DETECTION AND CLASSIFICATION OF DISTRIBUTED DENIAL-OF-SERVICE ATTACKS

Malathi Paladugu - 23M1068, Dhanvi Karanam - 23M1190

Department of Electrical Engineering, IIT Bombay

ABSTRACT

A denial-of-service attack(DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. Distributed DoS(DDoS) attacks originate from multiple hosts while DoS attacks originate from a single host. Machine learning has been widely used for intrusion detection and classification of the type of attack. This work provides the comparison of performance of different machine learning models used to detect and classify DDoS attacks. In this project, data is trained and tested using 4 different algorithms (extreme gradient boosting(XGBoost), Naive Bayes, K-nearest neighbor(KNN) and Stochastic Gradient Descent(SGD)) to detect and classify attacks and their performances were compared. The comparison shows that the XGBoost algorithm achieves the highest accuracy while the KNN algorithm gives a comparable figure.

1. INTRODUCTION

The internet connectivity has become ubiquitous across the world. The idea of Internet of Things(IoT) and connecting things with full freedom has increased the risk of cyber threats. One such threat is DDos attacks[1]. The multiple motives for these attacks are to have financial gain by threatening the organisations by service interruption, to gain competitive advantage by disrupting the services of other service providers, to promote political agenda, disrupting the infrastructure, military operations, government services of other nations, to gain notoriety within the hacking community. The common DDoS attack types are TCP flood, UDP flood, packet flood, SYN flood, RST flood, FIN flood, DNS reflection, DNSSec amplification. Many machine learning algorithms are designed to handle massive amounts of data[2]. The main purpose the classification algorithm in DDoS detection system is to identify and categorise requests resulting from a DDoS attack within regular traffic. The quantity of the dataset affects the model's precision and computational time. The pre-processing of the data can eliminate these affects and then the training can be done on the reduced dataset using different ML models.

EE769 - Introduction to Machine learning

2. TRADITIONAL DETECTION METHODS

Before the introduction of machine learning tools the DDoS attacks used to detect and prevent with several traditional methods and tools.

2.1. Threshold based detection

The network administrators used to set thresholds for various network metrics such as traffic volume, packet rate and so on. Based on the threshold it can be found the possibility of attack. But, it may not be correct because if the traffic is generated from actual source we may misinterpret it as a threat.

2.2. Signature-based detection

Signature-based detection look for known patterns or signatures of DDoS attacks. The signatures may be specific packet headers, payload characteristics or attack behaviors by analysing the previous incidents.

2.3. Anomaly-based detection

By analysing abnormalities in the traffic we may detect it as an attack.

2.4. Intrusion Detection Systems (IDS)

IDS not exclusively meant for DDoS detection but it can help finding abnormalities in traffic patterns Many machine learning algorithms are designed as distributed systems for scalability to handle huge data and to prevent DDoS attacks. The main motto of a classification algorithm is to recognise and categorise the requests from a DDoS attack within regular traffic in DDoS detection system. The performance of the model depends on the dataset and configuration of ML parameters. In this project, the data set is taken from Canadian Institute for Cybersecurity DDoS evaluation dataset (CIC-DDoS2019) which satisfies all the constraints of an intrusion detection dataset.

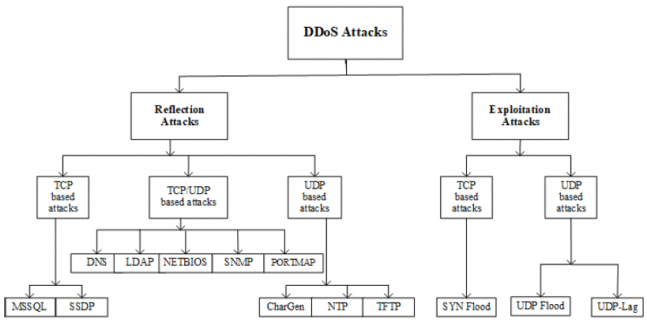

Fig. 1. Handling of data

3. DATA HANDLING

The data is acquired from the CIC-DDoS2019 dataset and is pre-processed. The pre-processing stages are mentioned below[3]. The pre-processing steps are shown in Fig.1

3.1. Data acquisition

The dataset in this project contains the information about 80 network traffic features and event logs for seven different kinds of DDoS attacks.[4]

3.2. Data pre-processing

The full dataset may have redundancies, missing values, zero values and noisy data which is unnecessary for processing while training the model. So, those data can be removed.[4]

3.2.1. Dealing missing values

The way we dealt the missing values in this project is by removing the rows of the dataset as we have abundance of data.

