Wireless Personal Communications (2024) 137:951–976
https://doi.org/10.1007/s11277-024-11419-2

Towards Detection of DDoS Attacks in IoT with Optimal Features Selection

Pooja Kumari[1] · Ankit Kumar Jain[1] · Yash Pal[1] · Kuldeep Singh[1] · Anubhav Singh[1]

Accepted: 19 June 2024 / Published online: 15 July 2024
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract
The exponential growth of internet-enabled devices and their interconnectedness heightens the vulnerability of technology to cyber threats. The simplicity of communication lures attackers to execute numerous attacks, with Distributed Denial of Service (DDoS) emerging as a major threat due to its challenging detectability. Over recent years, numerous machine learning mitigation methodologies have arisen to combat this issue. In this paper, we present an approach for detecting DDoS attacks, with a primary focus on optimal feature selection and data pre-processing to mitigate the risk of overfitting and enhance accuracy. We employ an embedded method utilizing a decision tree in Recursive Feature Elimination with Cross-Validation (RFECV) to select the most effective features. Subsequently, we apply Gradient Naïve Bayes (GNB), Decision Tree (DT), Random Forest (RF), and Binary Classification using deep neural network deep learning models. These models undergo validation using the CICDDoS2019 dataset. Performance evaluation reveals that the deep learning model surpasses others, achieving an accuracy of 99.72%.

Keywords Internet of things · DDoS · Machine learning · Deep learning · Feature selection

✉ Ankit Kumar Jain
ankit.jain2407@gmail.com

Pooja Kumari
poojakumari.cse7@gmail.com

Yash Pal
yashpalnss@gmail.com

Kuldeep Singh
ks517633@gmail.com

Anubhav Singh
anubhavsingh2690@gmail.com

[1] Department of Computer Engineering, National Institute of Technology, Kurukshetra, India

1 Introduction

The proliferation of internet-enabled devices, coupled with their increasing interconnectivity, has revolutionized modern living. Across various domains, from homes to factories, these devices offer unmatched convenience and effectiveness across different fields. The Internet of Things (IoT) is leading the way in today's digital age, presenting various compelling factors that drive its adoption. These include the emergence of novel business concepts and opportunities, the potential for revenue expansion in businesses, enhanced decision-making facilitated by data analysis, cost effectiveness by prioritizing focusing on safety and security, better user experiences with easy-to-use features, and improved infrastructure [1, 2]. As data innovation (IT) continues to converge, numerous data devices are becoming increasingly confusing. They are linked together and continue to create and save vast electronic data, accompanying in a time of big data. However, because they send a lot of data through constant interaction with one another, the chances of them exposing significant data are exceedingly high [3]. As more digital devices are connected, a framework becomes increasingly susceptible [4]. Most of the IoT devices have shortcomings, such as encryption and password security flaws [5, 6]. The fundamental reasons behind the insecurity of IoT devices include the scarcity of standardized security protocols, the heterogeneous nature of IoT devices, limited resources for security updates, reliance on centralized data storage, etc. which allow attackers to launch serious attacks such as DoS and Distributed DoS attacks [7]. Attackers are attempting these attacks to circumvent traditional security measures to cause damage, exploit intellectual property and compromise confidential information [8, 9].

DDoS attacks are experiencing significant growth, primarily because they originate from multiple sources, making them challenging to identify. Some Major outbreaks due to DDoS attacks include the Mirai botnet attack in 2016 [4, 10], the DDoS attack on Google in 2017 [11, 12], the GitHub attack in 2018 [10, 13], the attack on Amazon Web Services in 2020 [10, 13], the Yandex attack in 2021 [14] and many more. We have summarized this growth of DDoS attacks in Fig. 1 based on attacking power.

The attackers hack the IoT devices and make them work as zombies also called bots and later use them through handlers to launch the DDoS attacks. The attacks then disrupt the network services for legitimate users while accessing them [15]. While persuading the DDoS attack on IoT devices, there are individuals or groups acting as attackers, as well as controllers associated with counterfeited IoT devices and target/victim devices. The term 'distributed' in DDoS refers to the fact that the disrupted IoT devices utilized in the attack are distributed across various locations geographically [8, 16]. The botnets are controlled by controllers (handlers) who receive commands from the attackers. The commands comprise aspects such as the type of attack, the timing of the attack, the duration of the attack, and other relevant parameters [1, 17].

DDoS attacks encompass various forms, including volumetric DDoS attacks, protocol exploitation-based attacks, and amplified DDoS attacks [18]. The volumetric attacks are carried out by overwhelming the target machine with packet flooding that looks like legitimate traffic. The attack consumes the network bandwidth [8]. In protocol exploitation DDoS attacks, the attacker uses various protocols to perform the attack like, fragmentation attacks, ping of death attacks, smurf attacks, etc. [15, 17] These attacks distort the actual network resources and intermediate communicational systems like firewalls. In amplification attacks, the attacker transmits DNS/NTP requests towards the server and the size of the

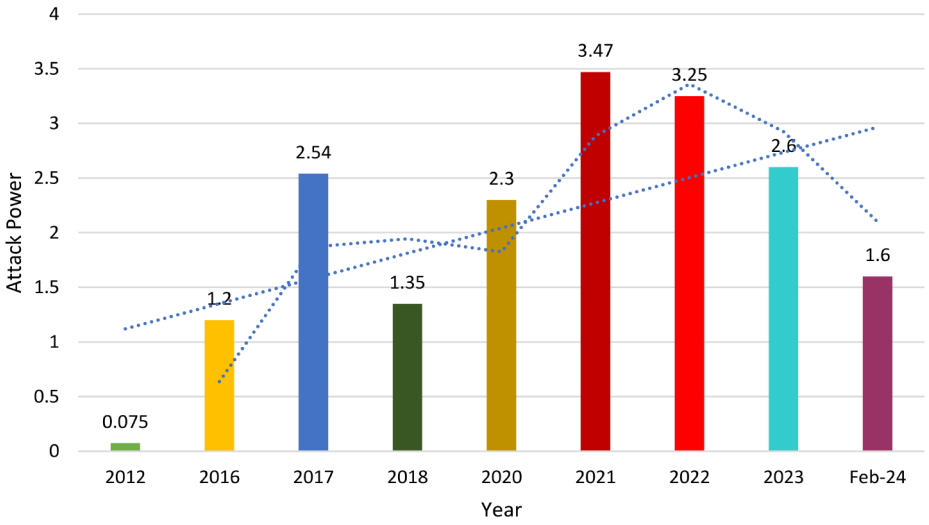

Fig. 1 Timeline of most significant DDoS attacks over previous years ranked by attack potency

payload of the response packet is way larger than the request packet which means that the target machine is bombarded by the amplified packets [19].

The DDoS attacks are constantly evolving in more prevalent and complex forms which makes them a lot more difficult to combat. Addressing the challenge of DDoS detection is paramount to safeguarding the integrity and reliability of networked systems. Traditional defense mechanisms struggle to keep pace with the evolving sophistication of DDoS tactics, underscoring the need for innovative approaches. Considering these circumstances, attack detection systems (ADS) should be smarter and more effective than before in combating cyber-attacks, which are constantly changing [20]. Consequently, it becomes imperative to identify the necessary security measures. Safeguarding IoT devices from attackers necessitates the ability to anticipate potential attacks. With such advancements in DDoS attacks, the adoption of machine learning-based defense mechanisms is essential to effectively combating such attacks. Machine learning presents a promising opportunity for enhancing DDoS detection capabilities along with its ability to distinguish patterns and anomalies in vast datasets. Machine learning algorithms stand out as an effective solution for addressing the liabilities of the IoT systems and IoT devices because these techniques can proficiently detect network anomalies and intrusions [20, 21]. Various researchers have extensively explored the application of machine learning in IoT to identify and mitigate malicious traffic traversing the network. Recent research studies have experimented with various machine learning classification algorithms such as SVM, Random Forest, Naïve Bayes, KNN, Deep Learning Neural networks, etc. [17, 22] with or without other detection techniques.

Although the machine learning models have considerable efficacy in attack prediction, however, they become relatively static as the size of the dataset rises. Deep learning comes to the rescue here as it keeps improving its efficiency even at a very large size of the dataset [23]. Hence, we propose a model wherein we conduct a comparative analysis of machine learning algorithms and deep learning algorithms using the CICDDoS-2019 dataset [24] so that we can easily compare the results and identify which approach performs better.

1.1 Problem Definition

The increasing frequency and complexity of DDoS attacks pose a significant threat to the security and resilience of internet-enabled systems. Despite the growing awareness of these attacks and efforts to mitigate their impact, detecting and defending against them remains a formidable challenge. Traditional defense mechanisms often struggle to handle the scale and sophistication of modern DDoS attacks. Hence, efficient and advanced detection approaches are imperative, capable of precisely identifying and countering DDoS attacks in real-time. By harnessing machine learning and data-driven methodologies, our aim is to develop a robust detection system capable of distinguishing between authentic and malicious traffic patterns. This endeavor will enhance the resilience of networked systems against DDoS attacks.

