Networks and Heterogeneous Media

http://www.aimspress.com/journal/nhm

NHM, 18(4): 1393–1408.
DOI: 10.3934/nhm.2023061
Received: 31 December 2022
Revised: 14 March 2023
Accepted: 13 April 2023
Published: 25 May 2023

Research article

Detection of DDoS attack in IoT traffic using ensemble machine learning techniques

Nimisha Pandey and Pramod Kumar Mishra*

Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi, Uttar Pradesh 221005, India

* **Correspondence:** Email: mishra@bhu.ac.in.

Abstract: A denial-of-service (DoS) attack aims to exhaust the resources of the victim by sending attack packets and ultimately stop the legitimate packets by various techniques. The paper discusses the consequences of distributed denial-of-service (DDoS) attacks in various application areas of Internet of Things (IoT). In this paper, we have analyzed the performance of machine learning(ML)-based classifiers including bagging and boosting techniques for the binary classification of attack traffic. For the analysis, we have used the benchmark CICDDoS2019 dataset which deals with DDoS attacks based on User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) in order to study new kinds of attacks. Since these protocols are widely used for communication in IoT networks, this data has been used for studying DDoS attacks in the IoT domain. Since the data is highly unbalanced, class balancing is done using an ensemble sampling approach comprising random under-sampler and ADAptive SYNthetic (ADASYN) oversampling technique. Feature selection is achieved using two methods, i.e., (a) Pearson correlation coefficient and (b) Extra Tree classifier. Further, performance is evaluated for ML classifiers viz. Random Forest (RF), Naïve Bayes (NB), support vector machine (SVM), AdaBoost, eXtreme Gradient Boosting (XGBoost) and Gradient Boosting (GB) algorithms. It is found that RF has given the best performance with the least training and prediction time. Further, it is found that feature selection using extra trees classifier is more efficient as compared to the Pearson correlation coefficient method in terms of total time required in training and prediction for most classifiers. It is found that RF has given best performance with least time along with feature selection using Pearson correlation coefficient in attack detection.

Keywords: DDoS attacks; random forest, gradient boosting; Pearson correlation coefficient; extra trees classifier; IoT; IoT security

Abbreviations: DoS: Denial of Service Attacks; IoT: Internet of Things; DDoS: Distributed Denial of Service Attacks; UDP: User Datagram Protocol; TCP: Transmission Control Protocol; ADASYN:

ADAptive SYNthetic Sampling; ML: Machine Learning; XGBoost: eXtreme Gradient Boosting; NB: Naïve Bayes; GB: Gradient Boosting; IIoT: Industrial Internet of Things; RF: Random Forest; SVM: Support Vector Machine; LOF: Local outlier factor; GODIT: Graph-based Outlier Detection in Internet of Things; RRCF: Robust Random Cut Forests; LSTM: Long-Short term Memory.

1. Introduction

IoT is expanding and flourishing over an extensive range of applications. DDoS attacks are a nimbly growing threat to IoT and its applications. With progressive adoption of IoT devices by people and organizations, these attacks can have far-reaching impact on human lives. Not only can these devices indirectly and unknowingly bombard small IoT-based systems, but they can also assault massive systems, too. Mirai malware used the IoT-based devices to create a botnet which was used to launch a huge attack on Dyn, a DNS service provider company, in October 2016. This attack ultimately killed the Internet service for a day. The volume of these attacks has aggravated over the years [22].

A DoS attack aims to stop or compromise the victim's performance through various measures like vulnerability attacks or flooding attacks. DDoS is an attempt to perform this attack on a victim collectively to ensure anonymity of attacker and increase the impact of the attack. This enhances the difficulty in identification of the source of DDoS attack. With the increase in number of IoT devices and presence of botnets on the Internet, launching attacks has become easier for the attackers. Therefore, the volume and frequency of these attacks have increased in the recent past years. The repercussions of these attacks are studied in various applications viz. healthcare [13, 17, 21, 27], smart homes [10], IIoT [8], agriculture [7], environment [7], logistics [28]. All these areas are close to the users, and resilience towards these attacks in these areas is essential.

Anomaly detection has been used to detect these attacks. ML algorithms enable the system to solve the problem by learning from the past data. ML-based classifiers promise the detection of new attacks also. Classification techniques can be used to differentiate between benign traffic and attack traffic based on the traffic features. We have also used this tendency to detect DDoS attacks from the dataset. In this paper, we applied different ML-based classification techniques in the detection of DDoS attacks and analyzed their performance. This study is specifically done to analyze these classifiers for the IoT environment. We have used a dataset comprising of TCP and UDP-based DDoS attacks for the performance analysis. These protocols are network/transport layer protocols used in IoT networks.

The rest of the paper is organized in the following manner. Section 2 discusses the related works while Section 3 describes the dataset. Section 4 discusses the methods used in our study while Section 5 discusses the experimental analysis done for the study. Section 6 discusses the results obtained while Section 7 concludes the paper.

2. Literature survey

Different ML algorithms have been used to detect DDoS attacks. Mostly researchers study the features of the genuine network traffic and perform anomaly detection on the incoming traffic. Doshi et. al. discussed DDoS attacks and differentiated the normal traffic and attack traffic in IoT network on the basis of many stateful and stateless features [6]. They analyzed the performance of 5 different ML algorithms, namely, k-nearest neighbor, SVM, decision tree using Gini impurity scores, random forests using Gini impurity score, neural networks. Performances of different ML algorithms in detecting the DDoS attacks in IoT environments have been analyzed in multiple research articles, as has been depicted in Tables 1 and 2.

Moreover, Meidan et al. have used autoencoders to detect anomalies in network traffic [19]. The deep autoencoder, being trained on the benign traffic, is unable to recreate the attack traffic. They analyzed the network traffic of nine IoT devices and created one autoencoder for each device. They also trained other three algorithms, namely, isolation forest, one class SVM and LOF, with the same traffic and found that the deep autoencoder performs better than all other algorithms, with 100% true positive rate, low false positive rate and low detection time. Summerville et al. proposed an ultra-lightweight deep packet anomaly detection approach which models the payloads using n-grams bit patterns and performs feature selection based on these bit patterns [26]. However, deep packet inspection can be more expensive for a resource-constrained device than flow-based features [6]. Doshi et al. proposed a pipelined anomaly detection mechanism in a smart home environment [6]. Their framework is protocol-agnostic and is deployed on the routers that connect IoT network to the Internet. They performed feature selection based on network flow statistics and used binary classification in order to perform anomaly detection.

