances and accurately detecting abrupt changes in the network flow distribution, we picked a 5 min time window. Also, the time window size has a direct impact on resources consumption. On one hand, using a large time window is costly in term of memory and it causes a smoothing of the estimated entropy curve as shown in Fig. 4 which prevent to see the abrupt changes of the network. Moreover, in the case of a large DDoS the victim services may fall within the time window. On the other hand, a small time window is costly in term of computation. More discussion of the time window setting is presented in Section 6.2.

5 Experiments

This section describes the experiments conducted and the performance metrics used to evaluate the proposed approach. To validate the contributions of our approach three datasets are used namely the NSL-KDD, the UNB ISCX IDS 2012 and the UNSW-NB15. The datasets were split into two subsets where 60% of each dataset is used to train the Extra-Trees ensemble classifiers and the remaining 40% is used to test the entire approach. The proposed approach is developed using the Python programming languages using several packages for machine learning, data analysis and data visualization including Scikit-learn [32], Numpy [33], Pandas [34] and Matplotlib [35]. The hardware used in the experiments is a core i3 2.4 GHz and 6 GB of memory running under Debian 8 ×64.

5.1 Performance metrics

The performance metrics used to assess obtained experimental results are computed based on the standard confusion matrix: True Positives (TP) refer to records correctly classified as attack, False Positives (FP) refer records incorrectly classified as attack, True Negatives (TN) are records correctly classified as normal and False Negatives (FN) are records incorrectly classified as normal. The performance metrics used are defined as follows:

$$\textbf{Accuracy} = 100 \times \frac{TP + TN}{total \;\; records} \tag{4}$$

$$\textbf{False Positive Rate (FPR)} = 100 \times \frac{FP}{FP + TN} \tag{5}$$

$$\textbf{F-measure} = 2 \times \frac{precision \times recall}{precision + recall} \tag{6}$$

Where: $precision = \frac{TP}{TP+FP}$, $recall = \frac{TP}{TP+FN}$.

$$\textbf{Cluster Purity} = \frac{1}{n} \sum_{i=1}^{k} \max_{j} (c_i \cap t_j) \tag{7}$$

Where n is the number of records in data inputed to the co-clustering algorithm, k is the number of clusters, c_i is a

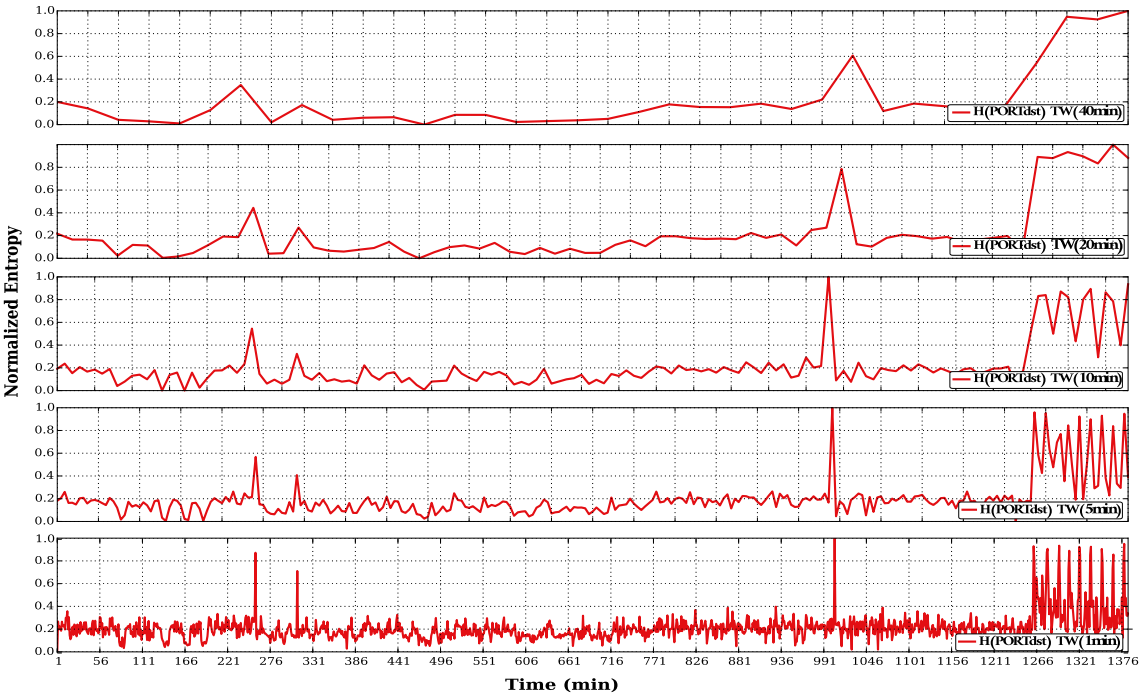

Fig. 4 Estimated entropy of PORTdst feature of UNB ISCX 12 dataset for different time window sizes

generated cluster and t_j represents the class which has the max count for cluster c_i.

