Western Graduate&PostdoctoralStudies

Western University

Scholarship@Western

Electronic Thesis and Dissertation Repository

8-16-2023 2:30 PM

Classification of DDoS Attack with Machine Learning Architectures and Exploratory Analysis

Amreen Anbar, *The University of Western Ontario*

Supervisor: Haque, Anwar, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

© Amreen Anbar 2023

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Digital Communications and Networking Commons, and the Other Computer Engineering Commons

Recommended Citation

Anbar, Amreen, "Classification of DDoS Attack with Machine Learning Architectures and Exploratory Analysis" (2023). *Electronic Thesis and Dissertation Repository*. 9622.
https://ir.lib.uwo.ca/etd/9622

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

The ever-increasing frequency of occurrence and sophistication of DDoS attacks pose a serious threat to network security. Accurate classification of DDoS attacks with efficiency is crucial in order to develop effective defense mechanisms. In this thesis, we propose a novel approach for DDoS classification using the CatBoost algorithm, on CICDDoS2019, a benchmark dataset containing 12 variations of DDoS attacks and legitimate traffic using real-world traffic traces. With a developed ensemble feature selection method and feature engineering, our model proves to be a good fit for DDoS attack type prediction. Our experimental results demonstrate that our proposed approach achieves high classification accuracy of 89.5% and outperforms several state-of-the-art machine learning algorithms in terms of both accuracy and computational efficiency. The thesis does not only limit itself to achieving a good prediction score, it also uses the recently introduced concept of explainable AI (XAI) as a tool for ensuring transparency and interpretability of the proposed approach. Our approach can be applied in real-world scenarios to enhance the security of network infrastructure against DDoS attacks.

Keywords

Cyber-attack, DDoS, Machine Learning, Deep Learning, Boosting algorithm, CatBoost, XAI

Summary for Lay Audience

Cyber-attacks have become increasingly sophisticated and can be considered to be a growing threat to the technology-dependent world. Distributed Denial of Service (DDoS) attack is a type of such attack that has an exponentially growing number of occurrences. Briefly, in a DDoS attack, attackers flood a server with traffic to overwhelm it, making it unavailable to the legitimate users. The attackers can be in different locations and cause chaos from sources difficult to trace.

To address this problem and to come up with better mitigation strategies, researchers have developed machine learning algorithms for classifying DDoS attacks with better efficiency. Deep learning is a type of machine learning that uses neural networks to learn patterns in data. By analyzing network traffic data, algorithms can learn to distinguish between benign traffic and malicious traffic that are sub-types of DDoS attack. With an aim to solve this problem, this thesis focuses on developing and improving machine learning algorithms by using different data pre-processing and selection techniques and examining different neural network architectures to enhance the performance of the algorithms.

The goal of the research is to provide network security professionals with better tools to detect and respond to DDoS attacks. By improving the classification of different types of attacks, the algorithms can help affected organizations to effectively identify and mitigate threats to their networks with specific mitigation techniques. The research has the potential to improve the security of computer networks and shield from the ever-growing threat of cyber-attacks. The use of machine learning algorithms in detecting and classifying DDoS attacks can help organizations to better safeguard their networks and ensure the reliability of their services. The developed model in this thesis outperforms the existing works on addressing the mentioned problem with stellar performance evaluation scores.

Acknowledgments

I would like to express my sincere gratitude to Dr. Anwar Haque, my thesis supervisor, for his invaluable guidance, support, and encouragement throughout my research. His expertise, patience, and constructive feedback have been instrumental in shaping my ideas and improving the quality of my work.

I am also grateful to Moinul Sayed, a fellow PhD student, for his generous help and insightful discussions that greatly enriched my research. I also had the pleasure of working alongside some outstanding graduate students from Dr. Haque's lab, Noshin Tasnim and Nusrat Samia, who made the research experience even more enjoyable, and I learned a great deal from them.

I am incredibly grateful to my family for being my constant support system and for giving me the strength and resilience to push through challenges and stay focused on my goals. I would like to extend a special thanks to my husband, Shahed Sabab, for his unwavering support, patience, and excellent mentorship.

Table of Contents

Abstract ii

Summary for Lay Audience iii

Acknowledgments iv

Table of Contents v

List of Tables ix

List of Figures x

Glossary xii

Chapter 1 1

1 Introduction 1

1.1 Thesis Contribution 3

1.2 Thesis Outline 3

Chapter 2 5

2 Background 5

2.1 Reflection on Cyber-Attacks: Threat or not? 5

2.1.1 Attack Taxonomy 5

2.1.2 DDoS Attack 6

2.1.3 DDoS Attack Types 6

2.1.4 Mitigation of DDoS Attacks 8

2.2 Machine Learning Algorithms 9

2.2.1 Supervised Learning Algorithms 9

2.2.2 Classification Algorithms 10

2.3 Ensemble Learning 10

2.3.1 Stacking 11

2.3.2 Blending 11

2.3.3 Bagging 12

2.3.4 Boosting 13

2.4 Gradient Boosting Algorithms 14

2.5 Classification Tasks in Machine Learning 14

2.6 Evaluation Methods for Classification Models 15

2.6.1 Accuracy 15

2.6.2 Precision, Recall, and F1-Score 16

2.6.3 Confusion Matrix 16

2.6.4 ROC and AUC 16

2.6.5 Cross-Validation 17

2.7 Deep Learning for Tabular Data 17

Chapter 3 18

3 Literature Review 18

3.1 Binary Classification 18

3.2 Multi-Class Classification 19

3.3 Research Gap 20

Chapter 4 22

4 Data 22

4.1 Data Description 22

4.2 Exploratory Data Analysis 24

4.2.1 Infinity/ null/ missing value 24

4.2.2 Object Type Data 25

4.2.3 Collinearity 25

4.2.4 Statistical Data Type 27

4.2.5 Data with No Useful Interpretation 29

4.3 Input Dataset Preparation 29

4.4 Feature Engineering 31

4.5 Feature Selection 31

4.5.1 Mutual Information 32

4.5.2 Analysis of Variance 33

4.5.3 Chi-squared 33

4.5.4 Ensemble Feature Selection 34

Chapter 5 38

5 Model Development and Evaluation 38

5.1 Development Resources 38

5.2 Deep Learning Algorithms 39

5.2.1 TabMLP 39

5.2.2 TabResnet 41

5.2.3 TabTransformer 43

5.2.4 FT- Transformer 45

5.2.5 FastFormer 47

5.2.6 Perceiver 48

5.3 Bagging Algorithms 49
5.4 Boosting Algorithms 51
5.4.1 XGBoost 51
5.4.2 LightGBM 52
5.4.3 CatBoost 53
5.5 Comparative Analysis 55
Chapter 6 60
6 Explainable Artificial Intelligence 60
Chapter 7 65
7 Conclusion 65
7.1 Contributions 65
7.2 Limitations 66
7.3 Future Research Direction 67
Bibliography 68
Curriculum Vitae 74

List of Tables

Table 4.1: Ensemble feature scores for the top 58 features. 34

Table 5.1: Hardware configuration of the server. 38

Table 5.2: Hardware configuration of the PC. 39

Table 5.3: Performance metrics for TabMLP. 40

Table 5.4: Performance metrics for TabResnet. 43

Table 5.5: Performance metrics for TabTransformer. 45

Table 5.6: Performance metrics for FT-Transformer. 47

Table 5.7: Performance metrics for FastFormer. 48

Table 5.8: Performance metrics for Perceiver. 49

Table 5.9: Performance metrics for RF. 50

Table 5.10: Performance metrics for XGBoost. 52

Table 5.11: Performance metrics for LGBM. 53

Table 5.12: Performance metrics for CatBoost. 55

Table 5.13: Performance comparison for deep learning algorithms. 56

Table 5.14: Performance comparison for the boosting algorithms. 58

List of Figures

Figure 2.1: Attack Vector, Operational Impact, Defense, Information Impact, and Target (AVOIDIT) – A cyber-attack taxonomy[18]. 7