3.2.2. Dealing categorical data

In this dataset, the IP address, timestamp and label are categorical in nature. IP address is represented in dotted decimal form, timestamp in its yy-mm-dd hh:mm:ss and upto 5 decimal values of ms, label has eight distinct values, 7 of them represents different types of attacks and the 8^{th} one represents benign data. So, IP address and timestamp is converted into its equivalent integer or float data, label is converted to multi-column binary representation using one-hot-encoding.

3.2.3. Non-varying data removal

The spread for every column is checked and wherever deviation is close to zero, those columns are removed since it indicates that the quantity is almost constant throughout.

3.2.4. correlation check

The degree to which two or more variables are linearly related is given by the correlation. This can be visualised by correlation heat map using the seaborn library in python. The features with correlation more than 0.9 are replaced by one of the feature.

3.2.5. Normalisation

The data is normalised to bring the magnitude of all the features in the dataset in the same range convert it to its standard normal variate form i.e., mean zero and variance 1. It is done to reduce the computation time and to prevent the misinterpretation of the importance of data by its values of higher scale.

3.2.6. Train Test Split

The dataset is split into two parts i.e., training and testing in the ratio of 75-25.

4. CLASSIFICATION USING MACHINE LEARNING ALGORITHMS

This works uses four different machine learning algorithms to detect and classify the types of DDoS attacks. The comparison of models is done mainly by comparing accuracy. Other metrics such as F1 score and recall have also been computed.[4]

4.1. Extreme Gradient Boosting

This algorithm is a tree boosting algorithm which makes use of a decision tree approach to learn the relationships between features. Based on the supervised input, this algorithm creates trees and reduces the errors by creating new trees for correcting the mistakes made by the base tree. Such new models are added until there are no further improvements. Minimisation of errors has been achieved through the gradient descent algorithm.

4.2. K-Nearest Neighbour

This classification algorithm is based on the idea that data points belonging to the same class would be present closer to each other than to the points from another class. In short, similarity between points is assumed within a class while different classes are expected to be dissimilar. The training process determines the training centres of each class which are then used to classify new points based on their distance from the class centres. A new data point is assigned to the class of a class centre from which it has the least distance. The number of nearest neighbours has been set to 8.

4.3. Stochastic Gradient Descent

Stochastic gradient descent uses a random data point from a batch to perform gradient descent . Gradient descent is a technique used to minimise the error of an algorithm by moving in the direction opposite to the maximum rise of the error. Gradient descent involves the calculation of gradients at each step and is therefore, computationally heavy. To make it more computationally friendly, SGD selects a random point from a batch to be used for error calculation and backpropagation. This results in a decrease in the number of computations and a good performance on big datasets. This work uses a log loss which gives logistic regression. The L2 penalty has been used along with a log loss.

4.4. Naive Bayes

The Naive Bayes classifier is a probabilistic classifier which works on Bayes theorem. It operates under the assumptions that the features are independent of other and that they have en equal effect on the outcome. The Naive Bayes classifier then computes the probability of an event given that another event has happened. A new data point is classified to the class for which the computed probability is the highest.

5. RESULTS

In this work, the performance of the four machine learning models has been evaluated using eight different metrics. These metrics are accuracy, confusion matrix, precision, recall, F1 score, support, macro-average and weighted average.

5.1. Result Analysis

Data pre-processing split the dataset in such a way that a part of the dataset was kept apart to be used only for testing the performance. It is observed that all the algorithms were able to classify 'Non-Attack', 'LDAP', 'MSSQL', 'Syn' and 'UDP' classes with higher confidence but struggled to identify the classes of 'NetBIOS', 'UDPLag' and 'Portmap'.

Fig.2 displays the performance evaluation of the XGBoost algorithm. It achieves the highest accuracy of 97.24%. XGBoost does an excellent job at classifying Benign, LDAP, MSSQL, Syn, UDP and UDPLag attacks with an F1 score of 1.0 for all the mentioned classes while it classifies NetBIOS and Portmap attacks with F1 scores of 0.95 and 0.91 respectively. The overall macro F1 score is 0.98.

