**Machine Learning Approach to Detect DDoS Attacks for Network Security and Protection**

**Abstract**

DDoS attacks aim to disrupt normal traffic flow to servers by overwhelming them with excessive internet traffic, resulting in significant impacts on businesses and individuals. This research focuses on detecting DDoS intrusions using machine learning and deep learning techniques. Various DDoS attack types include traffic floods and protocol exploitation, often executed through botnets. The present research utilizes a dataset from Kaggle containing 151,000 network transaction records, categorized into benign traffic and two DDoS types (DDoS-ACK and DDoS-PSH-ACK). Data preprocessing involved feature encoding, outlier detection using PCA, and normalization, reducing outliers from 99% to 60%. A hybrid feature selection method combined RFE and Chi-Squared to identify 14 optimal features for prediction. The dataset was split into training (75%) and testing (25%) sets, with models trained and compared. The CNN model achieved 100% accuracy in detecting DDoS intrusions, outperforming other models and demonstrating the research's superiority over existing studies.

Keywords: DDoS detection, Deep Learning, Hybrid Feature Selection, Outlier Reduction, Classification, Intrusion Detection

**Acknowledgement**

**Project Declaration**

**List of Abbreviations**

KDD: Knowledge Discovery in Databases

GBDT: Gradient-Boosted Decision Trees

DDoS: Distributed Denial-of-Service

CNN: Convolutional Neural Network

LSTM: Long Short-Term Memory

SDN: Software-Defined Network

SMOTE: Synthetic Minority Oversampling Technique

SVM: Support Vector Machine

IDS: Intrusion Detection System

ANN: Artificial Neural Network

NIDS: Network Intrusion Detection System

CIA: Central Intelligence Agency

ML: Machine Learning

**List of Figures**

[Figure 1 Gantt Chart 15](#_Toc174879771)

[Figure 2 Arrangement of Chapters and Key Contexts 15](#_Toc174879772)

[Figure 3 Proposed Framework 27](#_Toc174879773)

[Figure 4 APA DDoS Dataset 28](#_Toc174879774)

[Figure 5 Methods Chosen to Evaluate Algorithm's Performances 33](#_Toc174879775)

[Figure 6 Artefact Environment Setup 33](#_Toc174879776)

[Figure 7 Process of Library Import 34](#_Toc174879777)

[Figure 8 Data Outlook obtained in Jupyter Notebook 34](#_Toc174879778)

[Figure 9 Information on Data Features 35](#_Toc174879779)

[Figure 10 Missing Values Present in Data 35](#_Toc174879780)

[Figure 11 Encoded Data 36](#_Toc174879781)

[Figure 12 Outlier Detection and Reduction Process 36](#_Toc174879782)

[Figure 13 Hybrid Feature Selection Technique 37](#_Toc174879783)

[Figure 14 Final Hybrid Approach for Feature Selection 38](#_Toc174879784)

[Figure 15 Final Features Selected 38](#_Toc174879785)

[Figure 16 Queue for Machine Learning Models 39](#_Toc174879786)

[Figure 17 Model Summary and Structure 41](#_Toc174879787)

[Figure 18 Model Execution Queue 42](#_Toc174879788)

[Figure 19 Model Performance Evaluation 44](#_Toc174879789)

[Figure 20 Best-Performing Model 45](#_Toc174879790)

[Figure 21 Results of the Best-Performing Model 45](#_Toc174879791)

**List of Tables**

[Table 1 Feature Details 29](#_Toc174879792)

[Table 2 Data Classes 30](#_Toc174879793)

[Table 3 Source IP and Transactions of Packets 30](#_Toc174879794)

[Table 4 Planned Algorithms 32](#_Toc174879795)

[Table 5 Methods Applied for Hybrid Approach 38](#_Toc174879796)

[Table 6 Assigned Parameters to Machine Learning Models 40](#_Toc174879797)

[Table 7 CNN Layer Parameters 41](#_Toc174879798)

[Table 8 Data Split and Distribution of Class Instances 43](#_Toc174879799)

[Table 9 Detection Result 44](#_Toc174879800)

[Table 11 Research Comparison 47](#_Toc174879801)

Table of Contents

[Chapter-1. Introduction 10](#_Toc174879802)

[1.1 Project Overview 10](#_Toc174879803)

[1.1.1 Overview 10](#_Toc174879804)

[1.1.2 DDoS Attack and Affects on Network 10](#_Toc174879805)

[1.1.3 Application of Machine Learning and Deep Learning 11](#_Toc174879806)

[1.1.4 Scopes of Machine Learning and Deep Learning 11](#_Toc174879807)

[1.1.5 Challenges in Implementing Machine Learning and Deep Learning 12](#_Toc174879808)

[1.2 Project Particulars 12](#_Toc174879809)

[1.2.1 Research Questions 12](#_Toc174879810)

[1.2.2 Project Aim 12](#_Toc174879811)

[Chapter-2. Issues and Planning for Research 14](#_Toc174879812)

[2.1 Issues Related to Project 14](#_Toc174879813)

[2.1.1 Ethical Issues 14](#_Toc174879814)

[2.1.2 Social Issues 15](#_Toc174879815)

[2.1.3 Legal Issues 15](#_Toc174879816)

[2.1.4 Security Issues 15](#_Toc174879817)

[2.1.5 Professional Issues 16](#_Toc174879818)

[2.1.6 Organizational Issues 16](#_Toc174879819)

[2.2 Research Planning 16](#_Toc174879820)

[2.3 Arrangement of Chapters 17](#_Toc174879821)

[2.4 Resources for Research 18](#_Toc174879822)

[2.4.1 Hardware Resources 18](#_Toc174879823)

[2.4.2 Software Resources 19](#_Toc174879824)

[Chapter-3. Literature Review 20](#_Toc174879825)

[3.1 Data Preprocessing 20](#_Toc174879826)

[3.1.1 Outlier Detection in Data 20](#_Toc174879827)

[3.1.2 Feature Selection Techniques 21](#_Toc174879828)

[3.2 Intrusion Detection 22](#_Toc174879829)

[3.2.1 Application of Machine Learning 22](#_Toc174879830)

[3.2.2 Application of Deep Learning 24](#_Toc174879831)

[3.3 DDoS Detection 25](#_Toc174879832)

[3.3.1 Application of Machine Learning 25](#_Toc174879833)

[3.3.2 Application of Deep Learning 28](#_Toc174879834)

[Chapter-4. Methodology 30](#_Toc174879835)

[4.1 Proposed Framework 30](#_Toc174879836)

[4.2 Data Collection 30](#_Toc174879837)

[4.2.1 Data Source 30](#_Toc174879838)

[4.2.2 Feature Details 31](#_Toc174879839)

[4.2.3 Data Details 32](#_Toc174879840)

[4.3 Tool to be Used 33](#_Toc174879841)

[4.4 Planned Algorithms and Evaluation Methods 34](#_Toc174879842)

[4.4.1 Algorithms 34](#_Toc174879843)

[4.4.2 Evaluation Methods 35](#_Toc174879844)

[Chapter-5. Artefact Implementation and Result 36](#_Toc174879845)

[5.1 Preparing Artefact Environment 36](#_Toc174879846)

[5.2 Processing Data 36](#_Toc174879847)

[5.2.1 Library Import 36](#_Toc174879848)

[5.2.2 Dataset Reading 37](#_Toc174879849)

[5.2.3 Feature Information 37](#_Toc174879850)

[5.2.4 Missing Value 38](#_Toc174879851)

[5.3 Feature Preprocessing 38](#_Toc174879852)

[5.3.1 Encoding Categorical Features 38](#_Toc174879853)

[5.3.2 Treating Feature Outliers 39](#_Toc174879854)

[5.3.3 Selection of Optimum Features 40](#_Toc174879855)

[5.4 Model Preparation 42](#_Toc174879856)

[5.4.1 Preparing Machine Learning model 42](#_Toc174879857)

[5.4.2 Preparing CNN Model 43](#_Toc174879858)

[5.4.3 Model Execution Queue 44](#_Toc174879859)

[5.5 Data Preparation 45](#_Toc174879860)

[5.6 Result 45](#_Toc174879861)

[Chapter-6. Discussion and Conclusion 47](#_Toc174879862)

[6.1 Discussion 47](#_Toc174879863)

[6.1.1 Evaluation of Research Performances 47](#_Toc174879864)

[6.1.2 Discussion on Research Questions 49](#_Toc174879865)

[6.1.3 Discussion on Research Gap Filling 50](#_Toc174879866)

[6.1.4 Discussion on Challenges Faced 51](#_Toc174879867)

[6.2 Conclusion 51](#_Toc174879868)

[6.2.1 Research Finding 51](#_Toc174879869)

[6.2.2 Limitations 52](#_Toc174879870)

[6.2.3 Future Scopes 52](#_Toc174879871)

[References 53](#_Toc174879872)

[Appendix 60](#_Toc174879873)

# Introduction

## Project Overview

### Overview

In today's interconnected digital world, Distributed Denial of Service (DDoS) attacks have become a significant threat to businesses, organizations, and individuals (Devi, et al., 2023). These attacks aim to disrupt the normal functioning of a network or system by overwhelming it with a flood of traffic, causing services to become unavailable. Traditional methods of detecting and mitigating DDoS attacks are often insufficient due to the evolving nature of cyber threats. However, the integration of machine learning and deep learning algorithms has shown promise in enhancing DDoS attack detection capabilities (Li, et al., 2022).

### DDoS Attack and Affects on Network

#### DDoS Attacks

DDoS attacks are malicious attempts to disrupt normal traffic of a targeted server, service, or network by overwhelming it with a flood of internet traffic. These attacks can have severe consequences for businesses, organisations, and individuals, causing downtime, financial losses, and damage to reputation. In this essay, we will explore the description of DDoS attacks in networks and their effects (Ahmed, et al., 2009).

#### Types

DDoS attacks can take various forms, including volumetric attacks that flood the network with a high volume of traffic, protocol attacks that exploit vulnerabilities in network protocols, and application layer attacks that target specific applications or services (Khuphiran, et al., 2018). Attackers often use botnets, which are networks of compromised devices, to launch coordinated DDoS attacks, making it difficult to trace the source of the attack.

#### Effects

The effects of DDoS attacks on networks can be devastating. For businesses, DDoS attacks can result in downtime, leading to loss of revenue, decreased productivity, and damage to customer trust (Nurwarsito & Nadhif., 2021). In some cases, DDoS attacks are used as a diversion tactic to mask more serious security breaches, putting sensitive data at risk.

#### Possible Mitigation Process

To mitigate the impact of DDoS attacks, organizations can implement various security measures, such as deploying firewalls, intrusion detection systems, and DDoS mitigation services (Wang, et al., 2020). Regularly monitoring network traffic and conducting vulnerability assessments can also help identify and address potential weaknesses in the network infrastructure.

### Application of Machine Learning and Deep Learning

Machine learning and deep learning algorithms have revolutionized the field of cybersecurity by providing automated and intelligent solutions to detect and respond to cyber threats (Li, et al., 2022). In the context of DDoS attack detection, these algorithms can analyze network traffic patterns, identify anomalies, and differentiate between legitimate and malicious traffic in real-time. By training on historical data, machine learning models can learn to recognize patterns associated with DDoS attacks, enabling them to make accurate predictions and trigger timely responses to mitigate the impact of such attacks (Bhutia, et al., 2022).

Deep learning, a subset of machine learning that uses artificial neural networks to mimic the human brain's structure, has shown particular efficacy in handling complex and high-dimensional data, making it well-suited for DDoS attack detection (Nurwarsito & Nadhif., 2021). Deep learning models can automatically extract features from raw network traffic data, allowing them to adapt to new and previously unseen attack patterns without the need for manual reprogramming (Zhang, et al., 2017).

### Scopes of Machine Learning and Deep Learning

The integration of machine learning and deep learning in DDoS attack detection offers several benefits, including:

1. Improved Detection Accuracy: Machine learning models can analyze vast amounts of data quickly and accurately, enabling them to detect DDoS attacks with high precision.
2. Real-time Response: By continuously monitoring network traffic, machine learning algorithms can detect and respond to DDoS attacks in real-time, minimizing the impact on network performance.
3. Adaptability: Deep learning models can adapt to evolving attack strategies and variations, making them more resilient to new and sophisticated DDoS attack methods.

### Challenges in Implementing Machine Learning and Deep Learning

Despite their potential, the implementation of machine learning and deep learning for DDoS attack detection poses several challenges, including:

1. Data Quality and Quantity: Machine learning models require large and high-quality datasets for training, which may be challenging to obtain in the context of DDoS attacks due to their sporadic and unpredictable nature.
2. Model Interpretability: Deep learning models, in particular, are often considered black boxes, making it difficult to interpret their decision-making process, which is crucial for understanding and fine-tuning detection mechanisms.
3. Resource Intensiveness: Training and deploying machine learning and deep learning models for real-time DDoS attack detection may require substantial computational resources and expertise, posing barriers to implementation for smaller organizations.

## Project Particulars

### Research Questions

The research questions for the project are as follows:

1. How network can be protected with the application of deep learning by identifying DDoS attacks?
2. Can this research show an improved detection approach compared to previous research?

### Project Aim

#### Aim

The project aims to analyze network features to detect DDoS attacks with the application of Deep Learning.

#### Discussion

In this project, a variety of network features are extracted and analyzed to train the Deep Learning model. These features may include packet size, traffic volume, communication protocols, and more. By examining these features, the model can learn to differentiate between normal network behaviour and the patterns associated with DDoS attacks. This advanced level of analysis allows for swift and accurate detection of malicious activities, enabling network administrators to take proactive measures to protect their systems. The application of Deep Learning in DDoS attack detection represents a significant advancement in cybersecurity. By continuously learning and adapting to new threats, the system can stay ahead of evolving attack techniques. This proactive approach is essential in safeguarding networks against the growing complexity and sophistication of cyber threats.

#### Objectives of Project

The project objectives to detect network-based intrusions are discussed below:

1. To review the previous research papers concerning DDoS detection by emphasizing the application of deep learning models and understanding the application of algorithms and approaches.
2. To select the database containing the network transaction records and preprocess the features of the databases concerning data cleaning, encoding the feature, detecting & eliminating outliers etc.
3. To apply a hybrid feature selection method (by combining the selected features individually using Chi-squared and Recursive Feature Elimination) and prepare the data with the finally selected features.
4. To apply classifiers of machine learning and deep learning to detect DDoS attacks and determine the best-performing model using which the DDoS can be detected with the highest accuracy, least model overfit and less prediction time.
5. To identify the improvement in the detection approach compared to the existing research models.

# Issues and Planning for Research

## Issues Related to Project

The use of machine learning and deep learning in DDoS detection offers promising advancements in cybersecurity. However, it also presents a range of ethical, social, legal, security, professional, and organisational challenges. Addressing these issues requires a multidisciplinary approach involving continuous ethical evaluation, legal vigilance, professional development, and strategic organisational management to harness the full potential of AI technologies while mitigating associated risks (Barati & Rana, 2021).

### Ethical Issues

The implementation of machine learning and deep learning techniques in detecting Distributed Denial of Service (DDoS) attacks raises several ethical concerns. One primary issue is the potential for privacy infringement. ML and DL models often require massive amounts of data to train, which might include sensitive personal information (Dangheralou & Jahankhani, 2022). Ensuring that this data is anonymized and handled with strict privacy controls is crucial to maintaining user trust and complying with privacy laws such as the General Data Protection Regulation (GDPR). Another ethical concern is the accuracy and fairness of these models. Machine learning algorithms can sometimes exhibit bias, particularly if the training data is not representative of the overall network traffic. This can lead to false positives or false negatives, unfairly targeting or neglecting certain users or groups. It is essential for developers to continuously evaluate and update their models to ensure fairness and accuracy (European Commission, 2018).

### Social Issues

Social implications of deploying ML and DL for DDoS detection also merit consideration. The automation of security measures can lead to a reduction in jobs, as machines replace human roles in monitoring and responding to cybersecurity threats. This shift can have broader social impacts, including job displacement and the need for retraining programs within the cybersecurity industry (The Data Protection Act, 2019). Moreover, there is the issue of public perception and trust in AI technologies. False positives in DDoS detection can lead to disruptions for legitimate users, potentially eroding trust in not only the security systems but also the organizations that deploy them. Ensuring that ML and DL systems are transparent and their decisions are understandable to users can help mitigate these concerns (Ahmed, et al., 2020).