In machine learning and deep learning classification models, predictions are compiled on the basis of some specific relevant features. However, these models often face challenges while predicting the prominent features that will significantly influence the outcome. Consequently, the majority of the users include the complete feature set in the classifier model for attack prediction which leads to increased complexity and computational overhead in the classifier model. Therefore, our primary objective is to provide a way where the classification model only uses the most relevant features, removing unnecessary and redundant features using feature selection techniques. We have concentrated on feature selection and data pre-processing in the suggested methodology.

1.2 Contribution

Highlighting the key contributions of our proposed approach, we provide a comprehensive overview of the advancements and innovations driving our methodology forward. These contributions signify significant progress in the field, offering novel insights and solutions to address the challenges at hand:

- The suggested approach prioritizes data pre-processing to achieve a balanced and systematized dataset by analyzing and cleaning it.
- To reduce the complexity of the training model and standardize the features we employed feature scaling and feature selection.
- An embedded method is proposed for selecting the optimal features.
- The performance of different classifier models is validated on combinations of CICDDoS2019 dataset's day1 and day2 data i.e., the training is done by using day1 data and tested using day2 data.
- Lastly, we performed an exhaustive comparison of results obtained with and without feature selection, assessing their performance based on accuracy, precision, recall, and F1-score metrics.

1.3 Paper Organization

The remaining paper is structured into six sections. The second section entails an overview of related research concerning DDoS attack defense. In the third section, we detail our pro-

posed approach, including the machine learning and deep learning models employed, along with the processing steps involved. The fourth section delves into a comprehensive analysis of the results obtained through the proposed method, comparing it with other techniques. The effectiveness of the approach in achieving research objectives is analyzed, alongside its performance through different evaluation metrics. The fifth section concludes our work followed by the sixth section addressing the limitations of our approach and suggests areas for future development.

2 Related Work

This section provides an overview of various methodologies and strategies employed by researchers in defending against DDoS attacks using machine learning and deep learning models. This section renders some of the recently used approaches for defending DDoS attacks.

Manohar et al. [25] employed the C5.0 machine learning algorithm and compared the results with distinct machine learning techniques specifically the Naive Bayes classifier and the C4.5 decision tree classifier. The authors employed the techniques as an offline approach. N. G. et al. [26] employed an approach which is known as deep intelligence. The authors employed a radial basis function with varying levels of abstraction to gather intelligence and conducted the experimental evaluations on the widely recognized NSL KDD and UNSW NB15 datasets, each comprising 27 attributes. Dayanandam et al. [27] classified packets depending on their characteristics. The preventive mechanism analyzes the IP addresses by verifying the IP header. The approach utilizes these analyzed IP addresses for discerning the legitimate and counterfeit IP addresses. However, as the scale of attacks increases, conventional firewalls may become inadequate.

For bifurcating the regular and attacked traffic, Mallikarjunan et al. [28] used anomaly detection and machine learning approaches and real-time datasets for the experimental setup. The well-known naive Bayes ML approach was used for classification. The results of the Naïve Bayes algorithm were compared to those of J48 and random forest methods (RF). In a study by Aamir et al. [29], a clustering approach was employed to develop a feature selection technique. The paper presents an assessment of the performance of five distinct machine learning algorithms for training, which encompassed random forest (RF) and support vector machine (SVM). Among these algorithms, random forest exhibited the highest accuracy rate, reaching approximately 96%. Sharma, et al. [29] presented an IoT network anomaly detection architecture in which the detection takes place on the fog layer and uses local monitoring of the network traffic. The proposed algorithm is able to detect protocol-based DDoS attacks.

Batchu et al. [30] utilized a hybrid approach that involved feature selection and hyperparameter tuning. They applied this approach to various supervised learning algorithms, including Logistic Regression (LR), Decision Tree (DT), Gradient Boost (GB), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). Following this, the models were evaluated using the CICDDoS2019 dataset.

Rahman et al. [31] constructed a model to assess ML-based DDoS detection using Grid Search Hyperparameter Optimization. Their approach involves evaluating traditional Machine Learning (ML) methods to develop a DDoS detector. Additionally, it

employs exhaustive hyperparameter search to optimize the detection capability of each ML model.

Myneni et al. [10] introduced SmartDefense, an edge computing-based distributed DDoS detection and mitigation system designed to identify and counteract DDoS attacks originating from or near their source. Along with detecting DDoS attacks, the framework may also be used to prevent threats like bot devices and spamming.

Prasad, et al. [32], propose a voting-based multimodal framework to counter volumetric DDoS (VMFCVD) attacks, employing fast detection mode (FDM), defensive fast detection mode (DFDM), and high accuracy mode (HAM) methods. The model discards all predicted malicious packets to prevent the attack.

Gaurav, et al. [16] have proposed an approach for detecting DDoS attacks and distinguishing them from similar-looking legitimate traffic called Flash Crowd. They have used entropy-based statistical methods and machine learning models for this purpose.

3 Proposed Methodology

This section outlines the proposed approach, consisting of three main phases: dataset pre-processing, feature extraction, and classification models, as depicted in Fig. 2. In the initial step, during data pre-processing, we thoroughly analyzed the CICDDoS-2019 dataset to gain insights into its composition. Subsequently, data pre-processing techniques illustrated in Fig. 5 were implemented to enhance data quality and normalize feature ranges. Moving on to the second step, feature selection was conducted using the Recursive Feature Elimination with Cross-Validation (RFECV) technique. Once features were selected, both machine learning and deep learning models were utilized to identify and classify potential attacks. Following this, the models discerned between malicious and benign incoming traffic. A detailed explanation of each phase of the proposed approach is provided in the subsequent sections.

3.1 Dataset

We are using the CICDDoS2019 dataset to validate the proposed model. The dataset is released by the Canadian Institute for Cybersecurity in 2019 including the most recent variants of DDoS attacks. The dataset consists of both the PCAP files as well as the CSV files. The PCAP file resembles real-world data. Sharafaldin, et al. [24] extracted 88 features by using CICFlowMeter-V3. The dataset consists of two days of data which we have used for validating the proposed approach. The first day involves 7 variants of DDoS namely: PortMap, NetBIOS, MSSQL, LDAP, UDP, UDP-Lag, and SYN as shown in Fig. 3.

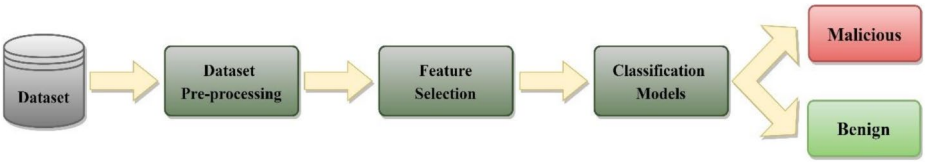

Fig. 2 Diagrammatic representation of the proposed methodology

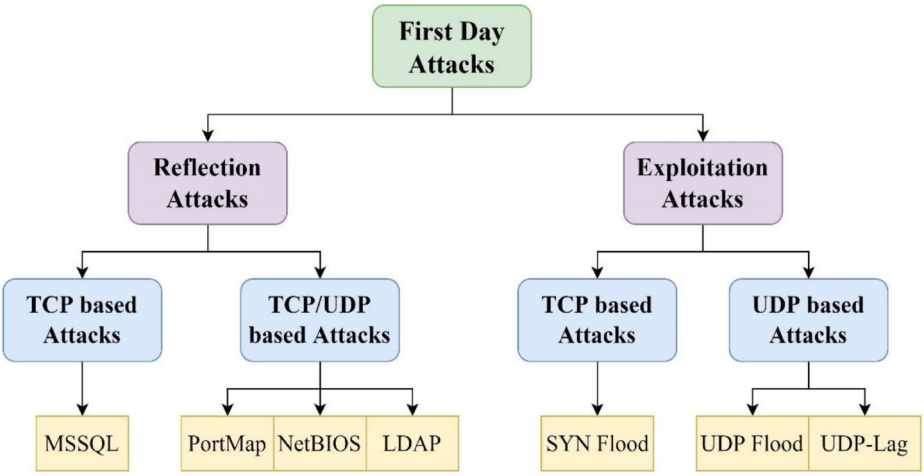

Fig. 3 DDoS attack variants on first day

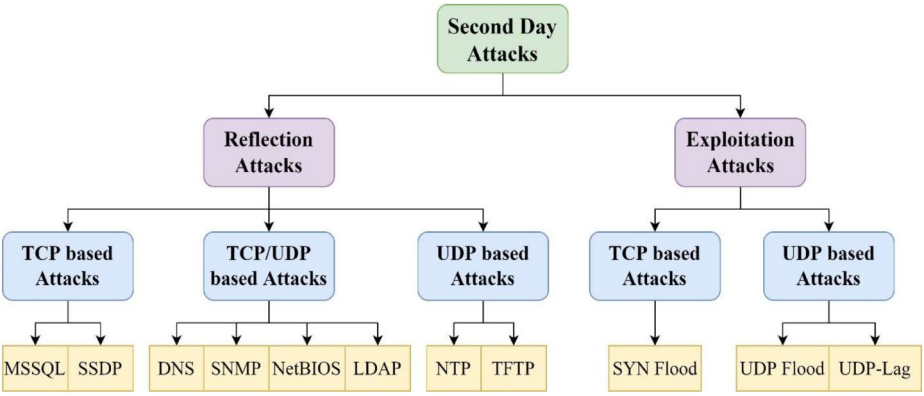

Fig. 4 DDoS attack variants on second day

The second day's data consists of 12 different types of DDoS which are, NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, TFTP as shown in Fig. 4.