Figure 2.2: DDoS attack taxonomy [15]. 8

Figure 2.3: Schematic illustration of the stacking [23]. 11

Figure 2.4: Schematic illustration of the stacking [23]. 12

Figure 2.5: Schematic illustration of the boosting [23]. 13

Figure 4.1: Attack percentages from the training set of CICDDoS2019. 23

Figure 4.2: Datatypes in the dataset CICDDoS2019. 25

Figure 4.3: Features with perfect collinearity. 26

Figure 4.4: Countplot for the feature ‘Protocol’. 27

Figure 4.5: Countplot for the feature ‘ACK Flag Count’. 28

Figure 4.6: Appropriate feature selection methods depending on variable types. 32

Figure 5.1: TabMLP model architecture. 40

Figure 5.2: Confusion Matrix for TabMLP. 41

Figure 5.3: TabResnet model architecture. 42

Figure 5.4: Confusion Matrix for TabResnet. 42

Figure 5.5: TabTranformer model architecture. 43

Figure 5.6: Confusion Matrix for TabTransformer 44

Figure 5.7: FT-Tranformer model architecture. 46

Figure 5.8: Confusion Matrix for FT-Transformer. 46

Figure 5.9: Confusion Matrix for FastFormer. 47

Figure 5.10: Confusion Matrix for Perceiver. 48

Figure 5.11: Schematic illustration of Random Forest. 49

Figure 5.12: Confusion Matrix for RF. 50

Figure 5.13: Confusion Matrix for XGBoost. 52

Figure 5.14: Tree expansion in LightGBM [62]. 53

Figure 5.15: Confusion Matrix for LGBM. 54

Figure 5.16: Confusion Matrix for CatBoost. 56

Figure 6.1: Relative importance of features. 61

Figure 6.2: Confusion Matrix for the CatBoost model performance. 61

Figure 6.3: SHAP summary plots. 62

Figure 6.4: SHAP summary plot with the feature values. 63

Figure 6.5: SHAP summary plot with the feature values. 64

Glossary

Term	Definition
Analysis of Variance (ANOVA)	A statistical technique for comparing means among different groups.
Artificial Intelligence (AI)	The simulation of human intelligence processes by machines, enabling them to perform tasks that typically require human cognitive abilities.
Bootstrap Aggregating (Bagging)	An ensemble method that combines multiple models to improve overall prediction accuracy.
CatBoost	A gradient boosting algorithm that handles categorical features naturally without preprocessing.
Character Generator (CharGen)	A network service for testing and measuring network performance.
Chi-Square (chi-2)	A statistical test used to determine the association between categorical variables.
Classification token (CLS)	The first token of every sequence that captures sentence level representations of sequences in natural language processing tasks.
Convolutional Neural Network (CNN)	A type of deep learning model often used for image recognition and processing.
Central Processing Unit (CPU)	The primary hardware component responsible for executing instructions in a computer.

Decision Tree (DT)	A flowchart-like model used for decision-making by breaking down a problem into smaller decisions.
Deep Learning (DL)	A subset of machine learning that involves training complex neural networks with multiple layers to automatically learn and represent patterns in data.
Distributed Denial of Service (DDoS)	A cyberattack that overwhelms a target system by flooding it with excessive traffic.
Domain Name System (DNS)	A system that translates human-readable domain names into IP addresses.
Explainable Artificial Intelligence (XAI)	AI models designed to provide transparent and understandable explanations for their decisions.
Extreme Gradient Boosting (XGBoost)	A machine learning algorithm known for its high performance in predictive modeling.
Gaussian Error Linear Unit (GeLU)	An activation function designed to model uncertainty and perform well in deep learning architectures.
Graphics Processing Unit (GPU)	A specialized hardware component capable of handling parallel processing tasks, often used in deep learning and graphics rendering.
Gated Recurrent Unit (GRU)	Another type of RNN variant that efficiently captures dependencies in sequential data.
Lightweight Directory Access Protocol (LDAP)	A protocol used for accessing and maintaining directory services.

LEaky ReLU	A variant of ReLU that allows a small gradient for negative input values, addressing the "dying ReLU" problem.
Logistic Regression (LR)	A statistical method for analyzing datasets with binary outcomes, predicting probabilities.
Long Short-Term Memory (LSTM)	A specific type of RNN that helps mitigate the vanishing gradient problem and handle longer sequences.
Machine Learning (ML)	A field of AI where computers learn from data to improve their performance on a specific task without being explicitly programmed.
Mutual Information (MI)	A measure of the dependence between two random variables in information theory.
Microsoft SQL Server (MSSQL)	A relational database management system developed by Microsoft.
Network Time Protocol (NTP)	A protocol for synchronizing clocks on computer systems over networks.
One-Hot Encoding (OHE)	A method of representing categorical data as binary vectors for machine learning algorithms.
Packet Capture (PCAP)	A file format used to store network traffic data for analysis.
Recurrent Neural Network (RNN)	A type of neural network architecture designed for sequence data analysis.

Random Forest (RF)	An ensemble technique that combines multiple decision trees to improve prediction accuracy and control overfitting.
Rectified Linear Unit (ReLU)	An activation function used in neural networks that returns zero for negative input values.
SHapley Additive exPlanations (SHAP)	A technique for explaining the output of machine learning models by attributing the contribution of each feature.
Simple Service Discovery Protocol (SSDP)	A network protocol for discovering devices and services in a local network.
SYN Flood	A type of DDoS attack that exploits the TCP handshake process by overwhelming a system with SYN requests.
Transmission Control Protocol (TCP)	A fundamental communication protocol that ensures reliable data transmission over networks.
Trivial File Transfer Protocol (TFTP)	A simple protocol for transferring files without authentication.
User Datagram Protocol (UDP)	A connectionless protocol for sending data without guarantee of delivery or order.
Variance Inflation Factor (VIF)	A measure used to detect multicollinearity in regression analysis.

Chapter 1

1 Introduction

In a time where the flow of information is controlling the motion of the world, affecting this flow is evidently a reason for concern. Turning towards the device screen and seeking solutions for problems has become a reflex action for any individual. But what if you turn to the network for information and you are denied the service you are seeking? For solving the majority of the problems today related to the important sectors ranging from healthcare industry to financial institutions, the probability of one going blank when there is a problem with the internet service is undeniably high. The Internet has become irreplaceable, and this motivates the growth of a force of resistance. The newest form of attacks to influence the world drastically are all cyberattacks. Any unauthorized cyber act aimed at violating the security policy of a cyber-asset and causing damage, disruption or disruption of the services or access to the information of the said national cyber asset is called cyber-attack [1]. Negatively affecting the network that the world is dependent on is no less than a weapon of destruction today. Types of cyber-attacks include malware, spyware, denial of service, sniffer, trojan etc.