### Legal Issues

Legally, the use of ML and DL in DDoS detection must navigate several complexities. Compliance with international and local laws concerning data protection and privacy is paramount. Organizations must ensure that their use of AI in cybersecurity adheres to these legal frameworks to avoid penalties and legal disputes. Additionally, there are concerns about accountability. When a DDoS attack is wrongly identified, or a legitimate activity is mistakenly blocked, determining who is liable—the software provider, the user, or the organization implementing the AI—can be challenging (Chen & Xu, 2022). Clear legal frameworks and guidelines are needed to address these accountability issues in the use of AI in cybersecurity.

### Security Issues

While ML and DL can enhance the ability to detect and respond to DDoS attacks, they also introduce new security vulnerabilities. Adversaries can attempt to manipulate the learning process by poisoning the training data with misleading information, leading to incorrect learning (adversarial attacks). Ensuring the integrity and security of the data used to train ML and DL models is critical. Moreover, the complexity of these models can sometimes be a double-edged sword (Cole & Schmitz, 2020). Complex models can be difficult to audit for vulnerabilities and can be opaque in terms of understanding how decisions are made (the "black box" problem). This opacity can make securing these systems against attacks more challenging.

### Professional Issues

From a professional standpoint, the use of ML and DL in cybersecurity requires a high level of expertise. There is a growing demand for professionals who are not only skilled in cybersecurity but are also proficient in AI and ML. This necessitates ongoing education and training programs to ensure that cybersecurity professionals can design, implement, and maintain sophisticated AI-based security systems (European Union, 2018). Additionally, there is a professional ethical responsibility to use AI in a manner that is just and beneficial for all. Cybersecurity professionals must adhere to ethical guidelines that govern the use of AI, ensuring that these technologies are used responsibly.

### Organizational Issues

Organizations implementing ML and DL for DDoS detection face several challenges. The integration of these technologies into existing IT systems can be complex and costly. Organizations must evaluate the cost-benefit ratio of adopting these advanced technologies and consider whether they have the necessary infrastructure and expertise to effectively utilize them (Barati & Rana, 2021). Furthermore, organizations must manage the change in organizational culture that comes with adopting new technologies. This includes training staff, adjusting to new operational processes, and managing the shift towards more data-driven decision-making processes in cybersecurity.

## Research Planning

The planning of the project is shown below:

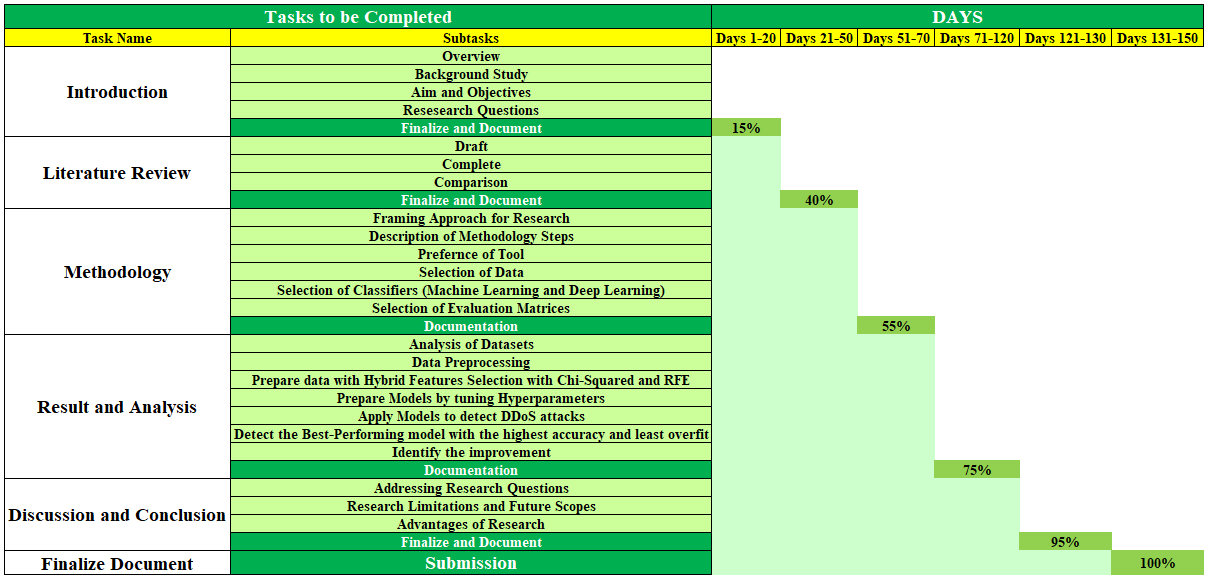


Figure 1 Gantt Chart

(Source: Self)

## Arrangement of Chapters

The arrangement and the key contexts of the chapters of this dissertation are shown below:

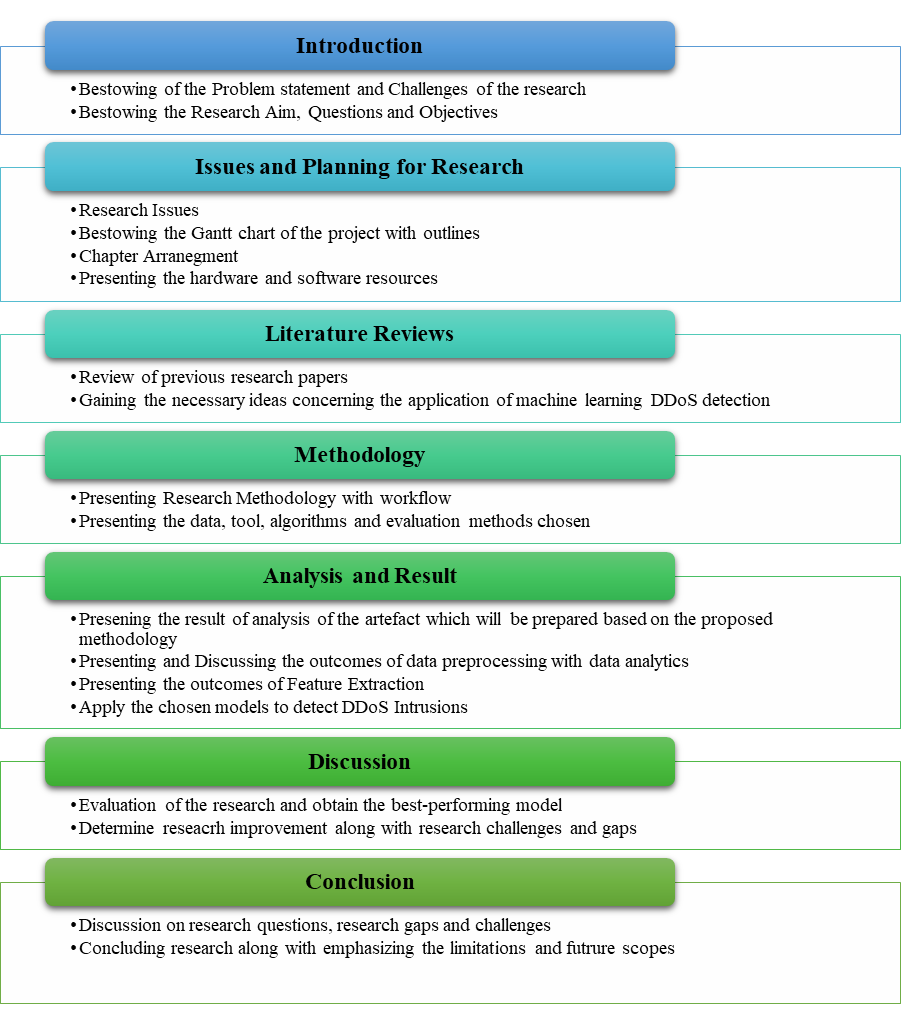


Figure 2 Arrangement of Chapters and Key Contexts

(Source: Self)

## Resources for Research

### Hardware Resources

The research requires the below-mentioned hardware without which the project cannot be commenced:

1. Computer or Laptop: For commencing the artefact and research
2. RAM: 16 GB or higher
3. ROM: 50 GB or higher
4. Processor: 5th Generation or higher
5. Camera: To attend the meeting

### Software Resources

The research requires the below-mentioned software without which the project cannot be commenced:

1. Anaconda: For the base of Python programming
2. Jupyter Notebook: To prove the IDE to Python for coding
3. Python: For coding (version 3.x)
4. Word Processor: For documentation
5. Text Processor: For other works

# Literature Review

## Data Preprocessing

### Outlier Detection in Data

(Prakobphol & Zhan, 2008) emphasized the value of anomaly detection within NIDS for providing a supplementary line of defence versus intrusions, particularly novel threats. This research presents a new cost-distribution-based outlier identification strategy as well as tests it using the KDD Cup1999 dataset. The results show that the suggested system is more effective than the current ones in identifying assaults that have a low rate of false detection.

According to (Mohy-Eddine, et al., 2023), despite its usefulness, IIoT is more susceptible to security breaches than the IoT. If want to protect the network from malicious attacks,  need an IDS. This research introduces an IDS model that makes use of feature engineering as well as ML; specifically, it optimizes computing efficiency by integrating IF with PCC. Results on the Bot-IoT, as well as NF-UNSW-NB15-v2 collections, show that the model outperforms the state-of-the-art methods in terms of accuracy along with prediction time.

With a focus on statistical analysis along with ML as well as deep learning,  (Priya & Pradeep, 2021) delved deeply into the crucial function of IDS in cybersecurity. People are starting to take notice of DRL because of the interest it might generate in intrusion detection systems. This research presented the IOD-ODRL method, which combines the outlier removal capabilities of iForest with the intrusion detection capabilities of Q-learning. Extensive simulations employing datasets of benchmark show that using SPO improves detection effectiveness by optimizing learning rates.

As per (Sahu, et al., 2021), outlier identification is crucial since outliers make it harder to train models with as few errors as possible when it comes to data preparation along with modelling. In a wide range of fields, different outlier detection approaches tackle different problems. In order to find outliers in intrusion datasets, this paper suggests an ensemble-based strategy that combines the SVM of one class along with the isolation forest, as well as the factor of local outlier. By using outlier indices, majority voting improves classification performance as well as drastically decreases training mistakes when compared with supervised approaches.

### Feature Selection Techniques

Optimal characteristics derived from the Canadian Institute for Cybersecurity ID set of information are selected by (Matsa, et al., 2021) using RFE inside a wrapper approach. A technique that combines RNN, as well as CNN architectures, is used to assess five characteristics that have been selected for deep learning. The results show that when identifying distributed DOS attacks across networks of software-defined, the accuracy represents 98.95% along with the precision stands at 99.45%, the recall is around 99.06% as well as the rate of false-positive represents 0.0112%, as well as the specificity represents 98.87%.

According to (Tonni & Mazumder, 2023), attackers take advantage of these weaknesses, endangering the very fabric of the network. Because of the massive amount of data flowing in, it is difficult to quickly detect breaches. Network security cannot be adequately assured without IDS. Nevertheless, efficiency may be hampered by their intricacy. With the use of Random Forest categorization, this research presents a two-layer selection of features strategy for streamlining IDS. It is evaluated on the information set of CSE-CIC-IDS-2018  as well as shows that it improves accuracy along with efficiency.

Using a simplistic feature selection technique, (Chanu, et al., 2023) investigated the difficulty of identifying DDoS assaults. In response, develop and test a new hybrid selection of features approach that relies on voting as well as compares it to three existing correlation techniques. This technique improves classification accuracy while simultaneously reducing feature size along with eliminating redundancy. A classifier that combines a multilayer perceptron utilizing MLP-GA achieves a 98.8% accuracy rate alongside a rate of false positives of just 0.6%, allowing for early identification.

As stated by (Zhou, et al., 2022), internet servers' availability is always at risk from DDoS assaults. To overcome this obstacle, it is necessary to identify these assaults and then correctly identify and block harmful traffic. Nevertheless, existing classification approaches have challenges when it comes to accurately differentiating between malicious as well as benign flows. The SAFE method for classifying DDoS attack flows at the network layer is presented in this paper. Extensive experimental results show that SAFE achieves better accuracy as well as efficiency than current approaches by refining features along with tweaking thresholds, as well as using weighted linear classification.

According to (Azmi, et al., 2021), the server along with the system has become increasingly vulnerable to DDoS assaults in recent years. DDoS attacks continue to occur, despite much previous research, calling for new ways to identify them. This research makes use of Information Gain along with Data Reduction techniques to pick features retrieved from the information set of  UNSW-NB 15. This strategy effectively improves network security by using classifiers like ANN along with Naïve Bayes, plus algorithms of Decision Table, which show better accuracy in differentiating attacks against regular traffic.

There has been an uptick in cyber-attacks like DDoS due to the increased demand for internet-based services caused by recent advances in communication as well as AI (Saha, et al., 2022). Using high-quality training data, DoS detection techniques based on AI have the potential to accurately identify threats with fewer false alarms. It is still not possible to determine an all-encompassing set of features that are optimal for ML, along with DL, as well as RL, despite previous attempts to do so. After analyzing fifteen different FS approaches, this research concludes that it is the best as well as most appropriate one.

## Intrusion Detection

### Application of Machine Learning

As stated by (Chia-Ying & Wei-Yang, 2009), The growing number of people using the Internet as well as the dangers of network assaults are both discussed in this overview of the literature. It emphasizes ID as a critical area of study for network security, with the goal of detecting as well as preventing illegal access. There hasn't been a thorough review article on using ML methods for intrusion detection, notwithstanding the abundance of such methods. It contrasts classifier designs the datasets, reviews 55 papers spanning 2000–2007, including talks about the results, limits, as well as potential future research path.

The increasing prevalence of Internet use as well as the dangers posed by network assaults are the subjects of this research (Lansky, et al., 2021). Finding along with stopping intruders is the goal of intrusion detection, the critical field of study for network security. It would be helpful to have a full review article on employing ML methods towards intrusion detection since there are already several ways. Comparing classifier designs as well as datasets, it also examines accomplishments along with limits, as well as future research goals after reviewing 55 papers spanning 2000 to 2007.

(Baklizi, et al., 2024) investigated ML techniques, including deep learning, for combating growing cyber risks in response to recent developments in network security for computers. An in-depth examination and categorization of IDS powered by deep learning are provided in this study. Using the methodologies used, it classifies schemes, describes architecture for IDS, as well as highlights the effectiveness of networks of deep learning towards intrusion identification. A comprehensive evaluation of the examined IDS frameworks as well as recommendations for further study constitute the review's last section.

As per (Kumari, et al., 2023), firms become more susceptible to attacks and exploitation. Thus, secure communication becomes even more important in the network. Despite the importance of IDS to network security, current solutions often fail to meet expectations. To improve the effectiveness of IDS, GA may be used. Conventional intrusion detection systems are not flexible enough to deal with the increasing number of cyber threats. AI classifiers as well as algorithms may improve intrusion detection while decreasing the number of false positives. In order to improve IDS capabilities along with guaranteeing effective threat detection, research analysis suggests using ML algorithms.

(Saranya, et al., 2020) reviewed the ever-changing technical world as well as the security issues that come along with it, with an emphasis on the growing cyber dangers caused by these developments. As a result, it stresses how important IDS are for improving data protection, especially when utilized alongside ML algorithms. Exploring various applications including fog computing along with IoT, as well as 5G networks, it examines how well ML algorithms including LDA along with CART, as well as Random Forest classify intrusions. It compares their results to those of recent studies using the dataset of KDD-CUP.

According to (Albulayhi, et al., 2022), data flow as well as complexity have been amplified by the proliferation of IoT. As important as IDSs are in protecting networks from cyberattacks, the diversity of the Internet of Things (IoT) presents significant challenges. Regarding anomaly-based intrusion detection systems, this research presents a new feature selection strategy. It finds the best features by using set theory as well as entropy-based methods. The model achieves an outstanding 99.98% accuracy in classification along with surpassing previous as tested using IoTID20 along with datasets of NSL-KDD.