3.2 Data Pre-Processing

The dataset consists of raw and unbalanced data, which is not directly compatible with machine learning or deep learning models. Therefore, data pre-processing emerges as an essential phase in organizing the data for training and evaluating the classification models. During this phase, the dataset undergoes crucial transformations to rectify imbalances, address missing values, and scale features. These actions result in a refined, organized, and balanced dataset, facilitating more effective training and evaluation of

models. While pre-processing the data we have followed three steps which are shown in Fig. 5.

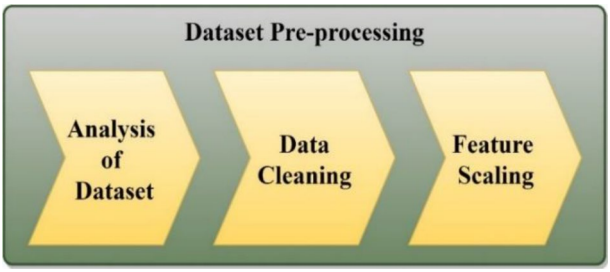

Fig. 5 Steps involved in data pre-processing

3.2.1 Analysis of Dataset

Since the data that we observe by visualizing it through our naked eyes is not necessarily accurate hence, to compile and interpret the precise information of the dataset we analyzed the dataset thoroughly. The analysis gives some necessary information about the dataset like the number of rows, maximum and minimum value for numerical data fields, categorical data, etc. The CICDDoS2019 dataset is considerably large, comprising 30GB of CSV files, each containing various types of attack traffic and spanning over 1 million rows. After a thorough analysis of the data, we found out that dataset contains:

- Infinite values.
- Missing Values or NaN values.
- Non-numeric field which is supposed to be having numerical data.
- Unnecessary Features.

These findings will be handled in the next step which is the data cleaning phase to get an organized and fair dataset.

3.2.2 Data Cleaning

After analyzing the dataset data cleaning is performed as shown in Fig. 6 to handle the null, infinite, and missing values.

To ensure a fair distribution of the dataset for feature selection, we implemented the following steps:

- Remove Null values.

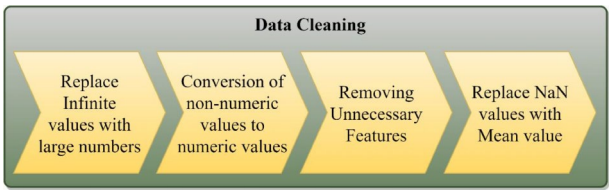

Fig. 6 Steps involved in data cleaning

- Substituting infinite values with significantly large numerical values in their respective fields.
- Some fields that were supposed to contain numerical data were falsely being detected as strings so we converted them into numerical data type.
- Some fields contained missing values, which were addressed by replacing them with the mean value of the corresponding column.
- To make computation efficient we dropped the following columns:

  – Timestamp
  – Flow ID
  – Source IP
  – Destination IP
  – SimillarHTTP
  – ID

- Ensure uniform labelling across both training and testing datasets. If the training and testing datasets have different labels then the model will not identify the data and hence the results will be affected.
- After that, we used Hot Encoding for label transformation for machine learning models which assigns a unique number starting from 0 to each class. Because the machine learning model understands numeric values.

3.2.3 Feature Scaling

The dataset has different scales for the values of each feature it contains which increases the complexity for training the model. Hence after cleaning the data and before the feature selection procedure we performed feature scaling. We used *StandardScaler* for feature scaling which standardizes the features by eliminating the mean and scaling to unit variance which improves the performance of supervised machine learning procedures by removing the skewness of the dataset. The standard scaler uses the following equation for feature scaling:

$$Scaled\,Data = \frac{(x - \mu)}{\sigma} \tag{1}$$

Where x is the data that is going to be scaled, μ and σ are the mean and standard deviation of the training samples respectively.

3.3 Feature Selection

Feature selection is an imperative step for generating the best performance model. The feature selection procedure helps in training the model faster, increasing its interpretability, improving its accuracy, and reducing the overfitting. Feature selection involves identifying the most prominent features for predictive modelling which can be performed based on certain statistical methods like filter, wrapper, and embedded methods.

- The filter methods work on ranking the feature methods on the basis of univariate metrics like variance, chi-square, correlation coefficients, information gain, etc.
- The wrapper method works on the prediction of the target variable by searching for the best subset of input features. Some wrapper methods are exhaustive search, forward selection, and backward selection. The wrapper methods select those features that provide the best accuracy but increase the computational cost.
- Embedded methods incorporate the qualities of both filter and wrapper methods. The model combines the filter and wrapper methods for feature selection. Some embedded methods like random forest perform the feature selection as their integral operation allowing them to execute feature selection and classification simultaneously.

The proposed approach utilizes the RFECV (Recursive Feature Elimination with Cross-Validation) method for feature selection, which employs any supervised machine learning model as the estimator. For this purpose, we employed the decision tree embedded feature selection method as the estimator. Here, it is necessary to provide a supervised learning estimator with a fit method that furnishes information about feature importance, either through a coef_ attribute or through a feature_importances_ attribute. In our proposed method, the Decision Tree classifier serves as the supervised estimator for RFECV, and the results are evaluated after running 10-fold cross-validation, as depicted in Fig. 7. The RFECV technique computes the optimal feature subsets by assessing cross-validation scores. It employs the recursive feature elimination method, which iteratively removes features from 0 to n (where n represents the total number of features in the dataset) to determine the best feature

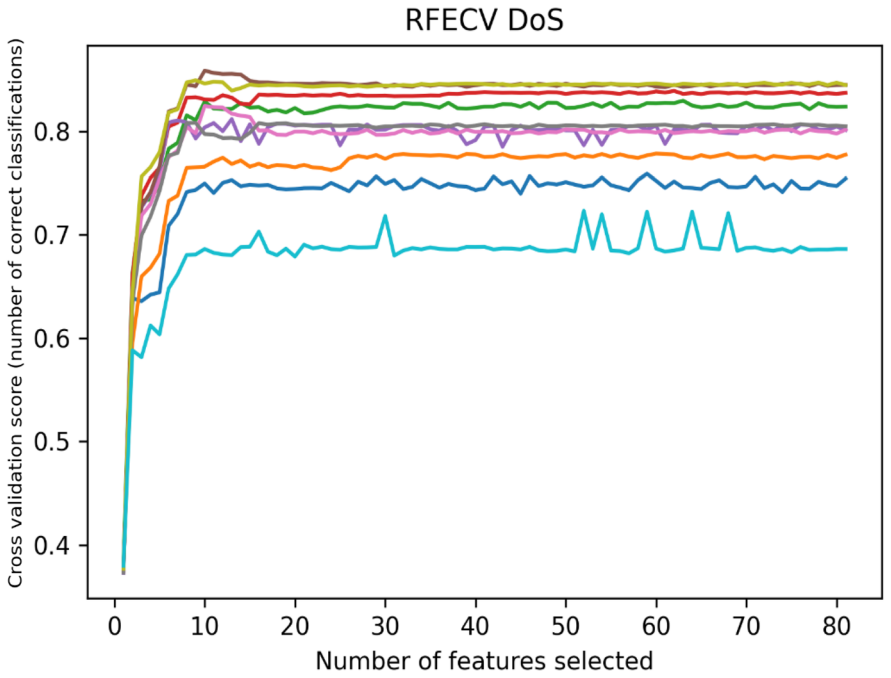

Fig. 7 Plot after running 10-Fold RFECV on all the features

set. RFECV fits the model multiple times, eliminating one weakest feature at each iteration, and provides the best optimal features.

After performing the feature selection procedure, the optimal resultant features are shown in Fig. 8.