6 Results and discussion

In this section we give the obtained results of the experiments. The obtained results illustrate the contribution of each component of the proposed approach and the entire approach. To validate the results we compared them with the state-of-the-art DDoS detection approaches.

6.1 Network entropy estimation

The Fig. 3 shows the estimated entropy of the FSD features in the experiment datasets. For each entropy time series the upper and lower thresholds are represented by the dash lines. The thresholds are estimated by using the max and the min entropy of normal intervals in train subsets of each dataset; portions of the datasets that contain only normal traffic. When the entropy time series curve breaches its lower or upper threshold the network traffic received within the current time window is selected for clustering.

Estimating entropy over a time sliding window is important to reduce the amount of data to preprocess and to classify by targeting only time windows that contain network anomalies. The test subsets of NSL-KDD, UNB ISCX 12 and UNSW-NB15 are reduced respectively by 26.6%, 23.5% and 22.8%.

6.2 Time window size setting

This section discusses the impact of the time window size on the entropy estimation algorithm and on the performance of the entire proposed approach in terms of accuracy and FPR.

The network traffic datasets used in this paper are flow-based records where each record is captured for a duration of 1 s [12] or less [13–15]. Therefore a time window represents a subset of network traffic containing a certain number of records. This implies that for each time window corresponds a different network traffic data distribution. In other words, the change of the time window size imply a change in the network traffic data distribution.

For each time window the entropy estimation algorithm gets a different network traffic distribution which produces different entropy values. Figure 4 illustrates this with different entropy estimation curves for different time window sizes. The PORTdst feature of the UNB ISCX 12 dataset is used to produce the entropy curves in Fig. 4. As shown in Fig. 4 the increase of the time window size produces low entropy values which causes a smoothing of the entropy curve. This smoothing effect prevent to detect the abrupt changes in the network traffic, hence the prevention of detecting DDoS attacks.

This effect may be due to the creation process of the used network traffic datasets where the DDoS traffic constitutes small groups of records separated by normal traffic. The fact of increasing the time window size during a DDoS attack imply the addition of more normal traffic. Therefore, the normal traffic distribution size increases and becomes dominant which causes a decrease of the uncertainty in the network traffic, hence a decrease of the estimated entropy which prevents to see the abrupt changes in network traffic.

The time window size has also a major impact on the performance of the proposed approach in terms of accuracy and false positive rates. As illustrated in Fig. 5 the accuracy of proposed approach continues to fluctuate slightly with the variation of the time window size for different datasets. Hence the accuracy of the proposed approach is not strongly related to the variation of the time window size. Whereas, Fig. 6 shows that the increase of the time window size implies the increase of the FPR. Therefore, the use of a small time window size is most reliable to produce good performance of the proposed approach.

For the reasons discussed above we picked up a 5 min time window for our proposed approach. As illustrated in Fig. 4 the 5 min time window allows to detect the abrupt changes in the network traffic. Also, this time window size

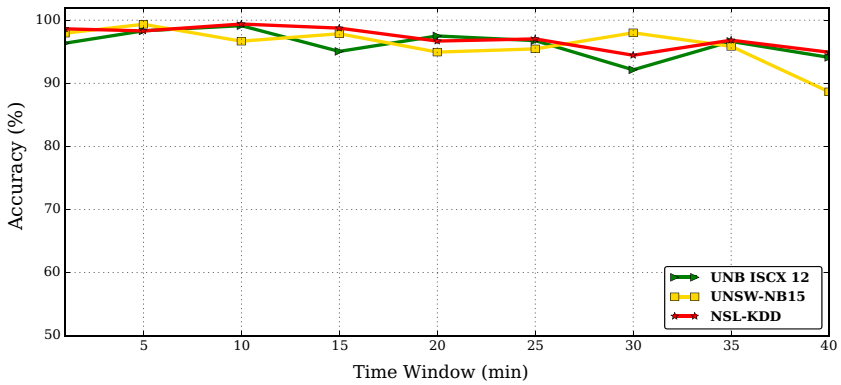

Fig. 5 Variations of accuracy of the proposed approach on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets for different time window sizes

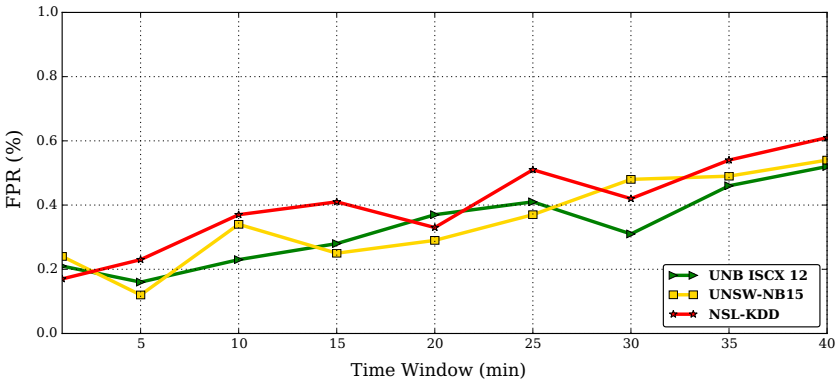

Fig. 6 Variations of FPR of the proposed approach on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets for different time window sizes

depicts good DDoS detection accuracy with low FPR of the proposed approach for different datasets as illustrated in Figs. 5 and 6.