There has been a dramatic increase in network security vulnerabilities due to the exponential growth of the Internet as well as the traffic it carries (Ahmed, et al., 2022). In order to steal data as well as disrupt resources, cybercriminals take advantage of security holes in networks. While keeping an eye out for suspicious activity across network traffic, NIDS use supervised as well as unsupervised ML. However, the efficiency of classic NIDS is being tested by new threats. This paper argues for updated NIDS training with recent datasets and suggests a classification framework utilizing five ML algorithms. Running on the dataset of UNSW-NB15, it shows encouraging outcomes, especially the Random Forest reaching 95.1% accuracy after applying SMOTE.

Due to the extensive computer networks and the ubiquitous IoT, cybersecurity worries have recently escalated (Sarker, et al., 2020). The need for effective IDS is driven by the critical nature of the detection of cyber threats. An intrusion detection system that uses ML, called "IntruDTree," is proposed in this study. It achieves excellent prediction accuracy while lowering computing complexity by prioritizing the significance of security features as well as using the tree-based framework. Its effectiveness is shown by experimental validation, which outperforms more conventional approaches such as logistic regression as well as naive Bayes.

### Application of Deep Learning

Robust security measures are required when control systems of industries are integrated with the external Internet, as discussed by (Kheddara, et al., 2023). It takes a look at the latest developments in AI methods, with a focus on IDS, especially those that use DTL. This research provides insights into the techniques and assessment criteria of IDS-only along with  DTL-only, as well as IDSi.e., DTL-based articles that were published after 2015. With its comprehensive coverage of datasets along with methodologies, as well as improvements, it is an invaluable resource for scholars aiming to comprehend DTL approaches with IDS across various networks.

According to (Hao, et al., 2020), network security is becoming more and more important due to the increasing number of cyber threats. To counter these dangers, ML algorithms such as SVM as well as ANN play a crucial role in IDS. Accuracy along with false positives as well as rates of false negatives, including other metrics of evaluation measure the performance of intrusion detection systems. This work suggests using ANN based on the sequential classifier method to handle their interconnectedness. It improves total detection efficiency, especially with four subclassifiers, through optimizing subclassifiers to obtain fewer false negatives not having reducing false positives.

Everyday activities rely on the pervasive integration as well as interoperability of computer systems, which also reveal risks that humans cannot manage (Ashiku & Dagli, 2021). In the face of ever-changing cyber dangers, cyber-security techniques are essential for safe communication. The authors of this research stress the need to build adaptive IDS using architectures based on deep learning. To identify both old as well as new network intrusions, DNNs are essential since they allow for learning-capable, versatile intrusion detection systems. Utilizing the dataset of UNSW-NB15, this article shows that these models may effectively reduce risks.

As per (Gamage & Samarabandu, 2020), there has been a lot of recent talk about how models based on deep learning may be useful for cybersecurity purposes in detecting intrusions into networks. Even though there are surveys that provide some information on this field, it a lack of data that objectively compares various models for deep learning. This work fills that need by providing a taxonomy of these models, reviewing relevant literature, along with comparing as well as contrasting them. The results show that deep neural networks i.e.,  feed-forward are the best model across all datasets, which could lead to some interesting new directions for study.

## DDoS Detection

### Application of Machine Learning

As stated by (Tambe, et al., 2023), DDoS attacks have become more common in the last decade, and this study shows how common they are and how important it is to have good defences. NIDS that can detect both old and new forms of distributed denial of service attacks are highly recommended. The suggested NIDS improves the efficiency of intrusion detection by using ensemble models that incorporate heterogeneous classifiers. It shows adaptation to new attack patterns by achieving an impressive 99.2% of the DDoS attack recognition rate via thorough analysis as well as validation.

As per (Vattikuti, et al., 2021), cybercrimes are becoming more common, thus it's more important than ever to strengthen security procedures to prevent breaches that might compromise the CIA. One of the most significant dangers is DDoS assault, which may cause major interruptions in service as well as, in turn, damage to your business's finances along with reputation. Mechanisms for timely as well as accurate detection including reaction are essential. In order to effectively identify and mitigate threats with little delay as well as false positives, this paper proposes an approach that uses ML ensemble algorithms in conjunction with anomaly detection methods.

According to (Bhargava R., et al., 2022), the goal of DoS as well as DDoS assaults is to interfere with legitimate users' access to internet services. In order to cripple servers with bogus requests as well as prevent legitimate ones, attackers use harmful websites as well as email attachments to distribute malware, which in turn causes economic losses. Research suggests a method for detecting prototype DDoS assaults employing a model with supervised learning, most especially SVM. Out of all the ML methods tested, Fuzzy clustering of c-means  emerged as the clear winner when it came to attacking detection.

A growing number of DDoS assaults are endangering critical internet services and putting network security at risk (Sujatha, et al., 2022). These assaults are becoming more complex and are causing problems for the system by flooding it with fake requests, which is preventing services from reaching clients. Because of this spike, sophisticated detection methods are required, but the use of ML is being considered as a potential solution. Considering the exponential growth of distributed DDoS assaults, it is critical to safeguard vital industries such as healthcare along with banking, as well as government services.

Innovations like 5G along with the IOT have expedited the spread of the internet, which in turn has increased its use (Özçam, et al., 2021). Nevertheless, cyber risks are intensified by this growth. There are substantial dangers associated with DDoS attacks, especially attacks of TCP-Flood. The focus of this research is on detecting these types of assaults in real time using various ML approaches. In the face of growing cyber vulnerabilities, the challenge is to quickly detect as well as lessen the impact of these attacks.

As stated by (Ashodia & Makadiya, 2022), SDN is a paradigm shift away from conventional networking that provides efficient as well as economical resource management via centralization as well as adaptability. Providers like the decreased CAPEX along with OPEX, while administrators appreciate the programmability as well as remote accessibility. However, security issues, such as DDoS attacks, arise from the centralized as well as complicated design of SDN. In order to tackle this, suggest using machine learning methods like Random Forest along with Decision Trees, as well as Naïve Bayes, which provide higher rates of accuracy along with decisions when it comes to identifying harmful traffic.

According to (Arya, et al., 2023), DDoS attacks have emerged as a significant danger, and the everyday security issue is only becoming worse due to the boom of devices connected to the internet. Their advanced nature makes them difficult to detect, which in turn disrupts networks. About 13 million  DDoS assaults occurred in 2022, in addition, attackers were good at getting past conventional defences. Four million assaults continued for at least a day, with 25% lasting more than twelve. The importance of adaptable DDoS solutions has been demonstrated by this. Here, provide an ensemble ML method that outperforms individual methods.

Cybersecurity concerns, with a particular emphasis on DDoS assaults, are the main focus of the research (Visetbunditkun & Srichavengsup, 2022). It presents RFE-algorithm-enabled ensemble Mlmodels to improve DDoS detection effectiveness. If compare the suggested approach to other methods, especially those that use neural networks, can see that it outperforms them in terms of accuracy along with precision as well as testing time, including CPU use. The results highlight the possibility of ensemble machine learning to optimize resource consumption while reducing DDoS threats.

As the internet continues to grow, so do the frequency and severity of DDoS assaults, making strong defences an absolute need (Prathiksha S. & Meenakshi Sundaram, 2023). In order to differentiate between DDoS as well as non-DDoS traffic, this study combines Decision Tree along with Naive Bayes algorithms. Decision Trees have been great with complicated datasets, whereas Naive Bayes provides interpretability through recursive partitioning.  Research suggests a hybrid strategy and uses a lot of data to see how well it works. It improves accuracy by highlighting the interaction among normal as well as attack patterns via feature engineering along with preprocessing.

(Naing & Thwel, 2023) emphasized that DDOS attacks continue to be a serious concern for the security and privacy of computers as well as systems. A huge obstacle persists in the form of DDOS, even though security methods have progressed. This research uses a publicly available DDOS assault dataset to compare and contrast different ML techniques. According to outcomes, Logistic Regression remains the best classifier for revealing patterns for network traffic. Effective mitigation of DDOS vulnerabilities requires enhanced preventive measures.

According to (Meriaux, et al., 2022), effective detection techniques are required because cyber networks are becoming more interdependent upon the grids of physical power, making them more susceptible to cyber-attacks. This research employs a range of ML methods as well as datasets to evaluate their effectiveness in detecting Distributed Denial of Service (DDOS) attacks in smart grids. The datasets used are KDDCup'99 along with CICIDS'17. Evaluate three machine learning algorithms—Decision Tree along with Random Forest, as well as SVM—with a focus on their computational efficiency along with storage economy, as well as accuracy. The results show that ML can be used to secure smart grids.

### Application of Deep Learning

Detecting DDoS assaults in this age of rapidly developing digital technologies is very difficult because of the computing complexity involved (Reddy, et al., 2021). In order to identify DDoS attacks early on, this article suggests a neural network with a hybrid design that uses models based on deep learning. An improvement in the accuracy of separating malicious as well as benign traffic is achieved by the use of GBDT along with CNN methods to classify spatial as well as temporal information.

Despite the risk of one point of failure that comes with decoupling control as well as data planes, examines the benefits of SDN for deploying services (Wang & Liu, 2020). Using information entropy as well as deep learning, it suggests a method for detecting distributed denial of service attacks. The controller uses entropy to detect malicious communication, along with a CNN that differentiates between malicious and benign traffic. There is the promise of efficient DDoS detection across SDN setups, as experimental findings show an encouraging 98.98% of accuracy.

Using small sample sizes as an example, (He, et al., 2020) looked at the difficulties of employing deep learning towards DDoS attack identification. It suggests a new way of using deep learning via transfer for small-sample identification, which is an innovative technique. The research shows a considerable improvement in detection performance after training as well as testing many neural networks. Results from experiments with the 8LANN network provide a 20.8% improvement, proving that deep learning via transfer is effective in reducing performance degradation towards DDoS detection.

A DDoS attack detection approach that utilizes a two-level approach of deep learning i.e., CNN-LSTM within an SDN network is proposed by (Li, et al., 2022). They primarily investigated the detection of DDoS attacks as well as defence within the 5G environment architecture of  SDN. It has the potential to both speed up the process of identifying as well as categorizing network traffic and substantially enhance accuracy for attack detection, allowing for the timely blocking of DDoS traffic that is being attacked as well as the subsequent maintenance of the availability of network service.

(Said & Askerzadea, 2023) drawn attention to the fact that conventional intrusion detection technologies aren't very good at spotting software-defined networks (SDNs). It highlights how susceptible SDN controllers are to intrusion. The suggested hybrid model incorporates an attention mechanism into CNN as well as BiLSTM to solve this problem. The model is able to accurately identify a wide range of intrusions and successfully captures patterns of network traffic. Its superior performance compared to Alexnet as well as CNN-LSTM demonstrates its promise for improving SDN security.

According to (Haider, et al., 2019), SDNs are a game-changer because they separate control from forwarding when it comes to satisfying the need for speedier networks. However, owing to centralized design, SDNs have security challenges, most notably distributed denial of service attacks. The effectiveness of countermeasures depends on the timely detection of such threats. To efficiently identify DDoS attacks in SDNs, this work presents a CNN ensemble that uses deep learning. The remarkable accuracy of 99.48% as well as minimum computing cost were shown during evaluation using the ISCX 2017 information set, resolving a critical issue regarding network security.

There are major security risks due to the interconnection of many intelligent sensors brought about by the fast growth of IoT (Jemal, et al., 2023). A possible answer might be derived from deep learning as well as its encouraging developments. This research explored how to use CNN to improve the security of the IoT. CNNs can identify and mitigate serious threats such as DOS as well as DDoS. An impressive 99.920% accuracy rate for detecting such assaults was shown experimentally using the dataset of Bot-IoT, demonstrating CNN's efficacy.

As the number of internet services grows, so does the likelihood of cyberattacks, particularly from more sophisticated DDoS attacks (Kumar, et al., 2023). This highlights the critical nature of early detection as well as data segregation in networks for defensive purposes. In order to identify DDoS threats, this research suggests using a based approach, which makes use of its strong feature extraction as well as selection capabilities. the model developed by LSTM outperforms conventional ML methods as well as attains an impressive 98%  accuracy when trained over an information set of CICDDoS2019.

# Methodology

## Proposed Framework

The proposed framework for the detection of DDoS attacks has been presented below:

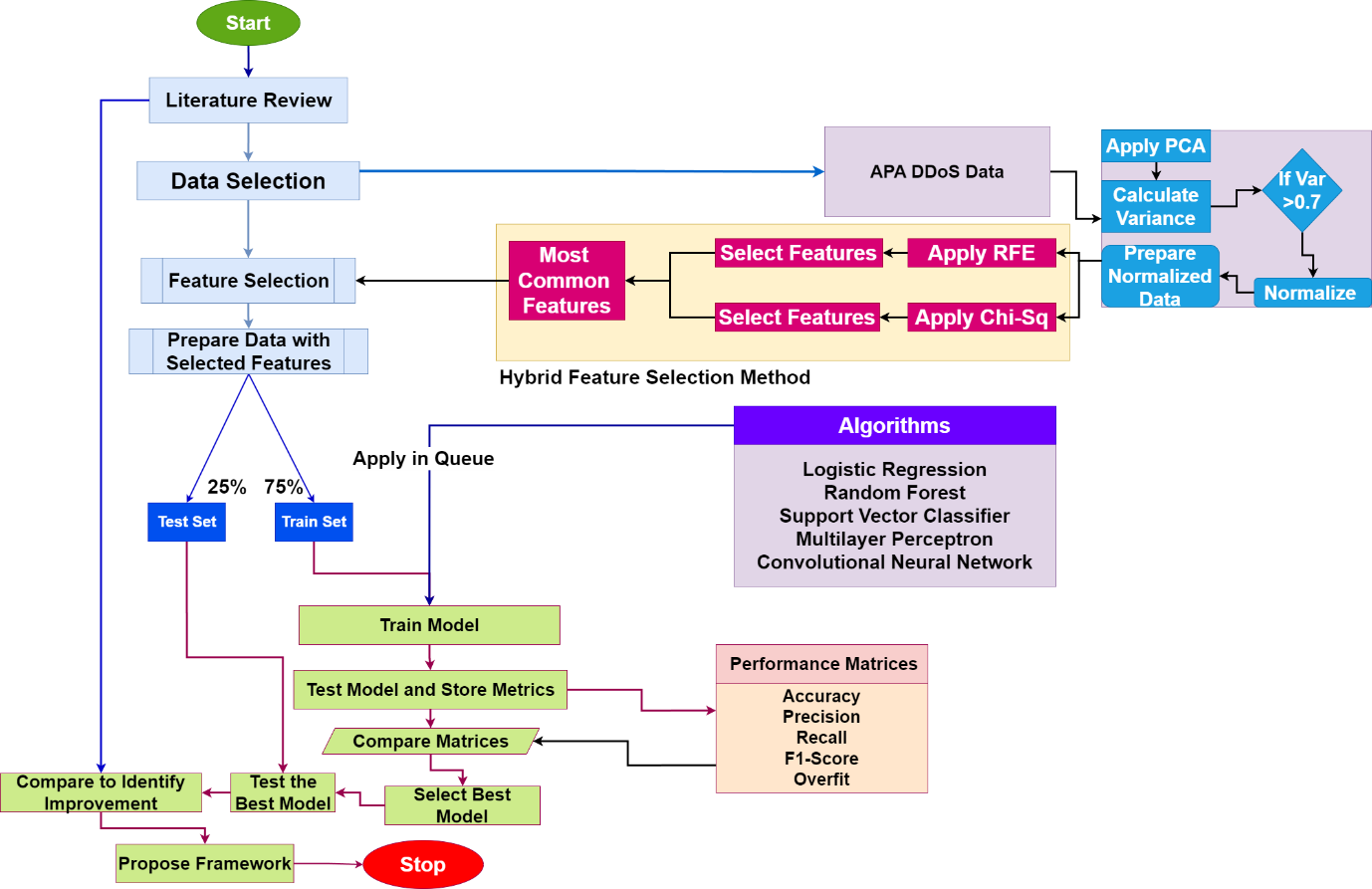


Figure 3 Proposed Framework

(Source: Self, using design tool, Draw.io)

## Data Collection

### Data Source

The dataset, employed in this project to detect DDoS attacks in the network will be collected from Kaggle (Kumbam, 2020). This dataset contains 151000 network transaction instances (records) with three types of network traffic namely Benign traffic, DDoS-ACK and DDoS-PSH-ACK. The last two types of network traffic belong to the DDoS category. The data snapshot is shown below:

