|| Volume 2|| Issue 08|| 2017|| ISO 3297:2007 Certified ISSN (Online) 2456-3293

OPEN ACCESS INTERNATIONAL JOURNAL OF SCIENCE & ENGINEERING

Design and Implementation of an Intelligent DDoS Attack Detection Framework Using Machine Learning

Pradeep Kundlik Deshmukh

Associate Professor, Department of Computer Engineering,
Government College of Engineering, Awasari - Kh, Pune, Maharashtra, India
pkdeshmukh9@gmail.com

Abstract: DDoS attacks have become an important concern in the security of networks because of the growing dependence on digital communication systems and cloud-based services. The paper gives the design and implementation of a smart DDoS attack detection framework using machine learning techniques. The major aim is to establish a system that will be able to classify network traffic data accurately to either normal or malicious traffic. The suggested framework also includes some critical steps such as preprocessing of data, extraction of features, and training of the model, which is necessary to achieve better detection. Several machine learning techniques, e.g. Support Vector Machine, Random Forest, and K-Nearest Neighbors are used to construct classification models. These models are tested and contrasted on the basis of the conventional performance measures such as accuracy, precision, recall and F1-score to establish the most effective way to detect DDoS. Through experimental findings, it is proved that ensemble-based and tree-based classifier are better than traditional classifiers in detection of complex attack patterns at lower false positive rates. The system will be a scalable and efficient system, which will facilitate real-time detection in an active network environment. The proposed framework offers beneficial information on the choice of the best models to be used in intrusion detection systems by making use of comparative analysis of alternative algorithms. The study helps in intensifying the security measures against cybercrimes by providing a trusted and intelligent mechanism to counter early detection and countermeasures of DDoS attacks in contemporary network systems.

Keywords: DDoS Attack Detection, Machine Learning, Intrusion Detection System, Network Security, Feature Selection, Real-Time Detection.

I.INTRODUCTION

In a digital age, network security has turned out to be a key issue since the number of internet-related services and interconnected systems has increased exponentially. Network infrastructures are essential to organizations in provision of services, storage of sensitive information and as a means of communication. Nonetheless, this heightened reliance has exposed networks to numerous cyber threats one of which is the Distributed Denial-of-Service (DDoS) attacks which are among the most disruptive. A DDoS attack is an attempt by an attacker to deny a target server, network, or application normal usage by flooding it with a huge amount of traffic utilizing a large number of distributed sources.

These attacks are commonly carried out with the help of botnets, i.e. compromised devices, including computers, servers, and internet of things devices run by attackers. The size and complexity of DDoS attacks have been increasingly advanced with time, and it has become more difficult to detect and counter. There is a constant changing of tactics by the attackers with low-rate attacks and application-layer flooding, some of which are employed to thwart the old methods of defense.

The implications of a DDoS attack are loss of money, loss of services, damaged reputation and even a data breach. With organizations constantly increasing their digital footprints, it has become imperative to have the ability to defend and eliminate DDoS attacks. Thus, there is an increasing necessity in smart and dynamic security systems that will be able to detect suspicious traffic patterns in real-time and protect network infrastructures in an efficient way.

Importance in Cloud, IoT, and Modern Systems (300 words)

The fast integration of cloud computing, Internet of Things (IoT) and up-to-date distributed systems have had an immense impact on how information is processed, stored, and transferred. Each of these technologies is scalable, flexible, and efficient, but at the same time, they come with new security threats, especially when it comes to DDoS attacks. Businesses all over the world have critical applications and services in cloud environments, which are the favorite targets of attackers. An effective DDoS attack on a cloud infrastructure can simultaneously impact several services, which affects many users and organizations.

Likewise, the presence of so many IoT devices has increased the attack surface exponentially. Most IoT devices lack sufficient computational capabilities, as well as have poor security settings that can be easily compromised. They end up being turned into botnets with these compromised devices and are then used to initiate mass DDoS attacks. Significant events have shown how botnets using the IoT can create huge amounts of traffic that can incapacitate even a well-established platform.

The network connectivity of modern systems (smart cities, autonomous vehicles, and industrial automation systems) is continuous and reliable. Any interference that DDoS attacks inflict may have grave consequences such as service problems to safety issues. Moreover, these systems are dynamic and distributed making centralized security solutions less effective. In this regard, it is important to have strong DDoS detection systems to provide the availability of the services, integrity of data, and user confidence. High-speed data streams, accommodating changing patterns of attack, and functioning effectively in distributed settings must be within the capability of more advanced detection systems. This points at the need to devise smart, scalable, and real-time solutions to secure the digital infrastructures of today.

Limitations of Traditional Detection Methods (150 words)

The main approaches to DDoS detection that have been traditionally used are signature-based and rule-based approaches that detect malicious traffic. Although this can be used to successfully combat familiar attack patterns, it cannot do the same with emerging and changing threats. Signature-based systems need regular updates on their databases and are therefore not so active in the case of zero-day attack or advanced version of existing attack. Also, rule-based systems are usually reliant on predefined thresholds, and these thresholds might not reflect the changing conditions in the network. This may result in great false positive and false negative rates, and thus decrease the detection reliability. The traditional techniques also do not have the capacity to process the large-scale and high-dimensional network traffic data effectively. Furthermore, the methods are not suitable to real-time detection in contemporary high-speed networks particularly, cloud and IoT. Due to this we are left with the need to have more adaptive and intelligent detection mechanisms that can address these limitations.