“Morris Worm”, introduced in 1988 by Robert Morris, a graduate student from Cornell, was the first major cyber-attack recorded and it successfully infected computer systems at Stanford, Princeton, Johns Hopkins, NASA, Lawrence Livermore Labs, and UC Berkeley, among other institutions. Starting from here, the series of attacks on the Air Force’s Rome Laboratory using a password sniffer by Kuji in 1994, a successful hacking attempt by Vladimir Levin on Citibank’s network in 1995, attack on Motorola and Nokia by Kevin Mitnick in 1995, attack on the U.S. government websites by Max Butler in 1998, and the incident of Melissa Virus attack on the Microsoft document files are some of the cyber-attacks that took place in world history [2]. The range of destruction caused by the attacks only went ramping up making it a significant problem requiring attention from the world scholars. “Mafiaboy” Michael Calse, a 15-year-old hacker launched a series of distributed denial of service (DDoS) attacks on world renowned sites like Amazon, Yahoo, CNN, and eBay in the year 2000, showing the new century the sight of an incoming war. In 2005, a security breach at a U.S. retailer led to the data leak of 1.4 million HSBC Bank MasterCard users and in 2008, Heartland Payment systems were attacked using a combination SQL injection, password sniffers, and malware, compromising the data of 134 million users [2] – these all reflect on the exponential growth of the destruction caused by cyber-attacks. Evidently, cyber-attacks that have only grown powerful over time which leaves us with a blank head space when asked the question mentioned at the beginning of the writing – "What if you turn to the network for information and you are denied of the service you are seeking?" DDoS attack is one of the major types of cyber-attacks that can make an individual face this. In November 2021, Microsoft mitigated a DDoS attack targeting an Azure customer with a throughput of 3.45 Tbps and a packet rate of 340 million PPS which is believed to be the largest DDoS attack ever recorded [3]. According to research from NETSCOUT's ATLAS Security Engineering & Response Team (ASERT), threat actors launched approximately 2.9 million DDoS attacks in the first quarter of 2021, a 31% increase from the same time in 2020 [4]. As per this source, there were 6,019,888 global DDoS attacks in 1st half of 2022 and globally, DDoS attacks are predicted to number over 15.4 million in 2023 – almost double that of 2018 [5]. This exponential growth is a reason enough for concern and it's important to have proper security measures in place to protect against DDoS attacks and prevent them from causing significant harm to online services. To protect networks against these types of attacks, detecting the attack sub-type accurately is a pre-requisite. This generates a new multi-classification problem to solve where the classes are the sub-types of the attack. Machine Learning (ML) techniques today have been proved to be a useful tool and quite literally, the only tool stands out for solving such multi-classification problems. Machine learning can be a valuable tool for classifying DDoS attacks, providing a scalable, flexible, and automated approach to detecting and mitigating these types of attacks.

The research aimed at solving the multiclass classification problem by investigating many of the 'state-of-the-art' deep learning models e.g., TabTransformer [6], FastFormer [7], Perceiver [8] to uncover the efficacy Attention [9] mechanism for network traffic data. In addition, ensemble tree-based approaches like Light Gradient Boosting Machine (LGBM) [10], eXtreme Gradient Boosting (XGBoost) [11] and CatBoost [12] were also investigated to solve the multiclass classification problem. Explainable Artificial Intelligence (XAI) was used as a tool to interpret model behavior.

1.1 Thesis Contribution

The research contributions of this thesis can be presented with the following points -

1. Applied feature engineering and created new features for the dataset CICDDoS-2019.
2. Proposed an ensemble feature selection technique that proved to be generating expected outcome while keeping the process computationally efficient.
3. Developed deep learning based models for tabular data and obtained evaluation metrices outperforming the existing related works.
4. Employed the recently introduced boosting techniques on network traffic data and showed improved results compared to the bagging technique employed.
5. Evaluated and presented comparison of the performance of the developed models, with other state-of-the-art machine learning models where the developed best performing model had the accuracy score of 0.895, exceeding the performances of the existing state-of-the-art models.
6. Analyzed the effect of the features on model prediction, bringing them under the umbrella of explainable AI.

1.2 Thesis Outline

The rest of this thesis is organized in 6 chapters and a brief idea on what these chapters will focus on is as follows.

Chapter 2 introduces the terminologies and technologies used for this thesis and contains the background knowledge working as a base for this work.

Chapter 3 records the literature review of the deep learning approaches for solving cyber-attack classification problems, highlighting mostly the works on DDoS attack classification.

Chapter 4 portrays the insights gained from the exploratory data analysis and the steps followed for pre-processing the data to use for model training.

Chapter 5 logs the resource requirement for model development and explains the development of the models and prediction performance obtained from each model. This chapter further evaluates the credibility of the work presenting a comparative analysis referencing the research works conducted with the same problem as of this thesis.

Chapter 6 focuses on interpreting predictions from the machine learning model in the light of Explainable AI (XAI).

Chapter 7 is the concluding chapter for the thesis that summarizes the findings of the proposed method and shares a high-level idea for scopes of the continuation of the work in future.

Chapter 2

2 Background

This chapter introduces the technical concepts and terminologies that are necessary for comprehending the ideas used as a foundation of the thesis.

2.1 Reflection on Cyber-Attacks: Threat or not?

Cyber-attacks are the emerging threats to the world and the impact of such attacks can be as devastating as the world being paralyzed. This is one big con of connecting everything. Any change in the system can affect the whole. The COVID-19 pandemic played a significant role in changing the cyber threat landscape. More organizations have shifted to hybrid mode to utilize the available resources to the fullest. 2022 saw cyberattacks reach an all-time high in response to the Russo-Ukrainian war - Education and Research remains the most targeted sector, but attacks on the healthcare sector registered a 74% increase year-on-year [13]. All the critical infrastructure services today are dependent on the mercy of the resistive force today. From the invasion of personal space to playing as a vulnerable base for the political and economic activities all over the world, cyber-attacks are a buzzword everywhere. As per the National Cyber Threat Assessment 2023-2024, it is estimated that 95% of all deepfake videos on the Internet contain non-consensual synthetic pornography and that about 90% of these depict women [14]. Starting from the attack that targeted nearly 500 people’s cryptocurrency wallets stealing approximately $18 million worth of Bitcoin and $15 million worth of Ethereum and other cryptocurrencies [15], 2022 is filled with incidences that shows us how cyber-attacks are a threat and evidently, the biggest threat that any sector of the world can face today.

2.1.1 Attack Taxonomy

A cyber-attack taxonomy is a classification system that categorizes different types of cyber-attacks based on various distinguishable characteristics such as their methods, objectives, targets, and impact. There are requirements that need to be met for a taxonomy to be universally accepted. To classify the cyber-attacks, different taxonomies have been proposed by Kjaerland [16], Hansman and Hunt [17] and many authors over time. Attack Vector, Operational Impact, Defense, Information Impact, and Target (AVOIDIT) [18] takes it a step further using five different classifiers and stands out as a complete taxonomy for cyber-attacks. The taxonomy classifies the attacks by five major classes or categories – Attack Vector, Operational Impact, Defense, Informational impact, Target. Figure 2.1 shows the cyber-attack taxonomy titled as AVOIDIT. It classifies attacks in a tree-like structure, providing the ability to classify the allusive blended attack [18].

2.1.2 DDoS Attack

DDoS or the Distributed Denial or Service attacks refer to the type of cyber-attacks in which the one attacking overwhelms the target flooding it with service requests. The flooding is done from multiple sources targeting a system or a network with a view to disrupting the service making it unavailable to the users using the service. Depriving legitimate users of the service by attacking using a botnet, which is a network of multiple compromised devices, is a reason for concern in a space where everything takes place over networks. The attacker exploits the device’s vulnerabilities to establish a botnet that sends a massive volume of service request simultaneously ultimately causing the system speed to decrease significantly or the overall system to crash. Since the botnets are only pretending to be service seekers, it is difficult to differentiate and filter out the malicious traffic from the legitimate ones.