6.3 Determination of optimal parameters of Extra-Trees and co-clustering

In this section the parameters of Extra-Trees and co-clustering algorithms are tuned in order to achieve high performances. For this purpose a procedure based on the Grid Search Cross Validation (GridSearchCV) [32] method is used. GridSearchCV performs an exhaustive search for the best parameters of the tuned model according to certain predefined performance metrics, for instance accuracy and mean error, or metrics defined manually. The procedure start by defining intervals of parameters' values, and then used them to construct a dictionary of the parameters and their corresponding values. For each combination of parameters' values of deferent elements in the dictionary a 10-fold cross validation procedure is performed.

The average of the performance obtained from 10-fold cross validation is computed for each element in the dictionary. Parameters' combination that produces high performance is then selected for initializing the ML algorithm. Despite that GridSearchCV allows to achieves high performance of the tunned ML algorithms, it is highly time-consuming especially for large dictionaries and datasets.

To find optimal parameters of the co-clustering algorithm the GridSearchCV procedure is applied on the NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets. The parameters used as well as the parameters' definition and dictionary are detailed in Table 1. A set of the co-clustering parameters are not used for tuning because of their nature such as the number of clusters to find, etc.

The optimal parameters values obtained using the GridSearchCV for the three datasets are tabulated in Table 2. Mean $Purity$ metric is the criterion used to select the best parameters values of the Co-Clustering algorithm. A high mean $Purity$ of 100% is noticed for all the dataset for many elements in the parameter dictionary of co-clustering.

The parameters that constitute the parameters' dictionary of the Extra-Trees algorithms are given in Table 3. We notice that only a partial set of Extra-Trees parameters are used in the tuning procedure, other parameters are avoided because of their nature or for reducing the execution time.

For each element in the parameters' dictionary of Extra-Trees algorithm GridSearchCV computes the mean accuracy and mean standard error. The best mean accuracy vlaues obtained on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets are respectively 99.85%, 99.99% and 99.22%. Whereas, the best mean standard error results obtained on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets are respectively 0.001%, 0.00008% and 0.009%. The optimal parameters of Extra-Trees for the three datasets are tabulated in Table 4.

6.4 Network traffic clustering

In this section we give the obtained results of the co-clustering algorithm when applied to each dataset. The

Table 1 Co-Clustering parameters tunned using the GridSearchCV procedure

Parameter	Definition	Values' interval
init	String, method for initialization of k-means algorithm	['k-means++','random']
svd_method	String, selects the algorithm for finding singular vectors	['randomized', 'arpack']
n_init	Integer, number of random initializations that are tried	[5, 10, 15, 20, 25, 30]
random_state	Integer, a pseudo random number generator used by the K-Means initialization.	[10, 15, 20, 25, 30]
mini_batch	Bool, whether to use mini-batch k-means	[False, True]

Table 2 Optimal Co-Clustering parameters set obtained using the GridSearchCV procedure

Dataset	init	svd_method	n_init	random_state	mini_batch
NSL-KDD	'random'	'arpack'	10	15	False
UNB ISCX 12	'k-means++'	'arpack'	5	15	True
UNSW-NB15	'random'	'arpack'	30	20	False

results are collected at the time windows containing anomalous traffic.

6.4.1 Co-clustering of the NSL-KDD dataset

The clustering results of the NSL-KDD dataset are shown in Fig. 7. It is obvious from Fig. 7 that the clustering $Purity$ of all the time windows exceeds 90%. This explains that among the three obtained clusters at each time window there is a dominant class whether normal or attack. Also, we noticed high FPR of the co-clustering algorithm for this dataset during the clustering phase, which is caused by the presence of a large number of normal records in the test subset. Hence the need to reduce the FPR during the classification stage. It is worth noting that 6.03% of the test subset is reduced in this step. Furthermore, as shown in Fig. 7 high $precision$ scores are obtained at the 1th, the 5th and the 7th time windows sizes. Whereas $F-measure$ and $recall$ scores continue to fluctuate during clustering phase.