Motivation for Using Machine Learning

The traditional methods of DDoS detection that have been mostly employed are the signature and rule-based methods where malicious traffic is identified. This can help effectively fight common attack patterns, but it cannot fight new and evolving threats. The signature based systems must be updated periodically in their databases, thus not very active in case of zero-day attack or advanced version of the available attack. In addition, rule-based systems tend to be relying on pre-defined thresholds, and such thresholds may not mirror the dynamic environment of the network. This can lead to very large false positive and false negative rates, and, hence, reduced detection reliability. The old methods also lack the ability to handle the intensive and huge network traffic data that are at high dimensions. Moreover, the techniques cannot be applicable in real-time detection in modern high-speed networks especially, cloud and IoT. As a result we are forced to consider having more accommodating and smarter detection processes that can overcome these restrictions.

Problem Statement

Although there are different methods of detecting a DDoS, it is still a big problem to detect malicious traffic on time. Conventional methods of detection are not usually able to support high scale, dynamic and high-speed networked environments like cloud computing and IoT systems. These systems find it difficult to detect both advanced and low rate DDoS attacks, which causes higher false positives and low response times. Besides, the complexity of network traffic data increases, and it is not easy to determine the legitimate and illegal activities based on the traditional methods. It is absent of effective structures that are capable of offering a high rate of detection, low computation, and dynamic response to changing attack patterns.

Objectives of the study

- To design a machine learning-based framework for DDoS attack detection
- To classify network traffic into normal and malicious categories
- To apply multiple machine learning algorithms for detection
- To evaluate and compare the performance of different models
- To improve detection accuracy while minimizing false positives.

The paper follows the following structure: the introduction has background, motivation, and problem statement, whereas the literature review is contained in the second section. The proposed system architecture and methodology is provided in Section 3, and the results and the performance analysis are presented in Section 4. Lastly, Section 5 is the conclusion of the paper and the future research directions are outlined in it.

II.LITERATURE REVIEW

The authors Saad, R. M. A. et al. (2016) [1], recommended an intelligent model v6IIDS to identify ICMPv6 based DDoS flooding attacks with the help of Back- Propagation Neural Network. The paper is aimed at increasing the accuracy of detection in IPv6 systems through the analysis of traffic patterns. The findings indicate that the neural network model is accurate and low in false alarm rates in detecting flooding attacks. This paper underscores how deep learning methods can be useful in improving intrusion detection systems. A comparative study by Suresh, M., and Anitha, R. (2011) [2] on different machine learning algorithms in the detection of DDoS attacks in network system was conducted. The study compares algorithms like the

Decision Trees, the Naive Bayes and Support Vector machines according to the detection performance. The paper concludes that machine learning solutions are much more effective in terms of detecting a problem than conventional ones. It has underscored the need to identify the right algorithms to attain high efficiency and reliability in detecting DDoS. Kulkarni, A. et al. (2016) [3] suggested a dynamic Trojan detection system based on machine learning algorithms. This system is concerned with detecting hardware level threats, dynamically by analyzing the behavior patterns. The framework is proven to be efficient in real time detection and has a better adaptiveness to changing threats. This research paper reveals how machine learning has been successful in improving hardware-level security. A collaborative method of shrew DDoS attacks detection and filtering of spectral analysis was proposed by Chen, Y. et al. (2006) [4]. The technique examines recurring attacks on the traffic over networks to detect low-rate attacks that are hard to detect under traditional techniques. The suggested system enhances the level of detection and minimizes false positives. This paper discusses the significance of signal processing methods in DDoS monitoring. An actual time DDoS detection system by Radial Basis Function (RBF) neural networks and statistical traffic characteristics was introduced by Gavrilis, D., et al. (2005) [5]. The methodology allows to identify attacks early in the network by detecting anomalies in network behavior. Experimental findings indicate that the detection rate is very high and the computation cost is minimal. This work establishes the possibility of the neural networks in the real time intrusion detection systems.

A strategy to discriminate between the DDoS attacks and flash crowds was suggested by Li, K. et al. (2009) [6] based on probability-based measurements. The method can evaluate the traffic properties that can distinguish between the malicious and legitimate high-traffic cases. The findings show enhanced precision in detecting DDoS attacks and not false declaring flash crowds. This article deals with a vital issue of network traffic analysis. Mirkovic, J., et al. (2004) [7] have given an extensive taxonomy of the types of DDoS attacks, as well as defense mechanisms. The research classifies different attack strategies and associated mitigation solutions, which provide an organized knowledge of the DDoS threats. It is used as a reference point by network security researchers. This publication emphasizes the changing nature and intricacy of DDoS attacks and defenses. A technique to determine denial-of-service activity on the Internet-scale was introduced by Moore, D., et al. (2006) [8], using large-scale traffic measurements. Backscatter analysis is applied in the study to determine the extent and the effects of DDoS attacks. Findings indicate that this type of attack is common and can be effortlessly tracked with network level measurements. This can be of great help in understanding the dynamics of DDoS attacks around the world. Siris, V. A.,et al. (2006) [9] investigated the application of SYN flooding attacks detection algorithms in network traffic. They do this by studying the abnormalities in the normal traffic patterns in order to identify abnormal behavior. The findings indicate that anomaly-based practices can be successfully used to detect attacks with acceptable precision. This paper brings out the significance of statistical analysis in identifying the network intrusions. The mechanism suggested by Wang, H., et al. (2002) [10] to identify SYN flooding attacks is to track traffic patterns and connection states. The approach is aimed at determining the abnormal SYN packet rates to determine the distinction between attacks and normal traffic. The experimental results indicate a higher detection ability with minimal overhead. This work has also added to early work in protection against TCP-based DDoS attacks. These entropy differences in network traffic proposed a traceback system of DDoS attacks (Yu, S., et al., 2011) [11]. The technique calculates variation in traffic distribution in order to determine the source of attacks. It allows efficient tracing of the attackers in a distributed setting. The research promotes detection and mitigation measures on DDoS attacks.