Authors in [23] proposed a novel GODIT approach. In their approach, they took source IP and destination IP of the IoT traffic from smart home and represented it as the graph stream. Then, they created n-shingles from the graphs. Then this graph information was converted to a sketch vector, and then anomaly detection is done using the RRCF algorithm. They have created a testbed of 28 IoT devices and then evaluated their algorithm in two scenarios, viz. static and real-time streaming. In the static setting, DDoS attack was carried out by five IoT devices for one day. In this setting, they used classification algorithms like DT, RF, SVM and GB for anomaly detection. They compared their results with that of Doshi et al. [6] and found that their approach gives better F-1 score. Particularly, decision tree and SVM outperformed RF and GB considering the F-1 score. In the real-time streaming setting, one week of data was used, and anomaly detection was done using RRCF algorithm. Here, they compared GODIT with SpotLight and STREAMSPOT. GODIT outperforms both the approaches. The limitation is that GODIT needs periodic training. GODIT was compared with a proposed approach of Doshi et al. [6]. They reported a better F-1 score as compared to Doshi's approach when using decision tree and SVM for the anomaly detection. Uprety et al. [29] have reviewed all the works in IoT security that use reinforcement learning.

Alimi et al. [1] proposed a refined LSTM-based IDS for DoS attack in IoT. They performed encoding, dimensionality reduction and normalization on the datasets as preprocessing. The datasets used to test the performance of their proposed approach are CICIDS-2017 and NSL-KDD datasets. Their proposed approach has shown 99.23% precision, 99.22% recall, 99.22% accuracy and 99.22% f1-score for the CICIDS-2017 dataset. However, it has achieved 98.6% precision, 98.6% recall, 98.6% accuracy and 98.6% f1-score for the NSL-KDD dataset. Mishra et al. [20] have performed analysis of ML classifiers in the detection and categorization of these attacks in the CICDDoS 2019 dataset. Feature selection is done using a tree-based approach and SelectKBest technique. In this analysis, the DT gives the best accuracy of 99.86%. Alieyan et al. [2] have proposed DNS-DB, a rule-based approach for botnet detection that utilizes abnormalities in DNS queries and responses. Feature selection was done using (a) information gain ratio, (b) PCA, (c) intersection of features

Table 1. Papers using ML approach on DDoS attack.

Paper	Purpose	Approach	Attacks considered	Dataset	Comparison	Result
[18]	detection of botnets	bidirectional LSTM-RNN (ML)	SYN, ACK, UDP, DNS flood	created a dataset	compared bidirectional LSTM-RNN with unidirectional LSTM-RNN	proposed approach achieved 98%, 99% and 98% accuracy for UDP, Mirai and dns attack vectors respectively.
[15]	DDoS mitigation	Fog computing, anomaly detection using k-NN, signature based detection using a database of attacks	DDoS attacks	CICDDoS2019	compared k-NN with decision tree and NB, compared the distance measurement techniques, Euclidean, Manhattan and Chebyshev for accuracy	kNN performed best with 99.99% accuracy, 100% recall, 100% precision, and 100% F1-score while the Euclidean and Manhattan had better accuracy.
[14]	detection of DDoS attacks	Entropy based features in classification; classifiers used: RNN, MLP, and ADT	low-intensity DDoS, high-intensity DDoS, mix-intensity DDoS attacks	University of Brunswick 2012 ISCX dataset, 1998 DARPA Intrusion Detection dataset	compared E3ML with other classifiers namely MLP, ADT, RNN, compared performance of their approach against existing algorithms called EMD-L_i and ESDA	gave higher accuracy than the existing approaches, namely EMD-L_i and ESDA
[12]	detection of HTTP DDoS attacks in cloud environment	Information theoretic entropy and RF ensemble learning algorithm	HTTP DDoS attack	CIDDS-001 dataset	compared the performance of five classifiers, namely, kNN, MLP, DT, RF, NB	achieved 99.54% accuracy, 0.4% false positive rate and claimed running time of 18.5s.
[11]	detection of DoS and DDoS attack	Deep residual network	Syn attack, NTP attack, UDP attack, UDP Lag attack, SNMP attack, SSDP attack, LDAP attack, MSSQL attack, NetBIOS attack, DNS attack, and TFTP attack	CICDDoS2019 dataset by University of Brunswick	compared their approach with [25]	gave 99% accuracy in binary classification, and 87.06% accuracy in multi-class classification, claimed 9% higher accuracy, 21% greater recall, and 17% more F1-score as compared to [25]
[23]	detection of DoS attack in smart home IoT devices in real time	introduced GODIT	DoS attack	data created in University of New South Wales Sydney (UNSW Sydney) smart home data	compared performances of different classifiers, namely, DT, SVM, GB, compared their approach with [6]	Their approach produced better results than [6] in terms of F1-score
[25]	creation of a DDoS attack dataset & detection of DDoS attack through ML algorithms	use of RadViz diagrams to determine the influence of different features on the detection of different DDoS attacks	MSSQL, SSDP, DNS, LDAP, NETBIOS, SNMP, PORTMAP, CharGen, NTP, TFTP, SYN flood, UDP flood, UDP-Lag attacks	Created CICDDoS2019	compared the performance of classifiers, ID3, RF, NB and logistic regression	ID3 performed the best result of 78%precision, 65% recall and 69% F1-score

Table 2. Papers using ML approach on DDoS attack.