6.4.2 Co-clustering of the UNB ISCX 12 dataset

Figure 8 shows the clustering results of the UNB ISCX 12 dataset. The co-clustering algorithm achieved its high $Purity$ of nearly 100% for all the suspected time windows. Whereas for this dataset the co-clustering algorithm achieved low $F-measure$, $precision$ and $precision$ scores. However, the co-clustering algorithm exhibits high FPR for this dataset, nearly an average of 50% is achieved for all the time windows. Moreover, it is worth noting that co-clustering algorithm has reduced a total of 5.86% of the normal network traffic in the test subset of the UNB ISCX 12 dataset.

6.4.3 Co-clustering of the UNSW-NB15 dataset

The obtained results from the clustering of the UNSW-NB15 dataset are presented in Fig. 9. As shown in Fig. 9 co-clustering algorithm achieves high $Purity$ scores nearly 100% for all the time windows except for the 1st and the 6th where the $Purity$ of 82.29% and 72.7% are respectively achieved. Also, for this dataset the co-clustering algorithm depicts a stable $precision$ scores and fluctuating $F-measure$ and $recall$ scores. Furthermore, a high FPR is also observed for this dataset. This may be due to two main reasons. First, the UNSW-NB15' size is larger compared to the other datasets. Second, the dataset contains lower percentage of DDoS attack traffic of 6.34% compared to 19.11% for the UNB ISCX 12 and 34.62% for the NSL-KDD. Also, it is worth noting that the co-clustering have reduced 6.77% of the normal network traffic in the test subset of this dataset.

6.4.4 Co-clustering and information gain ratio for data reduction

In this section data reduction results using co-clustering and information gain ratio algorithms are presented and discussed. Co-clustering algorithm is adopted in the proposed method to split the incoming network traffic into three clusters. Assuming that one cluster contains only DDoS traffic, a second one contains only normal traffic and a third one contains both types of traffic. As discussed in the previous sections co-clustering achieves high clustering performances especially high $Purity$ for the three datasets used in this paper. The high $Purity$ achieved during the suspicious time windows explains that there is a dominant class in the obtained clusters whether normal or attack. Information gain ratio algorithm is used to distinguish the class type of each cluster as explained in Section 4.4.2. The two clusters that contain DDoS traffic are selected for preprocessing and classification. Whereas the cluster that contains only normal traffic is filtered at this step allowing us to reduce important amount of network traffic data.

The results of the data reduction for NSL-KDD, ISCX UNB 12 and UNSW-BN15 datasets are presented in Fig. 10.

Table 3 Extra-Trees parameters tunned using the GridSearchCV procedure

Parameter	Definition	Values
n_estimators	Integer, number of trees in the forest	[5, 10, 15, 20, 25]
criterion	String, function to measure the quality of a split	[gini, entropy]
max_depth	Integer, maximum depth of the tree	[1, 5, 10, 15, 20, 25, 30]
random_state	Integer, seed used by the random number generator	[0, 5, 10, 15, 20, 25, 30]

Table 4 Optimal Extra-Trees parameters obtained using the GridSearchCV procedure

Dataset	n_estimators	random_state	criterion	max_depth
NSL-KDD	30	0	entropy	20
UNB ISCX 12	10	20	gini	10
UNSW-NB15	30	10	entropy	25

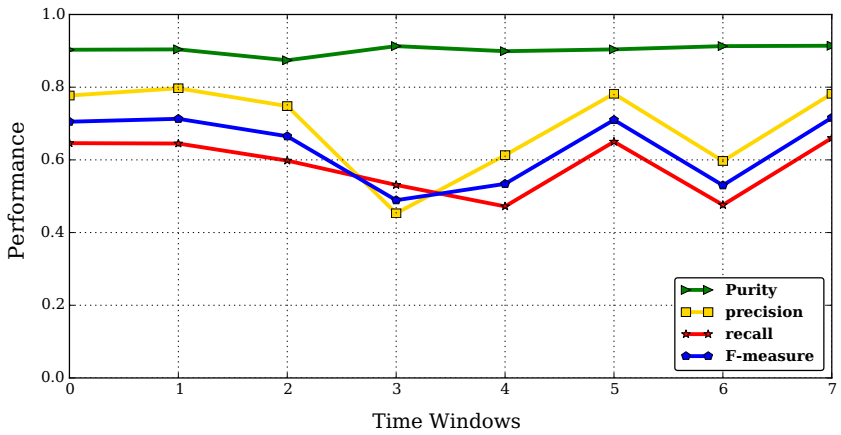

Fig. 7 Average co-clustering performance for each suspected time window in test set of NSL-KDD dataset