Table 1: Comparative Analysis of Literature Review

Author & Ref No.	Methodology Used	Dataset Used	Advantages	Results
Saad et al. [1]	Back-Propagation Neural Network for ICMPv6 DDoS detection	Simulated IPv6 traffic dataset	High detection accuracy, suitable for IPv6 networks	Improved accuracy with low false alarms
Suresh & Anitha [2]	Comparative ML algorithms (SVM, Decision Tree, Naïve Bayes)	KDD Cup dataset	Comparative evaluation of multiple models	ML models outperform traditional methods
Kulkarni et al. [3]	Adaptive ML-based Trojan detection	Hardware-level datasets	Real-time detection, adaptive framework	Efficient detection of hardware Trojans
Chen & Hwang [4]	Spectral analysis for shrew DDoS detection	Simulated network traffic	Detects low-rate attacks effectively	Reduced false positives, improved detection
Gavrilis & Dermatas [5]	RBF Neural Network with statistical features	Real-time traffic data	Fast detection with neural networks	High detection rate with low overhead
Li et al. [6]	Probability-based metrics for DDoS detection	Network traffic dataset	Differentiates flash crowds from attacks	Improved classification accuracy
Mirkovic & Reiher [7]	Taxonomy-based analysis of DDoS attacks	Not dataset-based (survey)	Comprehensive classification of attacks	Foundational framework for DDoS defense
Moore et al. [8]	Backscatter analysis for DDoS	Internet traffic data	Large-scale attack	Identified global DDoS

III.SYSTEM ARCHITECTURE

The suggested system layout of DDoS attack recognition is developed as a pipeline that organizes the raw network traffic data into actionable predictions employing machine learning models. The first step is the dataset layer with the use of the NSL-KDD dataset. This data is labeled network traffic traffic data of both normal and attack patterns, on which training and evaluation are based.

The second step is data preprocessing that checks the quality and consistency of data. Missing values are also addressed properly, categorical variables are converted into numerical data through one-hot encoding, and the labels of the classes will be changed to binary values (normal = 0, attack = 1). This step is essential to train the machine learning algorithms. After preprocessing, it is followed by feature engineering to increase model performance. Unnecessary or redundant features are eliminated and only the most informative features are considered to ensure dimensional reduction and enhance the level of computational efficiency. This creates a smooth feature set to be used in model training. This processed data is then split in train-test split phase where it is usually split into 80% training and 20% test. This will make sure that the models are tested with unknown data, and the ability to generalize is reliable as well. Three machine learning algorithms are adopted in the model training phase; Logistic Regression (linear based model with low accuracy) and Random Forest (high-accuracy based model which is an ensemble-based model) and K-Nearest Neighbors (distance-based classifier). Both models acquire the patterns based on the training data to differentiate between malicious and normal traffic. The stage of prediction consists of using the trained models on the test data to categorize the cases of traffic. Lastly, evaluation phase evaluates the performance of the model by comparing it with the measures of accuracy, precision, recall, F1-score, and confusion matrix, as a result of which the most productive detection model is identified.

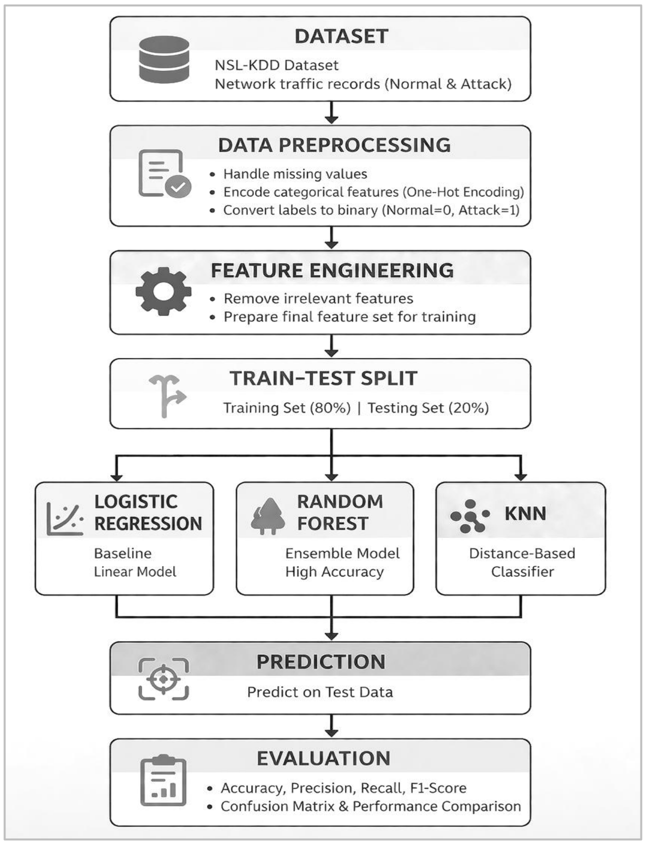

DATASET
NSL-KDD Dataset
Network traffic records (Normal & Attack)
DATA PREPROCESSING
Handle missing values
Encode categorical features (One-Hot Encoding)
Convert labels to binary (Normal=0, Attack=1)
FEATURE ENGINEERING
Remove irrelevant features
Prepare final feature set for training
TRAIN-TEST SPLIT
Training Set (80%) | Testing Set (20%)
LOGISTIC REGRESSION
Baseline
Linear Model
RANDOM FOREST
Ensemble Model
High Accuracy
KNN
Distance-Based
Classifier
PREDICTION
Predict on Test Data
EVALUATION
Accuracy, Precision, Recall, F1-Score
Confusion Matrix & Performance Comparison