After the feature selection process, the classifier model undergoes tuning. The dataset is divided into two parts: the training and testing data. Subsequently, each classifier model is trained using the training data, allowing the models to learn about the parameters. Following this training phase, the model is tested using the test data. In the proposed approach, the model was trained using data from the first day and validated using data from the second day as the test data.

3.4 Classification Models

This section entails the classification models that we have used for the detection of DDoS attacks. In machine learning-based approaches we have implemented Decision Tree, Naive Bayes, and Random Forest classifier while in deep learning-based approaches, we have implemented Binary classification using the Keras sequential model which uses a neural network. The dataset used is the latest CICDDoS-2019 dataset which has a huge amount of data related to DDoS attacks in IoT devices which will further enhance the deep learning model.

3.4.1 Gaussian Naive Bayes

The Gaussian model is based on the assumption that features follow a normal distribution. In other words, if the predictor takes continuous values rather than discrete values, the model predicts that the values will be drawn from a Gaussian distribution. Gaussian Naive Bayes is a variant of the well-known Naive Bayes algorithm designed for continuous data. Naive Bayes is a set of supervised machine learning classification algorithms that leverage Bayes' Theorem. To predict unknown attributes, the algorithm uses a training set of data and calculates probabilities using Bayes' theorem.

$$P\left(x_i \middle| y\right) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{\left(x_i - \mu_y\right)^2}{2\pi\sigma_y^2}\right) \tag{2}$$

```
Optimal number of features : 59
The selected features are:
['Source Port', 'Destination Port', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets', 'Total Length of Fwd Packets', 'Total Length of Bwd P
ackets', 'Fwd Packet Length Max', 'Fwd Packet Length Min', 'Fwd Packet Length Mean', 'Fwd Packet Length Std', 'Bwd Packet Length Mean', 'Bwd Packet Lengt
h Std', 'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean', 'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean', 'Fwd IAT St
d', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean', 'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd Header Length', 'Bwd Header Length',
'Fwd Packets/s', 'Bwd Packets/s', 'Min Packet Length', 'Max Packet Length', 'Packet Length Mean', 'Packet Length Std', 'Packet Length Variance', 'SYN Fla
g Count', 'RST Flag Count', 'PSH Flag Count', 'CWE Flag Count', 'Down/Up Ratio', 'Average Packet Size', 'Avg Fwd Segment Size', 'Avg Bwd Segment Size',
'Fwd Header Length.1', 'Subflow Fwd Packets', 'Subflow Fwd Bytes', 'Subflow Bwd Packets', 'Init_Win_bytes_forward', 'Init_Win_bytes_backward', 'act_data_
pkt_fwd', 'min_seg_size_forward', 'Active Mean', 'Active Min', 'Idle Std', 'Idle Min', 'Inbound']
```

Fig. 8 Optimal features selected by using RFECV

3.4.2 Decision Tree

The decision tree [33] is a supervised machine-learning model utilized for both classification and regression tasks, though it is predominantly employed for classification purposes. Named for its structure resembling a tree, it begins with a root node. Decisions or tests are based on the features of the specific dataset. Each branching of the tree signifies a decision, determined by calculating metrics that indicate the optimal feature at each step. Two commonly used metrics are Information Gain and the Gini Impurity Index. We utilized the Gini Index as the metric for selecting decision nodes. The tree construction is facilitated using the CART method, which stands for Classification and Regression Tree algorithm. It is an iterative process that includes splitting the data into partitions and then partitioning it further on each branch. Figure 9 depicts the basic structure of a decision tree.

3.4.3 Random Forest

Random Forest Classifier [3] also comes under a supervised set of machine learning algorithms. It works by producing more than one decision tree on many sub-samples of data. It makes a decision on considering the prediction result of many decision trees and based on the majority concludes the result. Thus, it avoids the prediction averaging and machine learning model overfitting issues. On the other side normal decision tree classifier makes a tree on the whole dataset which is supplied at the time of training. The Random Forest operates in two phases: first, it combines N decision trees to construct the random forest, and second, it generates predictions for each tree built in the initial phase. The Random Forest method is illustrated in Fig. 10.

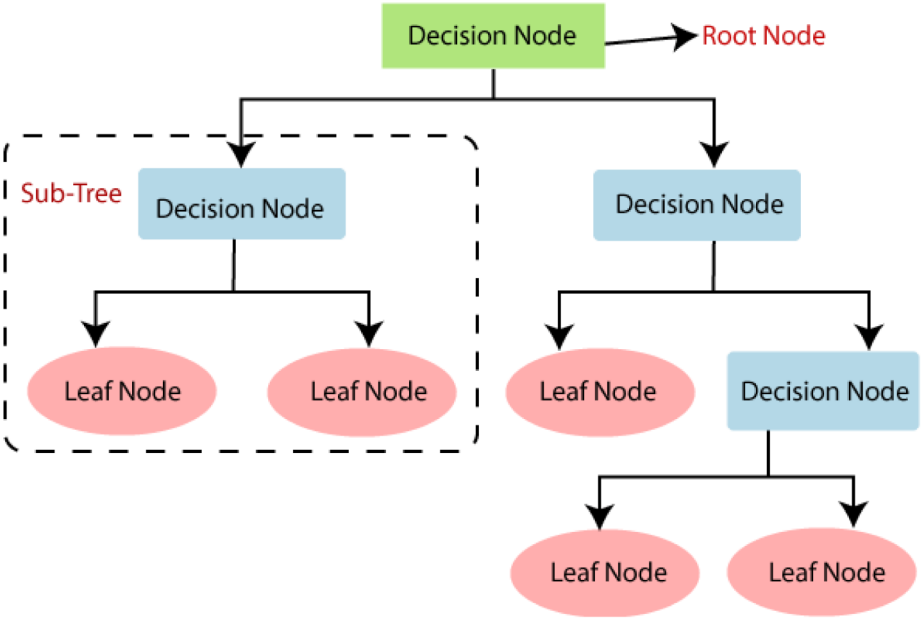

Fig. 9 Diagrammatic representation of decision tree classifier

3.4.4 Binary Classification Using Deep Learning Neural Network

Binary classification is used to classify the result into two categories, in our case these two categories are malicious traffic and benign traffic. Binary classification is the most common form of machine learning models but when the data amount is high, deep learning surpasses other techniques. Due to the presence of huge datasets in the IoT DDoS field, the application of deep learning is preferable. Binary classification using neural networks involves two simple changes which are, adding the activation function at the output layer and changing the loss function [34, 35].

The proposed approach utilizes the 'Keras' machine learning deep learning neural network library for Binary Classification. Keras is an open-source Python programming language library that provides API access to Artificial Neural Networks. It uses Google Tensorflow in the backend. It contains various commonly used neural network components like layers, activation functions, objectives, optimizers, etc. to work with different types of data like text and images. Keras also supports recurrent neural networks and convolutional and various commonly used utilities like pooling, dropout, and batch normalization.

For this binary classification, we are using two classes namely Benign (Normal Traffic) and Malicious (Another type of attacking Traffic) Traffic. We have taken 1 lakh rows from each CSV to train our neural network model and to make data equally distributed we are using Quantile Transformer. Quantile Transformer transforms the given features to follow a uniform/normal distribution. It helps in avoiding problems related to biased decision-making due to the presence of outliers in the dataset. Figure 11 shows an example of quantile transformation.

Coming to the most important part of the Artificial Neural Network model. We have used a sequential model from the Keras library having 4 layers.

- For the first layer, we have 20 sets of inputs and 64 outputs.
- For the second and third layers, we have the same 64 outputs and 64 inputs.
- Except for the last layer, we are using "relu" as our activation function.