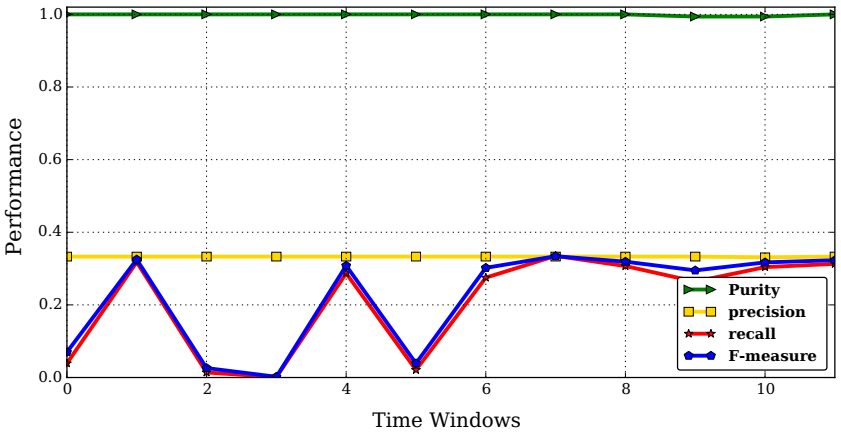

Fig. 8 Average co-clustering performance for each suspected time window in test set of UNB ISCX 12 dataset

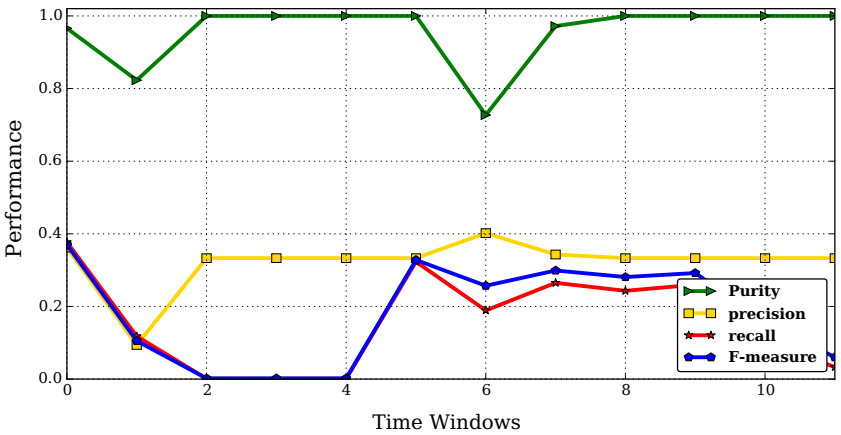

Fig. 9 Average co-clustering performance for each suspected time window in test set of UNSW-NB15 dataset

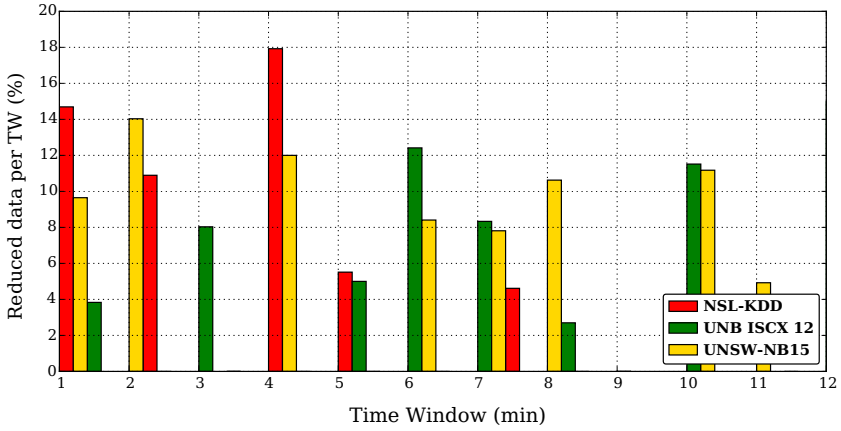

Fig. 10 Percentage data reduced for each suspected time window using Co-clustering and Information Gain Ratio

These results represent the percentage of the normal traffic data reduced during each suspected time window for each of the three datasets used in this paper. The highest reduction amount of normal traffic data is achieved for the NSL-KDD dataset during the 4th time window where 17.92% is reduced. Also, the amount of 15% and 12.41% were reduced respectively for UNB ISCX 12 dataset and UNSW-NB15 dataset. However, the amount of normal traffic data reduced compared to the whole test subsets of NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets are respectively 6.03%, 5.86% and 6.77%. Also, it is worth noting that the total amounts of data reduced by the Entropy estimation, the Co-clustering and the Information gain ratio algorithms combined are: 32.63% for NSL-KDD dataset, 29.36% for UNB ISCX 12 dataset and 29.57% for UNSW-NB15 dataset.

The obtained results show that important amounts of normal traffic data were reduced for the three datasets. The reason why Entropy estimation succeeded to reduce traffic data is due to the distribution of the DDoS traffic in the datasets, during DDoS time windows DDoS traffic increases compared to the normal traffic. As a result abrupt changes occur in the traffic distribution generating abnormal entropy values. This allows Entropy estimation algorithm to select efficiently suspected time windows that generate abnormal entropy. Also, the high $Purity$ achieved during the network traffic clustering of each suspected time window, which reflect the presence of a dominant class in each cluster, allows the co-clustering and information gain ratio algorithms to filter the normal traffic.