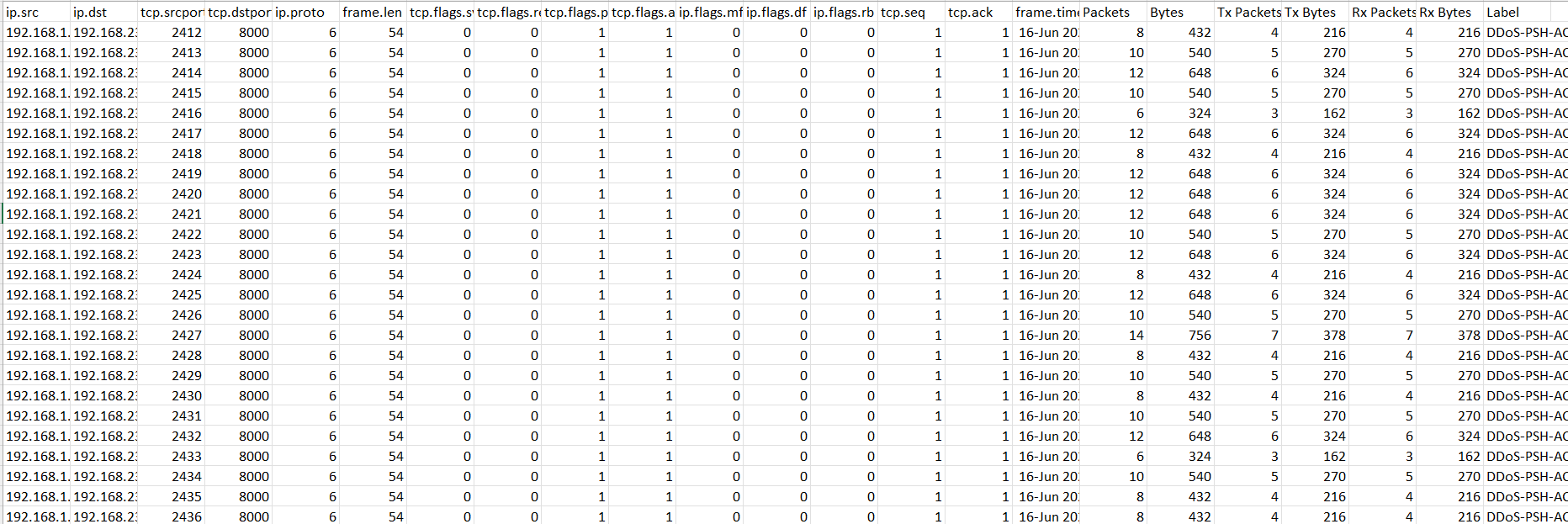


Figure 4 APA DDoS Dataset

(Source: Self, Data)

### Feature Details

The descriptions of the features are given in the table:

Table 1 Feature Details

(Source: Self, Data)

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Meaning** | **Feature** | **Meaning** |
| ip.src | Source IP for the network transactions | ip.flags.rb | IP Flag type (RB) |
| ip.dst | The destination IP address for the network transactions | tcp.seq | TCP sequence of the network packet |
| tcp.srcport | Source port from where the packet has been transacted | tcp.ack | Acknoledgement of the TCP |
| tcp.dstport | The destination port where the packet has been transferred | frame.time | Duration of the frame transmission |
| ip.proto | Protocol types of the IP | Packets | No. of packets transacted |
| frame.len | Length of the frame for the network packet | Bytes | Total byte transferred |
| tcp.flags.syn | TCP Flag type (SYN) | Tx Packets | No. of packets transacted while transmission |
| tcp.flags.reset | TCP Flag type (RESET) | Tx Bytes | Total bytes transferred while transmission |
| tcp.flags.push | TCP Flag type (PUSH) | Rx Packets | No. of packets received |
| tcp.flags.ack | TCP Flag type (ACK) | Rx Bytes | Total byte received |
| ip.flags.mf | IP Flag type (MF) | Label | Target feature with three classes namely Benign, DDoS-ACK and DDoS-PSH-ACK |
| ip.flags.df | IP Flag type (DF) |

### Data Details

The data contains three types of network packets as mentioned in the Lable feature. The distribution of the classes in the data is shown below:

Table 2 Data Classes

(Source: Self, Data)

|  |  |
| --- | --- |
| **Class** | **Distribution** |
| Benign | 75600 |
| DDoS-ACK | 37800 |
| DDoS-PSH-ACK | 37800 |

In this entire transition of packets, several source IPs have been attached from where the packets have been transferred to a single destination with the IP address 192.168.23.2. The number of packets transferred from the source IP addresses are listed below:

Table 3 Source IP and Transactions of Packets

(Source: Self, Data)

|  |  |
| --- | --- |
| **Source IP** | **Count of Packets** |
| 192.168.1.1 | 10800 |
| 192.168.10.1 | 10800 |
| 192.168.11.1 | 10800 |
| 192.168.13.1 | 10800 |
| 192.168.14.1 | 10800 |
| 192.168.16.1 | 10800 |
| 192.168.17.1 | 10800 |
| 192.168.19.1 | 10800 |
| 192.168.2.1 | 10800 |
| 192.168.20.1 | 10800 |
| 192.168.4.1 | 10800 |
| 192.168.5.1 | 10800 |
| 192.168.7.1 | 10800 |
| 192.168.8.1 | 10800 |

## Tool to be Used

Python programming language will be selected and the coding tool to prepare artefacts to detect network intrusions. When it comes to developing tools for detecting network intrusions, Python is a popular choice among professionals due to its versatility and ease of use. By leveraging the powerful libraries available in Python, cybersecurity experts can create effective detection mechanisms to safeguard networks from malicious activities. The process involves writing custom scripts that analyze network traffic patterns, identify anomalies, and raise alerts in real time to prevent potential security breaches (Hnamte & Hussain., 2023). This proactive approach to network security is essential in today's digital landscape where cyber threats are constantly evolving. By utilizing Python and specialized coding tools, cybersecurity professionals can stay ahead of potential intrusions and protect sensitive data from unauthorized access. The intrinsic reasons behind this choice are stated below:

1. One of the primary advantages of Python for data classification and machine learning is its extensive collection of libraries and frameworks. Libraries such as Scikit-learn, TensorFlow, and Keras provide robust support for implementing machine and deep learning algorithms with ease (Yuze, et al., 2018).
2. Python's simplicity and readability make it an ideal choice for rapid prototyping and experimentation in the field of cybersecurity (Matsa, et al., 2021). Its clear and concise syntax enables researchers and practitioners to quickly test different machine learning models, fine-tune parameters, and assess performance metrics.
3. Python's strong community support and active development ecosystem contribute to its advantages in the realm of cybersecurity. Developers and data scientists can leverage online forums, documentation, and tutorials to troubleshoot issues, seek advice, and stay updated on the latest advancements in machine learning for intrusion detection.
4. Python is an open-source coding language, meaning that its source code is freely available for anyone to use, modify, and distribute. This fosters a collaborative and supportive community of developers who contribute to the language's growth and improvement (Zhang, et al., 2023).
5. Python is an interpreted coding language, which means that code written in Python is executed line by line in real-time by the Python interpreter. This allows for quicker development and testing of code compared to compiled languages (Smith, 2016).
6. Python is known for being a platform-friendly programming language, meaning that it can run on various operating systems without requiring major modifications. This versatility makes Python a versatile language for developing applications across different platforms.

## Planned Algorithms and Evaluation Methods

### Algorithms

In this project, it has been planned to employ both machine learning state-of-the-art models and deep learning models to detect DDoS attacks. In this context, the choice of the algorithms has been done in two ways for the justification of choice. Those are:

1. Algorithms have been chosen from separate model families
2. Algorithms have been chosen from the literature review that the existing researchers have used for Intrusion and DDoS detection.

The planned algorithms are as follows:

Table 4 Planned Algorithms

(Source: Self, Data)

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm Name** | **Family** | **Associated Library** | **Previously Used By** |
| Logistic Regression | Linear Model | Scikit Learn | Lansky et al. (2021)  Albulayhi et al. (2022)  Sujatha et al. (2022) |
| Random Forest | Ensemble | Scikit Learn | Saranya et al. (2020)  Sarker et al. (2020)  Ashodia & Makadiya (2022) |
| Support Vector Classifier | Support Vector Machine | Scikit Learn | Ahmed et al. (2022)  Bhargava R. et al. (2022)  Sujatha et al. (2022)  Özçam et al. (2021) |
| Multilayer Perceptron | Neural Network | Scikit Learn | Hao et al. (2020)  He et al. (2020) |
| Convolutional Neural Network | Neural Network | Keras, TensorFlow | Reddy et al. (2021)  Wang & Liu (2020)  He et al. (2020)  Li et al. (2022)  Said & Askerzadea (2023)  Haider et al. (2019) |

### Evaluation Methods

The methods which have been chosen to evaluate the performances of the algorithms in detecting DDoS intrusions are as follows:

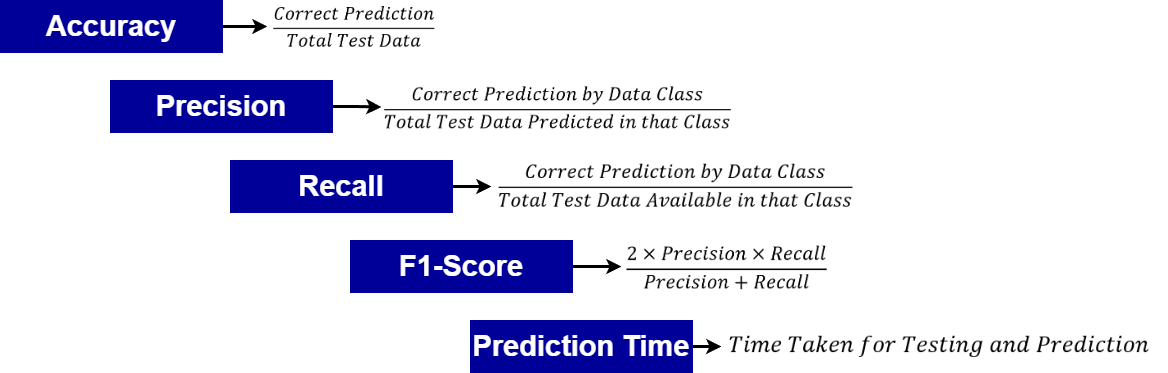


Figure 5 Methods Chosen to Evaluate Algorithm's Performances

(Source: Self, using design tool, Draw.io)

# Artefact Implementation and Result

## Preparing Artefact Environment

The environment of the artefact has been prepared by installing the Anaconda Navigator from where the Jupyter Notebook has been accessed. The data file has been stored in a directory and a notebook has been created from the same directory. The artefact for the research has been created in that notebook by using Python coding. The overall process is shown below:

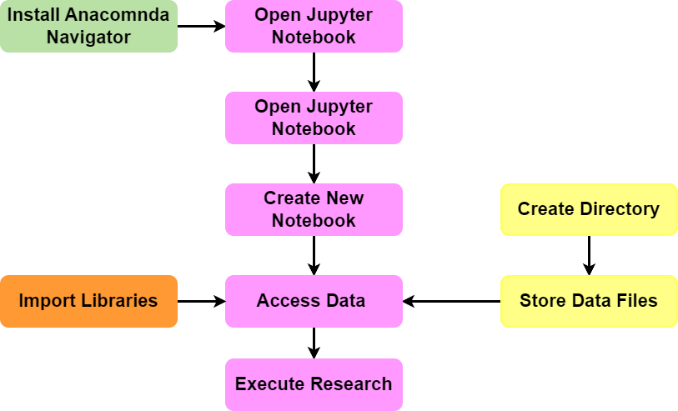


Figure 6 Artefact Environment Setup

(Source: Self, using the design tool, Draw.io)

## Processing Data

### Library Import

The first operation which has been done in the artefact is the library imports. Those libraries are essential in commencing data analytics, and the application of machine and deep learning models for DDoS detection. Hemncem the libraries which have been imported and the process is shown below:

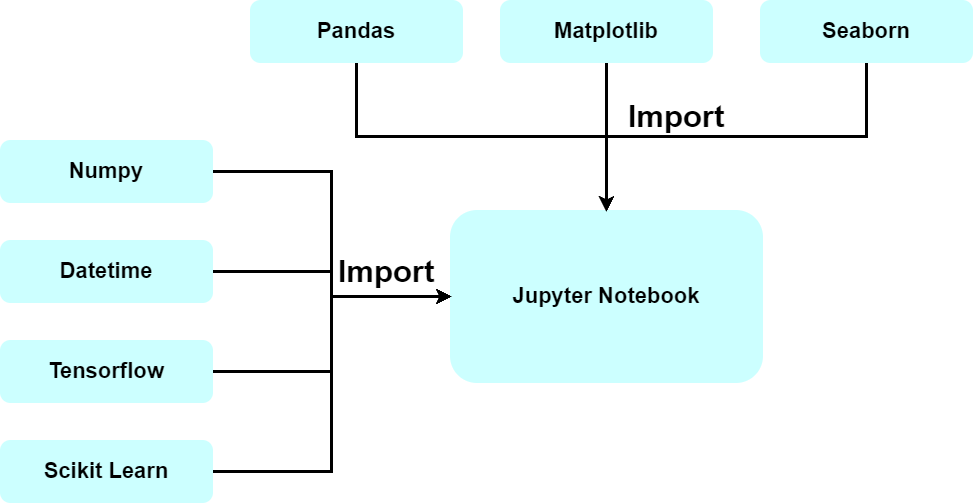


Figure 7 Process of Library Import

(Source: Self, using the design tool, Draw.io)

### Dataset Reading

The next operation of the artefact has been done by reading the dataset. After reading the data, the outcomes of data (data view) have been obtained in the Jupyter which is shown below:

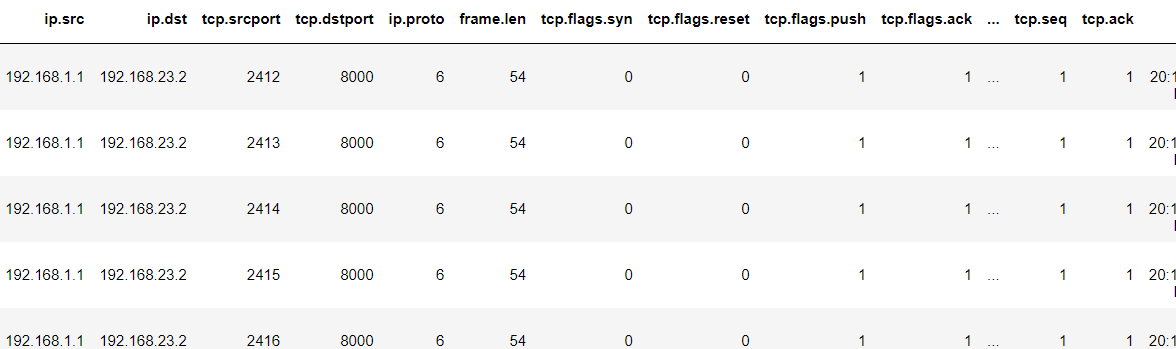


Figure 8 Data Outlook obtained in Jupyter Notebook

(Source: Self, Experiment in Jupyter Notebook)

### Feature Information

The features of the data have been investigated to extract necessary information. It has been found that the data contains 151200 network traffic records with 23 traffic features. Out of those 23 features, 19 features are numerical and the rest 4 features are categorical. The complete feature information is presented below:

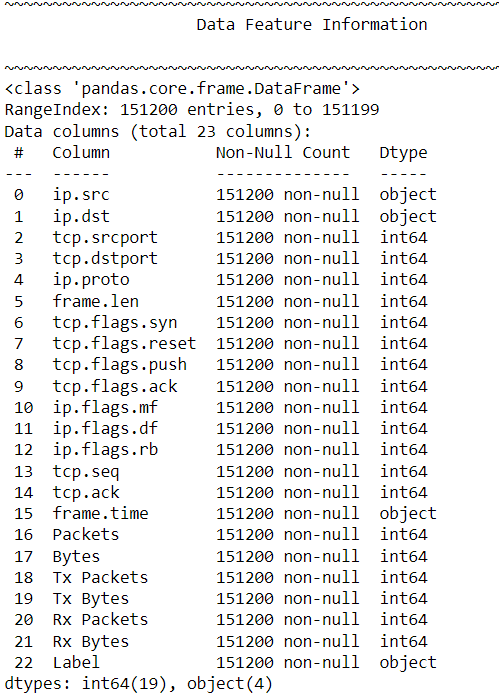


Figure 9 Information on Data Features

(Source: Self, Experiment in Jupyter Notebook)

### Missing Value

The check has been done on the missing values and it has been identified that the data does not contain any missing values. So, no data cleaning process is required to be done in the artefact.

