

**DISTRIBUTED DENIAL-OF-SERVICE ATTACK DETECTION USING  
ENSEMBLED BASED APPROACH**

**A PROJECT REPORT**

**Submitted by**

**PRABHU S  
(Reg. No: 23MCR068)**

**SANTHIYA S  
(Reg. No: 23MCR083)**

**YUVAPRIYAN K.M  
(Reg. No: 23MCR0126)**

*in partial fulfilment of the requirements  
for the award of the degree  
of*

**MASTER OF COMPUTER APPLICATIONS**

**DEPARTMENT OF COMPUTER APPLICATIONS**



**KONGU ENGINEERING COLLEGE**

**(Autonomous)**

**PERUNDURAI, ERODE – 638 060**

**MAY 2024**

**DEPARTMENT OF COMPUTER APPLICATIONS  
KONGU ENGINEERING COLLEGE**

**(Autonomous)**

**PERUNDURAI ERODE-638 060**

**MAY 2024**

**BONAFIDE CERTIFICATE**

This is to certify that the project report entitled “**DDOS ATTACK DETECTION USING ENSEMBLED APPROACH**” is the bonafied record of project work done by **PRABHU S(23MCR068),SANTHIYA S(23MCR083)** and **YUVAPRIYAN K.M(23MCR0126)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications of Anna University, Chennai during the year 2023-2024.

**SUPERVISOR**

**HEAD OF THE DEPARTMENT**

**(Signature with seal)**

**Date:**

Submitted for the end semester viva voce examination held on \_\_\_\_\_

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **DECLARATION**

We affirm that the project report entitled “**DDOS ATTACK DETECTION USING ENSEMBLED BASED APPROACH**” being submitted in partial fulfilment of the requirements for the award of Master of Computer Applications is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**Date:**

**PRABHU S**

**(Reg. No: 23MCR068)**

**SANTHIYA S**

**(Reg.No: 23MCR083)**

**YUVAPRIYAN K.M**

**(Reg.No:23MCR0126)**

I certify that the declaration made by the above candidate is true to the best of my knowledge.

**Date:**

**Name and signature of the supervisor with seal**

## ACKNOWLEDGMENT

We respect and thank our correspondent **Thiru.A.K. ILANGO BCom., MBA., LLB.**, and our Principal **Dr.V.BALUSAMY B.E(Hons)., M.Tech., Ph.D.** Kongu Engineering College, Perundurai for providing me with the facilities offered.

We convey our gratitude and a heartfelt thanks to our Head of the Department **Dr.A.TAMILARASI MSc., MPhil., PhD.**, Department of Computer Applications, Kongu Engineering College, for her perfect guidance and support that made this work to be completed successfully.

We also wish to express our gratitude and sincere thanks to our project coordinators **Dr.T.M.SARAVANAN MCA., MPhil., PhD.**, and **Dr.M.JAGADEESAN MCA., M.Phil., Ph.D.**, Associate Professor(s), Department of Computer Applications, who have motivated us in all aspects for completing the project in scheduled time.

We would like to express gratitude and sincere thanks to our project guide **Dr.T.M.SARAVANAN MCA., MPhil., PhD.**, Associate Professor, Department of Computer Applications, Kongu Engineering College for giving his valuable guidance and suggestions which helped us in the successful completion of the project.

We owe a great deal of gratitude to our parents for helping overwhelm in all proceedings. how our heart and head with heartfelt thanks to all those who thought our their warm services to succeed and achieve the work.

## ABSTRACT

The Distributed Denial of Service (DDOS) attack is a deliberate attempt to make an application or website unavailable to users such as flooding it with network traffic. The multiple computer systems are attack a target place and cause a denial of service for the users of the target resource. This attack causes harmful server outages and excessive stress on IT professionals for bringing the resources back to online. To detect the attack, we have to analysis three important phases available. They are optimal feature selection, classification and feature extraction. By using optimization algorithms, to optimally select the features of obtained feature sets.

In our work we propose, Random Forest a popular machine learning algorithm for classification and regression tasks. It's an ensemble learning method that operates by constructing a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

## TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	<b>ABSTRACT</b>	V
	<b>ACKNOWLEDGMENT</b>	IV
	<b>LIST OF FIGURES</b>	VIII
	<b>LIST OF ABBREVIATIONS</b>	IX
<b>1</b>	<b>INTRODUCTION</b>	
	1.1 OVERVIEW OF THE PROJECT	1
	1.2 MACHINE LEARNING FOR DDOS DETECTION	3
<b>2</b>	<b>LITERATURE REVIEW</b>	
	2.1 REVIEW ON DDOS ATTACK	4
<b>3</b>	<b>SYSTEM REQUIREMENTS</b>	
	3.1 HARDWARE REQUIREMENTS	11
	3.2 SOFTWARE REQUIREMENTS	11
	3.3 COIAB PYTHON	11
	3.4 APPLICATIONS OF GOOGLE	13
	3.5 KAGGLE	13
	3.6 FEATURES OF PYTHON	14
	3.7 READABILITY	14

<b>4</b>	<b>PROPOSED METHODOLOGY</b>	
	4.1 METHODOLOGY	16
	4.1.1 DATASET	16
	4.1.2 DATASET DESCRIPTION	17
	4.2 SUPPORT VECTOR MACHINE	18
	4.3 KNN ALGORITHM	19
	4.4 RANDOM FOREST CLASSIFIER	21
	4.5 ENSEMBLED METHOD	22
	4.5.1 BAGGING METHOD	23
	4.6 PERFORMANCE METRICS	24
	4.6.1 CONFUSION MATRIX	24
	4.6.2 ACCURACY	25
	4.6.3 PRECISION	25
	4.6.4 RECALL	25
	4.6.5 F1 SCORE	25
	4.6.6 SPECIFICITY	
<b>5</b>	<b>RESULT AND DISCUSSION</b>	
	5.1 PERFORMANCE MEASURES	26
<b>6</b>	<b>CONCLUSION AND FUTURE ENHANCEMENT</b>	
	6.1 CONCLUSION	31
	6.2 FUTURE ENHANCEMENT	31
	<b>APPENDICES</b>	
	A.SAMPLE CODE	32
	B.SCREENSHOT	37
	<b>REFERENCES</b>	40

**LIST OF FIGURES**

<b>FIGURE</b>	<b>NAME</b>	<b>PAGE NO</b>
4.1	Support Vector Machine	18
4.2	Pseudo code for SVM algorithm	19
4.3	Pseudo code for KNN algorithm	20
4.4	Pseudo code for Random Forest Regression	23
B.1	Importing Packages	37
B.2	Reading dataset	37
B.3	Dataset with attributes	38
B.4	Splitting training and testing data	38
B.5	K-Nearest Neighbour Model	39
B.6	Support Vector Machine model	39



## LIST OF ABBREVIATIONS

DDOS	Distributed Denial of Service
TCP	Transmission Control Protocol
SVM	Support Vector Machine
KNN	K-Nearest Neighbour
UDP	User Datagram Protocol
RF	Random Forest
MLP	Multi-Layer Perceptron
SDN	Software Defined Network
ML	Machine Learning
RL	Reinforcement Learning

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 OVERVIEW OF THE PROJECT**

##### **DDOS ATTACK MECHANISM:**

A Distributed Denial of Service (DDOS) attacks [1] to prevent legitimate users from accessing an online service or applications by suspending the hosting servers. To generate the attack, the attackers use numerous compromised or controlled sources to generate massive amounts of packets or requests. These requests cause the target system to become overburdened, causing it to operate poorly and become inaccessible to legitimate users. Based on TCP/UDP protocols, DDOS attacks are divided into reflection-based attacks and exploit-based attacks

A DDOS attack sends different requests (with IP spoofing) to the target web assets to exceed the site's ability to handle various requests, at a given time, and make the sustainable to operate effectively and efficiently – even for the legitimate users of the network. Typically, the target of various DDOS attacks are web applications and business websites; and the attacker may have different goals.

Tree- based ensemble models show better performance than linear models. This study shows excellent results, yet has several limitations. First, the approach is tested on a single dataset and Attackers send packets to reflector servers with the target victim's IP address as the source IP address to overwhelm the victim with response packets. The Transmission Control Protocol (TCP), the User Datagram Protocol (UDP), or a combination can be used in these attacks. SSDP and MSSQL are TCP-based attacks, while NTP TFTP.