In addition to the important data reduction achieved, we notice a lower memory usage of the Entropy estimation, the Co-clustering and the Information gain ratio algorithms for the three datasets as shown in Fig. 11b. Also, the results in Fig. 11a depict a lower CPU consumption that does not breach 32% compared to DDMA [19], DyProSD [20] that achieved high CPU load of respectively 70% and 92%.

6.5 Extra-Trees performance

The results presented in Table 5 correspond to accuracy and FPR of Extra-Trees algorithm tested directly on the NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets. Extra-Trees achieves highest accuracy of 86.57% on the UNSW-NB dataset. However, it is obvious from Table 5 that Extra-Trees has low FPR for the three datasets. This represents an important advantage of Extra-Trees.

6.6 Entire semi-supervised approach

Here, we present the obtained results of the entire proposed approach tested on the NSL-KDD, UNB ISCX 12 and

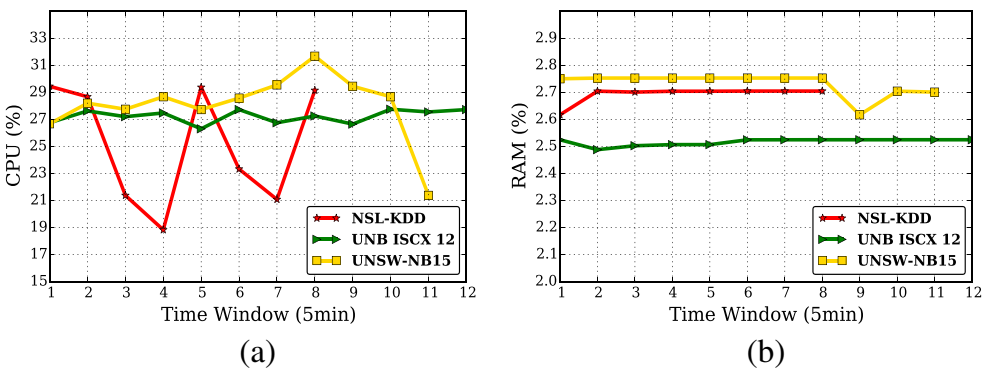

Fig. 11 CPU load (**a**) and RAM usage (**b**) of the Entropy estimation, the Co-clustering and the Information gain ratio algorithms

Table 5 Accuracy and FPR of Extra-Trees tested directly on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets

NSL-KDD		UNB ISCX 12		UNSW-NB15	
Accuracy (%)	FPR (%)	Accuracy (%)	FPR (%)	Accuracy (%)	FPR (%)
82.73	0.33	61.22	0.50	86.57	0.36

UNSW-NB15 datasets. The results in Table 6 represent the variations of accuracy and FPR over the time windows that contain anomalies for each datasets. The proposed approach achieves its highest overall accuracy on the UNB ISCX 12 dataset with 99.88%, followed by the NSL-KDD dataset with 98.23% and by the UNSW-NB15 with 93.74%. This lowest accuracy on UNSW-NB15 may be caused by the high similarity between the normal and the attack behaviors in the dataset. Also, the unbalanced normal and DDoS traffics between the train and test subsets, where the train set contains a small number of DDoS instances, leads to a weak detection model. Moreover, a high overall FPR of 0.46% is observed for this dataset. Whereas low FPR scores of 0.33% and 0.35% are achieved respectively on NSL-KDD and UNB ISCX 12 datasets.

The performance of the entire proposed approach are satisfactory first when compared to the results of the co-clustering and the Extra-Trees tested separately. Compared to the Extra-Trees performance illustrated in Table 5, the accuracy is enhanced by 15.5%, 38.66% and 7.14% for respectively NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets. Whereas, the FPR of Extra-Trees was reduced by 0.15% for the UNB ISCX 12 dataset and increased slightly by 0.1% for the UNSW-NB15 dataset. However, for the NSL-KDD dataset the overall FPR remains steady.

Moreover, compared to the co-clustering the FPR of the entire approach was drastically reduced by 33.67%, 49.65% and 29.33% for respectively NSL-KDD, UNB ISCX 12 and UNSW-NB15. Also, it is worth noting that the amount of the network traffic data is drastically reduced by both the entropy estimation algorithm and the co-clustering algorithm. The total amount of data reduced are 32.63%, 29.36% and 29.57% for respectively NSL-KDD, UNB ISCX 12 and UNSW-NB15.

The overall obtained results are consistent with our assumption that reducing irrelevant normal data from the network traffic enhances the DDoS detection accuracy and reduces the FPR.