2.1.3 DDoS Attack Types

Mirkovic et al. [19] proposed a taxonomy that categorizes attacks as automation, vulnerability, source address validity, attack rate dynamics, characterization, persistence of agents, victim, and impact on the victim. The authors also proposed defense mechanisms based on activity level, cooperation, and deployment location. Asosheh et al. [20] proposed a taxonomy based on known potential attacks and they categorized attacks based on these eight features- impact, architecture, strategy, degree of automation, attack rate dynamics, vulnerability, scanning propagation strategy, and packet content. Bhardwaj et al. [21] took

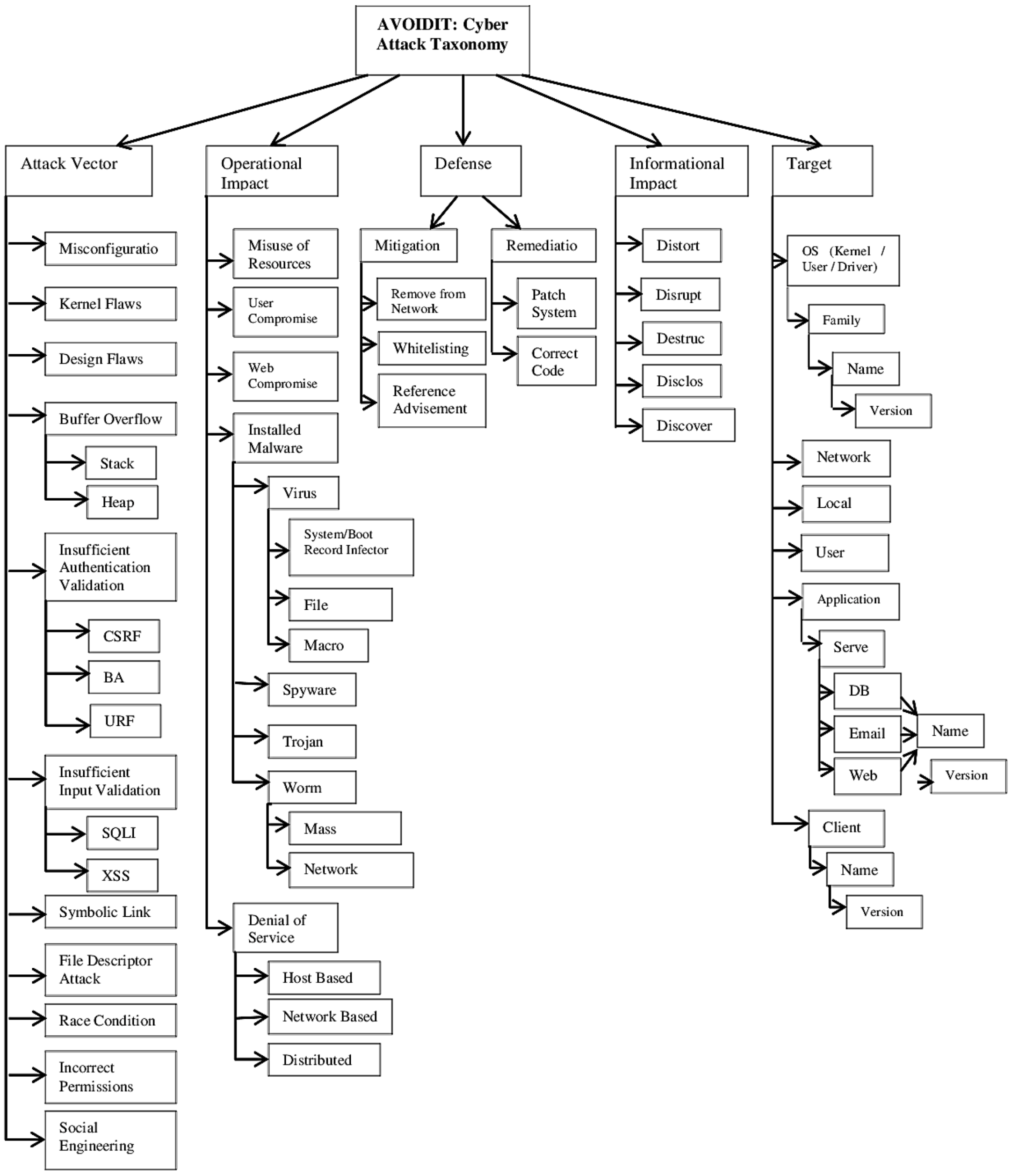


Figure 2.1: Attack Vector, Operational Impact, Defense, Information Impact, and Target (AVOIDIT) – A cyber-attack taxonomy[18]**.**

the cloud computing paradigm into consideration and proposed a taxonomy having four different categories of DDoS attacks - vulnerability, degree of automation, attack rate dynamics, and attack impact. Sharafaldin et al. [22] proposed a new taxonomy where the DDoS attacks that can be carried out using Transmission Control Protocol (TCP)/ User

Datagram Protocol (UDP) at the application layer are classified into two major categories – Reflection-based DDoS attacks and Exploitation-based DDoS attacks. Figure 2.2 sums it up.

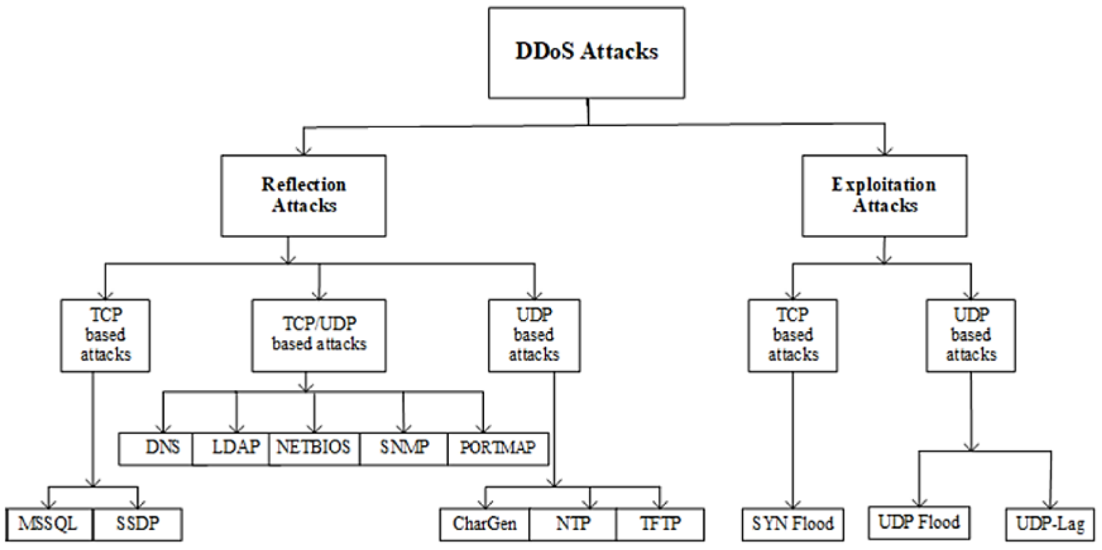


Figure 2.2: DDoS attack taxonomy [15].