Paper	Purpose	Attacks considered	Dataset	Comparison	Result
[9]	DDoS detection using data stream approach and incremental learning	NSL-KDD and 1 created dataset	UDP-Flood, Smurf, SYNDDOS, HTTP-FLOOD	NB, RF, DT, MLP, K-NN	RF performs best accuracy= 98.9% for NSL-KDD dataset, 98.8% for created data It gives precision, recall and f1-score of 99.6%,99.8% and 99.7% for attack traffic in NSL-KDD dataset respectively
[5]	Ranking of the features for better classification	CAIDA, MIT-DARPA, ISCX, TUDDoS dataset	ICMP flood, SYN flood, and UDP flood	Boosting, KNN, Linear Discriminant, Logistic Regression, and Complex Decision Tree	Using the 3 top features: KNN gave highest accuracy of 96.6% in MIT-DARPA dataset using 10 partitions Boosting performed best using 9 partitions with accuracy as follows : CAIDA dataset=98.2% ; ISCX dataset=97.6%; TUDDoS dataset=98.2%
[16]	analysis of performance of DDoS detection on various datasets	ddossim, DoS improved GET, DoS GET, slow body, slow read, slow headers, rudy, slowloris, TCP SYN flood, TCP SYN flood-light mode, TCP ACK flood, UDP flood, UDP flood-random dst port, range attack, and application layer attacks	ISCXIDS2012, CSE-CIC-IDS2018, CIC-DoS, CICIDS2017, one created dataset	performance evaluation of ML classifiers viz. RF, DT, Stochastic Gradient Descent, AdaBoost, Logistic Regression, Perceptron	RF has performed best among these classifiers. Their proposed approach has given best results in CSE-CIC-IDS2018 dataset, with 100% detection rate, 0.000% false alarm rate, and 100% precision
[6]	Finding the significance of lightweight stateless features in the detection of DDoS attacks	created	TCP SYN flood, UDP flood, and HTTP GET flood	K-nearest neighbors “KDTree” algorithm (KN), SVM with linear kernel, Decision tree using Gini impurity scores (DT), RF using Gini impurity scores, Neural Network	KN and DT gave best accuracy of 0.995
[20]	detection and classification of attack	UDP attack, UDP Lag attack, SSDP attack, MSSQL attack, SNMP attack, NetBIOS attack, Syn attack, NTP attack, LDAP attack, DNS attack, and TFTP attack	CICDDoS 2019 dataset	DT, NB, SVM, RF, AdaBoost, XGBoost	DT achieves the best accuracy of 99.86%
[2]	botnet detection using DNS queries and responses	botnet attacks	ISOT	with PsyBoG	DNS-DB achieves 99.35% accuracy and 0.25% FPR.

acquired by both. The detection of abnormality was done based on entropy values of individual domain and its average value. DNS-DB achieves an accuracy of 99.35% and false positive rate of 0.25%. Their approach outperformed a well-known detection approach called PsyBoG as it gave 91.25% accuracy and 0.3% FPR. Authors in [4] achieved near real-time detection by performing analysis every second. Their approach performs attack detection and mitigation of the identified attack. In the detection phase, they have employed both rule-based detection and anomaly-based detection. The rule-based detection is done using entropy of features source IP address, source port address, destination IP address and destination port address. 1D-CNN is used for achieving anomaly detection. Performance of CNN was compared with MLP, deep MLP and Logistic Regression. CNN approach gave the best f1-score of 92.8%. Moreover, mitigation was achieved using two modules. The first module defines the appropriate countermeasure policy utilizing the zero-sum games of game theory. The second module executes the suggested mitigation policy.

Authors in [24] proposed a clustering-based undersampling method for unbalanced datasets. It employs the k-means clustering and Mahalanobis distance to create clusters. Cluster weight is used to create clusters to maintain the data distribution in the clusters and dataset space. Further, the classification is achieved by C4.5 decision tree acting as a base classifier for the outcomes of boosting and bagging algorithms. The proposed ensemble techniques based on bagging and boosting gave a better AUC score as compared to SVM and KNN algorithms as shown using 44 benchmark datasets. Authors in [3] gave a Bayesian game theory-based detection approach that lets the service provider employ a Bayesian differentiated pricing strategy and auction method to build an incentive-based system. The legitimate users and service provider accumulate the probabilities of the maliciousness of the users. Further, a reputation score is updated to prioritize the legitimate users over malicious ones. Their approach achieves the Bayesian Nash equilibrium points.

3. Dataset

A benchmark dataset, CICDDoS2019, which was contributed by Sharafaldin et al. [25] from the Canadian Institute of Cybersecurity and University of Brunswick, has been utilized. Thirteen different reflection-based attacks and exploitation-based attacks leveraging UDP and TCP protocols are contained in this dataset. The dataset was recorded on two days, and two separate datasets were recorded, viz. training day data and testing day data. The contributor extracted 80 features from the collected data using CICFlowMeter and used Random Forest Regressor for the feature selection. We have analyzed these two datasets separately. In the training day data, the three attack files, namely, TFTP, SSNP and DNS are huge, and therefore their records are partially included for the analysis. Only 5%, 25% and 25% of TFTP, SNMP and DNS attacks were randomly selected for the analysis of training day dataset.

4. Methodology

4.1. Preprocessing

This dataset was further preprocessed to remove the missing values and convert the string data values into decimal form. The data contains a high proportion of attack traffic flows as compared to the normal traffic flows. Since the classes present in the dataset are highly imbalanced, i.e., the data for attack traffic is high in proportion as compared to the non-attack traffic, the data is resampled using a combination of undersampling and oversampling techniques since the ratio of imbalance is huge. The samples in majority class are many, and oversampling results in even more samples. Since the data in the minority class are very few, most of the data is lost in undersampling. Therefore, the combination of random undersampler and ADASYN algorithm was used to resample the dataset. Although, another oversampling technique, SMOTE, combines the existing data points through lines and predicts more synthetic data points on those imaginary lines. ADASYN generates the data points that are not linearly correlated also. In the ADASYN algorithm, the number of samples to be generated is calculated through Eq (4.1). Then, the neighbors of the minority class data points are observed to calculate the value of r_i. r_i is the ratio of number of neighbors belonging to majority class out of all the neighbors of a minority class sample and is calculated using Eq (4.2).

$$G = (m_l - m_s) * \beta \tag{4.1}$$

where m_l is number of samples in majority class, and m_s is number of samples in minority class. β is the level of balance desired between the classes.

$$r_i = \triangle_i / k, \quad i = 1, 2, 3, ..., m_s \tag{4.2}$$

Here, k is the number of nearest neighbors of a minority class sample based on the Euclidean distance, and $\triangle_i$ is the number of neighbors among the k nearest neighbors that belong to majority class. Now, these values are normalized to transform them into a density distribution using Eq (4.3).

$$\hat{r}_i = r_i / \sum_{i=1}^{m_s} (r_i) \tag{4.3}$$

Further, the number of synthetic observations to be generated is determined for all the data points in the minority class using Eq (4.4).

$$g_i = \hat{r}_i * G \tag{4.4}$$

Therefore, the number of synthetic observations generated for the border points or noise points is greater than the observations generated for other points. The random undersampler is supplied a sampling strategy of 0.125, and ADASYN is supplied the sampling strategy = 'minority' and n_neighbors = 5 for both datasets.