SVM, however, is selected for its established superior performance in binary classification tasks and its capacity to manage high-dimensional data efficiently. SVM are highly effective in accurately distinguishing between different classes in convoluted feature spaces, which makes them well suited for the complex nature of network traffic. Although decision trees and other machine learning models have shown effectiveness in specific applications, SVM's strong performance validates their choice for this particular task, especially in situations with non-linear bounds. In addition, the integration of GWO and SVM offers a unique and unexplored method for detecting DDoS attacks in SDNs. This highlights the originality and potential impact of this research.

#### **ADVANTAGE OF SUPPORT VECTOR MACHINE:**

- Handling large dataset: Support Vector Machines (SVM) efficiently handle large datasets due to their computational efficiency, memory efficiency, scalability, and robustness. By using a subset of training points, SVM reduces computational burden and memory requirements, making it suitable for high-dimensional datasets.
- Handling missing data: Support Vector Machines (SVM) can handle missing data by imputing values based on available features or using specialized algorithms to predict missing values. SVM's robustness to noise and focus on support vectors enable it to effectively manage missing or imperfect data while maintaining predictive accuracy

#### **DISADVANTAGE OF SUPPORT VECTOR MACHINE:**

- Computationally expensive: Particularly with large datasets or high-dimensional feature spaces. This complexity increases the computational cost, especially when working with big data or complex models.
- Training Time: Support Vector Machines (SVM) can be time-consuming compared to simpler models like logistic regression or decision trees.

## **RANDOM FOREST:**

**Data Collection:** The first step is to collect data related to network traffic, system logs, user behaviour, and any other relevant sources. This data should include both normal behaviour and instances of DDoS attacks if available.

**Model Training:** Train a Random Forest classifier using the labeled dataset. Random Forests are effective for this task because they can handle large datasets with high dimensionality and are robust against overfitting.

**Model Evaluation:** Evaluate the trained model using validation data to ensure it generalizes well to unseen instances. Metrics like accuracy, precision, recall, and F1-score can be used to assess the performance of the model.

## **1.2 MACHINE LEARNING FOR DDoS DETECTION:**

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on uses of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning is a growing technology which enables computers to learn information from the past data.

Machine Learning Algorithms are implemented in the datasets to predict the results and analysis. In unsupervised Learning, a machine finds the patterns in unknown objects by grouping similar objects together. For examples of such learning includes Recommender systems and Market Based Analysis.

DDoS can be integrated into the detection system to enhance the performance of SVM-based models. DDoS, as a metaheuristic optimization technique, can optimize the parameters of the SVM algorithm dynamically, ensuring that the model adapts effectively to changes in network traffic patterns. By continuously adjusting the SVM parameters based on real-time feedback from the network, DDoS-SVM hybrid models can achieve higher accuracy and robustness in detecting DDoS attacks.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 REVIEW ON DDOX ATTACK:

Bushra Alhijawi et al [1] included two main categories as presented a new classification of DOS attacks and mitigation techniques which is related to SDN. The classification that included mitigation methods to cope with DoS attacks on SDN and SDN-based solutions for mitigating DOS attacks. The first category indicates detection, mitigation, and prevention methods for DOS attacks against the SDN. The second category indicates the solutions that utilized SDN characteristics to handle DOS attacks in other networks such as cloud computing, legacy networks, and data centres. This paper analyzed the methods to cope with DoS attacks in SDN and classified them into six types, they are Table-Entry-based, Scheduling-based, Architectural-based, Flow statistics-based, Machine Learning-based, and Hybrid solutions.

In future research, there are several ways to identify this detection as an establishment in a good progress in Machine Learning-based detection for DOS/DDoS, then more contributions can be developed using KNN, clustering methods and graphs. Second, a promising future direction would be to develop additional SDN-based solutions for DoS/DDoS mitigation against other environments such as IoT, cloud computing paradigms, and ISP networks. The development of optimization-based method for detecting DoS/DDoS against

Raj Kumar Batchu et al [2] represented fast and efficient detection way for identifying real-world DDOS attacks in the networking. They had used the CICDDoS- 2019 traffic dataset for this detection that are closer to real-time traffic activities. The traffic activity contains noisy data, class imbalances, and irrelevant attributes. They had proposed a method that addresses all type of issues that are begin with class imbalance problem which are

dealing to avoid bias towards majority class instances. Furthermore research, the memory optimizations are directed to enhance the processing speed of massive traffic and conserve multiple resources. The Hybrid Attribute selection technique with parameter tunings are implemented to extracting the essential attributes. These attributes are retrieved to the ELM and obtained the best parameters that discriminate DDOS attacks from regular traffic. These results have shown good generalization performance with six features in less training time. As a result, the suggested model can be utilized as a detection system to discover DDOS attack patterns in any network. This work is focused on binary classification. In future, this work can be extended for multi-attack classification, identifying IOT network attacks, and Software-defined networks attack.

R.Amrish et al [4] proposed the machine learning algorithm for detecting DDOS attacks. This research study is evaluated using four distinct algorithm, they are ANN, KNN, Random Forest, and Decision Tree that are used to determine the classification model performs best in detecting malicious IP addresses. It is noted that the ANN model performs other classifier models with an accuracy of 99.95%. Furthermore results, these works can be expanded to develop and deploy in a block chain system to store and blacklist the IP Address of the node, if the traffic is classified as malicious by the machine learning algorithm. The use of the block chain provides an extra layer of security measure so that the data cannot be tampered.

James Halladay et al [5] analyzed the performance of eight machine learning algorithms and one deep learning model by using the dataset of CICDDoS2019. They had used these datasets for different two scenarios. Scenario A classified as benign versus DDOS traffic and Scenario B categorized as DDOS attack types. In these both scenarios, the models were initially trained with the control dataset consisting of 70 features. These models were trained on the proposed subset of time based features.

The top five models had accuracies of over 99% on the control features and over 98% on the time-based features in Scenario A. These results are as accurate as top DDOS classifiers seen in existing literature. Overall, the nine models had a median accuracy decrease of a mere 1.62% and a substantial median training time reduction of 36.28% when using only the proposed time-based features. Scenario B classified specific DDOS traffic as

MSSQL, SSDP, SYN Flood, PORTMAP, DNS, LDAP, NETBIOS, SNMP, TFTP, NTP, UDP Flood, or UDP-Lag. The top performing model, XGB, had an accuracy of 74.08% on the control features and 69.05% on the time-based features. The nine classifiers had a median decrease of 7.43% in accuracy with a median training time reduction of 25.29% using only time-based features. Our top classifier's performance metrics matched or exceeded the accuracies of previously discussed multiclass works, especially considering the uniquely large number of attack types in our experiments. Overall, XGB, RF, LD, LGBM, and KNN were the top-performing models.

Furqan Rustam et al [6] exploited the DOS/DDOS attacks and the number of attacks have step increased over the past few years. Due to advanced and sophisticated attack detection approaches, the network security remains challenging problem in day to day activities. This research study proposed a machine learning-based framework to detecting DDOS/DOS attacks at the application layer by using multi-features. For multi- features, the best features from PCA and SVD are extracted to obtain superior performance. Extensive experiments are carried out using LR, RF, GB, ETC, as well as, CNN, LSTM, CNN-LSTM, and RNN models using different dataset sizes and different features. Experimental results indicate that PCA features tend to show better performance as compared to SVD features. Despite the results being better with using all features, multi-features help optimize the models' performance.

The RF model outperforms all other models by obtaining 100% accuracy when used with multi-features. On average, the requires further experiments regarding attacks on other layers. Second, the impact of dataset size is not investigated. Increasing the size might produce better results for deep learning models. Third, the influence of feature set size is also not analyzed, which we intend to perform in future work.

Francesco Musumeci et al [7] evaluated ML assisted DDOS attack detection frameworks for application in SDN environment considering Standalone and Correlated DAD architectures. Leveraging the potential of data-plane programmability enabled by P4 language, They had evaluated the detection latency is reduced when performing features extraction at P4 switches. To do so, they had compared different ML classifiers in terms of accuracy and computational time, and deployed the algorithms in a real-time scenario in

which the P4 switch provides different types of data to the ML classifiers, namely, packet mirroring, header mirroring, and P4-metadata extraction. Numerical results show that attack detection can be performed with classification accuracy, precision, recall and F1- score higher than 98% in most cases, and with drastic time reduction in case of P4 is used for features extraction. As a future the work, they had planned to investigate attack-type identification by developing multiclass ML classifiers, and implementing attack detection exploiting ML algorithms which leverage historical data,such as Recurrent Neural Networks.