Figure 1. System Architecture

Working Flow

Pseudo Code:

```
BEGIN
1. Load NSL-KDD dataset into dataframe DF
2. Assign column names to dataset features
3. Perform data preprocessing
    a. Handle missing values
    b. Encode categorical features (protocol_type, service, flag)
    c. Convert categorical data into numeric format using one-hot encoding
4. Perform label encoding
    a. If label = normal → assign 0
    b. Else → assign 1
    c. Store result in target variable Y
5. Prepare feature matrix X
    a. Remove label columns from dataset
    b. Keep only feature columns
6. Split dataset
    a. Training set = 80%
    b. Testing set = 20%
7. Initialize machine learning models
    a. Logistic Regression
    b. Random Forest
    c. K-Nearest Neighbors
8. Train models using training data
9. Perform predictions on test data
    a. Predict using Logistic Regression
    b. Predict using Random Forest
    c. Predict using KNN
10. Evaluate models
    a. Calculate Accuracy
    b. Calculate Precision
    c. Calculate Recall
    d. Calculate F1-score
    e. Generate Confusion Matrix
11. Compare performance of all models
12. Select best performing model
END
```

IV.METHODOLOGY

The designed methodology has a step-by-step pipeline to develop a machine learning based DDoS attack detection engine. The stages have been well-developed to make sure that network traffic is properly classified and to compare models.

4.1 Dataset Description

The NSL-KDD data set has become a popular benchmark data set used to assess intrusion detection systems with specific emphasis on detection of Distributed Denial-of-Service (DDoS) and other network-based attacks. It is a modified version of the original KDD Cup 1999 data, with the problem of redundant records and data imbalance being solved and giving more accurate and objective evaluation results.

The dataset involved the records of network traffic that are categorized into two major groups namely normal and attack. The records are a representation of every network connection and are characterized using a set of 41 features which signify different properties of network behavior. All these features can be broadly divided into basic features (e.g., the duration, the type of protocol, the service), content-based features (e.g., the number of failed logins), and traffic-based features (e.g., the number of connections to the same host during the time window). The cases of attacks in the dataset are of various types like Denial-of-Service (DoS), Probe, User-to-Root (U2R), and Remote-to-Local (R2L) that can be clustered or transformed into a binary classification question (normal vs. attack) in order to simplify the analysis. In this experiment, the data is pre-processed to transform all the categories of attacks to one type of attack. NSL-KDD dataset is split into both training and testing data, such that the models are tested on the unknown data. Its balanced design and fewer redundancies make it appropriate to create and compare machine learning-based frameworks of DDoS detection.

4.2 Load the Dataset

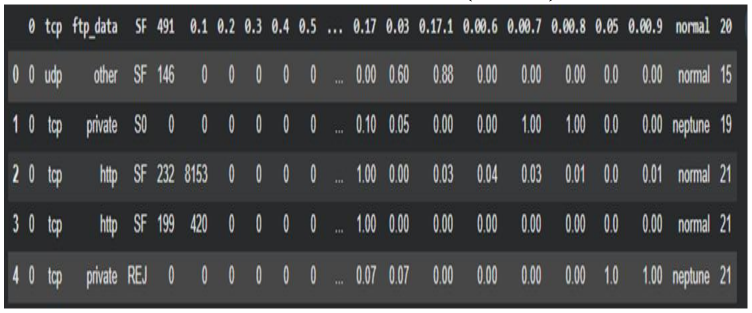
The main dataset of the study is NSL-KDD dataset. It is an enhanced form of the KDD Cup 1999 data set, which deals with the problems of redundancy and imbalance. The data is sourced in the form of a reputable online repository (Kaggle) and imported into the Python environment through Pandas. It comprises of labeled network traffic records that have been classified as normal or attack. KDDTrain+ and KDDTest+ are two primary files that are imported to achieve model training and assessment. The dataset might not have column headers and relevant feature names are added manually. The given step guarantees the appropriate interpretation of the data and it is the basis of the further processing.

4.3 Observing Sample Records

Before any preprocessing or modeling can be done, it is also a critical procedure to observe the data records in the sample to gain insights into the format and properties of the NSL-KDD dataset. With the presentation of several first rows of the dataset (which is provided in the figure) one will be able to draw some important conclusions about the data.

- Feature distribution: The data is a combination of both categorical and numerical attributes (protocol type, service, and flag, source bytes, destination bytes, and other statistics related to traffic) of the features. Analyzing the sample records, one can easier see how these values are allocated and to what extent they are different at various entries.
- Data types: It is apparent that based on the sample, certain characteristics are nominal (e.g., tcp, udp, http, private), and some are quantitative (e.g., number of bytes and rates). The importance of these types is that machine learning models need inputs that are numerical and categorical features need to be coded during preprocessing.
- Class imbalance: The label column (e.g., normal, neptune) denotes the normal and attack traffic. With the examination of several records, it is possible to evaluate the presence of more cases of an attack than a normal case and vice versa in the dataset, which may affect the performance of the model.