Reflection-based DDoS attacks are a type of DDoS attack that exploits the characteristics of certain network protocols to amplify the traffic sent to the target system or network. These attacks rely on the use of third-party servers or devices to reflect and amplify the attack traffic. TCP based attacks include MSSQL, SSDP while UDP based attacks include CharGen, NTP and TFTP. There are certain attacks that can be carried out using either TCP or UDP like DNS, LDAP, NETBIOS, and SNMP [22]. Exploitation-based DDoS attacks, also known as vulnerability-based DDoS attacks, are a type of DDoS attack that exploits vulnerabilities in the target system or network to overload or crash it. TCP based exploitation attacks include SYN flood and UDP based attacks include UDP flood and UDP- Lag [22].

2.1.4 Mitigation of DDoS Attacks

DDoS attacks can have a significant impact on the availability and performance of online services, applications, and networks and can cause a wide range of issues like website downtime, slow website loading times, and reduced availability of critical online services. All these can negatively impact businesses, organizations, and individuals. Implementing network security solutions, such as firewalls, intrusion detection and prevention systems can help mitigate DDoS attacks. Another strategy can be using content distribution networks (CDN) so that the clustered traffic can be distributed across multiple servers reducing the effect of DDoS attacks. Another approach is to use anomaly detection and behavior analysis to detect and block malicious traffic. This involves monitoring network traffic for unusual patterns and anomalies that may indicate a DDoS attack is underway. Once detected, the system can take action to block the traffic and mitigate the effects of the attack. Machine learning techniques can play a role here in proceeding with this approach.

2.2 Machine Learning Algorithms

Machine learning models have algorithms at the base level which receives the data it is to learn from, analyses the input data and provides a prediction on the basis of learned parameters. The data being fed as the learning material is the training data. The models learn from this data and optimize performance developing their own idea and intelligence on the knowledge provided.

Machine learning algorithms can be broadly categorized into four types: supervised, semi-supervised, unsupervised and reinforcement.

2.2.1 Supervised Learning Algorithms

Supervised learning uses labeled training data to learn the mapping function that takes the input variables and provides output variables. There are two main types of supervised learning algorithms – Classification and Regression and a modified type – Ensembling.

- Classification is used to predict the outcome of a given sample when the output variable is in the form of categories. Unconstrained, individual trees are prone to overfitting. Decision tree is a popular classification algorithm.

- Regression is used to predict the outcome of a given sample when the output variable is in the form of real values. Output values obtained have probabilistic interpretation and such algorithms can be regularized in order to avoid overfitting problems. Logistic Regression (LR) is an example.
- Ensembling is another modified algorithm in supervised learning. When a model performs good for some features and another performs good for a different set of features, combining the predictions of these individually weak models produce a more accurate prediction. Bagging and boosting are examples of ensemble techniques.

2.2.2 Classification Algorithms

Classification algorithms are a class of machine learning techniques that are used to automatically categorize data into predefined classes or categories. These algorithms are trained on labeled datasets, where each data point is assigned a target class label. The objective of a classification algorithm is to learn a decision boundary that separates different classes in the input feature space. Once trained, the algorithm can be used to predict the class labels of new, unseen data points based on their features. Decision tree, Random Forest (RF), SVM are some of the classification algorithms.

2.3 Ensemble Learning

Ensemble techniques are machine learning methods that combine multiple models to improve the overall predictive power and robustness of the algorithm. In an ensemble, individual models are trained on the same data using different techniques or parameters to produce a set of diverse models. The predictions of these models are then combined in some way to produce a final output that is often more accurate and less prone to overfitting than any individual model. Methods of ensembling include – bagging, boosting, stacking and blending.

2.3.1 Stacking

Stacking is an ensemble technique in machine learning that involves combining the predictions of multiple models by training a meta-model that learns to make predictions based on the output of the individual models. In this approach, multiple base models are trained on the training data, and their predictions are then used as input to a higher-level meta-model.

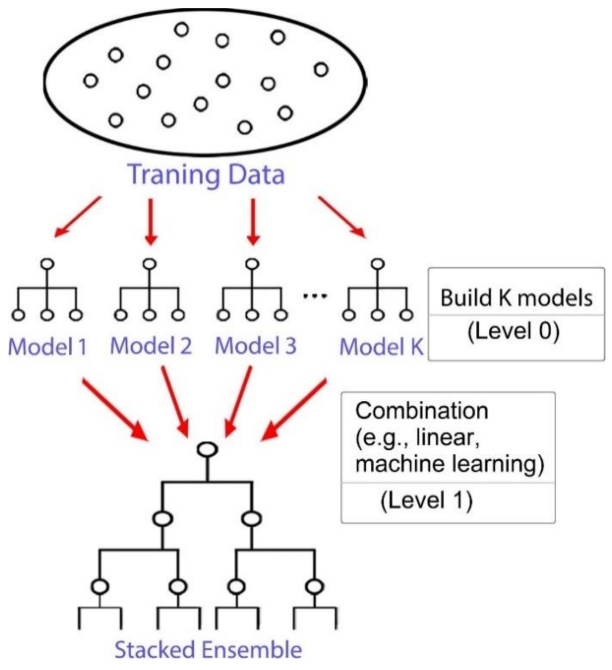


Figure 2.3: Schematic illustration of the stacking [23]**.**

The meta-model is trained on the predictions of the base models and learns to make final predictions based on the performance of the base models.

2.3.2 Blending

Blending is an ensemble technique in machine learning that is very similar to stacking but involves training a meta-model on a validation set rather than the entire training set. In a blending approach, the original training data is split into two parts: a training set and a validation set. Multiple base models are trained on the training set, and their predictions are then used as input to a higher-level meta-model, which is trained on the validation set.

The final predictions are then made by the meta-model on a test set that is independent of the training and validation sets.

2.3.3 Bagging

Bagging, short for Bootstrap Aggregating, is an ensemble technique in machine learning that involves training multiple instances of the same base model on

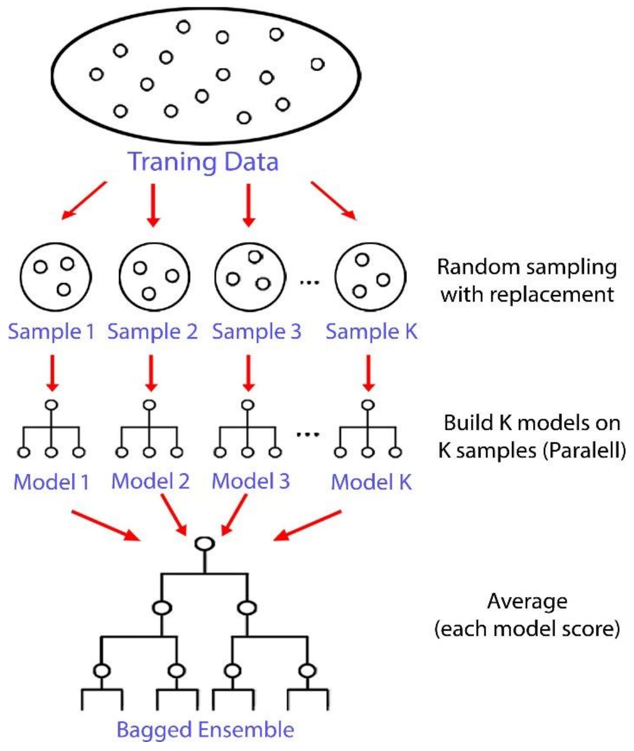

Traning Data
Random sampling with replacement
Sample 1
Sample 2
Sample 3
Sample K
Build K models on K samples (Paralell)
Model 1
Model 2
Model 3
Model K
Average (each model score)
Bagged Ensemble

Figure 2.4: Schematic illustration of the stacking [23]**.**

different random subsets of the training data. In a bagging approach, the training data is randomly sampled with replacement to generate multiple subsets of data, and then a base model is trained on each subset. The final prediction of the bagging ensemble is made by aggregating the predictions of the individual models, typically by averaging or taking a majority vote.