Ashfaq Ahmad Najar et al [8] had performed several classification models are used in our paper for detecting whether a packet is normal or an attack packet. We used the Random Forest model for the binary classification which showed an accuracy of 99.13% on train data 99.13% accuracy on validation data and 97% on test data with an f1 score of 0.9661% which is very effective. We used another model which is a Multi -Layer Perceptron model to detect which type of attack packet it is, the model showed an accuracy of 97.96% on train data and 98.53% on validation data and 74% on the test data. These models can be used to detect malicious activity on the network. Having shown great accuracy these models will be very beneficial for employing a defence mechanism on a network for detection of malicious activity.

Ismail et al [9] proposed a complete systematic approach for detection of this DDOS attack. They had applied the supervised machine learning techniques. This model generated the classification results from the supervised algorithm. They had used Random Forest and XG Boost classification algorithms. In their first classification, they had observed that both Random Forest Precision and Recall are getting approximately 89% accurate.

For their second classification, they had noted that both XG Boost Precision (PR) and Recall (RE) are approximately 90% accurate. By comparing these classification result proposal to existing research works. Looking into the future works, for functional applications, it is important to provide a more user-friendly, faster alternative to deep learning calculations, and produce better results with a shorter burning time. It is important to work on unsupervised learning toward supervised learning for unlabelled and labelled datasets. Moreover, we will investigate how non-supervised learning algorithms will affect the DDOS attack detection.



Kishore Babu Dasari et al [11] presented a comparison of the performance of six machine learning classification algorithms on eleven individual different DDOS attacks datasets. Unfortunately, the most common effective DDOS attack detection method for all DDOS attacks has yet to be identified. Some DDOS attacks have common effective methods and some attacks have different effective methods. Decision tree and random forest gave poorer results than others. Logistic regression, Ada Boost, KNN, and NB show good results. In this paper, classification algorithms applied to different individual DDOS attack datasets get the best scores in all metrics with Google COLAB. TPU processor which is a powerful hardware accelerator and 12GB RAM. This configuration is more expensive. All datasets are big data size. The idea of next research would be to use feature selection to reduce data [22] and detect DDOS attacks using low- cost hardware.

Mona Al dua ili j et al [12] the DDOS attack detection is a common problem in a distributed environment. This type of attack causes the unavailability of cloud service, which makes it essential to detect this attack. A machine learning model can be used to identify this type of attack. The research objective of this work is to detect a DDOS attack, with improved performance.

This experiment was performed on the CICIDS 2017 and CICDDOS 2019 datasets. Different files related to DDOS attack were included in experiments, from both datasets. They had selected the most relevant features, by applying the MI and the RFFI methods. The selected features are fed to machine learning algorithms including (RF, GB, WVE, KNN, LR). The overall prediction accuracy of RF with 16 features, is 0.99993, and with 19 features, is 0.999977, which is better, Symmetry 2022, 14, 1095 13 of 15 compared to other methods. They had concluded that RF, GB, WVE, KNN, and LR are achieving good results, by using MI and RFFI as feature selection techniques.

Tariq Emad Ali et al [13] represented difference between DDOS attacks with various rates and patterns and normal traffic. Over the years, many effective ML/DL methods for DDOS attack detection have been suggested by different researchers. Sadly, however, the applicability of these techniques is severely constrained due to attackers constantly changing their attack tactics. To find involving the SLR protocol are evaluated.

The literature has been summarized in Section 4 in accordance with the suggested taxonomy for DDOS attack detection using ML/DL techniques, with each study's respective advantages and disadvantages listed. The accuracy rate reported in much of the literature is over 99%. Because the majority of these studies assessed their models using offline data analysis for evaluation and comparison, certain metrics for performance may vary in a real-world or production settings. In particular, they had noted that the existing papers have generally not employed the same DS or assessment techniques, making comparisons between their results difficult.

LiX in long et al [14] improved detection of DDOS attacks in terms of their methodology, which is among the most essential and complicated concerns in information security. Dealing with these systems calls for highly developed computer programs that can use time sequences and other generally advanced intelligence qualities to conquer complex challenges. In this spirit, an innovative hybrid model of the HTM system is provided in this study. The system's architecture is modified by the addition of an LSTM cell so that the system can encode time sequences that comprise inflow data. Experiments showed that the proposed methodology successfully resolved the issue of accurately detecting DDOS attacks.

SVM learning was used for classification. Information of the switch from the flow table are obtained. DDoS attacks are observed where the entropy of the target is smaller and the entropy of the origin is exceeding the usual threshold of the traffic. To mitigate a DDoS threat, the DDoS attack defence dumps all table records. The proposed technique identifies attacks quickly and has a good detection rate and low FPR

Nada et al. [23] proposed a hybrid method which assigns a node trust value after detecting if any nodes are involved in DDoS attacks. Offending nodes are banned for certain period of time. But the flexibility of algorithm gives them an opportunity to reintegrate. But, if malevolent conduct is repeated, the nodes forever. The proposed approach is more robust as it takes into consideration the dynamics of current network traffic and can be adapted in real time as the traffic complexity of a network varies. The major drawback is the choice of threshold value, which is selected experimentally in this work. The threshold value may be calculated using machine learning.

Meenakshi Mittal et al [15] discriminated the DDOS attacks with different rates and pattern from benign traffic is a very challenging issue. Many efficient DL approaches have been proposed by fellow researchers for DDOS attack detection over the years. But unfortunately, the scope of these methods is very limited as the attackers are continuously updating their attack strategies and skills very rapidly to launch unknown or zero-day DDOS attacks with unique traffic patterns every time.

In this paper, we have used the SLR protocol to review the DDOS attacks detection system based on DL approaches and results of the SLR protocol are analyzed. Deepak Kumar et al [16] proposed that a Deep Learning model is used to classify DDOS attacks on the network, which is likely to be more effective than the machine learning model.

In this, research work, the LSTM model has been used for the classification of benign and threats on the CICDDOS 2019 dataset, the LSTM model, which is employed as a deep learning model, has approximately 98.6% for DDOS attacks classification which is large as compared to KNN and ANN model.

DDOS attack detection using ensemble-based approaches in machine learning reveals a growing interest in leveraging the combined power of multiple models for enhanced detection accuracy and robustness. Researchers have explored various ensemble techniques such as bagging, boosting, and stacking to effectively detect and mitigate DDOS attacks in network environments.

Studies have highlighted the advantages of ensemble methods in handling the complexity and variability of DDOS attack patterns by combining the strengths of different base learners. Ensemble models offer improved generalization, resilience to noisy data, and better performance in detecting both known and unknown attack types. Key components of ensemble-based DDOS detection approaches include feature selection, model diversity, ensemble aggregation methods, and evaluation metrics.

## **CHAPTER 3**

### **SYSTEM REQUIREMENTS**

#### **3.1 HARDWARE REQUIREMENTS**

- Memory: 4 GB
- CPU : AMD PRO
- File size : 2 GB
- OS : Windows 10

#### **3.2 SOFTWARE REQUIREMENTS**

- Operating System : Windows 10
- Software Tool : Google COLAB
- Coding Language : Python
- Dataset Library : Kaggle

#### **3.3 Colab Python**

These are the most important and key features of python as programming language and the python can be applied in the different types of environments. Colab stands for the colaboratory

which is a product from the Google Research. It is a free of charge to use the resources including the GPU'S. The resources in Colab are prioritized for the interactive use cases.

By using Colab we can share the Jupyter notebooks with others without downloading or installing or running. Anything colab notebooks are stored in Google drive or can be loaded from the Github. The uploaded file in the colab are removed When the Session is restarted because the Google colab does not provide a provide a persistent storage facility.

Python is a highly used general-purpose programming language with a variety of applications. It has high-level data structures, dynamic type, dynamic binding, and many other capabilities that make it suitable for both complex programme development and scripting or "glue code " that connects components. It can also be extended to make system calls to nearly all operating systems and run C or C++ code. Python is a universal language used in a variety of applications due to its ubiquity and ability to operate on practically every system architecture. As python language is easy to understand it is used to build learning models.

Python can be applied in the realworld applications such as in the field of Web development, Data science, Artificial Intelligence, Machine Learning, Enterprise Applications, Game development and also as well as the software development Python has the capability to be open in different software tools such as the Colab, Jupyter notebook.

It consists of the different data types namely Python numeric data type is used to store the numeric characters and python string data type is used to store the sequence of characters In python the tuple consists of an the Ordered set of values. Python is also utilised in other applications. Robotics, web scraping, scripting, artificial intelligence, data analysis, machine learning, face detection, colour detection, 3D CAD applications, console-based applications, audio-based applications, video-based applications, enterprise applications, and applications for images are some of the other applications for which Python is used.