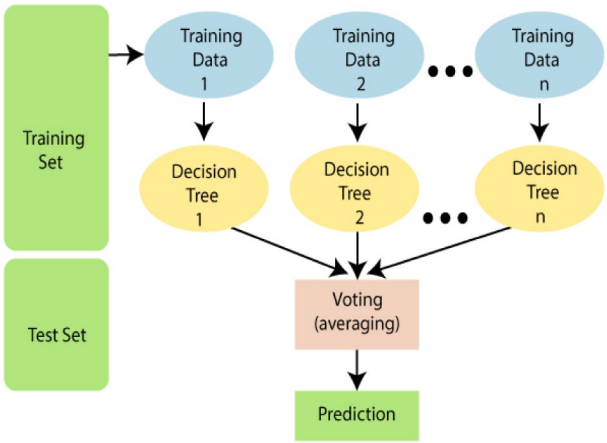

Fig. 10 Diagrammatic representation of random forest classifier method

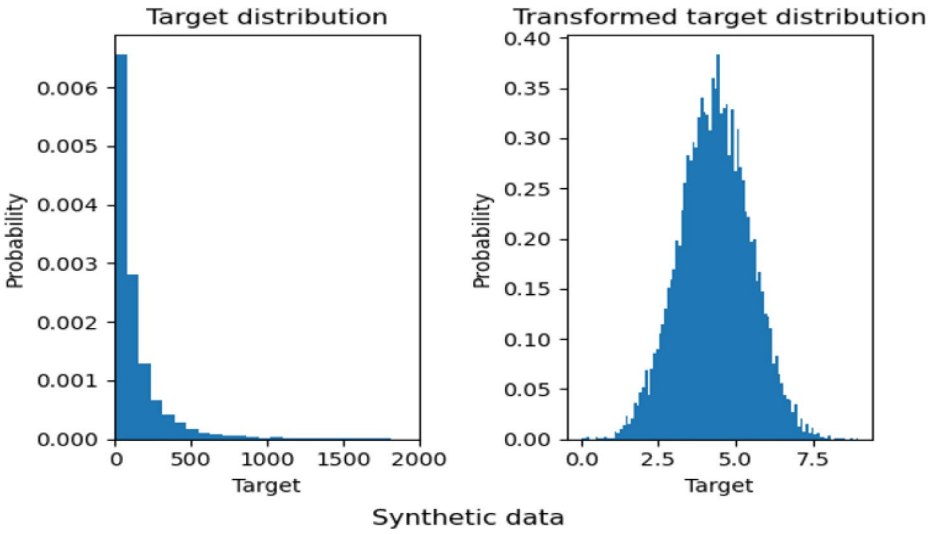

Fig. 11 An example of quantile transformation

4 Results and Analysis

This section outlines the experimental setup utilized for the proposed methodology, presents the obtained results, and compares them with other approaches. The comparison is performed using a comprehensive set of evaluation metrics, including confusion matrices, accuracy, precision, recall, and F1 score. Additionally, a comparative analysis has been conducted between the results achieved after feature selection and those obtained without feature selection.

4.1 Experimental Setup

We have used Google Colab to implement the proposed methodology. For our experiments, we utilized a range of essential Python libraries, including pandas, numpy, sys, scikit-learn (commonly known as sklearn), glob, and matplotlib. These libraries provided crucial functionalities for data manipulation, numerical computing, system-level interactions, machine learning algorithms, file path handling, and data visualization, respectively.

4.1.1 Subset Selection

The proposed approach utilizes the CICDDoS2019 dataset, which comprises 30GB of multiple CSV files with approximately 1 million rows each, loading the entire dataset into Google Colab was not feasible due to the substantial size of the dataset. Therefore, to address this issue we used a systematic approach to sample a representative subset from each CSV file. To ensure a fair distribution of data for feature selection, we randomly sampled 20,000 rows from each CSV file and proceeded with data preprocessing. The subset selection is utilized to reduce computational complexity while maintaining statistical validity. Random sampling involves selecting data rows from a population in such a way that each data row

has an equal chance of being chosen. In the context of the CICDDoS2019 dataset, each row in the CSV files represents the information held by a network packet. This approach helps in avoiding bias and ensures that the selected subset is a fair representation of the entire dataset.

4.1.1.1 Merging Steps for Combining the Selected Samples Selecting 20,000 rows as a subset from each file resulted in 380,000 rows for analysis from the CICDDoS2019 dataset which offers a balanced approach, providing sufficient statistical representation, variability, and computational efficiency. To combine samples from different CSV files into a unified training and test dataset, we followed some merging steps:

- **Load CSV Files**: Initially, all relevant CSV files containing samples of the CICDDoS2019 dataset are loaded into memory.
- **Merge Data**: After loading the CSV files, the data from these files is merged into two CSV files: one for the data from the first day and another for the data from the second day by concatenating the data frames along the rows axis which ensures that the samples are combined sequentially.
- **Shuffle Data**: The merged dataset is shuffled randomly to prevent any inherent ordering or bias in the dataset. This ensures that the samples are mixed uniformly which reduces the risk of creating unintended patterns or dependencies during model training and evaluation.

4.1.2 Data Preparation

During the merging process, it is essential to preprocess the dataset to handle any missing or null values present in the individual CSV files. Therefore, after sampling the subset, we conducted data cleaning and preprocessing steps to ensure data quality and consistency. This included handling missing values and standardizing features using sklearn. preprocessing.StandardScaler, and performing feature selection using RFECV and decision tree as the estimator. The *sklearn.preprocessing.StandardScaler* standardizes the features by removing the mean and scaling to unit variance. The RFECV, implemented in the Scikit-learn machine learning library, enabled us to select the optimal features for training our classification models while mitigating the risk of overfitting.

4.1.3 Training and Testing

Following data preparation, we partitioned the sampled subset into training and testing sets for model evaluation. The training set includes the first-day data which was used to train various classification algorithms including Gaussian Naïve Bayes, Decision Tree, Random Forest, and Multi-Layer Perception. Subsequently, we evaluated the trained models using the testing set which includes the second day data to assess their performance in detecting DDoS attacks.

After feature selection, we tuned the classifier models and pipelined them to validate the performance. For the machine learning classifiers, we have used the respective python

functions and for binary classification using deep learning neural network, we have used the keras python library to provide API access to Artificial Neural Network and quantile transformer for equal distribution of the data.

4.2 Performance Matrices

The Performance Metrics section provides a detailed analysis of various evaluation measures used to assess the effectiveness of classification models. Performance metrics such as confusion matrix, accuracy, precision, recall, and F1 score offer quantitative measures of a classification model's ability to accurately identify and classify instances. These metrics provide insights into the model's capacity to minimize false positives, false negatives, and overall classification errors, thereby guiding decisions regarding model selection and optimization.

4.2.1 Confusion Matrix

The confusion matrix illustrates the classifier model's performance in the matrix format. For any binary classifier, the confusion matrix uses four terms which are: 'True Positive', 'False Negative', 'False Positive', and 'True Negative'. The confusion matrix data can be used to calculate the other performance matrices.

- True Positive (TP) can be defined as the situation when the model identifies the malicious traffic while it is the malicious one.
- False Negative (FN) is when the model identifies the traffic as benign or non-attack but in reality, it is the attacking traffic.
- False Positive (FP) occurs when the traffic is identified as malicious but it is not.
- True Negative (TN) is the situation when the traffic is benign and correctly detected by the classifier model.

The confusion matrices for the classifiers we have used are shown in Figs. 12 and 13 without feature selection and with feature selection respectively.

4.2.2 Accuracy

Accuracy is used to determine how correctly the model identifies the malicious and benign traffic; it can be calculated as:

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

4.2.3 Precision

Precision is used to calculate the percentage of correctly identified packets from the total number of identified packets which can be calculated as:

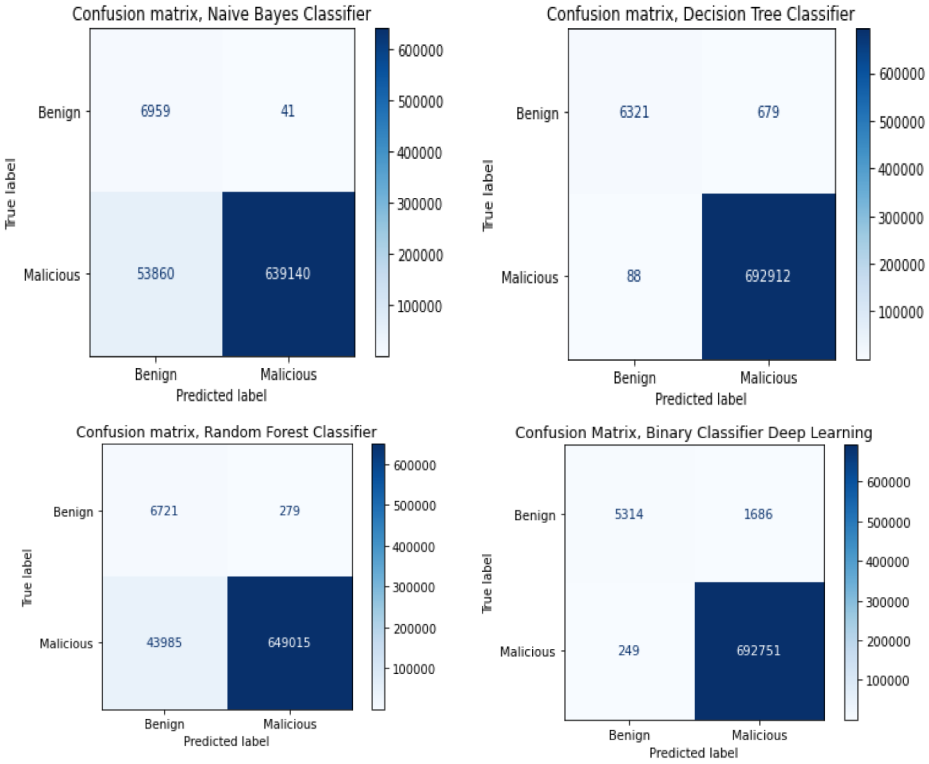

Fig. 12 Confusion matrices for the classifiers (without feature selection)

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

4.2.4 Recall

Recall calculates the percentage of correctly predicted packets and can be calculated by using Eq. 5.