4.2. Feature selection

Some features of the dataset can be redundant for the problem and can increase the computation time of a model. Therefore, feature selection supplies a compact set of relevant features and helps in reducing the time complexity of the algorithms. Feature selection is done on the basis of two methods:

- Pearson Correlation coefficient method: Pearson Correlation coefficient is the measure of magnitude and direction of correlation among two variables. Features having high correlation between them are considered redundant for the analysis and are removed from the dataset.
- Extra Tree Classifier: Extra Trees Classifier is an ensemble learning technique which combines the outcome of various de-correlated decision trees. The original training sample is used to

construct every decision tree in extra tree classifier. A sample of k features, selected from the feature-set, is provided at each test node in every tree. Out of k features, the best feature is provided for classification using mathematical criteria, like Gini index. The classifier sorts the features in descending order of their importance values and hence derives the most important features.

4.3. Classification techniques

We have used ML classification techniques for the detection, namely, Logistic regression, NB, SVM, RF, GB, AdaBoost and XGBoost. NB classifier uses the Bayes theorem assuming that the attributes are conditionally independent. Bernoulli NB is used to classify data that has multivariate Bernoulli distribution. SVM is a memory efficient ML classifier which is efficient for large feature sets. Logistic Regression is used to classify the categorical dependent variable. In logistic regression, an S-shaped logistic function (sigmoid function) is used for the classification of 0 and 1. The result of a sigmoid function is in the range of 0 and 1. Then, a threshold is used to classify these values among 0 and 1. All the values above this threshold are mapped to 1, and all the remaining values are mapped to 0. We have used the solver ‘saga’ in logistic regression since the dataset is huge.

Table 3. Performance of ML classifiers on testing day data when feature selection is done using Pearson correlation coefficient [20].

Classifiers	Training Time Complexity	Prediction Time Complexity
Naïve Bayes	O(sf)	O(f)
Random Forest	O(s^2ft)	O(ft)
Logistic Regression	O(sf)	O(f)
AdaBoost	O(sf)	O(ft)
Gradient Boosting	O(sft)	O(ft)
XGBoost	O(sft)	O(sft)
SVM	O($s^2f + s^3$)	O($n_{sv}f$)

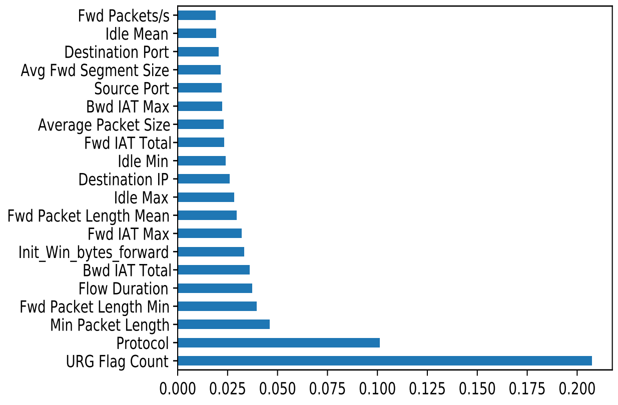
Boosting is an ensemble learning technique that creates an additive model using weak learners. It is based on the concept that weak learners can become better by improvement. AdaBoost classifier creates a strong classifier by sequentially combining a number of weak models. The wrongly predicted data of one model is supplied to the next model and so on. Finally, voting is performed among the outcomes of different models to get the final classification of the model. GB classifier is also a boosting technique that trains on the error created by last training. Here, the weights are not changed, but the models are trained using the residual errors of the last training. XGBoost, Extreme Gradient Boosting, optimizes the training part in the GB classifier. Bagging is an ensemble technique that supplies random subsets of the dataset to the models and then performs the final classification by voting. RF is a bagging technique which comprises various decision trees on different subsets of the dataset and yields their mean to improve the accuracy of the prediction. Boosting and bagging techniques avoid overfitting of data. The time complexities of training and prediction for these algorithms are given in Table 3. Here, f is the number of features, s is the number of samples in the dataset, t is the number of trees, and n_{sv} is the number of support vectors.

Considering the above-mentioned methods to get the relevant objectives, we have analyzed six algorithms, viz. RF, NB, AdaBoost, XGBoost, SVM, GB and logistic regression. The performance of these algorithms was computed, and the evaluation metrics were generated through Google Colab with Python. The details of the experiment and acquired results are discussed in the next section.

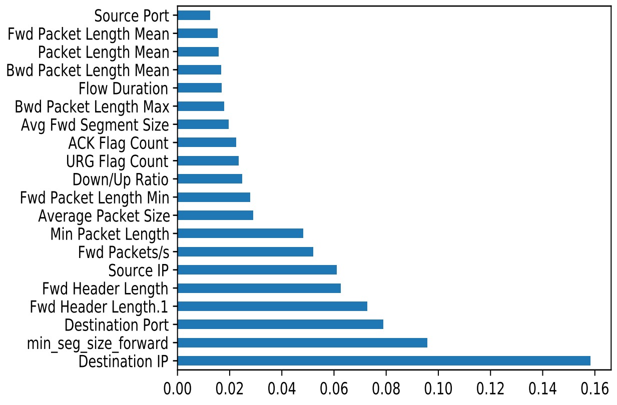
5. Experimental setup

ML based analysis on a huge dataset cannot be done without huge resources. We have used Google Colab Pro+ for this research, which provides 52 GB of RAM and 166 GB of disc space.

Feature Selection Pearson correlation coefficient was utilized for the feature selection, where the features having correlations of 0.85 and above were removed from the dataset. This method yielded 31 features and 34 features for the training day data and testing day dataset, respectively. Table 4 depicts the features given by this method for training day data and testing day data. When extra tree classifier is utilized, 20 most important features are selected according to their feature-importance values. These importance values are calculated based on the Gini impurity of the features. Figure 1a,b shows the selected features along with their importance values when extra tree classifier is used in cases of training day data and testing day data, respectively.



(a) 20 best features in training day data along with their importances



(b) 20 best features in testing day data with importances

Figure 1. Results of feature selection through Extra Trees Classifier in training day data and testing day dataset.

Further, the parameters used for the different techniques are discussed in Table 5. In order to analyze the performance of different parameters, we have calculated the precision, recall, accuracy and f1-score using Eqs (5.1)–(5.4).

$$precision = \frac{tp}{tp + fp} * 100 \tag{5.1}$$

$$recall = \frac{tp}{tp + fn} * 100 \tag{5.2}$$

$$accuracy = \frac{tp + tn}{tp + fp + fn + tn} * 100 \tag{5.3}$$

$$f1score = \frac{tp}{tp + fp} * 100 \tag{5.4}$$

Table 4. Features selected using Pearson correlation coefficient in training day dataset and testing day dataset.