### **3.4 Applications of Google Colab:**

Google Colab is widely used in different segments like;

- Business and Financial Analytics
- Marketing and Trading, Share market
- Education and Research
- Financial trend analysis
- Genetic engineering
- Space exploration
- Text Mining
- Machine Learning
- Predictive Analytics
- To meet the future demands like gold, oil, petroleum product.

### **3.5 Kaggle**

Kaggle is a part of Google LLC is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment work with other data scientists and machine learning engineers and enter competitions to solve data science challenges. Kaggle is a cloud-based workbench for data science and Artificial Intelligence education. Several academic papers have been published on the basis of findings made in Kaggle competitions.

### **3.6 Features of python:**

- It is very simple to use.
- Python is high level language.
- Python provides better structure and support.
- It provides a perfect interface to all commercial databases.

The aim of this platform is to help professionals and learners reach their goals in their data science journey with the powerful tools and resources it provides.

### **3.7 Readability:**

Python's syntax emphasizes readability, making it easy for developers to write and understand code. This feature is particularly beneficial for collaboration and maintenance.

#### **Application Example:**

Python is commonly used in web development frameworks like Django and Flask, where readability is crucial for efficient development and maintenance of complex web applications.

#### **Flexibility:**

Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming. This flexibility allows developers to choose the most suitable approach for their projects.

#### **Application Example:**

Python is widely used in scientific computing and data analysis libraries like NumPy and pandas, where developers can leverage functional programming concepts to manipulate large datasets efficiently.

**Extensive Standard Library:**

Python comes with a rich standard library that provides modules and packages for performing various tasks without the need for third-party dependencies.

**Application Example:**

Python's standard library includes modules for working with file I/O, networking, and regular expressions, making it well-suited for developing network applications like web servers and clients.

**Portability:**

Python code is highly portable and can run on different operating systems without modification. This portability simplifies the deployment of Python applications across diverse environments.

**Application Example:**

Python is used in cross-platform desktop application development frameworks like Tkinter and PyQt, enabling developers to create applications that run seamlessly on Windows, macOS, and Linux.

**Large Ecosystem:**

Python has a vast ecosystem of third-party libraries and frameworks that extend its capabilities for various domains, such as web development, data science, machine learning, and automation.

**Access Datasets:**

Kaggle hosts a wide range of datasets across various domains, including cybersecurity. You can search for datasets related to network traffic or cybersecurity threats, which may include features relevant to DDoS attack detection, such as network flow data, packet statistics, and anomaly indicators.



## CHAPTER 4

### PROPOSED METHODOLOGY

#### 4.1 METHODOLOGY

In this study, we've introduced a novel approach for detecting DDOS (Distributed Denial of Service) attacks. Our method leverages ensemble-based algorithms in conjunction with machine learning techniques, specifically Random Forest Regression, K-Nearest Neighbor (K- NN), and Support Vector Machine (SVM). To train and evaluate our models, we utilized the CIC2019 dataset sourced from Kaggle, a widely recognized platform for datasets. To enhance the efficiency of our model, we employed the improved Grey Wolf Optimization algorithm to select the most pertinent features from the dataset. This optimization process aids in streamlining the feature selection phase, thereby improving the overall performance of our DDOS attack detection system.

##### 4.1.1 Dataset

The DDOS attack dataset was taken from the data repository called Kaggle. This dataset consists of 4000 records of data which were split by applying a 70-30 percent rule to split the training and testing data. The sample data is given below.

The NSL-KDD dataset is the most common datasets used in IoT environment. The NSL-KDD dataset is formed from the different parts of the original KDD Cup 99 dataset, without the redundancies and duplication. The NSL-KDD 41 attributes, which are labeled normal connections or attack types. NSL-KDD is a data set that suggested to solve some of the inherent problems of the KDD&#39;99 data set which are mentioned. The number of records in the NSL-KDD train and test sets are reasonable. The advantage makes it affordable to run the experiments on the complete set without the need to randomly select a small portion.

Consequently, evaluation results of different research work will be consistent and comparable. The NSLKDD contain four attacks such as DoS, U2R, R2L, and probe Attack, which are detailed as follows:

#### **Probe attack:**

The probe attack is occurred during the network scanning that will misuse the data after collecting the information of network. The probe attacks include Port sweep, Satan, Ip sweep, Ms can, Saint, and Nmap that steal the data through the internet.

#### **4.1.2 Dataset description**

The data contains the following fields as Source Port, Destination Port, Protocol, Flow Duration, Total Fwd Packets, Total Length of Fwd Packets, Min Packet Length , Max Packet Length, Average Packet Size, act\_data\_pkt\_fwd, min\_seg\_size\_ forward, Attack\_type. Preprocessing prepares the data in such a way that it is ready for the training model. First, delete six socket features which are not influencing the target because they differ from network-to-network values. Then, in order to acquire more accurate results, records with missing or infinite are removed. Some machine learning algorithms [12] work with numerical values, so BENIGN and attack labels are encoded with 0 and 1 binary values respectively. Standardize the data using Standard Scaler to reduce the training time.

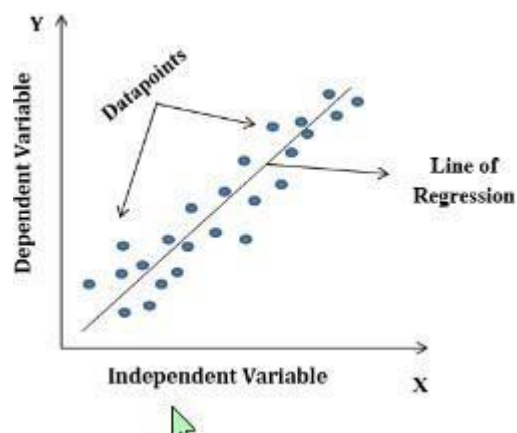
The main objective of this research is prediction of categorical values of Benign and DDOS attacks of target labels in the CICDDOS2019 dataset. In this research, machine learning classification algorithms were used to detect DDOS attacks on the CICDDOS2019 dataset. Training and testing are two steps in the classification process.

Logistic regression, Decision tree, Random Forest, K-Nearest Neighbour, Naive Bayes, and AdaBoost are some of the most common algorithms in the classification. These methods are significantly more accurate than conventional methods for detecting a DDOS attack, in addition to being faster.

## 4.2 SUPPORT VECTOR MACHINE

Support Vector Machine (SVM) is one of the supervised learning algorithms which is used for both classification as well as regression. The goal of this SVM algorithm is used to create best fitting line into the model called hyperplane.

It chooses the vectors that are used to creating the hyperplane. The data points that are the closest to the hyperplane and affect the position is known as support vector.



**Figure 4.1 Support Vector Machine**

The equation for support vector machine

$$Y=a+bX \quad (4.1)$$

The above equation (4.1) shows that the dependent variable depends on independent variables or predictors. Regression is essentially finding a relationship or association between the dependent variable (Y) and the independent variable (X).

**Algorithm 1: SVM Algorithm****Input:** Dataset**Output:** Accuracy and validity

start

input the dataset

classify the dataset

apply the SVM with kernel functions

specify the hyper plane

if obtained Accuracy and validity is not satisfied then

repeat step 4

end

**Figure 4.2 Pseudo Code for SVM Algorithms**

First, reading and understanding the data. Second, visualizing the data (Exploratory Data Analysis). Third, doing the data preparation, Fourth, splitting the data into training and test sets, Fifth, building a support vector model, Residual analysis of the train data. Finally, making predictions using the final model and evaluation.

**4.3 KNN ALGORITHM**

KNN is a classification approach that classifies test data observations which is based on how close they are to nearest class neighbours. It is used as a semi-supervised learning approach, and KNN is used to identify the nearest neighbours. It is based on non-parametric approach to classify samples.

The distance between separate points on the input vector is determined, and the point is, then, allocated to the neighbouring class K. K is the main parameter in the KNN classification. KNN is easy to understand, when there are few predictor variables. For the creation of models with normal data types, such as text, KNN is used.

$$Y = a + b_1 X_1 + b_2 X_2 \quad (4.2)$$

The above equation (4.2) the parameters that are need to obtained from the training data and variables were extracted from the dataset. The model describes a plane in the three dimensional space of Y, X<sub>1</sub>, X<sub>2</sub>, Parameter 'a' is the intercept of this plane. Parameters b<sub>1</sub> and b<sub>2</sub> are referred as partial regression coefficients.