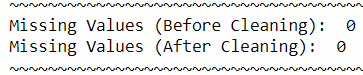


Figure 10 Missing Values Present in Data

(Source: Self, Experiment in Jupyter Notebook)

## Feature Preprocessing

### Encoding Categorical Features

The feature information has shown that the data contains categorical features. As the categorical features are ineligible for data analytics and machine learning, so, they need to be transformed into numerical values. Hence, the encoding has been done to convert the categorical features to numerical features. The outcome is shown below with the encoded data:

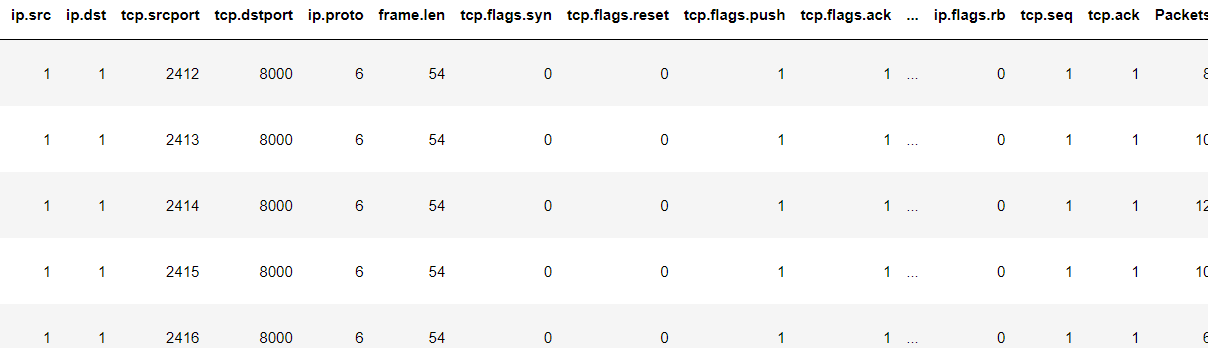


Figure 11 Encoded Data

(Source: Self, Experiment in Jupyter Notebook)

### Treating Feature Outliers

The outliers have been detected in the data and minimised. This operation has reduced the data noise. Hence, the steps taken to detect outliers and reduce those are described below:

1. Principal Component Analysis or PCA has been employed in the data to detect the variances in the PCA components. Those components are the dimensionality-reduced version of the data where the variance value can be achieved.
2. It has been observed that the data contains 99.89% variance or outliers which need to be reduced.
3. To reduce the variance value, the MinMaxScaler method has been employed for the data and the features have been normalized.
4. After normalization was done, the outliers were checked again and the variance value was found by 66%. So, 33% of outliers have been removed from data.

The overall process is presented below:

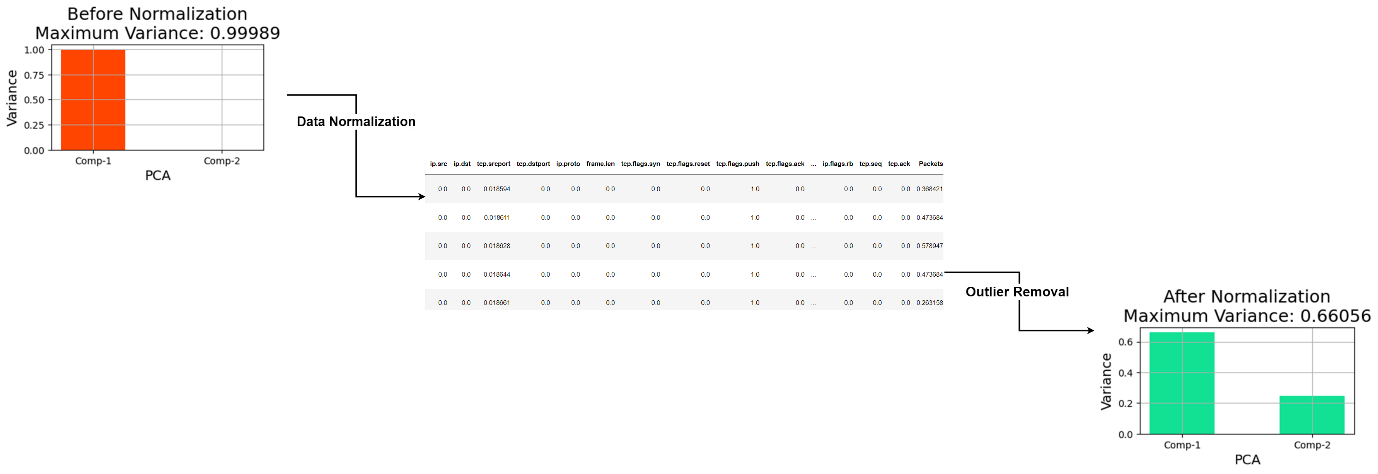


Figure 12 Outlier Detection and Reduction Process

(Source: Self, Experiment in Jupyter Notebook and Drawio)

### Selection of Optimum Features

#### Hybrid Approach

Feature selection is one of the most important feature preprocessing techniques using which the important data features can be selected and set to the data predictor for classification. There are various feature selection techniques available but using individual techniques, there will be disadvantages. If any single method is used for selecting features, the selected features cannot be judged whether those all are equally important. This issue can be resolved by employing more than one feature selection method and combining those to detect optimum features using the following technique:

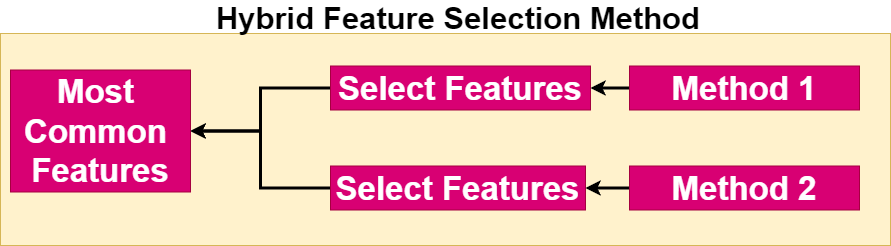


Figure 13 Hybrid Feature Selection Technique

(Source: Self, using design tool, Draw.io)

#### Process

Two features of selection methods have been applied which are Recursive Feature Elimination and Chi-Squared and the individual features have been selected. RFE works by recursively removing the least important features, while the Chi-Squared test evaluates the independence of each feature relative to the target variable. By integrating these two methods, we were able to identify and retain the most statistically significant features, enhancing the predictive power and efficiency of the model. After that, the common features which have been selected by both have been taken as the final feature set. The parameters used by those methods are listed below:

Table 5 Methods Applied for Hybrid Approach

(Source: Self, Experiment in Jupyter Notebook and Drawio)

|  |  |  |  |
| --- | --- | --- | --- |
| **Applied Method** | **Applied Estimator** | **Features to Select** | **Number of Features Selected** |
| Recursive Feature Elimination | LogisticRegression | 70% | 15 |
| SelectKBest | Chi2 (Chi-Squared) | 70% | 15 |

The hybrid approach using these two methods is shown below:

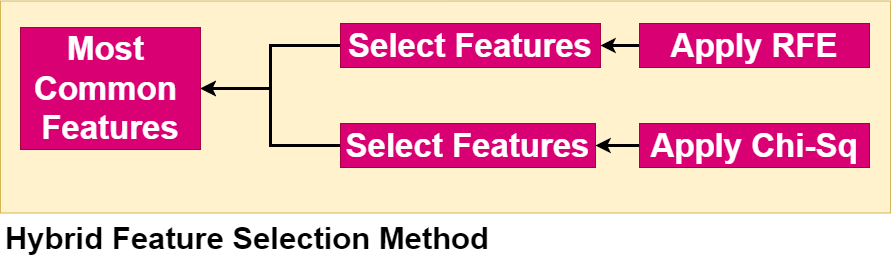


Figure 14 Final Hybrid Approach for Feature Selection

(Source: Self, using the design tool, Draw.io)

#### Outcome

Finally, the outcome has been achieved by obtaining the common features selected by both methods. The finally chosen features are shown below:

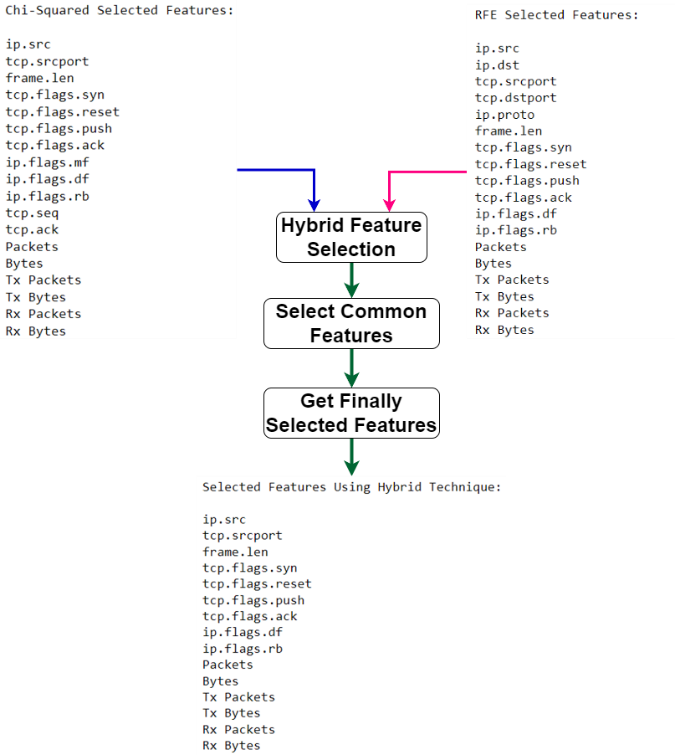


Figure 15 Final Features Selected

(Source: Self, using the design tool, Draw.io and Jupyter Notebook)

## Model Preparation

### Preparing Machine Learning model

The chosen machine learning models have been prepared with the necessary parameters which are listed below:

Table 6 Assigned Parameters to Machine Learning Models

(Source: Self, Experiment in Jupyter Notebook)

|  |  |
| --- | --- |
| **Model** | **Parameters** |
| Random Forest | n\_estimators=10, criterion='entropy', max\_depth=11, min\_weight\_fraction\_leaf=0.45,max\_features='log2' |
| Logistic Regression | tol=0.06, C=0.01,max\_iter=2,solver='liblinear',fit\_intercept=False |
| MLP Classifier | hidden\_layer\_sizes=(2,1,), learning\_rate\_init  power\_t=0.7, max\_fun=3, max\_iter=4 |
| Support Vector Classifier | C=0.2, kernel='sigmoid', degree=3,tol=0.01,max\_iter=20 |

By employing the parameters in the models, the queue has been prepared which is shown below:

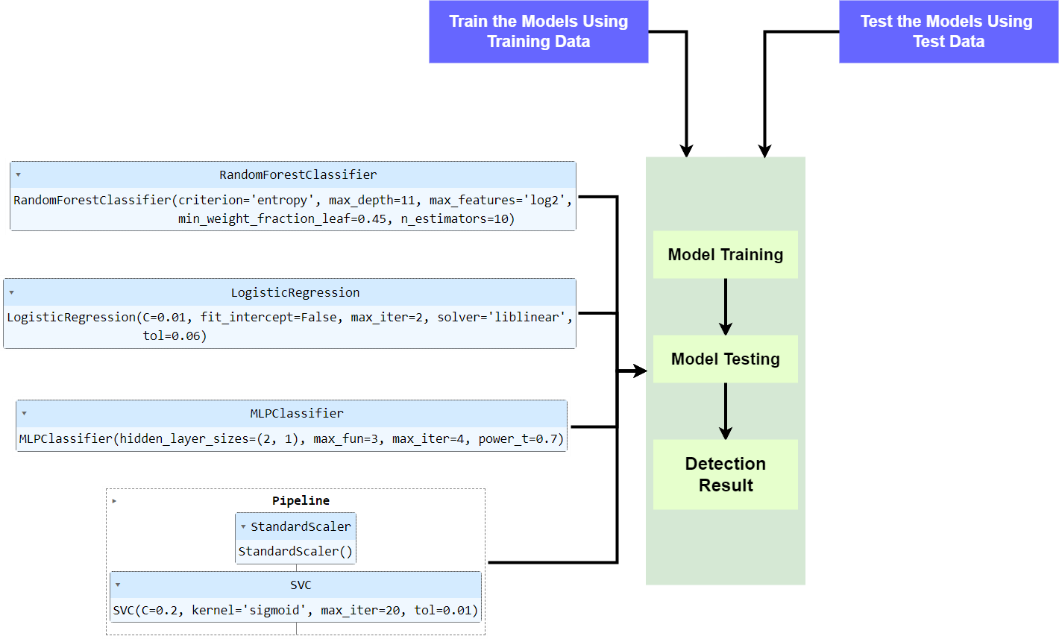


Figure 16 Queue for Machine Learning Models

(Source: Self, using the design tool, Draw.io and Jupyter Notebook)

### Preparing CNN Model

#### Assigning Layers

The Convolutional Neural Network has been prepared by assigning the necessary layers and parameters. The layer information is listed below:

Table 7 CNN Layer Parameters

(Source: Self, Experiment Jupyter Notebook)

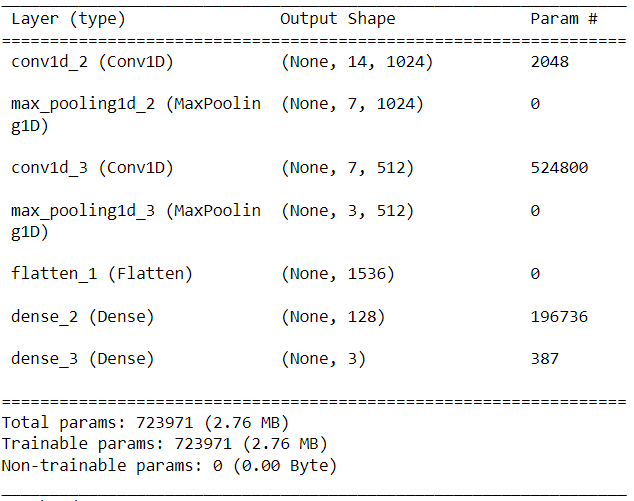
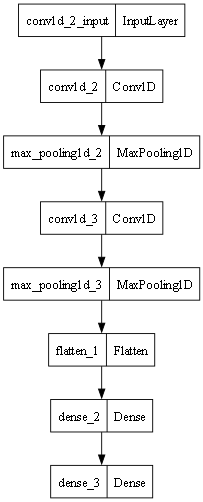
|  |  |  |  |
| --- | --- | --- | --- |
| **Layer Name** | **Parameters** | | **Description** |
| Conv1D | Filter | 1024 | The first layer is a 1D convolutional layer with 1024 filters, a kernel size of 1, and ReLU activation. This layer processes input data shaped (14, 1), which means it expects sequences of length 14 with 1 feature. |
| Kernel | 1 |
| Activation | ReLu |
| Input Shape | 14,1 |
| MaxPooling1D | Pool Size | 2 | Following the convolution, a max pooling layer reduces the dimensionality by taking the maximum value over a window of size 2, effectively downsampling the output. |
| Conv1D | Filter | 1024 | Another convolutional layer with 512 filters and ReLU activation follows, further extracting features from the pooled output. |
| Kernel | 1 |
| Activation | ReLu |
| MaxPooling1D | Pool Size | 2 | Another max pooling layer again downsamples the feature maps. |
| Flatten | Default | | This layer flattens the pooled feature maps into a one-dimensional vector, preparing it for the dense layers. |
| Dense | Neurones | 128 | The first dense layer has 128 units with ReLU activation. |
| Activation | ReLu |
| Output Dense | Neurones | 3 | The final dense layer has 3 units with a sigmoid activation function, used for multi-class classification for the present data. |
| Activation | Sigmoid |

This model will be compiled using the following parameters:

* Optimizer Function='adam'
* Loss Function='sparse\_categorical\_crossentropy'
* Metrics Function:'accuracy'