Classifiers	Features
Training day data	Source IP, Source Port, Destination IP, Destination Port, Protocol, Flow Duration, Total Fwd Packets, Total Backward Packets, Total Length of Bwd Packets, Fwd Packet Length Max, Fwd Packet Length Std, Bwd Packet Length Max, Bwd Packet Length Min, Bwd Packet Length Mean, Flow IAT Min, Bwd IAT Min, Fwd PSH Flags, Fwd Header Length, Bwd Header Length, Fwd Packets/s, Bwd Packets/s, Max Packet Length, SYN Flag Count, ACK Flag Count, CWE Flag Count, Down/Up Ratio, Init_Win_bytes_forward, min_seg_size_forward, Active Mean, Active Std, SimillarHTTP
Testing day data	Source IP, Source Port, Destination IP, Destination Port, Protocol, Flow Duration, Total Fwd Packets, Total Backward Packets, Total Length of Fwd Packets, Total Length of Bwd Packets, Fwd Packet Length Max, Fwd Packet Length Std, Bwd Packet Length Max, Bwd Packet Length Min, Bwd Packet Length Mean, Flow IAT Mean, Flow IAT Min, Bwd IAT Mean, Bwd IAT Min, Fwd PSH Flags, Bwd Header Length, Fwd Packets/s, Bwd Packets/s, SYN Flag Count, ACK Flag Count, URG Flag Count, CWE Flag Count, Down/Up Ratio, Init_Win_bytes_forward, Init_Win_bytes_backward, Active Mean, Active Std, Idle Std, SimillarHTTP

Table 5. Parameters used in different models in the analysis.

Models	Parameters
RandomUnderSampler	sampling_strategy =‘0.125’, random_state = 1, n_neighbors = 5, n_jobs=None
ADASYN	sampling_strategy =‘minority’, random_state = 1
Random Forest	n_estimators = 25, criterion = ‘gini’, max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_leaf_nodes = None, min_impurity_decrease = 0.0, random_state = 42
Logistic Regression	penalty = ‘l2’, C=1.0, fit_intercept = True, intercept_scaling = 1, class_weight = None, solver = ‘saga’, max_iter = 100
SVM	kernel = ‘rbf’, C = 1, degree = 3, gamma = ‘scale’, coef0 = 0.0
Gradient Boosting	n_estimators = 20, learning_rate = 0.5, max_depth = 2, random_state = 0

Table 6. Performance of ML classifiers on training day data when feature selection is done using Pearson correlation coefficient.

Classifiers	Training Time	Prediction Time	Total Time taken	Precision	Recall	Accuracy	F1-score
Naïve Bayes	0.252 s	0.114 s	0.366 s	0.9756	0.9893	0.9823	0.9824
Random Forest	6.315 s	6.523 s	12.838 s	1	0.9999	0.9999	0.9999
Logistic Regression	19.977 s	0.018 s	19.995 s	0.8342	0.9925	0.8976	0.9065
AdaBoost	25.3524 s	1.1057 s	26.4581 s	1	1	1	1
Gradient Boosting	15.725 s	15.81 s	31.535 s	1	0.9999	0.9999	0.9999
XGBoost	34.728 s	0.2084 s	34.9364 s	0.9999	1	0.9999	0.9999
SVM	2729.1 s	654.354 s	3383.454 s	0.9374	0.9909	0.9624	0.9635

When the Pearson correlation coefficient was used for the feature selection, Table 6 shows the performance of different ML classifiers used for the analysis. Table 7 depicts the performance analysis of these classifiers when Extra Trees classifier is used for the classification. Similarly, Tables 8 and 9 show the performance of these classifiers when correlation coefficient and extra trees classifiers were used for feature selection in testing day dataset, respectively. However, the performance of classifiers for attack detection is similar with both methods of feature selection.

Table 7. Performance of ML classifiers on training day data when feature selection is done using Extra Trees Classifier.

Classifiers	Training Time	Prediction Time	Total Time taken	Precision	Recall	Accuracy	F1-score
Naïve Bayes	↓0.159 s	↓0.066 s	↓0.225 s	0.9280	0.9845	0.9541	0.9554
Random Forest	↑8.255 s	↑8.471 s	↑16.726 s	1	0.9998	0.9999	0.9999
Logistic Regression	↓16.065 s	↓0.015 s	↓16.08 s	0.8963	0.9952	0.94	0.9432
AdaBoost	↓23.6873 s	↓0.8928 s	↓24.5801 s	0.9999	1	0.9999	0.9999
Gradient Boosting	↓14.531 s	↓14.601 s	↓29.132 s	0.9998	0.9997	0.9997	0.9997
XGBoost	↓28.8030 s	↓0.2805 s	↓29.0835 s	0.9997	0.9998	0.9998	0.9998
SVM	↓1188.997 s	↓325.694 s	↓1514.691 s	0.9728	0.9909	0.9816	0.9818

Table 8. Performance of ML classifiers on testing day data when feature selection is done using Pearson correlation coefficient.