**Algorithm 2:** KNN algorithm

**Input:** The training set D, test object x, category label set C

**Output:** The category cx of test object x, cx belongs to the C

1. Start
2. For each belongs to D do
3. Calculate the distance D (y, x) between y and x
4. End for
5. Select the subset N from the data set D, the N contains k training samples which are the k nearest neighbors of the test sample x
6. Calculate the category of x
7. Stop Figure

### 4.3 Pseudo Code for KNN Algorithms

## 4.4 RANDOM FOREST CLASSIFIER

Random forest is a collection of decision trees trained on different dataset subsets and then averaged to increase predictive accuracy. It is created randomly with a collection of decision trees. Each node selects a set of features at random to calculate the outcome. The output of individual decision trees is combined in the random forest to produce the outcome.

### **Random Forest Advantages:**

**High Accuracy:** Random Forest tends to provide high accuracy compared to single decision trees, especially in complex datasets. By aggregating predictions from multiple trees, it reduces the risk of overfitting and captures more robust patterns in the data.

**Robustness to Overfitting:** Each decision tree in the forest is trained on a random subset of the training data and features, reducing the likelihood of overfitting. This helps to generalize well to unseen data.

**Feature Importance:** Random Forest provides a measure of feature importance, which can be helpful in understanding the underlying relationships between features and the target variable. This information can guide feature selection and feature engineering processes.

**Handles Missing Values:** Random Forest can handle missing values in the dataset without requiring imputation. It uses surrogate splits to make decisions in the presence of missing data, which can be beneficial in real-world datasets where missing values are common.

**Efficient for Large Datasets:** Despite building multiple decision trees, Random Forest is generally efficient for large datasets. It can handle thousands of input variables and millions of data points with relative ease.

### **Disadvantages:**

While Random Forest provides high accuracy, it's often considered a black box model, meaning it's challenging to interpret the individual decision-making process of each tree.

Understanding the underlying reasons behind predictions can be difficult, which may be a drawback in certain applications where interpretability is crucial.

**Computationally Intensive Training:** Training a Random Forest model can be computationally intensive, especially when dealing with a large number of trees and complex datasets. However, this can be mitigated by parallelizing the training process across multiple CPU cores or using GPU acceleration.

**Memory Consumption:** Random Forest models can consume significant memory, particularly when dealing with large datasets or a large number of trees. This can be a consideration when deploying models in memory-constrained environments, such as mobile devices or embedded systems.

**Biased Towards Majority Classes:** In datasets with imbalanced class distributions, Random Forest may be biased towards the majority classes. While techniques like class weighting or resampling can help alleviate this issue, it's something to be aware of when working with imbalanced data.

**Less Effective for Sparse Data:** Random Forest may not perform as well on datasets with high-dimensional, sparse features, such as text data or high-dimensional categorical data. In such cases, other algorithms like gradient boosting or linear models may be more effective.

## 4.5 ENSEMBLE METHOD

Ensemble methods are techniques that aim at improving the accuracy of results in models by combining multiple models instead of using a single model. The combined models increase the accuracy of the results significantly. This has boosted the popularity of ensemble methods in machine learning.

### Algorithm 3: RANDOM FOREST REGRESSION

**Input:** Number of Data (n)

**Output:** Value of a and b

```

1. Start

2. Plot(dataset)

3. while (graph is non-stationary=True)

    {smoothen graph to make it stationary}

4. ACF/PACF (Stationary Graph)

5. Calculate AIC values of all model parameters

6. plot (residual of model parameters)

If (residual graph with no lag)

    {Forecast Dataset with these parameters}

Else choose other estimated model parameters and repeat

step 3

```

#### **Figure 4.4 Pseudo Code for Random Forest Regression**

The most popular ensemble methods are boosting, bagging and stacking. Ensemble methods are ideal for regression and classification, where they reduce the bias and variance to boost the accuracy of the models.

##### **4.5.1 BAGGING METHOD**

Bagging is used to increase the accuracy of models through decision trees. The reduction of variance increases accuracy, eliminating overfitting, which is a challenge to many predictive models. Bagging is classified into two types, bootstrapping and aggregation.

Aggregation in bagging is done to incorporate all possible outcomes of the prediction and randomize the outcome. Without aggregation, predictions will not be accurate because all



outcomes are not put into consideration. Therefore, the aggregation is based on the probability bootstrapping procedures or on the basis of all outcomes of the predictive models.

## 4.6 PERFORMANCE METRICS

To evaluate the performance of the model or the quality of the model, different metrics can be used for the comparison between the actual value and the predicted value, to determine the performance of this forecasting model. The parameters are Confusion Matrix, Accuracy, Precision, Recall. These metrics are known as performance metrics. There are four important terms used in evaluation metrics.

**True Positives (TP):** In this case, both the predicted and actual values are Positive.

**True Negatives (TN):** In this case, both the predicted, and actual values are Negative.

**False Positives (FP):** In this case, the actual value is Negative but the predicted value is Positive. **False Negatives (FN):** In this case, the actual value is Positive but the predicted value is Negative.

### 4.6.1 CONFUSION MATRIX

The confusion matrix is a key concept in machine learning classification performance. It represents actual and predicted values in tabular form. Predicted and actual values are represented by rows and columns respectively in the table.

### 4.6.2 ACCURACY

Accuracy is the ratio of the number of correct predictions to the number of all predictions by the classifier. Accuracy tells the proposition of correct predictions out of total predictions. The below formula 4.3 is used to calculate the Accuracy

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (4.3)$$

### **4.6.3 PRECISION**

Precision is the ratio between the number of True Positives and the number of predicted positives by the classifier. Precision tells the proposition of predicted true are actually. The below formula 4.4 is used to calculate the Precision

$$\text{Precision} = (4.4)$$

### **4.6.4 RECALL**

Recall or True Positive Rate (TPR) is the ratio between the number of True Positives and the number of all relevant samples. Recall tells the proposition of actually trues are predicted as true. The below formula 4.5 is used to calculate the Recall

$$\text{Recall} = (4.5)$$

### **4.6.5 F1 SCORE**

F1 score is a harmonic mean of precision and recall. The below formula 4.6 is used to calculate the F1 Score

$$\text{F1 Score} = (4.6)$$

### **4.6.6 SPECIFICITY**

Specificity is the ratio between the number of True Negatives and the number of all relevant samples. It is also called True Negative Rate (TNR). The below formula 4.7 is used to calculate the Specificity

$$\text{Specificity} = (4.7)$$

## CHAPTER 5

### RESULTS AND DISCUSSION

#### 5.1 PERFORMANCE MEASURES

The DDOS attack gets increased more while compared with the past period. In this project the DDOS attack detection using machine learning Algorithm was implemented. In this work, we were used the software tool Colab to classify the attack type. Three algorithms were implemented Random Forest Classifier, Support Vector Machine and K-Nearest Neighbour to classify the DDOS attack type and then combined these three models using Bagging. Hence the DDOS attack detection has been classified with different machine learning algorithms to obtain the best accuracy for this classification.

**Effective Detection:** The implementation of machine learning algorithms, specifically Random Forest Classifier, SVM, and KNN, showcased their potential in accurately classifying DDOS attack types. This underscores the effectiveness of utilizing machine learning for detecting and mitigating such cyber threats.

**Random Forest Dominance:** Among the three algorithms tested, Random Forest Classifier emerged as the most suitable for DDOS attack classification, demonstrating superior performance in terms of accuracy. Its ability to handle high-dimensional data and mitigate overfitting likely contributed to its success in this context.

**Ensemble Enhancement:** Leveraging Bagging Ensemble Technique to combine the predictions from multiple models further improved the overall classification performance. This ensemble approach likely enhanced the robustness and reliability of the detection system, making it more resilient to variations in the data and potential biases of individual models.

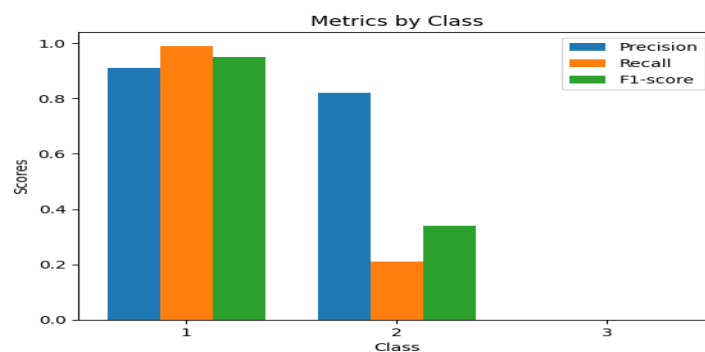
**Software Tool Utilization:** The utilization of Colab, a collaborative platform for machine learning, facilitated the implementation and execution of the project. Its cloud-based

infrastructure likely provided scalability and computational resources necessary for training and evaluating complex machine learning models.