Random Forest is an example of a bagging algorithm.

2.3.4 Boosting

Boosting is an ensemble technique in machine learning that involves training multiple weak models sequentially, where each subsequent model is trained to

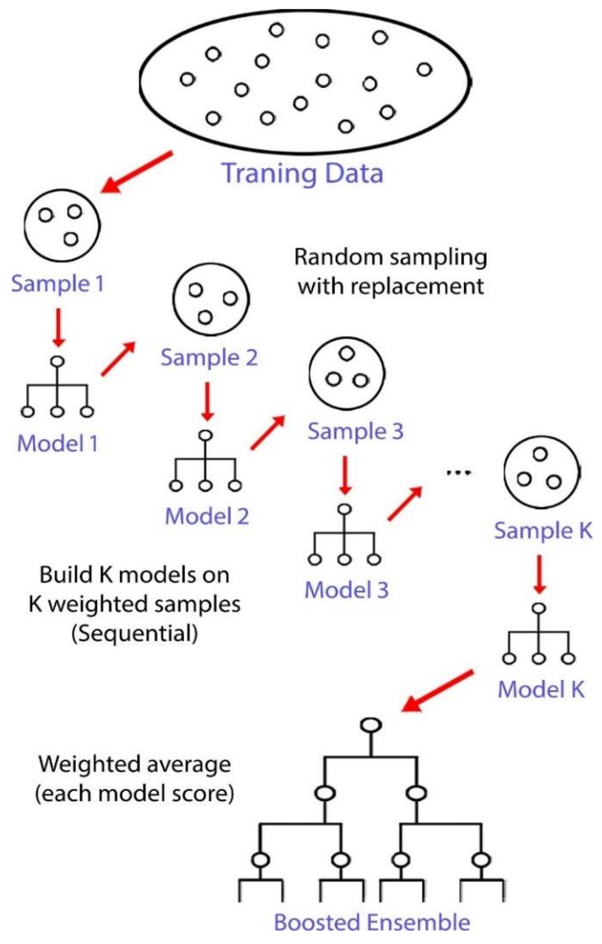

Traning Data
Random sampling with replacement
Sample 1
Model 1
Sample 2
Model 2
Sample 3
Model 3
...
Sample K
Model K
Build K models on K weighted samples (Sequential)
Weighted average (each model score)
Boosted Ensemble

Figure 2.5: Schematic illustration of the boosting [23]**.**

improve the weaknesses of the previous model. In a boosting approach, the training data is repeatedly reweighted to focus on the examples that are hard to classify, and each new model is trained to correctly classify the previously misclassified examples. The final prediction of the boosting ensemble is made by aggregating the predictions of the individual models, typically by weighted voting, where the weight of each model is determined based on its performance on the training data.

2.4 Gradient Boosting Algorithms

Gradient boosting is a machine learning algorithm that builds a predictive model by iteratively adding weak learners to a model in a way that minimizes a loss function. The "gradient" in gradient boosting refers to the use of gradient descent optimization to update the model's parameters. In this algorithm, the loss function is typically the difference between the predicted and actual values of the target variable, and the goal is to minimize this loss function by iteratively improving the model's predictions. To accomplish this, gradient boosting iteratively adds new models to the ensemble and trains them to correct the errors of the previous models. Specifically, at each iteration, the algorithm adds a new model to the ensemble that is trained on the residuals of the current predictions. The residuals are the differences between the predicted and actual values of the target variable. By training new models to correct the residuals, the algorithm gradually improves the accuracy of the predictions. The models added to the ensemble at each iteration are typically "weak learners," such as decision trees or linear regression models, that individually have limited predictive power. However, by combining the predictions of many weak learners, the algorithm can build a more accurate and robust predictive model.

The attributes that are tested for gradient boosting are learning rate, maximum depth of the tree, maximum number of features to consider, minimum number of samples required, and the subsampling rate.

Examples of such algorithms are, eXtreme Gradient Boosting (XGB), Light Gradient Boosting Machine (LGBM), and CatBoost.

2.5 Classification Tasks in Machine Learning

Classification tasks need predicting the target, which is the output variable based on the input fed which are the features. The output necessarily will not be a single variable for all the tasks.

Some of the commonly seen classification tasks are -

- Binary Classification: In this type of classification, the target variable has only two possible outcomes, typically represented as 0 or 1. Examples include spam detection, fraud detection, and disease diagnosis.
- Multi-class Classification: In this type of classification, the target variable has more than two possible outcomes, typically represented as distinct labels or classes. Examples include image classification, sentiment analysis, and speech recognition.
- Multi-label Classification: In this type of classification, each example can have multiple target labels, rather than being limited to a single label. Examples include image tagging, music genre classification, and text categorization.
- Imbalanced Classification: In this type of classification, the target variable has imbalanced class distribution, meaning that one class has significantly fewer examples than the other class. Examples include rare disease diagnosis, fraud detection, and anomaly detection.
- Hierarchical Classification: In this type of classification, the target variable has a hierarchical or nested structure, where the classes are organized in a tree-like structure. Examples include species classification and product categorization.

2.6 Evaluation Methods for Classification Models

There are several evaluation methods that are used to evaluate the machine learning model performance. Depending on the type of data, the evaluations methods to consider with more importance may vary. Some of the commonly used model performance indicators are- accuracy, ROC and AUC, precision, recall, f1 score, and confusion matrix.

2.6.1 Accuracy

This is the proportion of correctly classified examples out of the total number of examples. Accuracy is a simple and easy-to-understand evaluation metric, but it can be misleading in cases where the class distribution is imbalanced.

$$\textit{Accuracy} = \frac{\textit{True Positive} + \textit{True Negaive}}{\textit{True Positive} + \textit{True Negaive} + \textit{False Positive} + \textit{False Negative}} \quad \textbf{2.1}$$

2.6.2 Precision, Recall, and F1-Score

Precision measures the proportion of correctly classified positive examples out of all the positive examples.

$$\boldsymbol{Precision} = \frac{\boldsymbol{True\ Positive}}{\boldsymbol{True\ Positive + \ False\ Positive}} \qquad \mathbf{2.2}$$

Recall measures the proportion of correctly classified positive examples out of all the examples that should have been classified as positive.

$$\boldsymbol{Recall} = \frac{\boldsymbol{True\ Positive}}{\boldsymbol{True\ Positive + \ False\ Negative}} \qquad \mathbf{2.3}$$

F1-score is the harmonic mean of precision and recall, and it combines both measures into a single value.

$$\boldsymbol{F1\ score} = 2 * \frac{\boldsymbol{Precision * Recall}}{\boldsymbol{Precision + Recall}} \qquad \mathbf{2.4}$$

2.6.3 Confusion Matrix

This is a table that summarizes the performance of a classification model by counting the number of true positives, true negatives, false positives, and false negatives. It can be used to calculate various evaluation metrics, such as accuracy, precision, recall, and F1 score. A confusion matrix is an easily interpretable visual representation of the model’s performance for each feature provided as the input data.