6.7 Results comparison with state-of-the-art

For further validation of the obtained results the proposed approach is compared with the state-of-the-art DDoS detection methods, the comparison results are given here. For this purpose we used the performance of the proposed approach on the NSL-KDD dataset, as it is the

Table 6 Extra-Trees accuracy and FPR variations over time windows applied on NSL-KDD, UNB ISCX 12 and UNSW-NB15 datasets

NSL-KDD		UNB ISCX 12		UNSW-NB15	
Accuracy(%)	FPR(%)	Accuracy(%)	FPR(%)	Accuracy(%)	FPR(%)
98.41	0.33	99.97	0.33	66.28	0.34
98.72	0.33	99.96	0.33	81.65	0.50
98.21	0.33	99.97	0.33	100	0.50
98.46	0.33	100	0.50	100	0.50
98.21	0.33	99.97	0.33	100	0.50
97.74	0.34	99.99	0.33	100	0.50
98.67	0.33	99.96	0.33	79.48	0.36
97.39	0.34	99.96	0.33	97.22	0.50
NA	NA	99.97	0.33	99.92	0.33
NA	NA	99.41	0.33	100	0.50
NA	NA	99.43	0.33	100	0.50
NA	NA	99.93	0.33	100	0.50
98.23	**0.33**	**99.88**	**0.35**	**93.71**	**0.46**

Bold entries represent the average of the values in its corresponding column

Table 7 Comparison of the proposed approach with the state-of-the-art DDoS detection methods

Approach	Accuracy (%)	FPR (%)
Proposed approach	98.23	0.33
Hybrid-ADE	97.40	0.45
Marliboost	96.38	0.01

most benchmark dataset used in the state-of-the-art. The performances of the proposed approach are compared with Hybrid-ADE [21] and Marliboost [17]. The comparison results are summarized in Table 7.

It is clearly shown in Table 7 that the proposed approach achieves the highest accuracy of 98.23% exceeding the Hybrid-ADE that achieved 97.40%. Also, in term of FPR our approach exceeds the Hybrid-ADE by 0.12%. However, Marliboost achieved a lowest FPR of 0.01%. Although, Marliboost has the lowest FPR, it is limited in term of accuracy. Compared to the state-of-the-art the performance of the proposed semi-supervised approach are satisfactory.

7 Conclusion

In this paper, we have proposed a semi-supervised DDoS detection approach based on entropy estimation, co-clustering, information gain ratio and the Extra-Trees ensemble classifiers.

The entropy estimator estimates and analyzes the network traffic data entropy over a time-based sliding window. When the entropy exceeds its limits, the received network traffic during the current time window is split into three clusters using the co-clustering algorithm. Then, an information gain ratio is computed based on the average entropy of the network header features between the current time window subset and each one of the obtained clusters. The network traffic data clusters that produce high information gain ratio are considered as anomalous and selected for preprocessing and classification using an ensemble classifiers based on the Extra-Trees algorithm.

Various experiments were conducted in order to assess the performance of the proposed method using three public benchmark datasets namely the NSL-KDD, the UNB ISCX 12 and the UNSW-NB15. The experiment results, in terms of accuracy and false positive rate, are satisfactory when compared with the state-of-the-art DDoS detection methods.

Despite, that the proposed approach shows good performances with the public benchmark datasets, it is important to evaluate its performances in real world scenarios. For future work, we are planning to perform real world deployment of the proposed approach and evaluate it against several DDoS tools.