Lastly, this step would aid in checking the integrity of the dataset by determining whether there are missing values, inconsistencies, or any other unexpected pattern. In general, sample record observation serves as a guarantee of adequate comprehension of the data before preprocessing and model development.



	0	tcp	ftp_data	SF	491	0.1	0.2	0.3	0.4	0.5	...	0.17	0.03	0.17.1	0.00.6	0.00.7	0.00.8	0.05	0.00.9	normal	20
0	0	udp	other	SF	146	0	0	0	0	0	...	0.00	0.60	0.88	0.00	0.00	0.00	0.0	0.00	normal	15
1	0	tcp	private	S0	0	0	0	0	0	0	...	0.10	0.05	0.00	0.00	1.00	1.00	0.0	0.00	neptune	19
2	0	tcp	http	SF	232	8153	0	0	0	0	...	1.00	0.00	0.03	0.04	0.03	0.01	0.0	0.01	normal	21
3	0	tcp	http	SF	199	420	0	0	0	0	...	1.00	0.00	0.00	0.00	0.00	0.00	0.0	0.00	normal	21
4	0	tcp	private	REJ	0	0	0	0	0	0	...	0.07	0.07	0.00	0.00	0.00	0.00	1.0	1.00	neptune	21

Fig 2: Sample Dataset Records

Figure 2 shows a sample of NSL-KDD dataset which includes network traffic records which have categorical features as well as numeric ones. It contains features of type of protocol, service, flag as well as other traffic related statistics, which characterize network behavior. The last column is the class label that shows either the record is a normal traffic or an attack of a certain type (e.g., neptune).

4.4 Dataset Preparation

During the dataset preparation phase, the NSL-KDD dataset has been sorted and made into an orderly form to be used in processing and model generation. The data set contains one network connection described by each record, which has several features, which describe various facets of network behavior. They are protocol type (e.g., TCP, UDP), service (e.g., HTTP, FTP), and flag (i.e. whether the connection is open or closed), which are categorical. Also, the numerical attributes like source bytes and destination bytes describe the volume of the data sent between the destination and source, hence, giving the information about the volume of traffic and the pattern of communication. These properties are very important in the differentiation of normal and malicious activities. At this step, every feature is planned into a structured format in which input variables (features) and the target variable (label) are sorted out. This step makes sure that the dataset is clean and consistent and is in a state to be preprocessed, feature engineered and trained on a machine learning model.

4.5 Data Preprocessing

The first step is data preprocessing that is essential to be able to guarantee the clean, consistent, and machine learning-suited dataset. During this phase, a number of operations are done to convert the raw network traffic data into a numerical format that is structured.

- Missing Values Handling is done through detection of blank or undefined values in the dataset. These values are either deleted or substituted with the help of relevant methods like imputation to avoid the error of mistakes in the process of model training. Missing values are initially recognized with the functions of isnull () or info() in Python. Missing values are not critical in the NSL-KDD data, although checking is necessary.
- The features that are categorical like protocol type, service and flag are transformed into the numeric form via one-hot encoding. The method generates binary columns of every category so that machine learning

models cannot introduce bias or ordinal correlations to categorical data.

- Target labels are put in form of a binary value to make the classification problem simplistic. Any case that can be considered as normal is classified with a value 0, whereas all the attacks entrepreneurs are put in one inappropriate category with a value 1. This translation makes the system do binary classification, between benign and malicious network traffic. To simplify the problem, it is turned into binary classification:
    - Normal traffic → 0
    - All attack types → 1

Overall, these preprocessing steps enhance data quality and ensure compatibility with machine learning models, leading to improved detection performance.

4.6 Feature Engineering

The process of feature engineering is a significant procedure that leads to the enhancement of the quality of the input data and increase in the performance of machine learning models. It entails changing and picking out the most suitable elements in the dataset.

- Removal of Irrelevant Features: This step involves elimination of the features that are insignificant in the accuracy of predicting DDoS attacks. Noise can be presented by irrelevant or redundant features and can decrease the performance of the model. Such features are detected by means of techniques like correlation analysis and feature importance evaluation. Dropping redundant features can help in dimensional reduction, increase the computational power, and improve the accuracy of the model.
- Conversion of Categorical Features using *pd.get_dummies*() : The data includes categorical variables such as protocol type, service and flag, which cannot be directly consumed by machine learning models. These categorical variables are transformed into numerical format using one-hot encoding by the use of the pd.get_dummies() function. The categories are converted into independent binary columns, making no ordinal assumptions of the relationship between the categories. This change enables models to be able to process and learn categorical data.
- Feature Scaling (if required): Feature scaling is used to scale up or down the numerical features in order to make them of the same scale. This is of particular significance to such algorithms as K-Nearest Neighbors and Logistic Regression that are sensitive to the size of features. The most popular methods are Min-Max scaling and standardization. Scaling is used to make sure that there is no domineering feature of the model as it has a higher value range which results in better model performance and convergence.

4.7 Train-Test Split

The data set after the processing is separated into training and testing data to effectively test the performance of the model. In most cases, the data will be divided into 80 and 20 percent of training and testing respectively. The model is built and learned through the training set and then the testing set is used to evaluate the performance of the model on the unseen data. The separation will guarantee that the model is general and will not be overfitting the training data. Random shuffling is used to ensure that the data is random and then divided.