2.6.4 ROC and AUC

The receiver operating characteristic (ROC) curve is a plot of the true positive rate (sensitivity) against the false positive rate (1 - specificity) at different classification thresholds. The area under the ROC curve (AUC) is a measure of the model's overall performance, and it ranges from 0 to 1, with higher values indicating better performance.

2.6.5 Cross-Validation

This technique involves splitting the data into multiple subsets and training the model on different subsets while testing on the remaining subset. Cross-validation can help to evaluate the model's generalization performance and avoid overfitting.

2.7 Deep Learning for Tabular Data

Structured data or data organized in a tabular format can be referred to as tabular data. For analysis of such, deep learning has grown popularity in the recent years for their ability to efficiently learn the hierarchical representations of the input data and handle the complex relationships between the input features regardless of these being non-linear or non-monotonic. Using deep learning models for tabular data can reduce the need for feature engineering. Although for larger datasets the overfitting problem can arise, with tuning the hyper-parameters in the right direction, these models produce considerably good performance score.

In addition to the traditional deep learning algorithms like DNN, RNN, the recently introduced transformer-based models are also a potential fit for the model choice for learning from tabular data. Some of such models are TabTransformer, FT-Transformer, Perceiver.

Chapter 3

3 Literature Review

This chapter presents the works on DDoS detection and classification with machine learning approaches. Machine learning techniques have been applied to DDoS detection for quite some time now, and the first use of machine learning in DDoS detection can be traced back to the mid-2000s.

3.1 Binary Classification

Mirkovic et al. [19] used a supervised learning algorithm to identify the characteristics of DDoS attacks based on packet header information. They trained their algorithm using a labeled dataset of benign and attack traffic and evaluated its performance showing a satisfactory performance considering the works published around that time.

Hasan et al. [24] used Deep Convolutional Neural Network (DCNN) on an Optical Burst Switching (OBS) Network dataset which outperformed most other machine learning techniques performance with the maximum classification accuracy of 99% where k-Nearest Neighbor (KNN), SVM and NB's classification accuracy were 93%, 88% and 79% respectively. The dataset consists of 21 attributes and 4 class variables which makes it a multi-class classification problem. Amma et al. [25] proposed a Vector Convolutional Deep Feature Learning (VCDeepFL) approach, combining Vector Convolutional Neural Network (VCNN) and Fully Connected Neural Network (FCNN) for identifying DDoS attacks. VCNN downsampled the input vector and FCNN boosted the system performance by calculating the best weight set from the training data. On the NSL KDD dataset, VCDeepFL (99.3%) exceeded the accuracy of Multi-layer Perceptron (MLP) (98.9%) and SVM (99%). In the proposed model, the output layer of FCNN contained 6 nodes as it was a 6-class classification problem. On the KDDCUP99 dataset, Virupakshar [26] et al. obtained 97% accuracy with Deep Neural Network (DNN) and 96% accuracy with DT DAD-MCNN, a multi-channel CNN(MC-CNN) based DDoS attack detection framework proposed by Chen et al. [27] showed better performance on CICIDS2017 (83 features) and

KDDCUP99 (41 features) datasets than the conventional machine learning models like Long Short-Term Memory (LSTM), RF. The accuracy of the MC-CNN for binary classification on the CICIDS2017 dataset was 98.87% and for KDDCUP99 dataset it was 99.18% exceeding the performance of the conventional models. Shaaban et al. [28] used CNN to detect and classify the DDoS traffic into benign and malicious information with an accuracy of 99% using two different datasets- one was captures using wireshark on Mission control center (MCC) network and another one was NSL-KDD dataset. Wang et al. [29] also used on CICIDS2017 and obtained an accuracy of 98.98% in an Software Defined Networking (SDN) environment. Sabeel et al. [30] also worked with binary classification of DDoS traffic on a benchmark datasets- CICIDS2017 which resulted in an accuracy score of 99.8% and 99.9% DNN and LSTM respectively. Assis et al. [31] used CNN on CICDDoS2019 keeping all 12 attack classes present in the dataset but for binary classification that led to an accuracy of 95.4%. Even though it was binary classification, keeping all the classes influenced the prediction.

3.2 Multi-Class Classification

The next level of DDoS detection and classification was not only separating the malicious traffic from the benign but also identifying the subtype of the DDoS attack. CICDDoS2019 [32] dataset, prepared by Sharafaldin et al. [22] is a benchmark dataset for including a lot more subtypes of DDoS attacks than the previously mentioned datasets. The author [22] used 12 attacks from the training data and 7 attacks from the testing data of the original dataset and obtained precisions of 78%, 77%, 41% and 25% for the algorithms Intrusion Detection (ID3), RF, NB, and LR respectively. 91.16% precision was achived by Can et al. [33] but it was only for 6 classes from the dataset. Ferrag et al. [34] took all the 13 classes from the CICDDoS2019 dataset and achieved precisions of 72% using RNN and 66% using DNN. DNN was also used by Chartuni et al. [35] taking all the 13 labels from the original dataset which resulted in precision score of 83.1%.

Shurman et al. [36] used only the reflection based attacks from the dataset (9 attack classes) on LSTM and achieved an accuracy of 91.54%. Cil et al. [37] used DNN for the same dataset and keeping 12 classes excluding the benign attacks, achieved a precision score of

80.49%. an ensemble DNN method was proposed by Sayed et al. [38] stacking CNN, LSTM and GRU which resulted in precision of 84.8% for 12 classes.

From all the reviewed research mentioned here, a scope of improvement could be figured as none of these works put emphasis on reading beyond the type-wise feature division from the dataset. Moreover, the recently introduced deep learning models based on attention and transformer, that understand context from within what apparently looks like mere textual information, so far have not been developed and used on network traffic data. This research gap led to the aim for this thesis.

3.3 Research Gap

Research has been conducted to address DDoS attack classification problem and as seen in the previous sections of this chapter, majority of the existing research uses the supervised instance learning models like Deep Neural Network (DNN)[33][34][35][37] and Convolutional Neural Network (CNN)[34] or the supervised sequence learning models that are Recurrent Neural Network (RNN)[34] e.g., Long Short-Term Memory (LSTM)[36]. There are extensive surveys of the existing research addressing the research problem of predicting cyber-attacks as accurately as possible. Even though no two works are completely similar with regards to how they pre-processed data, they all looked at the data from the same point of view i.e., considering the samples as a sequence of numerical instances. This shows a scope of looking at the solution spectrum from a different point of view in regard to how the data will be processed for numerical features and categorical features. There are features in the network traffic datasets which, even though, are represented by numbers, the numbers don't convey meaningful quantitative information. Numeric values here represent categories or labels and can be considered as ordinal data. Several model architectures have been proposed recently [6][39][8][7], that process numerical and categorical features differently. They proved to be effective while preserving the interrelation of the categorical feature better than considering them as a sequence of numerical input. Attention mechanism paved the direction to capture more semantic meaning from these categorical features [9]. The potential of these architectures remains unexplored for network traffic data. In addition, ensemble tree-based approaches

[10][11][12] that offers an advantage of processing categorical columns differently (i.e., generating embeddings) for classification tasks. Also remains uninvestigated.

Considering the weight and the impact of misclassifying the attack type, the performance scores obtained from the existing works directs towards room for improvement. Creating new features/ feature engineering proved to have a positive influence on model performance [40]. By creating new features that are more meaningful and understandable can help to better understand the underlying patterns and relationships in the data. This aspect shows potential that was not sufficiently explored. Moreover, combinations of feature selection techniques like chi-2 [41], Analysis of Variance (ANOVA) [42], Mutual Information (MI) [43] etc. could be leveraged to improve the learning performance, lower computational complexity, and build better generalizable models [44].