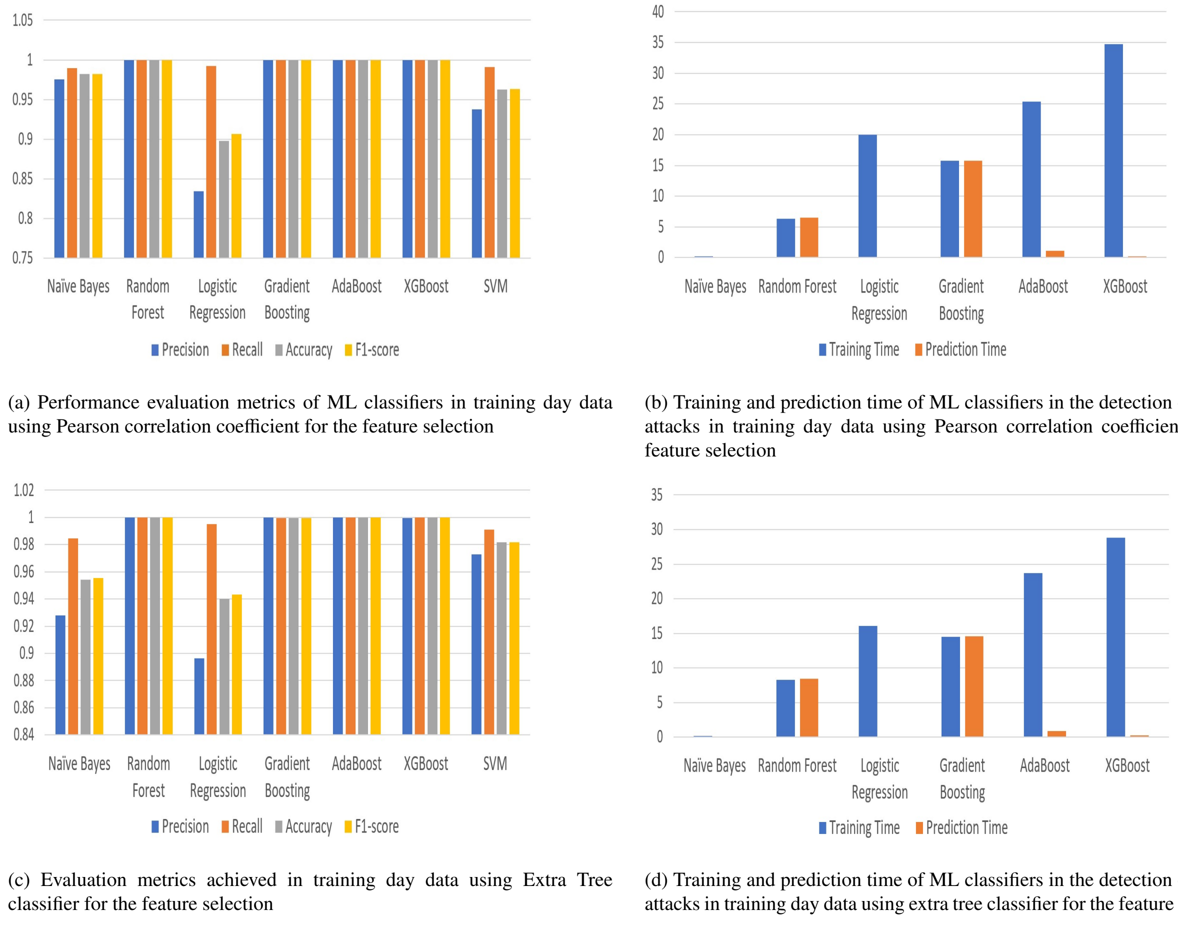
Classifiers	Training Time	Prediction Time	Total Time taken	Precision	Recall	Accuracy	F1-score
Naïve Bayes	0.425 s	0.187 s	0.612 s	0.9620	0.9003	0.9324	0.9301
Random Forest	13.175 s	13.579 s	26.754 s	1	0.9999	0.9999	0.9999
Logistic Regression	45.667 s	0.033 s	45.7 s	0.9478	0.9999	0.9724	0.9732
AdaBoost	64.454 s	2.180 s	66.634 s	1	1	1	1
Gradient Boosting	35.555 s	35.734 s	71.289 s	1	1	1	1
XGBoost	74.457 s	0.3965 s	74.8535 s	1	1	1	1
SVM	3919.818 s	185.188 s	4105.006 s	0.9894	0.9996	0.9944	0.9944

Table 9. Performance analysis of ML classifiers on testing day data when feature selection is done using Extra Trees Classifier.

Classifiers	Training Time	Prediction Time	Total Time taken	Precision	Recall	Accuracy	F1-score
Naïve Bayes	↓0.273 s	↓0.134 s	↓0.407 s	0.9753	0.9997	0.9872	0.9873
Random Forest	↑14.147 s	↑14.523s	↑28.670 s	1	1	1	1
Logistic Regression	↓32.654 s	↓0.021 s	↓32.675 s	0.9472	0.9998	0.9720	0.9728
AdaBoost	↓55.3852	↓1.7406	↓57.1258 s	1	1	1	1
Gradient Boosting	↓30.647 s	↓30.766 s	↓61.413 s	1	0.9999	0.9999	0.9999
XGBoost	↓56.143 s	↑0.4137 s	↓56.5567 s	1	1	1	1
SVM	↓1013.899 s	↑367.462 s	↓1381.361 s	0.9792	0.9995	0.9892	0.9893

In Tables 7 and 9, the increase and decrease in time taken is shown using ↑ and ↓ symbols, respectively. Tables 6–9 show that there is a decrease in time taken by most classifiers when the feature selection is done using extra tree classifier as compared to Pearson correlation coefficient. This is because the number of features selected by the Extra trees classifier is less than that by Pearson correlation coefficient method.

Figure 2b,d and Figure 3b,d show that the only RF algorithm takes comparatively less time with Pearson correlation coefficient. It is evident from Table 6 that the RF algorithm along with Pearson correlation coefficient gives best performance with least time in training data. Similarly, the same combination performs best for testing data, as shown in Table 8.



(a) Performance evaluation metrics of ML classifiers in training day data using Pearson correlation coefficient for the feature selection

(b) Training and prediction time of ML classifiers in the detection of DDoS attacks in training day data using Pearson correlation coefficient for the feature selection

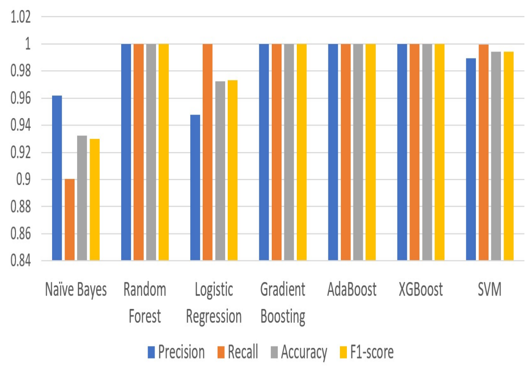
(c) Evaluation metrics achieved in training day data using Extra Tree classifier for the feature selection

(d) Training and prediction time of ML classifiers in the detection of DDoS attacks in training day data using extra tree classifier for the feature selection

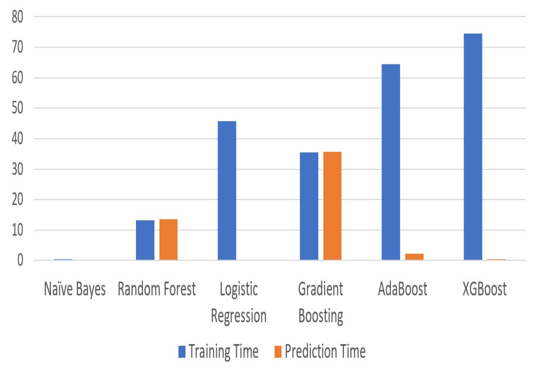
Figure 2. Performance evaluation of ML-based classifiers in the detection of DDoS attacks in training day data.