Fig.3 displays the performance evaluation of the stochastic gradient descent algorithm. It achieves an accuracy of 90.02%. It does a good job at classifying Benign, LDAP, MSSQL, Syn and UDP attacks with F1 scores of 0.974, 0.931, 0.963, 0.999 and 0.992 respectively. It struggles at classifying NetBIOS and Portmap attacks with F1 scores of 0.838 and 0.671. It does a poor job at classifying UDPLag attacks with an F1 score of 0.352. The overall macro F1 score comes out to be 0.84.

```
----------------------------------
Performance Results using XGBoost
----------------------------------

Accuracy is: 97.24307607408514
Recall is : 0.9724307607408514
Confusion matrix :
 [[ 2751     0     0     0     0     0     0     1]
 [    0 90471     0     0  9354     0     0     0]
 [    0     0  2446     0     0     0     0     0]
 [    0     0     0 53676     0     0     0     0]
 [    0   236     0     0 46894     0     0     0]
 [    0     0     0     0     0 70130     0     0]
 [    1     0     0     0     0     0 71519     0]
 [    0     0     0     0     0     0     0   445]]
Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      2752
           1       1.00      0.91      0.95     99825
           2       1.00      1.00      1.00      2446
           3       1.00      1.00      1.00     53676
           4       0.83      0.99      0.91     47130
           5       1.00      1.00      1.00     70130
           6       1.00      1.00      1.00     71520
           7       1.00      1.00      1.00       445

    accuracy                           0.97    347924
   macro avg       0.98      0.99      0.98    347924
weighted avg       0.98      0.97      0.97    347924

F1 Score :  0.9819260246548239
```

Fig. 2. Performance results using XGBoost algorithm

Fig.4 displays the performance evaluation of the K-nearest neighbour classification algorithm. It achieves an overall accuracy of 95.76%. It performs better than SGD at classifying NetBIOS, Portmap and UDPLag attacks with F1 scores of 0.924, 0.858 and 0.919 respectively. It does an exceptional job at classifying Benign, LDAP, MSSQL, Syn and UDP attacks with F1 scores of 0.997, 0.986, 0.999, 1.000 and 0.999 respectively. The overall macro F1 score is 0.960. The disadvantage with using the K-nearest neighbour algorithm is its complexity which makes it unsuitable for large datasets.

Fig.5 displays the performance evaluation of the Naive Bayes classifier. The overall accuracy obtained for this classifier is 86.12%. It does a good job at classifying LDAP, MSSQL, Syn and UDP attacks with F1 scores of 0.987, 0.995, 1.000 and 0.995 respectively. It struggles to classify Benign, NetBIOS and UDPLag attacks with F1 scores of 0.867, 0.810 and 0.623. It does a bad job at classifying Portmap attacks (F1 score = 0.000) mistaking most of them as NetBIOS attacks. It manages to get an overall macro F1 score of 0.784.

Fig.6 is a bar-graph which compares the overall accuracies of the four different machine learning algorithms while Fig.7 is a bar-graph comparing the overall macro F1 scores of the four different machine learning algorithms.

```
------------------------------------------------------
Performance Results using Stochastic Gradient Descent
------------------------------------------------------

Accuracy is  90.02540784769087
Recall :  0.9002540784769087
Confusion Matrix :
 [[ 2688     4     0     7    29    11    10     3]
 [    1 85151     0    54 14619     0     0     0]
 [    2     3  2291   148     0     2     0     0]
 [   17  2505   178 50390     0    10   576     0]
 [   28 15732     6   166 31198     0     0     0]
 [    4     0     0     0     0 70112     8     6]
 [    6    10     0   182     0    18 71290    14]
 [   21     0     0     0     0    26   298   100]]
Classification Report :
              precision    recall  f1-score   support

           0      0.971     0.977     0.974      2752
           1      0.823     0.853     0.838     99825
           2      0.926     0.937     0.931      2446
           3      0.989     0.939     0.963     53676
           4      0.680     0.662     0.671     47130
           5      0.999     1.000     0.999     70130
           6      0.988     0.997     0.992     71520
           7      0.813     0.225     0.352       445

    accuracy                          0.900    347924
   macro avg      0.899     0.824     0.840    347924
weighted avg      0.901     0.900     0.900    347924

F1 Score :  0.8401553533610508
```