#### Model Details

After preparing the CNN model, the summary and the model structure have been observed which is shown below:

**(a) CNN Model Summary (b) CNN Model Structure**

Figure 17 Model Summary and Structure

(Source: Self, Experiment in Jupyter Notebook)

### Model Execution Queue

Finally, all the models will be employed to the date following the sequence of operations which are training and testing with the training and test data to detect DDoS intrusions. The final model queue is shown below:

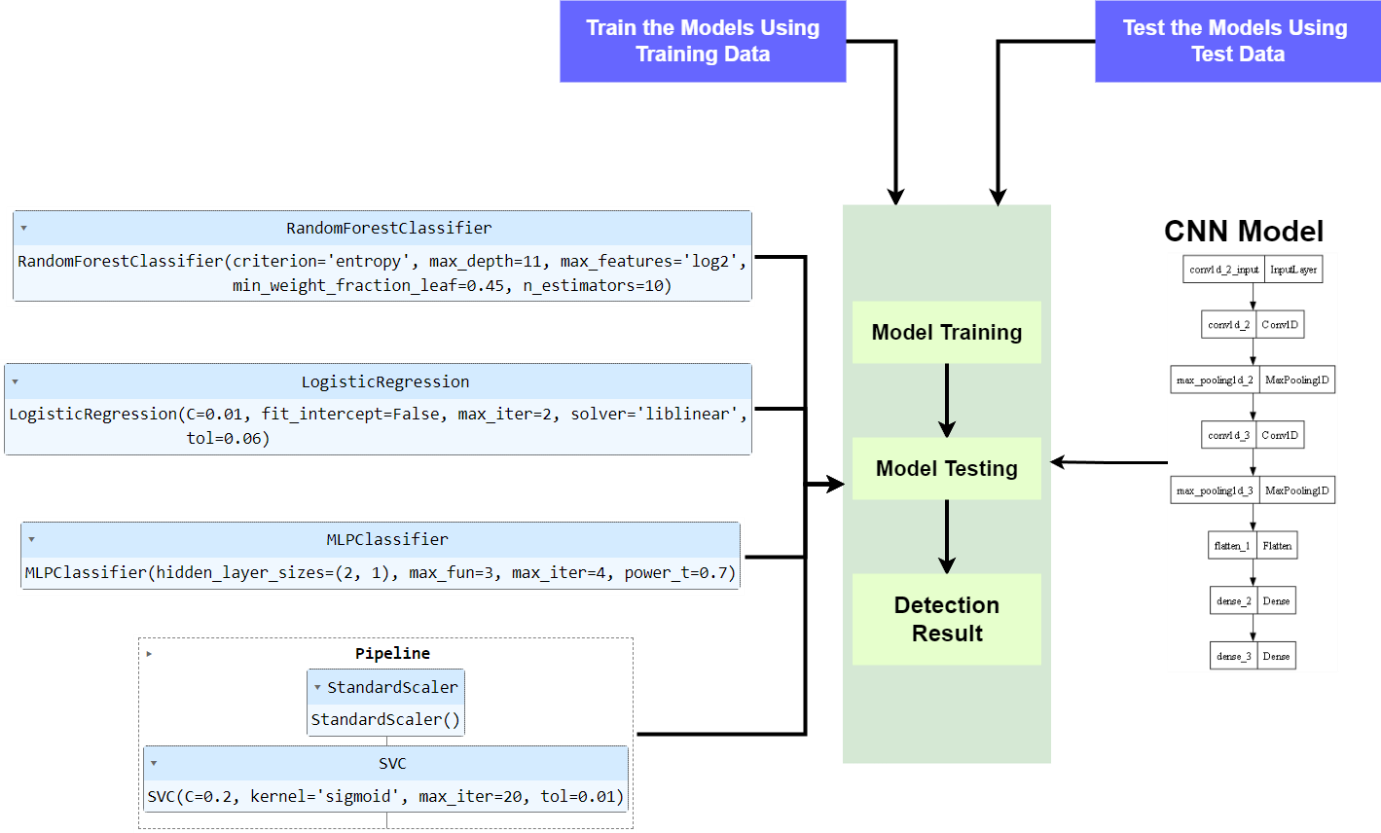


Figure 18 Model Execution Queue

(Source: Self, using the design tool, Draw.io and Jupyter Notebook)

## Data Preparation

The dataset has been split into two parts namely training and testing where the training data contains 75% of the overall instances and the test data contains 25% instances. The actual class distribution along with the distribution of classes in the training and test parts are listed below:

Table 8 Data Split and Distribution of Class Instances

(Source: Self, Experiment in Jupyter Notebook)

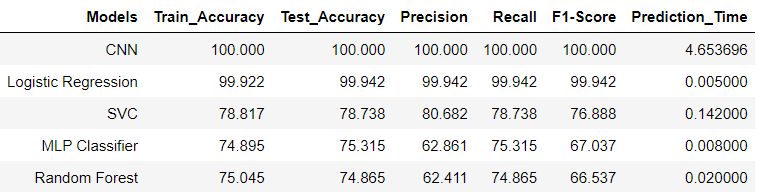
|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Actual Instances** | **Training Instances** | **Test Instances** |
| Benign | 75600 | 56632 | 18968 |
| DDoS-PSH-ACK | 37800 | 28469 | 9331 |
| DDoS-ACK | 37800 | 28299 | 9501 |

## Result

Finally, all the models have been applied to the chosen dataset to detect DDoS intrusions. The result of detection is shown below:

Table 9 Detection Result

(Source: Self, Experiment in Jupyter Notebook)



The result shows that CNN has gained the highest accuracy in detecting DDoS with 100% test accuracy by outperforming the highest seen accuracy of the machine learning category for Logistic Regression with 99.94% accuracy.

# Discussion and Conclusion

## Discussion

### Evaluation of Research Performances

The algorithms have been employed in the data to detect DDoS intrusions. The results have been obtained for DDoS intrusion detection and presented in the previous chapter in section 5.6. The overall evaluation of the research will be done and discussed in this section.

#### Model Performances

The performance of the models has been compared in terms of classification metrics and presented below in the form of bar charts:

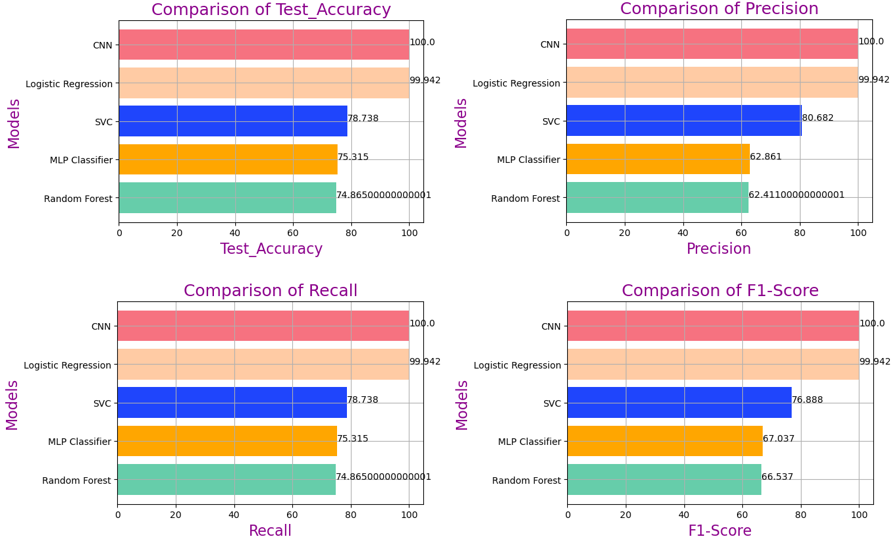


Figure 19 Model Performance Evaluation

(Source: Self, Experiment in Jupyter Notebook)

The comparison shows that the accuracy, precision, recall and f1-scores are highest for the CNN model compared to the other models applied.

#### Best-Performing Model

In the overall applications of the models, it has been seen that the CNN model has gained 100% accuracy in detecting DDoS intrusions. This accuracy outperformed the other chosen models showing the superiority. So, the CNN model can be referred to as the best-performing model to detect DDoS intrusions. The model structure of CNN is shown below:

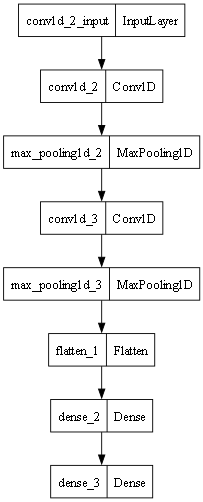


Figure 20 Best-Performing Model

(Source: Self, Experiment in Jupyter Notebook)

The results of the Best-Performing Model are presented below:

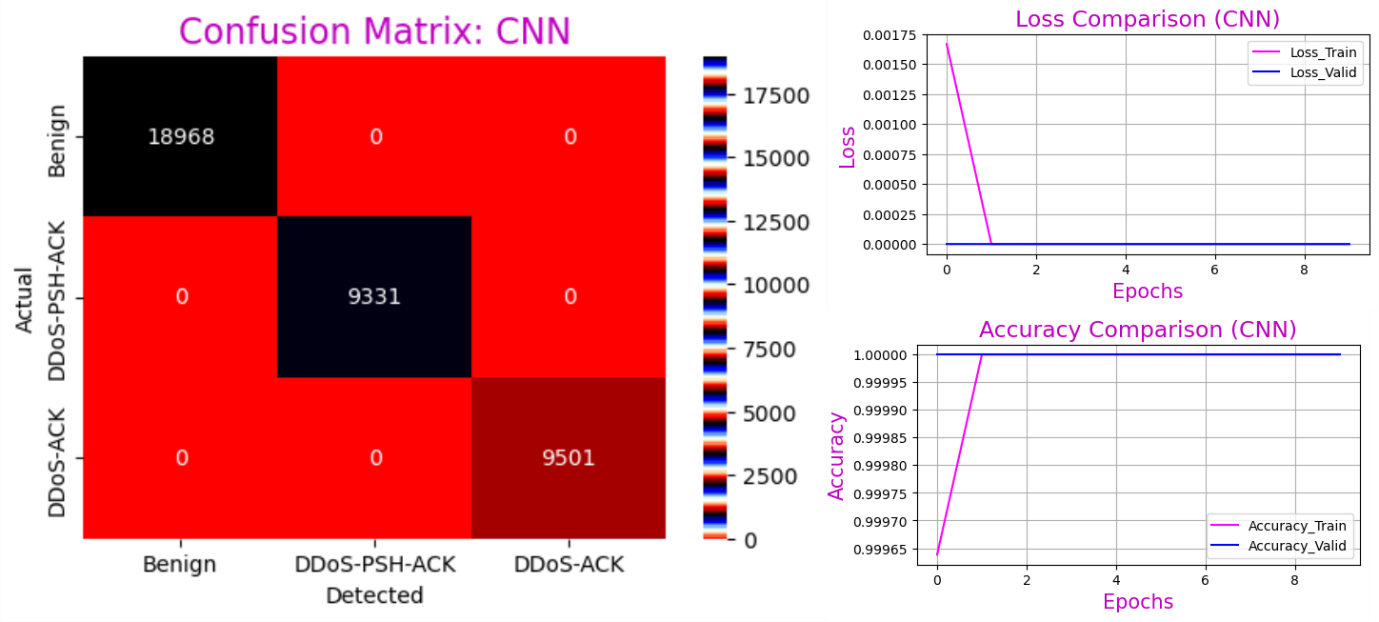


Figure 21 Results of the Best-Performing Model

(Source: Self, Experiment in Jupyter Notebook)

From the result, the following outcomes can be stated to justify the 100% accuracy of CNN:

* It has detected 18968 Benign out of the total 18968 test classes with 0 classifications.
* It has detected 9331 DDoS-PSH-ACK out of the total 9331 test classes with 0 classifications.
* It has detected 9501 DDoS-ACK out of the total 9501 test classes with 0 classifications.

#### Improvement in the Present Approach

The current research shows a significant improvement in the accuracy of detecting intrusions and DDoS attacks using a Convolutional Neural Network (CNN) model. When compared to prior studies, the CNN model in this research outperforms previous methods, achieving higher accuracy rates. This advancement is attributed to the application of a hybrid feature selection approach in conjunction with the CNN model. Therefore, it can be concluded that this research offers a more effective solution for DDoS attack detection. The comparison is listed below:

Table 11 Research Comparison

(Source: Self, Literature Review)

|  |  |  |  |
| --- | --- | --- | --- |
| **Research** | **Domain** | **Model Applied** | **Accuracy Achieved** |
| Matsa et al. (2021) | Intrusion detection | CNN | 98.95 |
| Chanu et al. (2023) | Intrusion detection | MLP | 98.8 |
| Albulayhi et al. (2022) | Intrusion detection | Random Forest | 99.98 |
| Ahmed et al. (2022) | Intrusion detection | Decision Tree | 95.1 |
| Tambe et al. (2023) | DDoS Detection | CNN | 99.2 |
| Wang & Liu (2020) | DDoS Detection | CNN | 98.98 |
| Haider et al. (2019) | DDoS Detection | CNN | 99.48 |
| Jemal et al. (2023) | DDoS Detection | CNN | 99.9 |
| Kumar et al. (2023) | DDoS Detection | CNN | 98 |
| Present Research | DDoS Detection | CNN | 100 |

### Discussion on Research Questions

**RQ-1**

The detection of DDoS attacks has been done in this research with the applications of both machine learning and deep learning. By comparing the performances through the classification metrics such as accuracy, precision etc., it has been observed that CNN has achieved the highest accuracy (100%). So, it can be said that the network can be protected with the application of deep learning by identifying DDoS attacks using the CNN Model.

**RQ-2**

The accuracy of the best-performing model which is the CNN at present has been compared with the previous research dealt with the detection of intrusions and DDoS attacks. In that comparison, it has been observed that the CNN model has performed optimally by obtaining higher accuracy compared to the existing research. Hence, it can be said that the present research has an improved approach to detecting DDoS attacks with the application of a hybrid feature selection approach and the CNN model.

### Discussion on Research Gap Filling

* The first gap that has been seen in the previous research was the lack of the application of a hybrid feature selection method for DDoS detection. In the previous research, it has been seen that RFE was applied by the following authors:
  + Matsa et al. (2021)
  + Tonni & Mazumder (2023)
  + Chanu et al. (2023)

Similarly, it has been seen that Chi-Squared was applied by the following authors:

* + Azmi et al. (2021)
  + Saha et al. (2022)

So, to fill the gap, these two methods have been combined to design the hybrid approach using which the optimum features have been chosen.

* The second gap that has been seen was the lack of the application of both machine learning and deep learning models for DDoS detection. The researchers who have employed machine learning are as follows:
  + Chia-Ying & Wei-Yang (2009)
  + Lansky et al. (2021)
  + Kumari et al. (2023)
  + Tambe et al. (2023)
  + Vattikuti et al. (2021)

The researchers who have employed deep learning are as follows:

* + Matsa et al. (2021)
  + Wang & Liu (2020)
  + Haider et al. (2019)
  + Jemal et al. (2023)
  + Kumar et al. (2023)

So, to fill this gap, the detection of DDoS has been done in this research by employing both machine learning and deep learning models.

### Discussion on Challenges Faced

While executing the artefact, the challenges have been faced and resolved by employing the required technical measures. Those are discussed below:

* The primary challenge which has been faced was due to the incorporation of data outliers which has been resolved by employing PCA and feature normalization. Principal Component Analysis (PCA) was employed to detect outliers in the data. Feature normalization has been used to reduce the amount of outliers. These methods, when applied to the data, the outliers have been reduced successfully.
* The second challenge that has been faced was due to the application of feature selection methods. To resolve this issue, Recursive Feature Elimination (RFE) and Chi-Squared methods, both of which have been observed and applied by previous researchers, were combined. This hybrid method was designed to leverage the strengths of both techniques. This approach not only streamlined the feature selection process but also contributed to more accurate and interpretable results.