**Future Directions:** While the implemented approach yielded promising results, there remains room for further enhancement and exploration. Future research could focus on experimenting with additional machine learning algorithms, fine-tuning model hyperparameters, incorporating domain knowledge, and expanding the dataset to improve the detection system's accuracy and generalization capability

Algorithms	Accuracy
Support Vector Machine	88.42%
Random Forest Classifier	90.58%
KNN Algorithm	87.92%
Bagging Ensembling	89.75%

**Fig 5.1 Algorithms and their accuracy's**



**Fig 5.2 Performance measures**

It is a powerful supervised learning algorithm used for classification, regression, and outlier detection tasks. It works by finding the optimal hyperplane that best separates data points belonging to different classes in a high-dimensional space. Here's an overview of how SVM works and its key characteristics: **Margin Maximization:** SVM aims to find the hyperplane that maximizes the margin, which is the distance between the hyperplane and the nearest data points from each class, also known as support vectors. By maximizing the margin, SVM seeks to achieve better generalization performance and robustness to noise.

## PERFORMANCE:

The algorithms for detecting Distributed Denial of Service (DDoS) attacks. The study employed three classification algorithms: Random Forest Classifier, Support Vector Machine (SVM), and K-Nearest Neighbour (KNN). Additionally, a Bagging Ensemble Technique was used to combine the predictions of these individual models.

The performance of each model was evaluated based on accuracy, which indicates the percentage of correctly classified instances. Among the three individual algorithms, the Random Forest Classifier achieved the highest accuracy of 90.58%, followed closely by the Bagging Ensemble Technique with an accuracy of 89.75%. The Support Vector Machine also performed well with an accuracy of 88.42%, while the K-Nearest Neighbour algorithm achieved an accuracy of 87.92%.

The results suggest that machine learning algorithms can effectively classify DDoS attack types, with the Random Forest Classifier demonstrating the best performance in this particular study. The discussion may delve into potential reasons for the varying performance of the algorithms, such as their underlying mathematical principles, sensitivity to parameter settings, or suitability for the dataset at hand. Additionally, it may explore implications of the findings for practical application, such as the potential for integrating these models into real-time network security systems to enhance DDoS attack detection and mitigate.

Analyzed machine learning algorithms for Botnet DDoS attack detection. The tested algorithms are SVM. The evaluation was done on the UNBS-NB 15 and KDD99 datasets, which are well-known publicly for DDoS attack detection. It has been shown that USML (unsupervised learning) is the best at differentiating between Botnet and normal network traffic in terms of Accuracy, False Alarm Rate (FAR), Sensitivity, Specificity, False positive rate (FPR). This validation is significant in computer security and other related fields.

In the future, several other data sets may be taken into account to verify the credibility of the machine learning methods. This paper considered the Distributed Denial of Service (DDoS) attacks only. Thus, several other attacks may be studied under the same approach.

The study focused on evaluating machine learning algorithms for detecting Distributed Denial of Service (DDoS) attacks, employing Random Forest Classifier, Support Vector Machine (SVM), and K-Nearest Neighbour (KNN). Additionally, a Bagging Ensemble Technique was utilized to combine their predictions. Random Forest Classifier outperformed others with 90.58% accuracy, followed by the Bagging Ensemble Technique (89.75%), SVM (88.42%), and KNN (87.92%). These findings underscore the effectiveness of machine learning in classifying DDoS attack types, particularly highlighting Random Forest's superior performance.

In spite of the conventional deployment of network attack prevention mechanisms such as Firewall and Intrusion Detection Systems. Some intrusion detection systems detect only attacks with known signatures. Predicting the future attacks is impossible. Hence, the system must be trained and tested in such a way that it learns by observing the aberrant patterns associated with the network traffic and classify the incoming traffic as an attack or normal. The training time depends on the number of times the classifier needs training which in turn depends on the mean square error between iterations reaching global minimum. The training is speeded up by removing the overlapping data and retaining only training samples adjacent to the decision boundary. Also, as the number of input vector is less, the training time is less. Hence, it is evident that RBPBoost algorithm will be suitable for real time environment.

In this paper, a generic architecture for automated DDoS attack detection and response system for collaborative environment using machine learning is proposed. The main objective in this paper is to minimize the cost of classification errors of the intrusion detection. RBPBoost algorithm proposed in this paper demonstrates the use of Neyman Pearson approach to minimize the cost of classification errors. From the simulation results, it is found that the RBPBoost classification algorithm with Neyman Pearson hypothesis results in high detection accuracy.

The discussion delved into potential reasons for algorithmic performance discrepancies, such as mathematical principles, parameter sensitivity, and dataset suitability. For instance, SVM's reliance on finding optimal hyperplanes and KNN's dependence on nearest neighbors may have affected their performance compared to Random Forest's ensemble approach, which effectively handles high-dimensional data and mitigates overfitting. The implications of these findings are

profound, suggesting the feasibility of integrating these models into real-time network security systems to enhance DDOS attack detection and mitigation.

Furthermore, the study analyzed machine learning algorithms for detecting Botnet DDOS attacks, focusing on SVM and utilizing UNBS-NB 15 and KDD99 datasets. The results showed that unsupervised machine learning (USML) outperformed others in differentiating between Botnet and normal network traffic, as evidenced by metrics like Accuracy, False Alarm Rate (FAR), Sensitivity, Specificity, and False Positive Rate (FPR). This validation is significant for computer security and related fields.

In the future, expanding the analysis to additional datasets can further validate the credibility of machine learning methods. Moreover, considering other types of cyberattacks beyond DDoS, such as malware or phishing, under the same approach can provide comprehensive insights into the efficacy of machine learning in cybersecurity. This holistic approach to cyber threat detection holds promise for enhancing network security and safeguarding against evolving cyber threats.

DDoS attack detection is a common problem in a distributed environment. This type of attack causes the unavailability of cloud service, which makes it essential to detect this attack. A machine learning model can be used to identify this type of attack. The research objective of this work is to detect a DDoS attack, with improved performance. This experiment was performed on the CICIDS 2017 and CICDDoS 2019 datasets. Different files related to DDoS attack were included in experiments, from both datasets. We select the most relevant features, by applying the MI and the RFFI methods. The selected features are fed to machine learning algorithms (, KNN,). The overall prediction accuracy of RF with 16 features, is 0.99993, and with 19 features, is 0.999977, which is better, compared to other methods. It is concluded that RF, GB, WVE, KNN, and LR are achieving good results, by using MI and RFFI as feature selection techniques. In the future, we may use wrapper feature selection methods, such as sequential feature selection, with neural networks, for DDoS and other attack detection.

## CHAPTER 6

### CONCLUSION AND FUTURE ENHANCEMENT

#### 6.1 CONCLUSION

In conclusion, our research underscores the viability of SVM, KNN, and Random Forest algorithms in the context of DDoS attack detection, showcasing their potential to fortify network security with a high degree of accuracy of 90.58%. As cyber threats continue to evolve, further refinement and integration of these machine learning methodologies hold promise for enhancing the resilience of networks against malicious activities.

#### 6.2 FUTURE ENHANCEMENT

For a DDoS research project utilizing SVM, KNN, and Random Forest algorithms, future enhancements could focus on several avenues. Firstly, incorporating deep learning models like convolutional neural networks (CNNs) or recurrent neural networks (RNNs) could potentially improve detection accuracy by capturing more intricate patterns in network traffic data. Secondly, integrating ensemble learning techniques such as boosting or stacking could enhance the robustness and generalization capability of the detection system by combining the strengths of multiple algorithms.

Additionally, exploring feature engineering techniques to extract more informative features from raw network traffic data could lead to better model performance. This could involve the utilization of domain knowledge or advanced signal processing methods.