Fig. 3. Performance results using SGD algorithm

```
------------------------------
Performance Results using KNN
------------------------------

Accuracy is :  95.76689162000896
Recall is :  0.9576689162000896
Confusion Matrix :
 [[ 2746     0     0     0     2     4     0     0]
 [    0 88401     0     1 11422     1     0     0]
 [    0     0  2427    18     0     1     0     0]
 [    1     2    48 53600     5     0    19     1]
 [    4  3098     0     0 44026     2     0     0]
 [    1     0     0     0     0 70114     0    15]
 [    2     0     0    23     0     4 71484     7]
 [    3     0     0     5     3    15    21   398]]
Classification Report :
              precision    recall  f1-score   support

           0      0.996     0.998     0.997      2752
           1      0.966     0.886     0.924     99825
           2      0.981     0.992     0.986      2446
           3      0.999     0.999     0.999     53676
           4      0.794     0.934     0.858     47130
           5      1.000     1.000     1.000     70130
           6      0.999     0.999     0.999     71520
           7      0.945     0.894     0.919       445

    accuracy                          0.958    347924
   macro avg      0.960     0.963     0.960    347924
weighted avg      0.962     0.958     0.958    347924

F1 Score :  0.9603597978156014
```

Fig. 4. Performance results using KNN algorithm

```
-----------------------------------
Performance Results using Naive-bayes
-----------------------------------

Accuracy is : 86.12081948931376
Recall is :  0.8612081948931376
Confusion Matrix :
 [[ 2614     0     0     0     0     0   138]
 [   59 99754     0     1     1     0    10     0]
 [   40     2  2383    21     0     0     0     0]
 [  197     1     0 53323     0     0   155     0]
 [  249 46776     0     0     0     0   105     0]
 [   23     0     0     0     0 70090     3    14]
 [   80     0     0   172     0    12 71155   101]
 [   15     0     0     0     0    16    98   316]]
Classification Report :
              precision    recall  f1-score   support

           0      0.798     0.950     0.867      2752
           1      0.681     0.999     0.810     99825
           2      1.000     0.974     0.987      2446
           3      0.996     0.993     0.995     53676
           4      0.000     0.000     0.000     47130
           5      1.000     0.999     1.000     70130
           6      0.995     0.995     0.995     71520
           7      0.555     0.710     0.623       445

    accuracy                          0.861    347924
   macro avg      0.753     0.828     0.785    347924
weighted avg      0.769     0.861     0.806    347924

F1 Score :  0.7845583409588428
```

Fig. 5. Performance results using Naive-Bayes algorithm

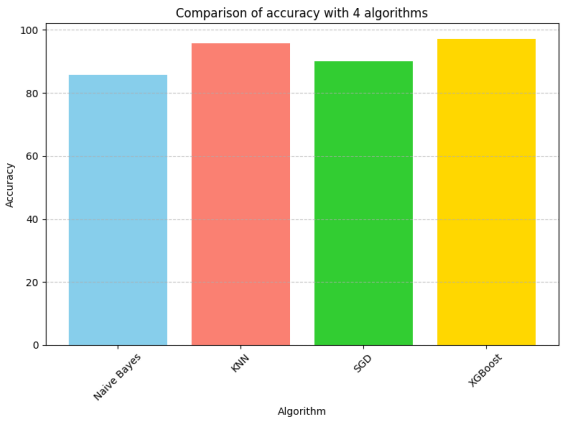

Fig. 6. Comparison of accuracy among 4 algorithms

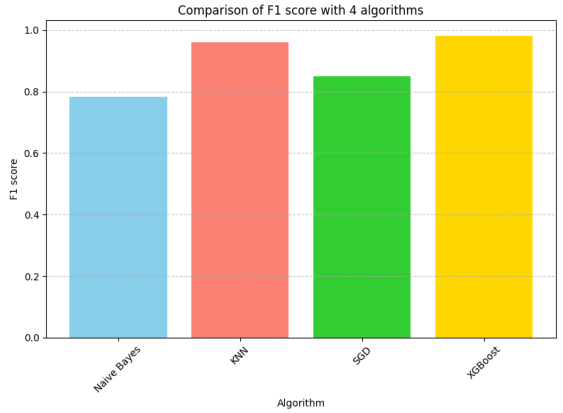

Fig. 7. Comparison of F1 score among 4 algorithms

6. REFERENCES

[1] M. Ahmim S. Namane A. Ahmim, F. Maazouzi and I. B. Dhaou, "Distributed denial of service attack detection for the internet of things using hybrid deep learning model," 2023, pp. 119862–119875.

[2] M. Sambath S. Santhosh and J. Thangakumar, "Detection of ddos attack using machine learning models," 2023, pp. 1–6.

[3] S. Hakak I. Sharafaldin, A. H. Lashkari and A. A. Ghorbani, "Developing realistic distributed denial of service (ddos) attack dataset and taxonomy," 2019 international carnahan conference on security technology (iccst)," 2019.

[4] Narang M. Kumar A Usha, G., "Detection and classification of distributed dos attacks using machine learning," 2021.