4.8 Applying Machine Learning Models

- Logistic Regression: Logistic Regression is a supervised machine learning algorithm that is applicable to binary classification issues hence it is applicable in detecting DDoS attacks. It approximates the likelihood of an input to fall into a specific class utilizing a sigmoid-shaped function, which scales the data between 0 and 1. This model operates by establishing the best correlation among features of input and the desired variable.
- Random Forest: Random Forest is an ensemble learning algorithm that constructs numerous decision trees and integrates them to enhance the level of classification. All trees are trained on a random sample of data and features which minimizes overfitting and enhances strength. Random Forest has been very effective in DDoS detection since it is capable of detecting complicated patterns of network traffic. It manages categorical and numerical data effectively and gives scores of features importance.
- K-Nearest Neighbors (KNN): KNN is a simple, non-parametric and instance-based learning algorithm that is applied to classification. It categorizes a point according to the majority class of the closest point in the feature space. The number of neighbors to be used in the classification process is based on value of k. KNN is used in DDoS detection to identify patterns according to the similarity between network traffic records.

4.9 Model Training

Model training is the process of enabling machine learning algorithms to learn patterns from the dataset so they can accurately classify network traffic as normal or malicious. In this stage, the prepared dataset is divided into input features and target labels.

- The input to the model is the feature matrix (X_train), which contains all selected and processed features representing network behavior.
- The output is the target variable (Y_train), which indicates whether the traffic is normal (0) or an attack (1).

All machine learning models (Logistic Regression, Random Forest and K-Nearest Neighbors) are trained using this training data.

The models are used during training to discover relationships among features and labels that define pattern of DDoS attacks. The models modify their inner parameters so as to reduce the errors of prediction and enhance accuracy. As an illustration, the coefficients of each feature are learned using Logistic Regression, random trees are constructed using the Forest, and training instances are stored as distances to each other in KNN.

Proper training makes the models generalize well and helps to predict the models at the testing stage with the models able to predict the unseen network traffic.

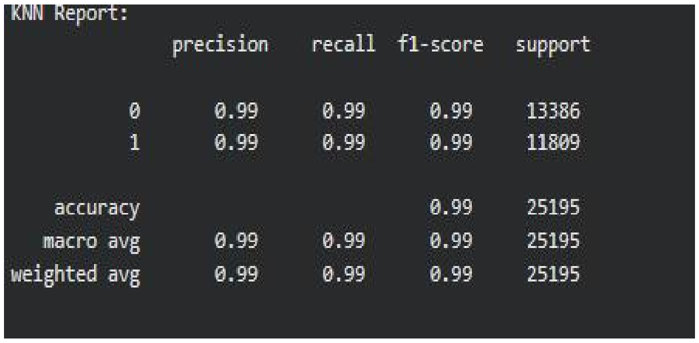
Results and Analysis

Three machine learning models, including the Logistic Regression, Random Forest, and K-Nearest Neighbors (KNN), were used to test the performance of the proposed DDoS detection framework.

The findings indicate that Random Forest and KNN had better performance in terms of accuracy, precision, recall, and the F1-score of about 0.99. On the contrary, Logistic Regression showed relatively poorer performance with an approximate value of 0.89. According to the confusion matrices, the misclassifications of random forest and KNN are minimum whereas logistic regression records a larger number of false positives and false negatives.

All in all, the ensemble and distance-based models are better to use than the linear model in detecting DDoS precisely.

Random Forest:



```
KNN Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     13386
           1       0.99      0.99      0.99     11809

    accuracy                           0.99     25195
   macro avg       0.99      0.99      0.99     25195
weighted avg       0.99      0.99      0.99     25195
```

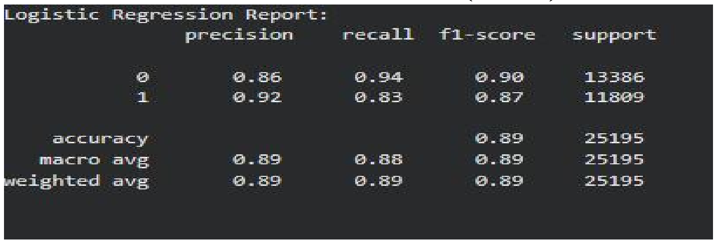
Fig 3: Classification report of Random Forest

The figure 3 presents the classification report of the Random Forest model for DDoS attack detection.

It shows excellent performance with precision, recall, and F1-score values of 0.99 for both normal (0) and attack (1) classes. The overall accuracy of the model is also 0.99, indicating highly reliable predictions.

The macro and weighted averages further confirm consistent performance across all classes.

Logistic Regression:

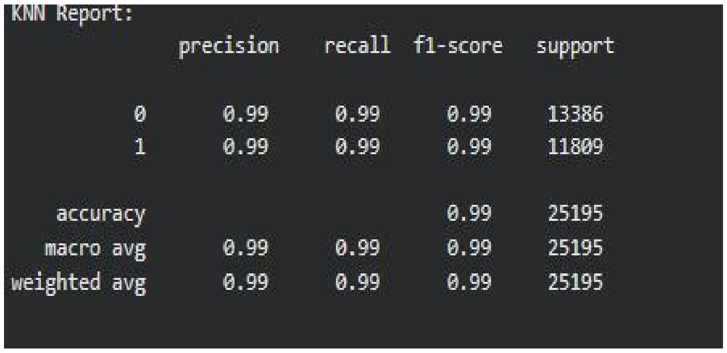


```
Logistic Regression Report:
              precision    recall  f1-score   support

           0       0.86      0.94      0.90     13386
           1       0.92      0.83      0.87     11809

    accuracy                           0.89     25195
   macro avg       0.89      0.88      0.89     25195
weighted avg       0.89      0.89      0.89     25195
```

Fig 4: Classification report of Logistic Regression

The figure presents the report of the classification of the Logistic Regression model used to detect DDoS. The overall accuracy of the model is 0.89 which is not very high as compared to that of other models. It shows a good recall of normal traffic (0.94) but poor recall (0.83) of attack detection implying that there can be some misclassifications of attacks. Precision and F1-scores are comparable, and slightly better results are achieved regarding the attack class.