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

4.2.5 F1 Score

F1 score is defined as the harmonic mean of precision and recall matrices and gives their combined result. The F1 score can be calculated by using Eq. 6.

$$F1\,Score = \frac{2}{1/Recall + 1/Precision} \text{ or } F1\,Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

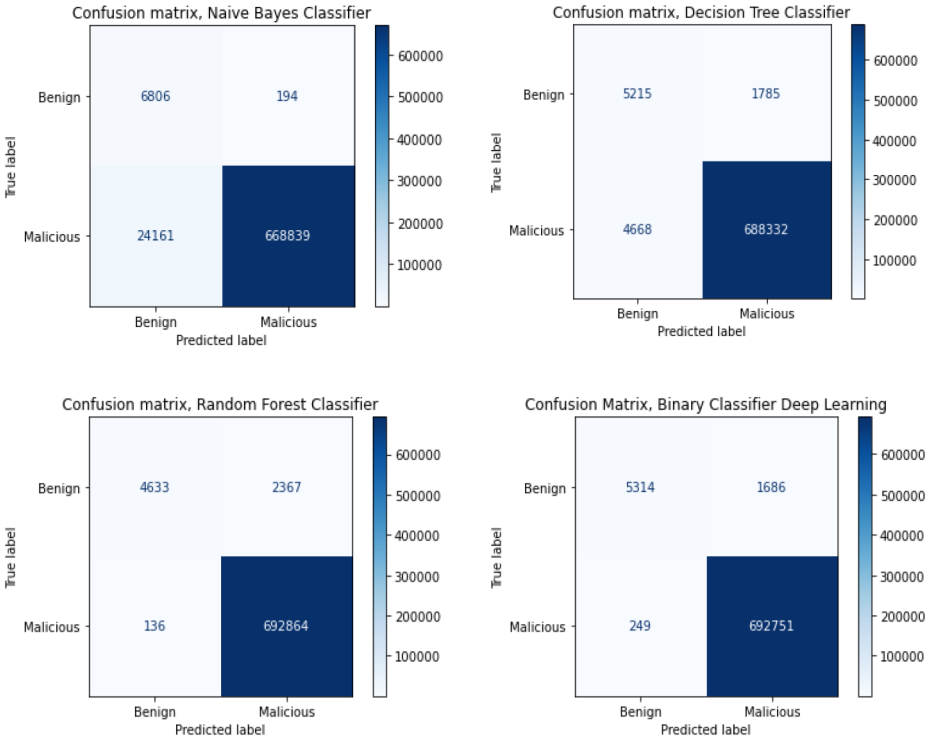

Fig. 13 Confusion matrices for the classifiers (with feature selection)

4.3 Comparative Analysis of Obtained Results

This section provides a comprehensive comparison of the results obtained from different classifiers before feature selection and after optimal feature selection. Table 1; Fig. 14 represent the comparative analysis of results obtained by different classifier models without feature selection based on the stated parameters and Table 2; Fig. 15 represent the comparative analysis of results obtained by using the RFECV feature selection method. Table 3 provides the comparison of our proposed approach with the existing approaches which are validated on the same dataset i.e., the CICDDoS2019 dataset.

Figure 16 represents the comparative analysis of accuracy, recall, precision, and F1 score results obtained by the classifiers before and after applying the feature selection method.

To ensure the statistical soundness of the obtained results despite the similarities in performance, we employed a variety of robust statistical measures. These measures provide a comprehensive assessment of model performance and mitigate the potential for bias or uncertainty in our findings. The k-fold cross-validation method is used to evaluate the robustness of the proposed model and reduce the risk of overfitting. The confusion matrix provides a detailed analysis of model predictions and actual results, evaluates model performance across classes, and identifies potential weaknesses and areas of misclassification. Furthermore, the performance metrics, accuracy, precision, recall and F1 score provide a quantitative assessment of the model's accuracy, precision in positive predictions, recall

Table 1 Comparison of classification models (without feature selection)

Classifier	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
Gaussian Naïve Bayes	92.2998	92.2279	99.9935	95.9539
Decision Tree	99.8904	99.9873	99.9021	99.9446
Random Forest	93.6765	93.6529	99.9570	96.7023
Multi-Layer Perception	99.958	99.9984	99.9591	99.9787

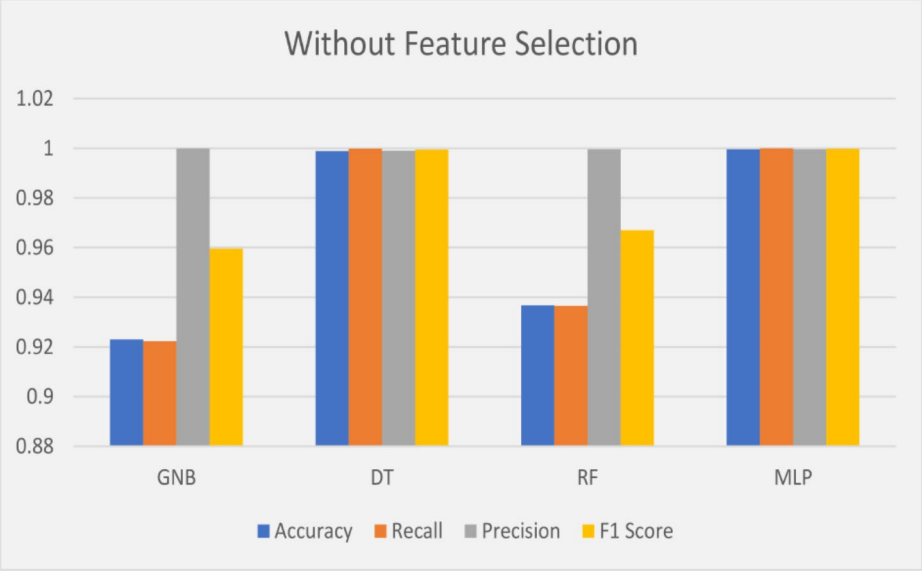

Fig. 14 Performance matrices for the classifiers functions (without feature selection)

Table 2 Comparison of classification models (with feature selection)

Classifier	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
Gaussian Naïve Bayes	96.5207	96.5135	99.9710	98.2118
Decision Tree	99.0781	99.3264	99.7413	99.5334
Random Forest	99.6424	99.9803	99.6595	99.8196
Multi-Layer Perception	99.7235	99.9640	99.7572	99.8605

or sensitivity to detect positive instances, and the harmonic mean of precision and recall, respectively.

4.3.1 Performance Discrepancies among Models

The disparities in performance among different classification models highlight the critical importance of selecting suitable algorithms and optimizing feature selection techniques for DDoS attack detection.

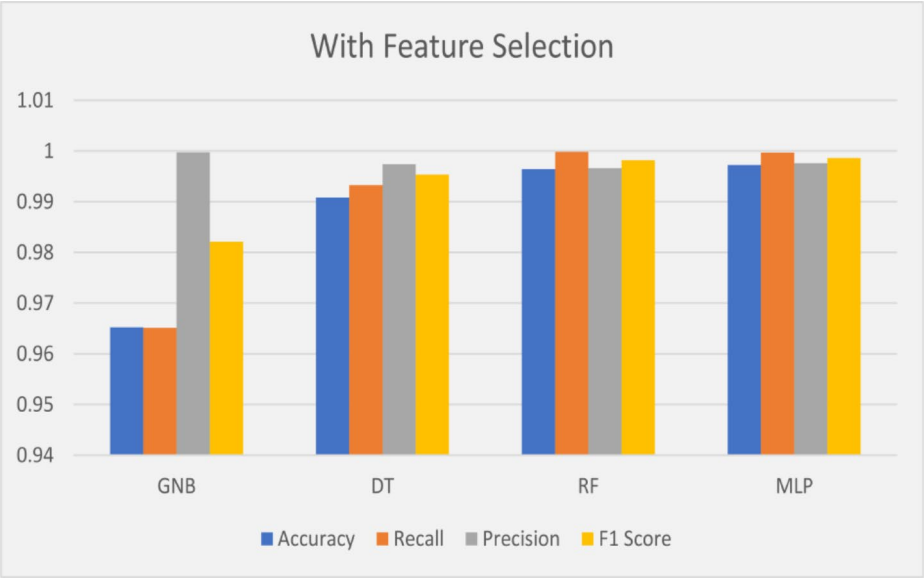

Fig. 15 Performance matrices for the classifiers functions (with feature selection)