## Conclusion

### Research Finding

DDoS attacks involve trying to disrupt the regular flow of traffic to a specific server, service, or network by inundating it with a large amount of internet traffic. The impact of these attacks on businesses, organizations, and individuals can be significant, leading to reduced productivity, monetary setbacks, and harm to one's public image. The present research has been conducted to detect DDoS intrusions with the application of machine learning and deep learning. DDoS attacks come in different types such as floods of traffic to overwhelm the network, exploiting protocol vulnerabilities, and targeting specific applications or services. Attackers frequently utilize botnets, which consist of hacked devices, to carry out synchronized DDoS attacks, making it challenging to identify the origin of the attack. In this research, the DDoS attack data has been collected from Kaggle. The data contains 151000 network transaction instances (records) with three types of network traffic namely Benign traffic, DDoS-ACK and DDoS-PSH-ACK. The last two types of network traffic belong to the DDoS category. The dataset has been preprocessed so that the features will be prepared for analytics. Firstly, feature encoding has been done to transform the categorical features into numerical ones. Next, the outliers in the features have been detected using the PCA method. After detecting outliers, those have been reduced using feature normalization by employing the MinMaxScaler where the outliers have been reduced from 99% to 60%. The hybrid feature selection technique has been employed by combining RFE and Chi-Squared using which 14 optimum featureless have been chosen and set to the final data predictor. The data has been split to prepare training and test sets with a 75:25 split ratio. The chosen models have been prepared and enqueued in a list for the sequential execution for training, testing and obtaining the prediction output. The performances of the employed models have been compared and it has been observed that the CNN model has gained 100% accuracy in detecting DDoS intrusions. This accuracy outperformed the other chosen models showing the superiority. Finally, the accuracy of CNN has been compared with the existing research where the superiority of the present research was found.

### Limitations

The present limitations of the research are stated below:

* The research has been done by collecting a DDoS attack dataset from one source.
* The data from diverse sources have not been employed.
* Only four machine learning models have been applied along with one deep learning model (CNN) for DDoS intrusion detection.
* The application of a hybrid model has not been done in this research.
* To design the hybrid feature selection method, only RFE and Chi-Squared methods have been combined.

### Future Scopes

The research can be extended further by employing the following implications:

* The application of multiple feature selection methods can be done to detect DDoS attacks.
* The hybrid model can be used to design multiple hybrid models.

# References

Ahmed, H. A., Hameed, A. & Bawany, N. Z., 2022. Network intrusion detection using oversampling technique and machine learning algorithms. *PeerJ Computer Science, 8, e820..*

Ahmed, J. et al., 2020. GDPR Compliant Consent Driven Data Protection in Online Social Networks: A Blockchain-Based Approach. *3rd International Conference on Information and Computer Technologies (ICICT),* pp. 1-6.

Ahmed, M. et al., 2009. NIDS: A network based approach to intrusion detection and prevention.. *In 2009 International Association of Computer Science and Information Technology-Spring Conference ,* pp. 141-144.

Albulayhi, K. et al., 2022. IoT intrusion detection using machine learning with a novel high performing feature selection method. *Applied Sciences, 12(10).*

Arya, A., Kumar, A. & Ahmad, S. S., 2023. DDoS Attack Detection Using Ensemble Machine Learning Approach.. *In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT),* pp. 1-5.

Ashiku, L. & Dagli, C., 2021. Network intrusion detection system using deep learning.. *Procedia Computer Science,,* pp. 239-247.

Ashodia, N. & Makadiya, K., 2022. Detection of DDoS attacks in SDN using Machine Learning.. *In 2022 International Conference on Electronics and Renewable Systems (ICEARS),* pp. 1322-1327.

Azmi, M. A. H., Sukri, K. A. M., Abdullah, N. A. & Hamid, I. R. A., 2021. Feature Selection Approach to Detect DDoS Attack Using Machine Learning Algorithms.. *JOIV: International Journal on Informatics Visualization, 5(4), ,* pp. 395-401..

Baklizi, M. K. et al., 2024. Web Attack Intrusion Detection System Using Machine Learning Techniques.. *International Journal of Online & Biomedical Engineering.*

Barati, M. & Rana, O., 2021. Checking GDPR Compliance for Cloud-based Services. *IEEE World Congress on Services (SERVICES),* pp. 1-5.

Bhargava R., Singh, Y. P. & Narawade, N. S., 2022. . Implementation of machine learning based ddos attack detection system.. *In 2022 3rd International Conference for Emerging Technology (INCET),* pp. 1-5.

Bhutia, N. T., Verma, H., Chauhan, N. & Kumar., L., 2022. DDoS Attacks Detection in ‘Internet of Medical Things’ Using Machine Learning Techniques.. *In 2022 IEEE Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI),* pp. 1-6.

Chanu, U. S., Singh, K. J. & Chanu, Y. J., 2023. A dynamic feature selection technique to detect DDoS attack.. *Journal of Information Security and Applications.*

Chen, M. J. & Xu, J., 2022. Data Privacy Protection from the Perspective of GDPR - A Case Study on E-learning Platform “SHCneo”. *15th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI),* pp. 1-6.

Chia-Ying, L. & Wei-Yang, L., 2009. Intrusion detection by machine learning: A review. *expert systems with applications, 36(10),* pp. 11994-12000..

Cole, M. D. & Schmitz, S., 2020. The Interplay Between the NIS Directive and the GDPR in a Cybersecurity Threat Landscape. *SSRN,* pp. 1-21.

Dangheralou, A. & Jahankhani, H., 2022. The Impact of GDPR Regulations on Cyber Security Effectiveness Whilst Working Remotely. *Artificial Intelligence in Cyber Security: Impact and Implications,* pp. 1-7.

Devi, R. S., Bharathi, R. & Kumar., P. K., 2023. Investigation on Efficient Machine Learning Algorithm for DDoS Attack Detection.. *In 2023 International Conference on Computer, Electrical & Communication Engineering (ICCECE),* pp. 1-5.

European Commission, 2018. *Data protection in the EU.* [Online]   
Available at: https://ec.europa.eu/info/law/law-topic/data-protection/data-protection-eu\_en

European Union, 2018. General Data Protection Regulation. Issue https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679.

Gamage, S. & Samarabandu, J., 2020. Deep learning methods in network intrusion detection: A survey and an objective comparison.. *Journal of Network and Computer Applications, 169, 102767..*

Haider, S., Akhunzada, A., Ahmed, G. & Raza, M., 2019. Deep learning based ensemble convolutional neural network solution for distributed denial of service detection in SDNs.. *In 2019 UK/China Emerging Technologies (UCET),* pp. 1-4.

Hao, Z., Feng, Y., Koide, H. & Sakurai, K., 2020. A sequential detection method for intrusion detection system based on artificial neural networks.. *International Journal of Networking and Computing,* pp. 213-226..

He, J., Tan, Y., Guo, W. & Xian, M., 2020. A small sample DDoS attack detection method based on deep transfer learning.. *In 2020 International Conference on Computer Communication and Network Security (CCNS,* pp. 47-50.

Hnamte, V. & Hussain., J., 2023. DDoS Detection Using Hybrid Deep Neural Network Approaches.. *In 2023 IEEE 8th International Conference for Convergence in Technology (I2CT),* pp. 1-8.

Jemal, I., Cheikhrouhou, O. & Haddar, M. A., 2023. IoT DOS and DDOS Attacks Detection Using an Effective Convolutional Neural Network.. *In 2023 International Conference on Cyberworlds (CW) ,* pp. 373-379.

Kheddara, H., Himeurb, Y. & Awad, A. I., 2023. Deep transfer learning for intrusion detection in industrial control networks: A comprehensive review.. *arXiv preprint arXiv:2304.10550..*

Khuphiran, P. et al., 2018. Performance comparison of machine learning models for DDoS attacks detection.. *In 2018 22nd International Computer Science and Engineering Conference (ICSEC),* pp. 1-4.

Kumar, D. et al., 2023. DDoS detection using deep learning.. *Procedia Computer Science, 21,* pp. 2420-2429..

Kumari, M., Choudhary, N. & Parihar, S., 2023. Impact of novel machine learning approaches on intrusion detection system.. *In AIP Conference Proceedings (Vol. 2782, No. 1). AIP Publishing..*

Kumbam, Y. R., 2020. *APA-DDoS Dataset.* [Online]   
Available at: https://www.kaggle.com/datasets/yashwanthkumbam/apaddos-dataset

Lansky, J. et al., 2021. Deep learning-based intrusion detection systems: a systematic review. *IEEE Access, 9,,* pp. 101574-101599..

Li, M. et al., 2022. A DDoS attack detection method based on deep learning two-level model CNN-LSTM in SDN network.. *In 2022 International Conference on Cloud Computing, Big Data Applications and Software Engineering (CBASE),* pp. 282-287.

Li, M. et al., 2022. A DDoS attack detection method based on deep learning two-level model CNN-LSTM in SDN network.. *In 2022 International Conference on Cloud Computing, Big Data Applications and Software Engineering (CBASE),* pp. 282-287.

Matsa, L. S., Zodi-Lusilao, G.-A. & Shava., F. B., 2021. Recursive Feature Elimination for DDoS Detection on Software Define Network.. *In 2021 IST-Africa Conference (IST-Africa),* pp. 1-10.

Matsa, L. S., Zodi-Lusilao, G.-A. & Shava, F. B., 2021. Recursive Feature Elimination for DDoS Detection on Software Define Network.. *In 2021 IST-Africa Conference (IST-Africa),* pp. 1-10.

Meriaux, E. et al., 2022. Performance Comparison of Machine Learning Methods in DDoS Attack Detection in Smart Grids. *In 2022 IEEE MIT Undergraduate Research Technology Conference (URTC),* pp. 1-5.

Mohy-Eddine, M. et al., 2023. An ensemble learning based intrusion detection model for industrial IoT security.. *Big Data Mining and Analytic,* pp. 273-287.

Naing, S. K. & Thwel, T. T., 2023. A Study of DDOS Attack Classification Using Machine Learning Classifiers.. *In 2023 IEEE Conference on Computer Applications (ICCA),* pp. 108-112.

Nurwarsito, H. & Nadhif., M. F., 2021. DDoS attack early detection and mitigation system on SDN using random forest algorithm and Ryu framework.. *In 2021 8th International Conference on Computer and Communication Engineering (ICCCE),* pp. 178-183.

Özçam, B., Kilinc, H. H. & Zaim, A. H., 2021. Detecting tcp flood ddos attack by anomaly detection based on machine learning algorithms. *In 2021 6th International Conference on Computer Science and Engineering (UBMK),* pp. 512-516.

Prakobphol, K. & Zhan, J., 2008. A novel outlier detection scheme for network intrusion detection systems.. *In 2008 International Conference on Information Security and Assuranc,* pp. 555-560.

Prathiksha S. & Meenakshi Sundaram, 2023. Detection of DDoS Attack using Machine Learning.. *Global Conference on Information Technologies and Communications (GCITC),* pp. 1-5.

Priya, S. & Pradeep, M. K., 2021. Intelligent outlier detection with optimal deep reinforcement learning model for intrusion detection.. *In 2021 4th International Conference on Computing and Communications Technologies (ICCCT),* pp. 336-341.

Reddy, K. P. et al., 2021. A hybrid neural network architecture for early detection of DDOS attacks using deep learning models. *In 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC),* pp. 323-327.

Saha, S., Priyoti, A. T., Sharma, A. & Haque, A., 2022. Towards an optimized ensemble feature selection for DDoS detection using both supervised and unsupervised method. *Sensors, 22(23),.*

Sahu, S. K., Mohapatra, D. P. & Ray, N. K., 2021. An ensemble-based outlier detection approach on intrusion detection.. *In 2021 19th OITS International Conference on Information Technology (OCIT,* pp. 404-409.

Said, R. B. & Askerzadea, I., 2023. Attention-Based CNN-BiLSTM Deep Learning Approach for Network Intrusion Detection System in Software Defined Networks.. *In 2023 5th International Conference on Problems of Cybernetics and Informatics (PCI),* pp. 1-5.

Saranya, T. et al., 2020. Performance analysis of machine learning algorithms in intrusion detection system: A review.. *Procedia Computer Science,* pp. 1251-1260..

Sarker, I. H., Abushark, Y. B., Alsolami, F. & Khan, A. I., 2020. Intrudtree: a machine learning based cyber security intrusion detection model. Symmetry,. *Symmetry, 12(5).*

Smith, R., 2016. Performance of MPI Codes Written in Python with NumPy and mpi4py. *6th Workshop on Python for High-Performance and Scientific Computing (PyHPC),* pp. 1-6.

Sujatha, G., Kanchhal, Y. & George, G., 2022. An advanced approach for detection of distributed denial of service (DDoS) attacks using machine learning techniques.. *In 2022 3rd International Conference on Smart Electronics and Communication,* pp. 821-827.

Tambe, R. S., Dand, H. & Salunke, M. D., 2023. Role of Machine Learning Ensemble in DDoS Intrusion Detection.. *In 2023 IEEE 5th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA),* pp. 145-149.

The Data Protection Act, 2019. *The Data Protection Act.* [Online]   
Available at: https://www.gov.uk/data-protection#:~:text=The%20Data%20Protection%20Act%202018%20is%20the%20UK's%20implementation%20of,used%20fairly%2C%20lawfully%20and%20transparently

Tonni, Z. A. & Mazumder, R., 2023. A Novel Feature Selection Technique for Intrusion Detection System Using RF-RFE and Bio-inspired Optimization. *In 2023 57th Annual Conference on Information Sciences and Systems (CISS),* pp. 1-6.

Vattikuti, S. et al., 2021. Ddos attack detection and mitigation using anomaly detection and machine learning models.. *In 2021 IEEE International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) ,* pp. 1-6.

Visetbunditkun, T. & Srichavengsup, W., 2022. Ddos attack detection using ensemble machine learning models with rfe algorithm.. *In 2022 7th International Conference on Business and Industrial Research (ICBIR),* pp. 269-273.

Wang, J., Liu, Y., Su, W. & Feng, H., 2020. A DDoS attack detection based on deep learning in software-defined Internet of things. *IEEE 92nd Vehicular Technology Conference (VTC2020-Fall),* pp. 1-5.

Wang, L. & Liu, Y., 2020. A DDoS attack detection method based on information entropy and deep learning in SDN.. *In 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC,* pp. 1084-1088.

Yuze, S., MENG, X., MENG, Q. & HAN., X., 2018. DDoS attack detection algorithm based on hybrid traffic prediction model.. *In 2018 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC),* pp. 1-5.

Zhang, B., Zhang, T. & Yu, Z., 2017. DDoS detection and prevention based on artificial intelligence techniques. *3rd IEEE International Conference on Computer and Communications (ICCC),* pp. 1276-1280.

Zhang, Z., Xing, Z., Xu, X. & Zhu, L., 2023. RIdiom: Automatically Refactoring Non-Idiomatic Python Code with Pythonic Idioms. *IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion),* pp. 1-7.

Zhou, L., Zhu, Y., Zong, T. & Xiang, Y., 2022. A feature selection-based method for DDoS attack flow classification.. *Future Generation Computer Systems, 132,* pp. 67-79.