K-Nearest Neighbors (KNN):



```
KNN Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     13386
           1       0.99      0.99      0.99     11809

    accuracy                           0.99     25195
   macro avg       0.99      0.99      0.99     25195
weighted avg       0.99      0.99      0.99     25195
```

Fig 5: Classification report of KNN

The figure lists the classification report of the KNN model of K-DDoS attack detection. The model gives a high performance with the precision, recall, and F1-score of 0.99 when the model is tested on both normal (0) and attack (1) classes.

Confusion Matrices

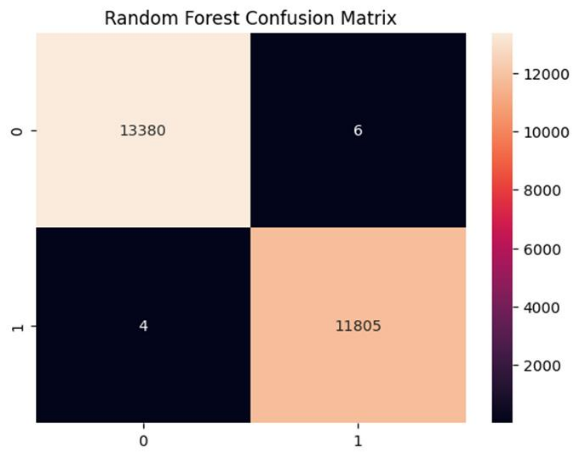


Fig 6: Confusion Matrix of Random Forest

The figure 6 demonstrates confusion matrix of the DDoS attack detection model using the Random Forest. It demonstrates that there are many correct predictions with 13,380 true negatives and 11,805 true positives. The model has minimal number of misclassifications as it has 6 false positives and 4 false negatives. This indicates that it is very capable of detection with few errors.

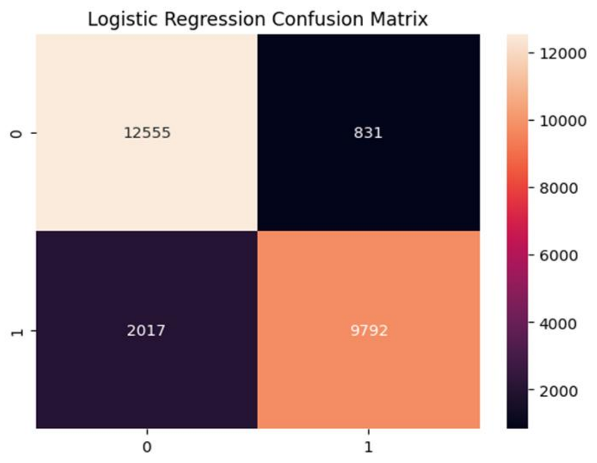


Fig 7: Confusion Matrix of Logistic Regression

The table 7 shows the confusion matrix of the Logistic Regression model used to detect DDoS. The results it gives are 12,555 true negatives and 9,792 true positives, which means that it classifies both normal and attack traffic correctly. But the model gives more amount of misclassifications with 831 false positives and 2,017 false negatives. This implies that the model is not as effective as identifying the instances of attacks.

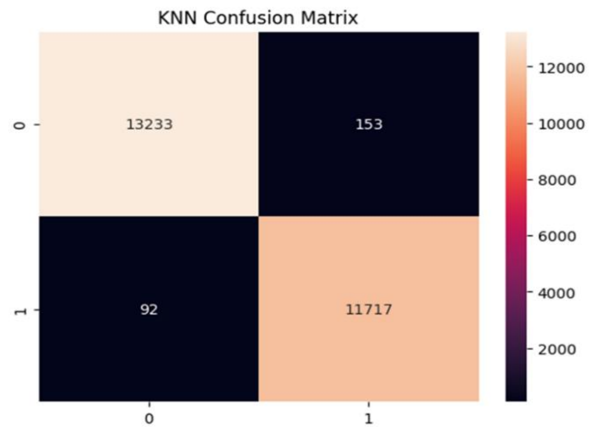


Fig 8: Confusion Matrix of Logistic Regression

The confusion matrix of the KNN model in detecting DDoS is presented in the figure 8. It makes the right call on 13,233 normal (true negative) and 11,717 attack (true positive) cases. The model has few errors in that 153 false positives and 92 false negatives were obtained.

Comparative Analysis of Models

The three machine learning models of Logistic Regression, Random Forest, and K-Nearest Neighbors (KNN) to detect DDoS attacks were compared and analyzed to determine their performance. Key performance metrics such as accuracy, precision, recall and F1-score are used to analyze it. Random Forest and KNN have better performance as they show almost 0.99 in all assessment measures. These models are successful in capturing complex patterns in network traffic hence high rate of detection and low rate of false positives and false negatives. The Logistic Regression, in turn, performs relatively lower with the accuracy approximately 0.89. Although it works excellently on normal traffic detection, it does not work well on identifying the instances of attacks because it is linear. On balance, the findings show that ensemble (Random forest) and distance-based (KNN) models perform better in comparison to linear model. It is crucial to note that, in real-life situations, advanced algorithms should be chosen to detect DDoS attacks accurately and reliably.