6. Results and discussion

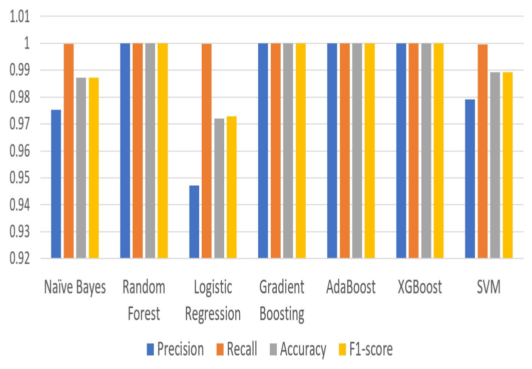
It is evident from the above analysis that RF has given the best performance with the Pearson correlation coefficient method. Figures 2 and 3 show the results of experiments for training day dataset and testing day dataset, respectively. Figure 2a shows the performance evaluation metrics achieved by the classifiers when correlation coefficient is used for the feature selection. The time taken by the classifiers in this experiment is shown in Figure 2b. Figure 2c shows the performance evaluation metrics achieved by the classifiers when feature selection was done using Extra tree classifier. Figure 2d depicts the time taken by the classifiers in the same experiment. When the same parameters were employed for testing data, the performance and time taken are recorded in Figure 3a,b. Further, the feature selection was done using Extra Tree classifier in testing data. When the Extra tree classifier was used for feature selection in the testing day dataset, Figure 3c,d shows the achieved evaluation metrics and time taken by the classifiers. Figure 2a,c shows that RF and boosting techniques yielded high performance metrics for both cases in training data. Figure 3a,c shows similar performances by these classifiers in case of testing data. Figures 2b,d show that boosting techniques take more time in classification as compared to RF. Figure 2a shows that RF has performed best along with Pearson correlation coefficient, with 100% precision, 99.99% recall, 99.99% accuracy, 99.99% f1-score in case of training day data. The total computation time taken by RF in this case is 12.838 seconds. Figure 3a shows that RF along with Pearson correlation coefficient has yielded 100% precision, 99.99% recall, 99.99% accuracy, 99.99% f1-score in testing data. The total time consumed is 26.754 seconds, as shown in Figure 3b. From Figures 2a,c and 3a,c, it is also seen that GB has performed best with the least amount of time among the boosting techniques. Hence, it is found that RF along with Pearson correlation coefficient for feature selection has performed best in both datasets with the least amount of time.



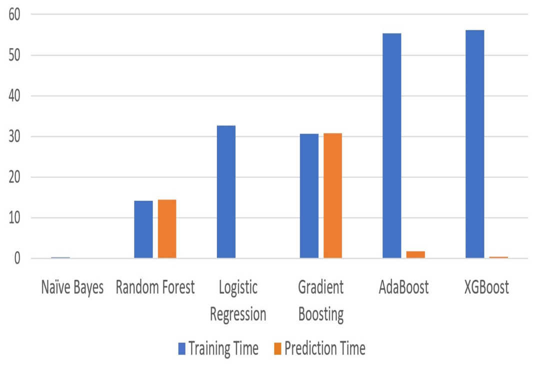
(a) Performance evaluation metrics of ML classifiers in testing day data using Pearson correlation coefficient for the feature selection



(b) Training and prediction time of ML classifiers when using Pearson correlation coefficient for the feature selection in testing day data



(c) Evaluation metrics achieved in testing day data using Extra Tree classifier for the feature selection



(d) Training and prediction time of ML classifiers in the detection of DDoS attacks in testing day data using extra tree classifier for the feature selection

Figure 3. Performance evaluation of ML-based classifiers in the detection of DDoS attacks in testing day data.

7. Conclusion and future scope

With the advent of botnets and availability of their code online, IoT devices have become vulnerable to DDoS attacks. ML-based approaches can play a vital role in the detection of these attacks. In this paper, we have studied the ML-based detection of DDoS attacks in IoT environment. We have analyzed the detection of TCP and UDP-based DDoS attacks using ML-based classification techniques. The classes in the dataset were first balanced using RandomUnderSampler and ADASYN techniques. The feature selection was done using Pearson correlation coefficient and Extra Trees Classifier. The classifiers used are GB, RF, NB, logistic regression, AdaBoost and XGBoost. We have observed that boosting techniques like GB, AdaBoost and XGBoost classifiers along with a bagging technique, i.e., RF classifier, has performed better than the traditional NB, Logistic Regression and SVM. Boosting techniques yielded good performance in the detection but had high training time. Among these techniques, RF has yielded the best performance in the least amount of computation time. Therefore, we conclude that the combination of feature selection using Pearson correlation coefficient and classification using RF algorithm has performed the best with the least amount of time. In future work, we aim to design a deep learning model for the detection of these attacks in an IDS.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The first author acknowledges the financial support from University Grants commission, New Delhi (UGC-JRF Reference Number-3680/ (NET-JULY 2018)). The second author gratefully acknowledges the financial support from IoE grant of Banaras Hindu University, Varanasi.

Conflicts of interest

The authors declare that they have no conflict of interest.

References

1. K. O. Adefemi Alimi, K. Ouahada, A. M. Abu-Mahfouz, S. Rimer, O. A. Alimi, Refined lstm based intrusion detection for denial-of-service attack in internet of things, *J. Sens. Actuator Networks*, **11** (2022), 32. https://doi.org/10.3390/jsan11030032

2. K. Alieyan, A. Almomani, M. Anbar, M. Alauthman, R. Abdullah, B. B. Gupta, Dns rule-based schema to botnet detection, *Enterp. Inf. Syst.*, **15** (2021), 545–564. https://doi.org/10.1080/17517575.2019.1644673

3. A. Dahiya, B. B. Gupta, A reputation score policy and bayesian game theory based incentivized mechanism for ddos attacks mitigation and cyber defense, *Future Gener. Comput. Syst.*, **117** (2021), 193–204. https://doi.org/10.1016/j.future.2020.11.027

4. M. V. de Assis, L. F. Carvalho, J. J. Rodrigues, J. Lloret, M. L. Proença Jr, Near real-time security system applied to sdn environments in IoT networks using convolutional neural network, *Comput. Electr. Eng.*, **86** (2020), 106738. https://doi.org/10.1016/j.compeleceng.2020.106738