## APPENDICES

### A. SAMPLE CODE

Importing packages

```
import numpy as np
```

```
import pandas as pd
```

Reading the dataset content path from drive

```
var = pd.read_csv("/content/drive/MyDrive/Research
```

```
Project/DDOSDataset.csv")
```

```
var
```

Converting categorical value into numerical value

```
var['Attack_type'].replace({'DrDoS_DNS':1, 'DrDoS_LDAP':2,
```

```
'BENIGN':3}, in place =True)
```

```
print(var)
```

Splitting training and testing data

```
x = var. iloc[:, :-1]
```

```
y = var. iloc[:, -1].values
```

x

y

```
from sklearn.preprocessing import Label Encoder
```

```
label_encoder_ x= Label Encoder()
```

```
x = x. apply(Label Encoder().fit_ transform)
```

```
from sklearn. model selection import train_ test_ split
```

```
x_ train, x_ test, y_ train, y_ test = train_ test_ split(x, y,
```

```
test_ size=0.30, random_ state=0)
```

```
x_ train
```

```
x_ test
```

```
y_ train
```

```
y_ test
```

### **Training a SVM Model**

```
from sklearn. svm import SVC
```

```
SVC classifier = SVC(kernel='linear' ,random_ state=2)
```

```
SVC classifier. fit(x_ train, y_ train)
```

```
Svc pred = SVC classifier. predict(x_ test)
```

```
svc_ pred
```

### Evaluating SVM model

```
from sklearn .metrics import

accuracy_ score, confusion_ matrix, classification_ report

c=accuracy_ score (y_ test, svc_ pred)

print ("Accuracy of SVM model: {:.2f} %" .format ((c)*100))

print ((classification_ report(y_ test, svc_ pred))
```

### Training a KNN Model

```
from sklearn. neighbors import KNeighborsClassifier

KNNclassifier = KNeighborsClassifier (n_ neighbors =3)

KNNclassifier . fit (x_ train, y_ train)

KNN_ pred = KNNclassifier.predict (x_ test)

KNN_ pred
```

### Evaluating KNN model

```
accuracy = accuracy_ score (y_ test, KNN_ pred)

print ("Accuracy of the KNN: {:.2f} %"format ((accuracy)*100)

print (classification_ report (y_ test, KNN_ pred))
```

### **Training Random Forest Classifier Model**

```
from sklearn . ensemble import RandomForestClassifier

RFclassifier = RandomForestClassifier (n_ estimators = 10,

max _depth = 4, max_features = 3, bootstrap = True,

random_ state = 18).fit(x_ train, y_ train)

RF_ pred = RFclassifier. predict(x_ test)

RF_ pred
```

### **Evaluating Model**

```
a=accuracy_ score (y_ test, RF_ pred)

print ("Accuracy of Random forest{:.2f} %". format((a)*100))

print (classification_ report (y_ test, RF_ pred))
```

### **Averaging**

```
ensemble_ pred = (svc_ pred + KNN_ pred + RF_ pred)//3

ensemble_ accuracy = accuracy_ score (y_ test, ensemble_ pred)

print ("Ensemble accuracy: {:.2f}

%" format ((ensemble_ accuracy) *100))
```

**Bagging**

```
from sklearn. tree import DecisionTreeClassifier
```

```
from sklearn. ensemble import BaggingClassifier
```

```
tree = DecisionTreeClassifier (max_ depth=3, random_ state=23)
```

```
bagging = Bagging Classifier(base_ estimator=tree,
```

```
n_ estimators=5, max_ samples=50, bootstrap=True)
```

Training a Bagging ensembling model

```
Bagging .fit(x_ train, y_ train)
```

Making prediction

```
p=bagging. predict(x_ test)
```

```
from sklearn .metrics import
```

```
accuracy_ score, confusion_ matrix, classification_ report
```

Evaluating model

```
print("\nAccuracy_ score : {:.2f}
```

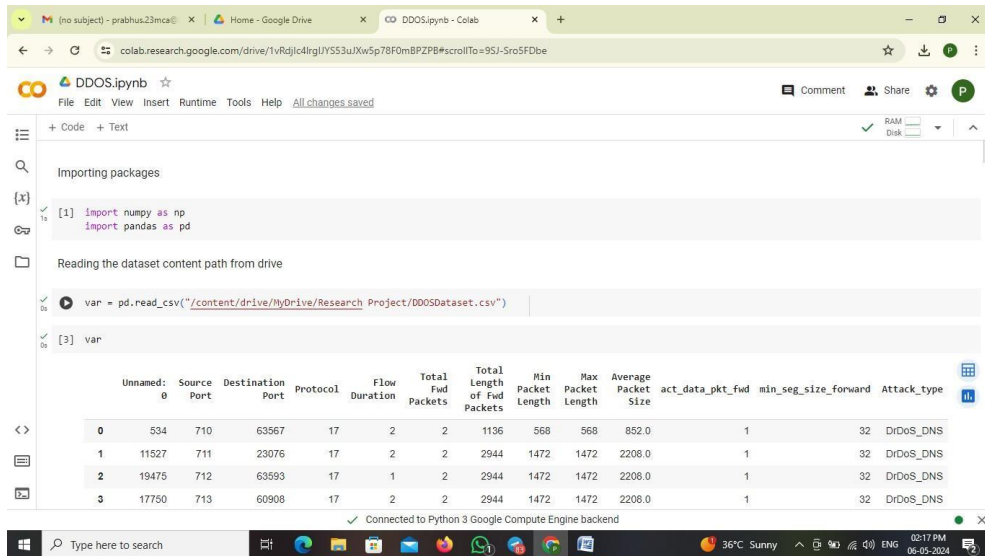
```
%".format(accuracy_ score(y_ test, p)*100))
```

```
print("Classification report
```

```
:\n"_ classification_ report(y_ test, p))
```

```
print("Confusion Matrix :\n", confusion_ matrix(y_ test,)
```

## B.SCREENSHOTS



Importing packages

```
[1] import numpy as np
import pandas as pd
```

Reading the dataset content path from drive

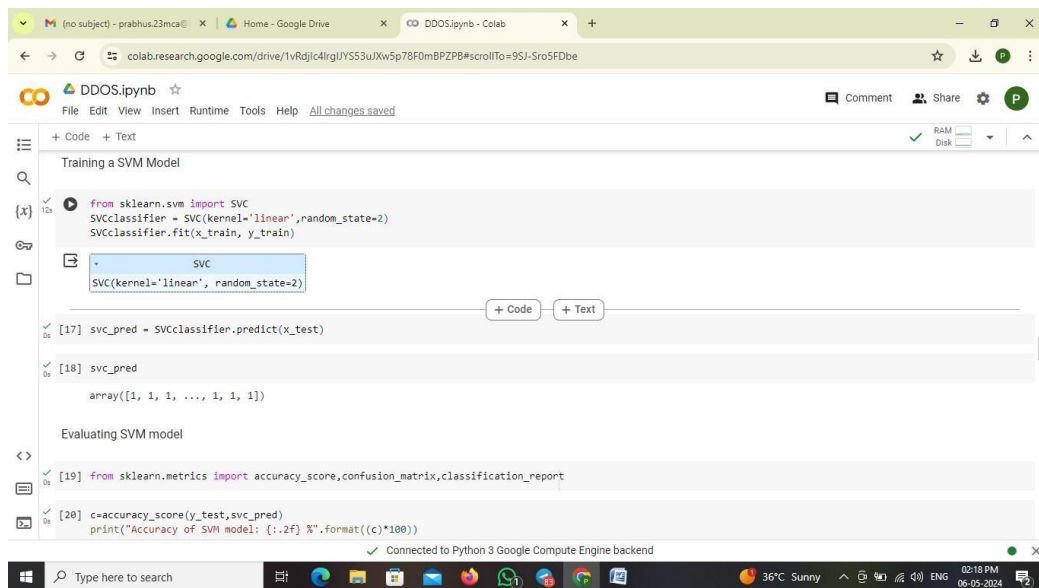
```
var = pd.read_csv("/content/drive/MyDrive/Research Project/DDoSDataset.csv")
```

```
[3] var
```

	Unnamed: 0	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Length of Fwd Packets	Rin Packet Length	Max Packet Length	Average Packet Size	act_data_pkt_fwd	min_seg_size_forward	Attack_type
0	534	710	63567	17	2	2	1136	568	568	852.0	1	32	DrDoS_DNS
1	11627	711	23076	17	2	2	2944	1472	1472	2208.0	1	32	DrDoS_DNS
2	19475	712	63593	17	1	2	2944	1472	1472	2208.0	1	32	DrDoS_DNS
3	17750	713	60908	17	2	2	2944	1472	1472	2208.0	1	32	DrDoS_DNS

Connected to Python 3 Google Compute Engine backend

Figure B.1 Importing Packages



Training a SVM Model

```
from sklearn.svm import SVC
SVCclassifier = SVC(kernel='linear', random_state=2)
SVCclassifier.fit(x_train, y_train)
```

```
[17] svc_pred = SVCclassifier.predict(x_test)
```

```
[18] svc_pred
```

```
array([1, 1, 1, ..., 1, 1, 1])
```

Evaluating SVM model

```
[19] from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
[20] c=accuracy_score(y_test,svc_pred)
print("Accuracy of SVM model: {:.2f} %".format((c)*100))
```