# Appendix

# Attaching Libraries

import warnings

warnings.filterwarnings("ignore")

import numpy as np,pandas as pd,seaborn,datetime,os

import matplotlib.pyplot as vsap

import matplotlib as mpvs

import plotly.express as pxids

from sklearn.pipeline import make\_pipeline

from sklearn import preprocessing, utils,metrics,pipeline,feature\_selection, model\_selection, decomposition

from sklearn import tree, linear\_model,svm, ensemble, neural\_network

import visualkeras

from PIL import ImageFont

import tensorflow

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D

from keras.layers import Dense

from keras.layers import Flatten

from keras.layers import Dropout

from keras.layers import Conv1D

from keras.layers import MaxPooling1D

from keras.utils import to\_categorical

import os

## Data Reading

def ReadAPA():

apa=pd.read\_csv("APA-DDoS-Dataset.csv")

print("Total Records of Data: {}".format(apa.shape[0]))

print("Total Features of Data: {}".format(apa.shape[1]))

return apa

APADT=ReadAPA()

APADT.head()

## Data Cleaning

def CleanAPA(apa):

print(apa.info())

print("Missing Values (Before Cleaning): ",sum(apa.isna().sum()))

if sum(apa.isna().sum())>0:

apa=apa.dropna()

print("Missing Values (After Cleaning): ",sum(apa.isna().sum()))

return apa

APADT=CleanAPA(APADT)

## Data Visualization

pd.crosstab(APADT['ip.src'],APADT['Label']).plot(kind='barh',

title="Network Traffic Types by Source IP", figsize=(8,3))

vsap.grid()

vsap.show()

apaft=['frame.len','Packets','Bytes']

apanmft=['Frame Length','Packets','Bytes Transacted']

for c in range(len(apaft)):

APADT.groupby('Label').mean('{}'.format(apaft[c]))['{}'.format(apaft[c])].plot(

kind='bar',figsize=(5,3), color=['g','c','m'], title="{} for Network Traffics".format(apanmft[c]))

vsap.ylabel('{}'.format(apaft[c]))

vsap.xlabel('Traffic Type')

vsap.grid()

vsap.show()

## Feature Engineering

### Feature Encoding

def EncodeAttr(dt):

apa=dt.copy()

apa=apa.drop(['frame.time','Label'],axis=1)

lbapa=dt['Label']

apa\_cts=apa.dtypes[apa.dtypes=='object'].index.tolist()

print("Detected Object Type Features: \n", \*apa\_cts, sep="\n")

if len(apa\_cts)==0:

apa['Label']=lbapa

return apa

else:

for c in range(len(apa\_cts)):

apa[apa\_cts[c]]=apa[apa\_cts[c]].replace(apa[apa\_cts[c]].unique(),[i+1 for i in range(len(apa[apa\_cts[c]].unique()))])

apa['Label']=lbapa

return apa

Enc\_APADT=EncodeAttr(APADT)

Enc\_APADT.head()

### Outlier Treatement

def OutlierChecking(apa,n,col,tx):

arrapa=np.array(apa)

pcapa = decomposition.PCA(n\_components=n)

pcapa.fit(arrapa)

pcapacm=["Comp-{}".format(i+1) for i in range(len(pcapa.explained\_variance\_ratio\_.tolist()))]

vsap.figure(figsize=(4,2))

vsap.title("{} Normalization\nMaximum Variance: {}".format(tx,round(max(pcapa.explained\_variance\_ratio\_),5)),fontsize=18)

vsap.bar(pcapacm,pcapa.explained\_variance\_ratio\_.tolist(),width=0.5,color=col)

vsap.xlabel("PCA",fontsize=14)

vsap.ylabel("Variance",fontsize=14)

vsap.grid()

vsap.show()

return pcapa.explained\_variance\_ratio\_

def DataScaling(apa):

ppc = preprocessing.MinMaxScaler()

nrmapa=ppc.fit\_transform(apa)

return nrmapa

outapa=[]

Enc\_APADT = Enc\_APADT.replace([np.inf, -np.inf], np.finfo('float32').max)

outapa.append(OutlierChecking(Enc\_APADT.drop('Label',axis=1),2,"#FF4500","Before"))

fledt=[]

apapca\_flag=[]

for rp in outapa:

for r in rp:

if r>0.7:

apapca\_flag.append(True)

if len(apapca\_flag)==1 and True in apapca\_flag:

APA\_Norm=DataScaling(Enc\_APADT.drop('Label',axis=1))

APA\_Norm=pd.DataFrame(APA\_Norm,columns=Enc\_APADT.drop('Label',axis=1).columns.tolist())

APA\_Norm['Label']=Enc\_APADT['Label']

OutlierChecking(APA\_Norm.drop('Label',axis=1),2,"#12E193","After")

APA\_Norm.head()

### Hybrid Feature Selection By Combining RFE and Chi-Sq

def FetRFE(apa):

Xapa=apa.drop([apa.columns.tolist()[-1]],axis=1)

Yapa=apa[apa.columns.tolist()[-1]]

Yapa=Yapa.replace(Yapa.unique(),[x for x in range(len(Yapa.unique()))])

rfe\_apa = feature\_selection.RFE(estimator=linear\_model.LogisticRegression(),n\_features\_to\_select = int(len(apa.columns)\*0.7), step = 0.7)

rfe\_trnd=rfe\_apa.fit(Xapa,Yapa)

print("RFE Selected Features: \n",\*Xapa.columns[rfe\_trnd.get\_support()],sep="\n")

return Xapa.columns[rfe\_trnd.get\_support()]

def FetChisq(apa):

Xapa=apa.drop([apa.columns.tolist()[-1]],axis=1)

Yapa=apa[apa.columns.tolist()[-1]]

Yapa=Yapa.replace(Yapa.unique(),[x for x in range(len(Yapa.unique()))])

chi\_apa=feature\_selection.SelectKBest(feature\_selection.chi2, k=int(len(apa.columns)\*0.7))

chi\_apa.fit(Xapa, Yapa)

print("Chi-Squared Selected Features: \n",\*Xapa.columns[chi\_apa.get\_support()],sep="\n")

return Xapa.columns[chi\_apa.get\_support()]

ApaFets=[]

ftc=FetChisq(APA\_Norm)

ftr=FetRFE(APA\_Norm)

for x in ftc:

if x in ftr:

ApaFets.append(x)

print("\nSelected Features Using Hybrid Technique: \n")

print(\*ApaFets, sep="\n")

## Data Preparation

### Preparing Predictor and Target Data

Predictor=APA\_Norm[ApaFets]

Predictor.head()

Target=APA\_Norm['Label']

print(Target.value\_counts())

### Data Split

def SegData(PredDt, TgrDt):

TrnDDSX,TstDDSX,TrnDDSy,TstDDSy=model\_selection.train\_test\_split(PredDt,TgrDt, test\_size=0.25, random\_state=0)

print("Test Class Distribution: ",TstDDSy.value\_counts(),"\n")

print("Training Class Distribution: ",TrnDDSy.value\_counts())

return TrnDDSX,TstDDSX,TrnDDSy,TstDDSy

TrnDDSX,TstDDSX,TrnDDSy,TstDDSy=SegData(Predictor, Target)

## Assigning Algorithms

ClfDDS=[

ensemble.RandomForestClassifier(n\_estimators=10, criterion='entropy', max\_depth=11,min\_weight\_fraction\_leaf=0.45,max\_features='log2'),

linear\_model.LogisticRegression(tol=0.06, C=0.01,max\_iter=2,solver='liblinear',fit\_intercept=False),

neural\_network.MLPClassifier(hidden\_layer\_sizes=(2,1,), learning\_rate\_init=0.001, power\_t=0.7,max\_fun=3,max\_iter=4),

make\_pipeline(preprocessing.StandardScaler(),svm.SVC(C=0.2, kernel='sigmoid', degree=3,tol=0.01,max\_iter=20))

]

Clfs=[

"Random Forest",

"Logistic Regression",

"MLP Classifier",

"SVC"

]

ClfDDS[0]

ClfDDS[1]

ClfDDS[2]

ClfDDS[3]

## DDoS Detection

def ConfMatVs(yAct,yPreds,ModelDDS):

CLSS=np.unique(np.array(yAct))

ids\_cnf=pd.crosstab(yAct,yPreds,rownames=['True'], colnames=['Predicted'], margins=True)

vsap.figure(figsize=(5,3))

vsap.title("{}".format(ModelDDS), fontsize=16,color="m")

seaborn.heatmap(ids\_cnf.iloc[:len(CLSS),:len(CLSS)],fmt="d",annot=True,cmap="plasma")

vsap.show()

return ids\_cnf

DDSResultData=[[],[],[],[],[],[]]

for i in range(len(ClfDDS)):

print(" {} ".format(Clfs[i]))

PrsClf = ClfDDS[i]

StartTime\_1 = datetime.datetime.now()

PrsClf.fit(TrnDDSX, TrnDDSy)

StartTime\_2 = datetime.datetime.now()

TimeDifference = StartTime\_2 - StartTime\_1

TimeTrnSec=TimeDifference.total\_seconds()

print("Training Time: {} Seconds".format(round(TimeTrnSec,3)))

StartTime\_3 = datetime.datetime.now()

DDSPredTst=PrsClf.predict(TstDDSX)

StartTime\_4 = datetime.datetime.now()

TimeDifference = StartTime\_4 - StartTime\_3

TimeTstSec=TimeDifference.total\_seconds()

print("Training Time: {} Seconds".format(round(TimeTstSec,3)))

AccDDSTst=metrics.accuracy\_score(TstDDSy,DDSPredTst)

print("Train Accuracy: ",round(PrsClf.score(TrnDDSX,TrnDDSy),5)\*100)

print("Test Accuracy: ",round(PrsClf.score(TstDDSX,TstDDSy),5)\*100)

DDSResultData[0].append(round(AccDDSTst,5)\*100)

DDSResultData[1].append(round(metrics.precision\_score(TstDDSy,DDSPredTst,average="weighted"),5)\*100)

DDSResultData[2].append(round(metrics.recall\_score(TstDDSy,DDSPredTst,average="weighted"),5)\*100)

DDSResultData[3].append(round(metrics.f1\_score(TstDDSy,DDSPredTst,average="weighted"),5)\*100)

DDSResultData[4].append(round(TimeTstSec,3))

DDSResultData[5].append(round(PrsClf.score(TrnDDSX,TrnDDSy),5)\*100)

ConfMatVs(TstDDSy,DDSPredTst,Clfs[i])

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Classification report for \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ {} \_\_\_\_\_\_\_\_\_\_\_\_\_\_".format(Clfs[i]))

print(metrics.classification\_report(TstDDSy,DDSPredTst))

DDoSDf=pd.DataFrame({

"Models":Clfs,

"Train\_Accuracy":DDSResultData[5],

"Test\_Accuracy":DDSResultData[0],

"Precision":DDSResultData[1],

"Recall":DDSResultData[2],

"F1-Score":DDSResultData[3],

"Prediction\_Time":DDSResultData[4]

})

DDoSDf=DDoSDf.sort\_values(by="Test\_Accuracy",ascending=False)

DDoSDf

for i in DDoSDf.columns.tolist()[2:]:

figres = pxids.bar(DDoSDf, y=i, x="Models",

text=i,color="Models",title="Comparison of {}".format(i),height=400,width=600)

figres.show()

## CNN

def ResultVisualizer(ffanhs,md,strt,en,gprhknd,yax,eps):

reshs=ffanhs.history

fithst=pd.DataFrame({

"Iteration":[i+1 for i in range(eps)],

"Loss\_Train":reshs['loss'],

"Loss\_Valid":reshs['val\_loss'],

"Accuracy\_Train":reshs['accuracy'],

"Accuracy\_Valid":reshs['val\_accuracy']

})

clscl=["#FF00FF","#0002FF"]

fithst.iloc[:,strt:en].plot(kind=gprhknd,figsize=(6,3),color=clscl)

vsap.title("{} Comparison ({})".format(yax,md),fontsize=17,color="m")

vsap.xlabel("Epochs",fontsize=15,color="m")

vsap.ylabel("{}".format(yax),fontsize=15,color="m")

vsap.grid()

vsap.show()

TrUn=TrnDDSy.unique()

TstUn=TstDDSy.unique()

print(TrnDDSy.unique())

print(TstDDSy.unique())

clss=TrnDDSy.unique().tolist()

TrnDDSyUN=TrnDDSy.replace(TrnDDSy.unique(),[x for x in range(len(TrnDDSy.unique()))])

TstDDSyUN=TstDDSy.replace(TstDDSy.unique(),[2,0,1])

print(TrnDDSyUN.unique())

print(TstDDSyUN.unique())

scaler = preprocessing.StandardScaler()

x\_train = scaler.fit\_transform(TrnDDSX)

x\_test = scaler.transform(TstDDSX)

# Reshape data to fit into the Conv1D layer

x\_train = x\_train.reshape((x\_train.shape[0], x\_train.shape[1], 1))

x\_test = x\_test.reshape((x\_test.shape[0], x\_test.shape[1], 1))

DOSCNN = Sequential([

Conv1D(1024, 1, activation='relu', input\_shape=(14, 1)),

MaxPooling1D(2),

Conv1D(512, 1, activation='relu'),

MaxPooling1D(2),

Flatten(),

Dense(128, activation='relu'),

Dense(3, activation='sigmoid')

])

DOSCNN.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

DOSCNN.summary()

TrnTimeStrt = datetime.datetime.now()

DOSCNN\_Hist=DOSCNN.fit(x\_train,TrnDDSyUN, epochs=10, validation\_data=(x\_test,TstDDSyUN))

TrnTimeEnd = datetime.datetime.now()

TrnTm = TrnTimeEnd - TrnTimeStrt

TrnTimeSecCNN=TrnTm.total\_seconds()

ResultVisualizer(DOSCNN\_Hist,"CNN",3,5,'line','Accuracy',10)

ResultVisualizer(DOSCNN\_Hist,"CNN",1,3,'line','Loss',10)

pred\_train=DOSCNN.predict(x\_train)

trnpred = pred\_train.argmax(axis=1)

dftrn=pd.DataFrame({

"Actual":TrnDDSyUN,

"Predicted":trnpred

})

acctrcnn=metrics.accuracy\_score(TrnDDSyUN,trnpred)

AcrFinCNNTr=round(acctrcnn,4)\*100

print("Training Accuracy for CNN: {}%".format(AcrFinCNNTr))

TstTimeStrt = datetime.datetime.now()

pred\_test=DOSCNN.predict(x\_test)

TstTimeEnd = datetime.datetime.now()

PrdTime = TstTimeEnd - TstTimeStrt

TstTimeSecCNN=PrdTime.total\_seconds()

tstpred = pred\_test.argmax(axis=1)

dftest=pd.DataFrame({

"Actual":TstDDSyUN,

"Predicted":tstpred

})

cnfcnn=pd.crosstab(TstDDSyUN,tstpred,rownames=['Actual'], colnames=['Detected'], margins=True)

cnfcnncls=cnfcnn.iloc[:3,:3]

print("--------\n{}\n-----------".format(cnfcnncls))

vsap.figure(figsize=(6,4))

vsap.title("Confusion Matrix: {}".format("CNN"), fontsize=16,color="m")

seaborn.heatmap(cnfcnncls,fmt="d",annot=True,cmap="flag",xticklabels=clss, yticklabels=clss)

vsap.show()

acr\_cnn\_test=metrics.accuracy\_score(TstDDSyUN,tstpred)

prec\_cnn\_test=metrics.precision\_score(TstDDSyUN,tstpred,average='weighted')

recl\_cnn\_test=metrics.recall\_score(TstDDSyUN,tstpred,average='weighted')

f1s\_cnn\_test=metrics.f1\_score(TstDDSyUN,tstpred,average='weighted')

AcrFinCNN=round(acr\_cnn\_test,4)\*100

PrcFinCNN=round(prec\_cnn\_test,4)\*100

RclFinCNN=round(recl\_cnn\_test,4)\*100

FsFinCNN=round(f1s\_cnn\_test,4)\*100

print("Accuracy: {}%\nPrecision: {}%\nRecall: {}%\nF1-Score: {}%".format(AcrFinCNN,PrcFinCNN,RclFinCNN,FsFinCNN))

tensorflow.keras.utils.plot\_model(

DOSCNN,

to\_file="CNNModel.png",

rankdir="TB",

dpi=65

)

DDoSDfFinal=pd.DataFrame({

"Models":Clfs+['CNN'],

"Train\_Accuracy":DDSResultData[5]+[AcrFinCNNTr],

"Test\_Accuracy":DDSResultData[0]+[AcrFinCNN],

"Precision":DDSResultData[1]+[PrcFinCNN],

"Recall":DDSResultData[2]+[RclFinCNN],

"F1-Score":DDSResultData[3]+[FsFinCNN],

"Prediction\_Time":DDSResultData[4]+[TstTimeSecCNN]

})

DDoSDfFinal=DDoSDfFinal.sort\_values(by='Test\_Accuracy',ascending=False).reset\_index(drop=True)

DDoSDfFinal

bc\_colrs=["#66CDAA","#FFA600","#1F45FC","#FFCBA4","#F67280","#6A0DAD"]

for rh in DDoSDfFinal.columns.tolist()[2:]:

DDoSDfFinal=DDoSDfFinal.sort\_values(by=rh,ascending=True)

vsap.figure(figsize=(6,4))

vsap.title("Comparison of {}".format(rh),fontsize=18,color="#8B008B")

vsap.barh(DDoSDfFinal['Models'],DDoSDfFinal[rh],color=bc\_colrs)

vsap.ylabel("Models",fontsize=16,color="#8B008B")

vsap.xlabel("{}".format(rh),fontsize=16,color="#8B008B")

for sr, val in enumerate(DDoSDfFinal["{}".format(rh)]):

vsap.text(val, sr, str(val))

vsap.grid()

vsap.show()