Table 2: Performance evaluation of Models

Model	Accuracy	Precision	Recall	F1 Score
Random Forest	0.99	0.99	0.99	0.99
Logistic Regression	0.89	0.89	0.88	0.89
KNN	0.99	0.99	0.99	0.99

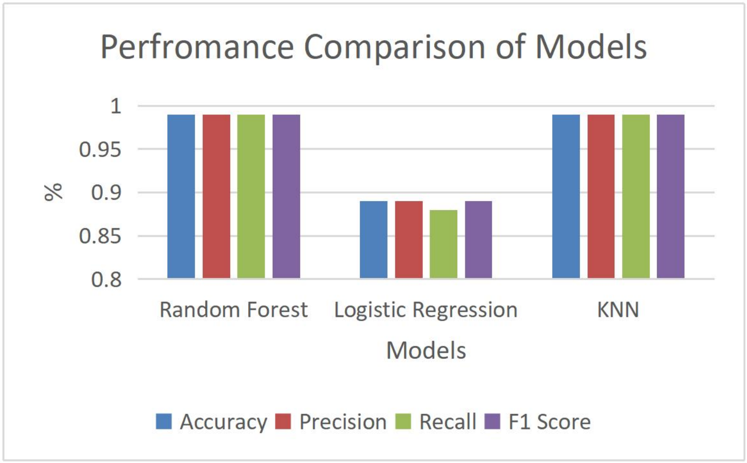


Fig 9: Performance Comparison of Models

The figure 9 illustrates the performance comparison of three machine learning models: Random Forest, Logistic Regression, and KNN.

V.CONCLUSION

The current paper is an overview of the design and implementation of an intelligent machine learning-based attack detector to detect Distributed Denial-of-Service (DDoS) attacks using network traffic data. The proposed system can successfully process data, exert the most crucial features, and use several machine learning models, such as Logistic Regression, the Random Forest, and the K-Nearest Neighbors to categorize network traffic into normal and malicious. The results of the experiments have shown that Random Forest and KNN models are modeled with high accuracy, precision, recall and F1-scores than the Logistic Regression model. The framework is able to detect complex attack patterns and the false positive rate is also low thus it can be used in real time. Also, the comparative analysis demonstrates the significance of the choice of proper algorithms in the case of intrusion detection systems. In general, the suggested solution is a scalable, efficient and reliable tool to ensure network security. It aids in enhancing defense against changing DDoS attacks in contemporary cloud and IoT settings.

VI.REFERENCES

[1] Saad, R. M. A., Anbar, M., Manickam, S., & Alomari, E. (2016). An Intelligent ICMPv6 DDoS Flooding-Attack Detection Framework (v6IIDS) using Back-Propagation Neural Network. IETE Technical Review, 33(3), 244–255. https://doi.org/10.1080/02564602.2015.1098576

[2] Suresh, M., Anitha, R. (2011). Evaluating Machine Learning Algorithms for Detecting DDoS Attacks. In: Wyld, D.C., Wozniak, M., Chaki, N., Meghanathan, N., Nagamalai, D. (eds) Advances in Network Security and Applications. CNSA 2011. Communications in Computer and Information Science, vol 196. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-22540-6_42

[3] A Kulkarni, Y. Pino and T. Mohsenin, "Adaptive real-time Trojan detection framework through machine learning," 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), McLean, VA, USA, 2016, pp. 120-123, https://doi.org/10.1109/HST.2016.7495568

[4] Yu Chen, Kai Hwang, Collaborative detection and filtering of shrew DDoS attacks using spectral analysis, Journal of Parallel and Distributed Computing, Volume 66, Issue 9, 2006, Pages 1137-1151, ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2006.04.007

[5] Dimitris Gavrilis, Evangelos Dermatas, Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features, Computer Networks, Volume 48, Issue 2, 2005,Pages 235-245, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2004.08.014

[6] Li Ke & Zhou Wanlei & Li, Ping & Hai, Jing & Liu, Jianwen. (2009). Distinguishing DDoS Attacks from Flash Crowds Using Probability Metrics. 9-17. 10.1109/NSS.2009.35. https://doi.org/10.1109/NSS.2009.35

[7] Jelena Mirkovic and Peter Reiher. 2004. A taxonomy of DDoS attack and DDoS defense mechanisms. SIGCOMM Comput. Commun. Rev. 34, 2 (April 2004), 39–53. https://doi.org/10.1145/997150.997156

[8] David Moore, Colleen Shannon, Douglas J. Brown, Geoffrey M. Voelker, and Stefan Savage. 2006. Inferring Internet denial-of-service activity. ACM Trans. Comput. Syst. 24, 2 (May 2006), 115–139. https://doi.org/10.1145/1132026.1132027

[9] Vasilios A. Siris and Fotini Papagalou. 2006. Application of anomaly detection algorithms for detecting SYN flooding attacks. Comput. Commun. 29, 9 (May, 2006), 1433–1442. https://doi.org/10.1016/j.comcom.2005.09.008

[10] Haining Wang, Danlu Zhang and Kang G. Shin, "Detecting SYN flooding attacks," Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, New York, NY, USA, 2002, pp. 1530-1539, https://doi.org/10.1109/INFCOM.2002.1019404

[11] S. Yu, W. Zhou, R. Doss and W. Jia, "Traceback of DDoS Attacks Using Entropy Variations," in IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 3, pp. 412-425, March 2011, https://doi.org/10.1109/TPDS.2010.97