5. R. K. Deka, D. K. Bhattacharyya, J. K. Kalita, Active learning to detect ddos attack using ranked features, *Comput. Commun.*, **145** (2019), 203–222. https://doi.org/10.1016/j.comcom.2019.06.010

6. R. Doshi, N. Apthorpe, N. Feamster, Machine learning ddos detection for consumer internet of things devices, in *2018 IEEE Security and Privacy Workshops (SPW)*, IEEE, (2018), 29–35. https://doi.org/10.1109/SPW.2018.00013

7. V. Hassija, V. Chamola, V. Saxena and D. Jain, A survey on IoT security: application areas, security threats, and solution architectures, *IEEE Access*, **7** (2019), 82721–82743. https://doi.org/10.1109/ACCESS.2019.2924045

8. T. Horak, P. Strelec, L. Huraj, P. Tanuska, A. Vaclavova, M. Kebisek, The vulnerability of the production line using industrial Iot systems under DDoS attack, *Electronics*, **10** (2021), 381. https://doi.org/10.3390/electronics10040381

9. S. Hosseini, M. Azizi, The hybrid technique for DDoS detection with supervised learning algorithms, *Comput. Net.*, **158** (2019), 35–45. https://doi.org/10.1016/j.comnet.2019.04.027

10. L. Huraj, M. Šimon, T. Horák, Resistance of IoT sensors against DDoS attack in smart home environment, *Sensors*, **20** (2020), 1–23. https://doi.org/10.3390/s20185298

11. F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, G. A. Shah, IoT DoS and DDoS attack detection using ResNet, in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, IEEE, 2020. https://doi.org/10.1109/INMIC50486.2020.9318216

12. M. Idhammad, K. Afdel, M. Belouch, Detection system of HTTP DDoS attacks in a Cloud environment based on Information Theoretic Entropy and Random Forest, *Secur. Commun. Net.*, **2018** (2018), 1–13. https://doi.org/10.1155/2018/1263123

13. Y. Jung, Hybrid-aware model for senior wellness service in smart home, *Sensors*, **17** (2017). https://doi.org/10.3390/s17051182

14. A. Koay, A. Chen, I. Welch, W. K. Seah, A new multi classifier system using entropy-based features in DDoS attack detection, in *2018 International Conference on Information Networking (ICOIN)*, (2018), 162–167. https://doi.org/10.1109/ICOIN.2018.8343104

15. M. A. Lawal, R. A. Shaikh, S. R. Hassan, A DDoS attack mitigation framework for IoT networks using fog computing, *Procedia Comput. Sci.*, **182** (2021), 13–20. https://doi.org/10.1016/j.procs.2021.02.003

16. F. S. d. Lima Filho, F. A. Silveira, A. de Medeiros Brito Junior, G. Vargas-Solar, L. F. Silveira, Smart detection: an online approach for dos/ddos attack detection using machine learning, *Secur. Commun. Net.*, 2019.

17. L. Liu, E. Stroulia, I. Nikolaidis, A. Miguel-Cruz, A. Rios Rincon, Smart homes and home health monitoring technologies for older adults: A systematic review, *Int. J. Med. Inf.*, **91** (2016), 44–59. https://doi.org/10.1016/j.ijmedinf.2016.04.007

18. C. D. McDermott, F. Majdani, A. V. Petrovski, Botnet detection in the Internet of Things using deep learning approaches, in *2018 International Joint Conference on Neural Networks (IJCNN)*, (2018), 1–8. http://dx.doi.org/10.1109/IJCNN.2018.8489489

19. Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, et al., N-baiot—network-based detection of Iot botnet attacks using deep autoencoders, *IEEE Pervas. Comput.*, **17** (2018), 12–22. http://dx.doi.org/10.1109/MPRV.2018.03367731

20. A. Mishra, N. Gupta, B. B. Gupta, Defensive mechanism against DDoS attack based on feature selection and multi-classifier algorithms, *Telecommun. Sys.*, **82** (2023), 229–244. https://doi.org/10.1007/s11235-022-00981-4

21. L. Nauha, N. S. Keränen, M. Kangas, T. Jämsä, J. Reponen, Assistive technologies at home for people with a memory disorder, *Dementia*, **17** (2018), 909–923. https://doi.org/10.1177/1471301216674816

22. N. Pandey, P. K. Mishra, Taxonomy of DDoS attacks and their defense mechanisms in IoT, *J. Sci. Res.*, **65** (2021), 197–207.

23. R. Paudel, T. Muncy, W. Eberle, Detecting DoS attack in Smart Home IoT devices using a graph-based approach, in *2019 IEEE International Conference on Big Data (Big Data)*, (2019), 5249–5258. https://doi.org/10.1109/BigData47090.2019.9006156

24. M. S. E. Shahabadi, H. Tabrizchi, M. K. Rafsanjani, B. Gupta, F. Palmieri, A combination of clustering-based under-sampling with ensemble methods for solving imbalanced class problem in intelligent systems, *Technol. Forecast. Soc. Change*, **169** (2021), 120796. https://doi.org/10.1016/j.techfore.2021.120796

25. I. Sharafaldin, A. H. Lashkari, S. Hakak, A. A. Ghorbani, Developing realistic Distributed Denial of Service (DDoS) attack dataset and taxonomy, in *2019 International Carnahan Conference on Security Technology (ICCST)*, IEEE, (2019), 1–8. https://doi.org/10.1109/CCST.2019.8888419

26. D. H. Summerville, K. M. Zach, Y. Chen, Ultra-lightweight deep packet anomaly detection for Internet of things devices, in *2015 IEEE 34th international performance computing and communications conference (IPCCC)*, IEEE, (2015), 1–8. https://doi.org/10.1109/PCCC.2015.7410342

27. R. Turjamaa, A. Pehkonen, M. Kangasniemi, How smart homes are used to support older people: an integrative review, *Int. J. Older People Nurs.*, **14** (2019), 1–15. https://doi.org/10.1111/opn.12260

28. D. Uckelmann, A definition approach to smart logistics, in *International Conference on Next Generation Wired/Wireless Networking*, Springer, (2008), 273–284.

29. A. Uprety, D. B. Rawat, Reinforcement learning for IoT security: a comprehensive survey, *IEEE Int. Thing. J.*, **4662** (2020), 1–14. https://doi.org/10.1109/JIOT.2020.3040957