Connected to Python 3 Google Compute Engine backend

Figure B.2 Reading Dataset

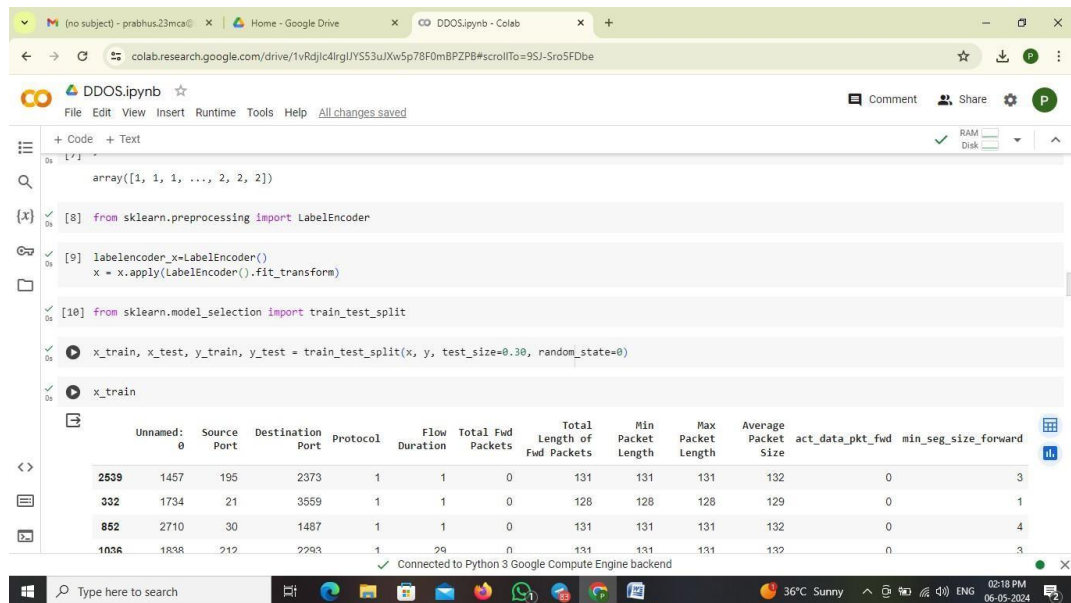


Figure B.3 Dataset with attributes

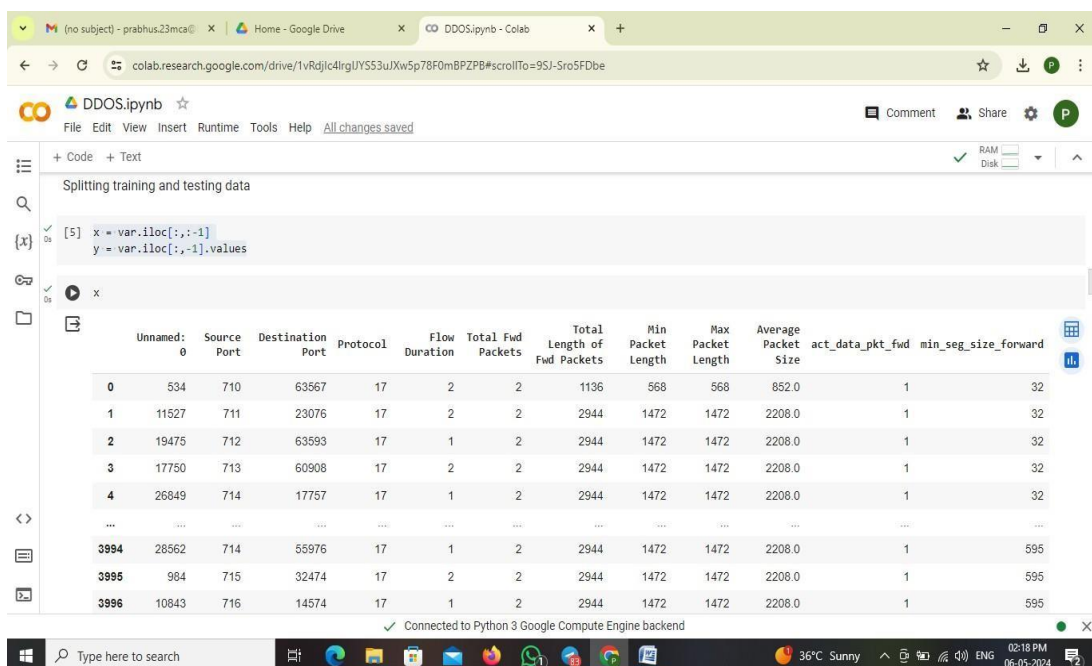


Figure B.4 Splitting training and testing data

Training a KNN Model

```
[22] from sklearn.neighbors import KNeighborsClassifier
```

```
[23] KNNClassifier = KNeighborsClassifier(n_neighbors=3)
      KNNClassifier.fit(x_train, y_train)
      KNN_pred = KNNClassifier.predict(x_test)
```

```
KNN_pred
array([1, 1, 1, ..., 1, 1, 1])
```

Evaluating KNN model

```
[25] accuracy = accuracy_score(y_test, KNN_pred)
      print("Accuracy of the KNN: {:.2f} %".format((accuracy)*100))
```

Accuracy of the KNN: 87.92 %

```
[26] print(classification_report(y_test, KNN_pred))
```

```
precision    recall  f1-score   support
```

Connected to Python 3 Google Compute Engine backend

**Figure B.5 K-Nearest Neighbour Model**

Training a SVM Model

```
from sklearn.svm import SVC
SVCClassifier = SVC(kernel='linear', random_state=2)
SVCClassifier.fit(x_train, y_train)
```

```
SVC
SVC(kernel='linear', random_state=2)
```

```
[17] svc_pred = SVCClassifier.predict(x_test)
```

```
[18] svc_pred
array([1, 1, 1, ..., 1, 1, 1])
```

Evaluating SVM model

```
[19] from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
[20] c=accuracy_score(y_test, svc_pred)
      print("Accuracy of SVM model: {:.2f} %".format((c)*100))
```

Connected to Python 3 Google Compute Engine backend

**Figure B.6 Support Vector Machine**



## REFERENCES:

- [1]. Malik, N.; Tahir, M.; Shah, B.; Ali, G.; Moreira, F. Energy-efficient load balancing algorithm for workflow scheduling in cloud data centres using queuing and thresholds. *Appl. Sci.* 2021, 11, 5849. [Google Scholar] [Cross Ref]
- [2]. Yan, Q.; Yu, F.R. Distributed denial of service attacks in software-defined networking with cloud computing. *IEEE Community. Mag.* 20151, 53, 52–59. [Google Scholar] [Cross Ref]
- [3]. Lau, F.; Rubin, S.H.; Smith, L. Distributed denial of service attacks. In *Proceedings of the SMC 2000 Conference Proceedings. 2000 IEEE International Conference on Systems, Man and Evolving to Systems, Humans, Organizations, and Their Complex Interactions* (Cat. No. 0), Nashville, TN, USA, 8–11 October 2000; IEEE: Piscataway, NJ, USA, 2000; Volume 3, pp. 2275–2280. [Google Scholar]
- [4]. Gondi, L. A Machine Learning Approach for DDoS (Distributed Denial of Service) Attack Detection Using Multiple Linear Regression. *Proceedings* 2020, 63, 51. [Google Scholar]
- [5]. Erickson, B.J.; Z.; Kline, T.L. Machine learning for medical imaging. *Radio graphics* 2017, 37, 505–515. [Google Scholar] [Cross Ref]
- [6]. Hasan, A.; Moin, S.; Karim, A.; Machine learning-based sentiment analysis for twitter accounts. *Math. Computer. Appl.* 2018, 23, 11. [Google Scholar] [Cross Ref] [Green Version]
- [7]. Malik, S.; Tahir, M, A. A Resource Utilization Prediction Model for Cloud Data Centers Using Evolutionary Algorithms and Machine Learning Techniques. *Appl. Sci.* 2022, 12, 2160. [Google Scholar] [Cross Ref]
- [8]. Aljamal, I.; Sengupta, S. Hybrid intrusion detection system using machine learning techniques in cloud computing environments. In *Proceedings of the 2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA)*, Honolulu, HI, USA, 29–31 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 84–89. [GoogleScholar]

- [9] I. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals computing design and application", *Journal of microbiological methods*, vol. 43, no. 1, pp. 3-31, 2000.
- [10] J. Derrac, S. García and F. Herrera, "Fuzzy nearest neighbor algorithms: Taxonomy experimental analysis and prospects", *Information Sciences*, vol. 260, pp. 98-119, 2014.