Table 3 Comparison of proposed approach with existing approaches

Author	Methodology	Accuracy	Precision	Recall	F1-score
Myneni, et al. [10]	LSTM with Edge Computing	0.97	-	-	-
Batchu, et al. [30]	Gradient Boost with Hybrid Feature Selection and Hyperparameter Optimization	0.99	0.98	0.99	0.99
Cheng, et al. [36]	M.S-CNN	0.94	-	-	-
Cil, et al. [37]	Feed Forward DL	0.94	0.80	0.95	0.87
Daniyal, et al. [38]	CNN+BILSTM with improved FS	0.94	0.94	0.92	0.93
Proposed Approach	MLP+RFECV	0.99	0.99	0.99	0.99

- **Gaussian Naïve Bayes (GNB)**: In both scenarios, GNB exhibited relatively lower performance compared to other models. The simplistic assumption of feature independence in GNB limits its ability to capture the complex dependencies present in DDoS attack patterns. As a result, GNB struggles to capture the underlying relationships and dependencies in the data, leading to suboptimal performance.
- **Decision Tree and Random Forest**: The decision tree and random forest classifiers exhibited high accuracy in both scenarios, outperforming GNB. This is attributed to their ability to capture non-linear relationships and interactions among features. While both of the models demonstrated high accuracy, their performance may be impacted by the overfitting in the absence of feature selection. Without feature optimization, Random Forest and Decision Tree models may prioritize less relevant features, resulting in suboptimal classification performance.
- **Multi-Layer Perception (MLP)**: MLP consistently performed exceptionally well across both scenarios, achieving the highest accuracy and F1 score. This is due to its

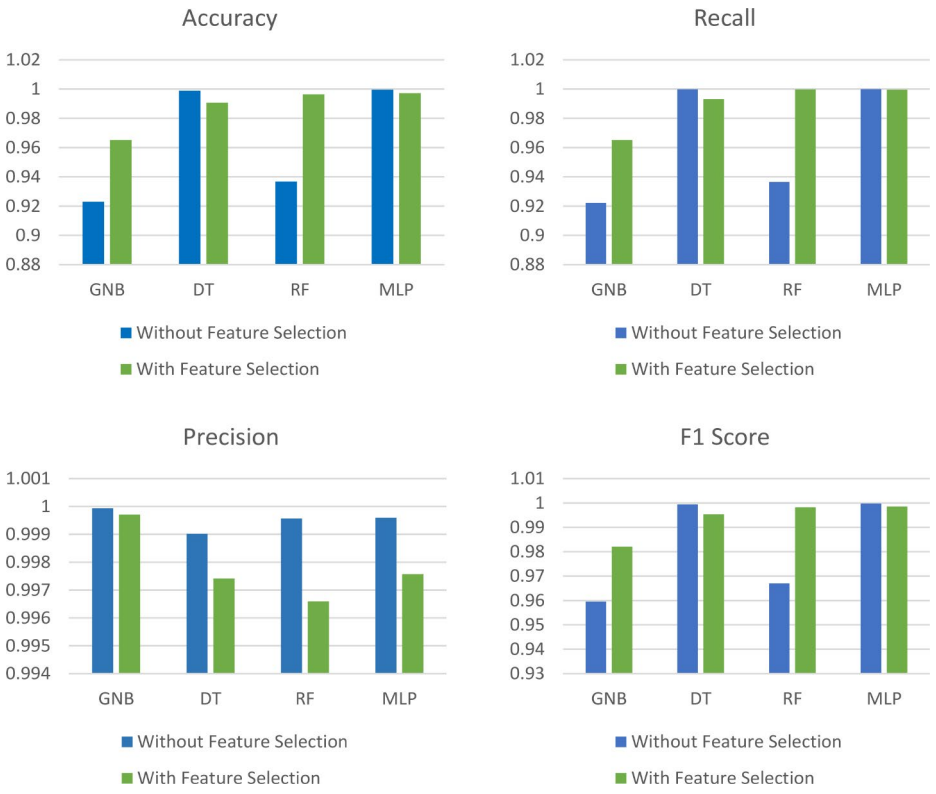

Fig. 16 Comparative analysis of results obtained by with feature selection and without feature selection

ability to learn complex patterns and relationships in data through multiple hidden layers. Additionally, the inclusion of feature selection further enhanced the performance of MLP, underscoring the importance of feature optimization in Deep learning model.

5 Conclusion

In this work, we have analyzed the CICDDoS-2019 dataset and trained our model on a subset of the same dataset taking features of over 10 lakh data packets. We have used the RFECV (with decision tree as the estimator) feature selection algorithm to get the optimal features and provide the obtained optimal features to the classifier models. We got the highest accuracy of 99.72% with the binary classification using deep learning with recursive feature elimination using a decision tree classifier as an estimator function. We also compared the results obtained after feature selection using RFECV with the results obtained without feature selection using various machine learning models like Naïve Bayes, Decision Tree, and Random Forest with the accuracy of 96.52%, 99.07%, and 99.64% respectively.

The collected results will help select a detection strategy for DDoS attacks in IoT devices across various use cases. For a detection system inside the IoT device itself, we might also

look at factors such as the resource consumption of the model and thus might give some free hand on the accuracy part and opt for the machine learning models while if we are looking for a highly accurate solution, we might go to deep learning-based solutions and instead of installing the detection mechanism on each IoT device, we can install the system on a hub device which will filter out the malicious packets.

6 Limitations and Future Directions

while the research represents a significant step toward improving DDoS attack detection in IoT devices, there are inherent limitations and opportunities for further exploration. By addressing these limitations and pursuing future research directions, we can advance the state-of-the-art in cybersecurity and contribute to the development of more robust and adaptive defense mechanisms against DDoS attacks.

- **Dataset Augmentation**: While our study relies on the CICDDoS-2019 dataset, its limited diversity may not fully represent all possible DDoS attack scenarios and network conditions. Augmenting the dataset with a larger number of attack situations and network conditions is essential to enhance the robustness and generalizability of our detection models.
- **Resource Consumption**: The resource requirements of our proposed detection system, especially for deep learning-based approaches, may be prohibitive for resource-constrained IoT devices. Optimizing models for reduced memory and computational requirements while maintaining detection accuracy is crucial for practical deployment in IoT environments.
- **Scalability**: Scaling our approach to larger datasets and real-time monitoring scenarios presents significant challenges. Further research is needed to address scalability issues and ensure efficient processing of data streams in dynamic IoT environments.
- **Dynamic Adaptation**: Developing adaptive detection mechanisms capable of dynamically adjusting to evolving attack patterns and network conditions is critical. Techniques such as online learning, reinforcement learning, and anomaly detection can enable continuous model updates and refinement in real time.
- **Edge Computing**: Exploring the feasibility of deploying detection mechanisms directly on IoT devices or at the network edge can improve detection efficiency and reduce latency. Edge computing architectures enable localized processing and analysis of network traffic, minimizing reliance on centralized systems and mitigating communication overhead.