References

1. Bhuyan MH, Bhattacharyya DK, Kalita JK (2015) An empirical evaluation of information metrics for low-rate and high-rate ddos attack detection. Pattern Recogn Lett 51:1–7
2. Lin S-C, Tseng S-S (2004) Constructing detection knowledge for ddos intrusion tolerance. Exp Syst Appl 27(3):379–390
3. Chang RKC (2002) Defending against flooding-based distributed denial-of-service attacks: a tutorial. IEEE Commun Mag 40(10):42–51
4. Yu S (2014) Distributed denial of service attack and defense. Springer, Berlin
5. Wikipedia (2016) 2016 dyn cyberattack. https://en.wikipedia.org/wiki/2016_Dyn_cyberattack. (Online; accessed 10 Apr 2017)
6. theguardian (2016) Ddos attack that disrupted internet was largest of its kind in history, experts say. https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet. (Online; accessed 10 Apr 2017)
7. Kalegele K, Sasai K, Takahashi H, Kitagata G, Kinoshita T (2015) Four decades of data mining in network and systems management. IEEE Trans Knowl Data Eng 27(10):2700–2716
8. Han J, Pei J, Kamber M (2006) What is data mining. Data mining: concepts and techniques. Morgan Kaufinann
9. Berkhin P (2006) A survey of clustering data mining techniques. In: Grouping multidimensional data. Springer, pp 25–71
10. Mori T (2002) Information gain ratio as term weight: the case of summarization of ir results. In: Proceedings of the 19th international conference on computational linguistics, vol 1. Association for Computational Linguistics, pp 1–7
11. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42
12. Tavallaee M, Bagheri E, Lu W, Ghorbani A-A (2009) A detailed analysis of the kdd cup 99 data set. In: Proceedings of the second IEEE symposium on computational intelligence for security and defence applications 2009
13. Shiravi A, Shiravi H, Tavallaee M, Ghorbani AA (2012) Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Comput Secur 31:357–374
14. Moustafa N, Slay J (2015) Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: Military communications and information systems conference (MilCIS), 2015. IEEE, pp 1–6
15. Moustafa N, Slay J (2016) The evaluation of network anomaly detection systems: statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. Inf Secur J: Glob Perspect 25:18–31
16. Akilandeswari V, Shalinie SM (2012) Probabilistic neural network based attack traffic classification. In: 2012 fourth international conference on advanced computing (ICoAC). IEEE, pp 1–8
17. Boroujerdi AS, Ayat S (2013) A robust ensemble of neuro-fuzzy classifiers for ddos attack detection. In: 2013 3rd international conference on computer science and network technology (ICCSNT). IEEE, pp 484–487
18. Ahmed M, Mahmood AN (2015) Novel approach for network traffic pattern analysis using clustering-based collective anomaly detection. Ann Data Sci 2(1):111–130
19. Saied A, Overill RE, Radzik T (2016) Detection of known and unknown ddos attacks using artificial neural networks. Neurocomputing 172:385–393
20. Boro D, Bhattacharyya DK (2016) Dyprosd: a dynamic protocol specific defense for high-rate ddos flooding attacks. Microsyst Technol 23:1–19
21. Nicolau M, McDermott J et al (2016) A hybrid autoencoder and density estimation model for anomaly detection. In: International

conference on parallel problem solving from nature. Springer, pp 717–726
22. Idhammad M, Afdel K, Belouch M (2017) Dos detection method based on artificial neural networks. Int J Adv Comput Sci Appl (ijacsa) 8(4):465–471
23. Mustapha B, Salah EH, Mohamed I (2017) A two-stage classifier approach using reptree algorithm for network intrusion detection. Int J Adv Comput Sci Appl (ijacsa) 8(6):389–394
24. Lakhina A, Crovella M, Diot C (2005) Mining anomalies using traffic feature distributions. In: ACM SIGCOMM computer communication review, vol 35. ACM, pp 217–228
25. Wagner A, Plattner B (2005) Entropy based worm and anomaly detection in fast ip networks. In: 14th IEEE international workshops on enabling technologies: infrastructure for collaborative enterprise (WETICE'05). IEEE, pp 172–177
26. Liu T, Wang Z, Wang H, Lu K (2014) An entropy-based method for attack detection in large scale network. Int J Comput Commun Control 7(3):509–517
27. Papalexakis EE, Beutel A, Steenkiste P (2014) Network anomaly detection using co-clustering. In: Encyclopedia of social network analysis and mining. Springer, Berlin, pp 1054–1068
28. Ahmed M, Mahmood AN (2014) Network traffic pattern analysis using improved information theoretic co-clustering based collective anomaly detection. In: International conference on security and privacy in communication systems. Springer, Berlin, pp 204–219
29. Ahmad A (2014) Decision tree ensembles based on kernel features. Appl Intell 41(3):855–869
30. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
31. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
32. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
33. van der Walt S, Colbert CS, Varoquaux G (2011) The numpy array: a structure for efficient numerical computation. Comput Sci Eng 13(2):22–30
34. McKinney W (2014) Pandas, python data analysis library. 2015. Reference Source
35. Hunter JD (2007) Matplotlib: a 2d graphics environment. Comput Sci Eng 9(3):90–95

Mohamed Idhammad received his master's degree in Computer Science and Distributed Systems in 2013 from the Faculty of Science, University Ibn Zohr, Agadir, Morocco. He is currently pursuing his PhD in Information Systems and Vision Laboratory, Agadir, Morocco. His research interests include Computer Science Security, Information Theory, Machine Learning and Cloud Computing.

Karim Afdel received his PhD in Digital processing of Medical images in University of Aix Marseille II, in 1998. He is now a full professor of Computer Sciences at Faculty of Sciences, University of Ibn Zohr, Agadir – Morocco. He is the Director of the Laboratory of IT Systems and Vision (LabSiv). His research interests include: Images Processing, Computer Vision, Machine Learning, Distributed Algorithms, Cloud Computing, Cryptography, Computer Security and Cloud Computing Security.

Mustapha Belouch received his master's degree in Network and Systems in 2013 from the Faculty of Science, University Ibn Zohr, Agadir, Morocco. He is currently pursuing his PhD in Laboratory of Applied Mathematics and Computer Science (LAMAI), Faculty of Science and Technology (FSTG), University Cadi Ayyad, Marrakech, Morocco. His research interests include Networks, Computer Science Security, Machine Learning and Cloud Computing.