Author Contributions All authors contributed to the study conception and design. The first draft of the manuscript was written by Pooja Kumari, Yash Pal, Kuldeep Singh, and Anubhav Singh. Ankit Kumar Jain provided supervision. All authors read and approved the final manuscript.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data Availability The datasets analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing Interests We declare that we have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Kumari, P., & Jain, A. K. (2024). Timely detection of DDoS attacks in IoT with dimensionality reduction. *Cluster Computing*, pp. 1–19.
2. Gopi, R., Selvakumar, S., Sathiyamoorthi, V., Manikandan, R., Chatterjee, P., Jhanjhi, N. Z., & Luhach, A. K. (2021) Enhanced method of ANN based model for detection of DDoS attacks on multimedia internet of things. *Multimedia Tools and Applications*, 1–19.
3. Mishra, B. B., Gupta, D., Perakoviċ, F. J. G., Peñalvo, & Hsu, C. H. (2021), Classification based machine learning for detection of DDos attack in cloud computing, *IEEE International Conference on Consumer* Electronics *(ICCE)*, Las Vegas, NV, USA, 2021.
4. Sanchez, O. R., Repetto, M., Carrega, A., & Bolla, R. (2021). Evaluating ML-based DDoS Detection with Grid Search Hyperparameter Optimization, in *IEEE 7th International Conference on Network Softwarization (NetSoft)*, 2021.
5. Zagrouba, R., & Alhajri, R. M. (2021). Machine learning based Attacks Detection and countermeasures in IoT. *International Journal of Communication Networks and Information Security (IJCNIS)*, *13*(2), 158–167.
6. Zhao, K., Lu, B., Shi, H., Ren, G., & Zhang, Y. (2021). A DDoS attack detection and defense mechanism based on the self-organizing mapping in SDN. *Internet Technology Letters*, *e305*, 1–6.
7. Kumari, P., & Jain, A. K. (2023). A comprehensive study of DDoS attacks over IoT network and their countermeasures. *Computers & Security*, *127*(103096), 1–23.
8. Wani, S., Imthiyas, M., Almohamedh, H., Alhamed, K. M., Almotairi, S., & Gulzar, Y. (2021). Distributed denial of service (DDoS) mitigation using Blockchain—A Comprehensive Insight. *Symmetry*, *13*(227), 1–21.
9. Ye, J., Cheng, X., Zhu, J., Feng, L., & Song, L. (2018). A DDoS attack detection method based on SVM in Software defined Network. *Security and Communication Networks*, *9804061*, 1–8.
10. Myneni, S., Chowdhary, A., Huang, D., & Alshamrani, A (2022). SmartDefense: A distributed deep defense against DDoS attacks with edge computing. *Computer Networks*, *209*, 108874, 1–12.
11. Kovacs, E. (2020) *Google Targeted in Record-Breaking 2.5 Tbps DDoS Attack in 2017*, 19 October 2020. [Online]. Available: https://www.securityweek.com/google-targeted-record-breaking-25-tbps-ddos-attack-2017. [Accessed 2021].
12. Devdiscourse (2020). *Google absorbed record-breaking 2.5 Tbps DDoS attack in September 2017*, 17 October 2020. [Online]. Available: https://www.devdiscourse.com/article/technology/1264631-google-absorbed-record-breaking-25-tbps-ddos-attack-in-september-2017. [Accessed 2021].
13. Aljuhani, A. (2021). Machine learning approaches for combating distributed denial of service attacks in Modern networking environments. *IEEE Access: Practical Innovations, Open Solutions*, 9, 42236–42264.
14. Raza, A. (2021). *Russian Internet Giant Suffers Largest DDoS Attack in History*, Koddos, 17 September 2021. [Online]. Available: https://blog.koddos.net/russian-internet-giant-suffers-largest-ddos-attack-in-history/. [Accessed 4 October 2021].
15. Tuan, N. N., Hung, P. H., Nghia, N. D., Tho, N. V., & Phan, T. V. (2020). A DDoS Attack Mitigation Scheme in ISP networks using machine learning based on SDN. *Electronics*, *9*(413), 1–19.
16. Gaurav, B. B., Gupta, & Panigrahi, P. K. (2022). A novel approach for DDoS attacks detection in COVID-19 scenario for small entrepreneurs. *Technological Forecasting & Social Change*, *177*(121554), 1–11.
17. Tuan, T. A., Long, H. V., Son, L. H., Kumar, R., Priyadarshini, I., & Son, N. T. K. (2020). Performance evaluation of Botnet DDoS attack detection using machine learning. *Evolutionary Intelligence*, *13*(2), 283–294.
18. Prasad, & Chandra, S. (2022). VMFCVD: An optimized Framework to combat volumetric DDoS attacks using machine learning. *Arabian Journal for Science and Engineering*, pp. 1–19.
19. Kshirsagar, D., & Kumar, S. (2022). A feature reduction based reflected and exploited DDoS attacks detection system. *Journal of Ambient Intelligence and Humanized Computing*, *13*(1), 393–405.

20. Kebede, S. D., Tiwari, B., Tiwari, V., & Chandravanshi, K. (2022). Predictive machine learningbased integrated approach for DDoS detection and prevention. *Multimedia Tools and Applications*, *81*(3), 4185–4211.
21. Kumari, P., Jain, A. K., Seth, A., & Raghav (2024). Leveraging blockchain and machine learning to counter DDoS attacks over IoT network. *Multimedia Tools and Applications*, pp. 1–25.
22. Yang, L., & Zhao, H. (2018). DDoS Attack Identification and Defense using SDN based on Machine Learning Method, in *15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN)*, Yichang, China, 2018.
23. Mittal, M., Kumar, K., & Behal, S. (2022). Deep learning approaches for detecting DDoS attacks: A systematic review. *Soft Computing*, pp. 1–37.
24. Sharafaldin, A. H., Lashkari, S., Hakak, & Ghorbani, A. A. (2019). Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy, in *International Carnahan Conference on Security Technology (ICCST)*, Chennai, India, 2019.
25. Manohar, H., K, A. H., & Prasad, B. G. (2019). DDoS Attack Detection using C5.0 machine learning algorithm. *International Journal of Wireless and Microwave Technologies*, *9*(1), 52–59.
26. Bhuvaneswari Amma, N.G. & Selvakumar, S. (2019), Deep radial intelligence with cumulative incarnation approach for detecting denial of service attacks. *Neurocomputing*, 340, 294–308,
27. Dayanandam, G., Rao, T.V., Bujji Babu, D., & Nalini Durga, S. (2019). DDoS Attacks—Analysis and Prevention, *Innovations in Computer Science and Engineering. Lecture Notes in Networks and Systems*, vol. 32, pp. 1–10.
28. Mallikarjunan, N., Bhuvaneshwaran, A., Sundarakantham, K., & Shalinie, S. M. (2019). Detecting DDoS attacks using Machine Learning Approach. *Computational Intelligence: Theories, Applications and Future Directions*, *1*, 261–273.
29. Sharma, K., Dhankhar, T., Agrawal, G., Singh, S. K., Gupta, D., Nebhen, J., & Razzak, I. (July 2021). Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks. *Ad Hoc Networks*, *121*, 1–9.
30. Batchu, R. K., & Seetha, H (2021). A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning. *Computer Networks*, 200, 108498, 1389–1286.
31. Rahman, O., Quraishi, M. A. G., & Lung, C. H. (2019). DDoS Attacks Detection and Mitigation in SDN using Machine Learning, in *2019 IEEE World Congress on Services (SERVICES)*, Milan, Italy.
32. Prasad, A., & Chandra, S (2022). VMFCVD: An optimized Framework to combat volumetric DDoS attacks using machine learning. *Arabian Journal for Science and Engineering*, pp. 1–19.
33. Sangodoyin, O., Akinsolu, M. O., Pillai, P., & Grout, V. (2021). Detection and classification of DDoS flooding attacks on Software-defined networks: A Case Study for the application of machine learning. *IEEE Access: Practical Innovations, Open Solutions*, *9*, 122495–122508.
34. Pérez-Díaz, I. A., Valdovinos, K. K. R., Choo, & Zhu, D (2020). A flexible SDN-Based Architecture for identifying and mitigating low-rate DDoS attacks using machine learning. *IEEE Access: Practical Innovations, Open Solutions*, 8, pp. 155859–155872, 25 August 2020.
35. Wei, Y., Jaccard, J. J., Sabrina, F., Singh, A., Xu, W., & Camtepe, S. (2021). AE-MLP: A Hybrid Deep Learning Approach for DDoS detection and classification. *Ieee Access: Practical Innovations, Open Solutions*, *9*, 146810–146821.
36. Cheng, J., Liu, Y., Tang, X., Sheng, V. S., Li, M., & Li, J. (2020). DDoS Attack Detection via Multiscale Convolutional neural network. *Computers Materials & Continua*, *62*(3), 1317–1333.
37. Cil, E., Yildiz, K., & Buldu, A. (2021). Detection of DDoS attacks with feed forward based deep neural network model. *Expert Systems with Applications*, *169*, 114520.
38. Alghazzawi, O., Bamasag, H., Ullah, & Asghar, M. Z. (2021). Efficient detection of DDoS attacks using a Hybrid Deep Learning Model with Improved feature selection. *Applied Sciences*, *11*(24), 11634.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Pooja Kumari is a Ph.D. scholar in National Institute of Technology, Kurukshetra, India. She has received her M.Tech degree in Computer Science and Technology (Cyber Security) from Central University of Punjab, Bathinda, India. Her research interests include, IoT Security, Machine Learning and Deep Learning, Network and Information Security.

Ankit Kumar Jain is presently working as Assistant Professor in National Institute of Technology, Kurukshetra, since September 2013. He received Master of technology from Indian Institute of Information Technology Allahabad (IIIT) India. Dr. Jain received PhD degree from National Institute of Technology, Kurukshetra in the area of Information and Cyber Security. He has published more than 50 research papers in International journals and conferences of high repute including Elsevier, Springer, Taylor & Francis, Inderscience, IEEE, etc. His general research interest is in the area of Information and Cyber security, Phishing Website Detection, Web security, Mobile Security, IoT Security, Online Social Networks and Machine Learning.

Yash Pal currently working as Software Engineer with keen interest in problem solving. He completed his B.Tech. in Information Technology from National Institute of Technology Kurukshetra, India.

Kuldeep Singh pursued his early education in Hisar, India. He completed his graduation from National Institute of Technology Kurukshetra, India with Information Technology specialization in 2022. During his graduation, He also got the chance to work as a research and development Intern at Logic Fruit Technologies, Gurugram. Currently he is working as a Software Development Engineer at Adobe Systems, India.

Anubhav Singh pursued his early education in Jaipur, Rajasthan, India. Later, He completed his graduation from National Institute of Technology, Kurukshetra with Information Technology specialization in 2022. Currently he is working as Software Engineer.