For interpreting the machine learning models, interpretation tools [45] like SHapley Additive exPlanations (SHAP) [46], Local Interpretable Model-Agnostic Explanations (LIME) [47] , Anchors [48] etc. could be leveraged in the existing works but this aspect also shows insufficient exploration.

Chapter 4

4 Data

The set of observation of recorded measurements we feed to the machine learning networks for them to learn from and test the gathered knowledge with is referred to as data. Data is an integral part of the task utilizing machine learning. The quantity and quality of data has a huge role to play when it comes to the performance of the models.

Data can be collected in any format- textual, visual, audio etc. and various forms like numerical, categorical, time- series data, text data; the collected data is later processed based on the input requirements of the model to be used. Processed data is usually split into training, testing and validation sets for the purpose of training the model, evaluating the model performance, and improving model performance by hyperparameter tuning respectively. The models learn from the patterns and correlations existing in the data points and come up with a relationship between the train and test data to be used for prediction. Data can be labeled or unlabeled to be used for supervised and unsupervised machine learning respectively. Labeled data includes a target variable that the model is trying to predict, unlike unlabeled data.

Some popular and reliable resources for datasets are Microsoft research open data, Google's dataset search, government datasets, Amazon datasets, UCI machine learning repository etc. The dataset used for this thesis is from the University of New Brunswick (UNB).

4.1 Data Description

The CICDDoS2019 [32] dataset, created by the Canadian Institute for Cybersecurity (CIC), is a publicly available benchmark dataset for the evaluation of DDoS detection and classification methods. This dataset is a collection of various DDoS attacks that were generated in a controlled lab environment to simulate real-world attacks. The CICDDoS2019 dataset is one of the largest and most comprehensive DDoS datasets available, with over 80 million records and 80 features extracted from network packets.

This dataset was gathered over 2 days as both Packet Capture (PCAP) file format and flow format based for training and testing evaluation. Training dataset contains 12 different types of DDoS attacks- NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP. The testing dataset contains 7 types of DDoS attacks- PortMap, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, and SYN. Total number of attacks are 50063112 and 20364525 respectively in the training and testing set. The attacks were categorized into two types- reflection-based and exploitation-based attacks from the transport and application layer. 80 traffic features were extracted from the PCAP files of the dataset using CICFlowMeter-V3 [49][50] and saved as CSV files. CICFlowMeter-V3 is a open source toolkit written in Java for flow based feature extractor that can extract 80 features from PCAP files.

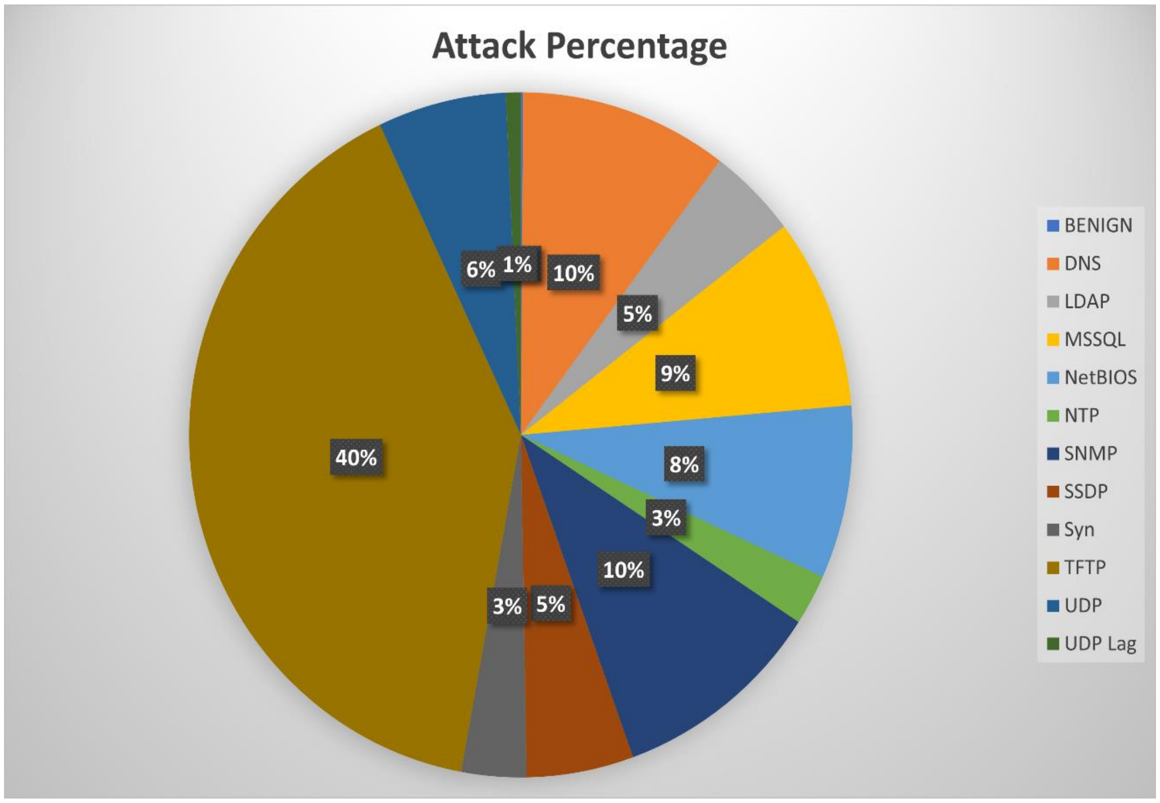


Figure 4.1: Attack percentages from the training set of CICDDoS2019.

Figure 4.1 shows the attack percentages from the training set of the dataset excluding WebDDoS which had 439 attacks recorded leading to a negligible percentage. Portmap was recorded only on the testing day and the number of such attacks was 186960, which is

0.92% of the testing set. The pie chart presented shows that the dataset is an imbalanced dataset, which can lead to a biasness of the model.

4.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) refers to the initial investigations on data prior to usage. EDA enables data usage in the most efficient way possible giving insights on the potential of the data. With this data focused approach, patterns and anomalies are looked for, data is visualized for a better understanding of feature significance. The process starts with questioning the data, followed by visualizing, analyzing, and modelling data. EDA techniques can be broadly classified into 2 types –

- Graphical Exploratory Data Analysis Techniques i.e., boxplot, histogram, scatter plot.
- Quantitative Exploratory Data Analysis Techniques i.e., variation, hypothesis testing.

The graphical and the non-graphical data analysis techniques can be univariate or multivariate, based on the number of variables.

In the thesis, the benchmark dataset CICDDoS-2019 being a significantly large one with a reasonable number of classes representing attack sub-types offered a wide range of scope for exploration of the data to gain insight.

4.2.1 Infinity/ null/ missing value

Detecting the missing values or the infinity values is a mandatory exploration for EDA. In the dataset chosen, among the original features, the feature 'Flow Bytes/s' had 12939 missing values. ' Flow Packets/s' also had values that are missing and infinite. Missing or infinite values are redundant data that can be handled with imputation.

4.2.2 Object Type Data

Two features, ‘SimillarHTTP’ and ‘Label’ had object data type and the rest of the features in the dataset were integer or float types. ‘Label’ being of object datatype here raises no concern.

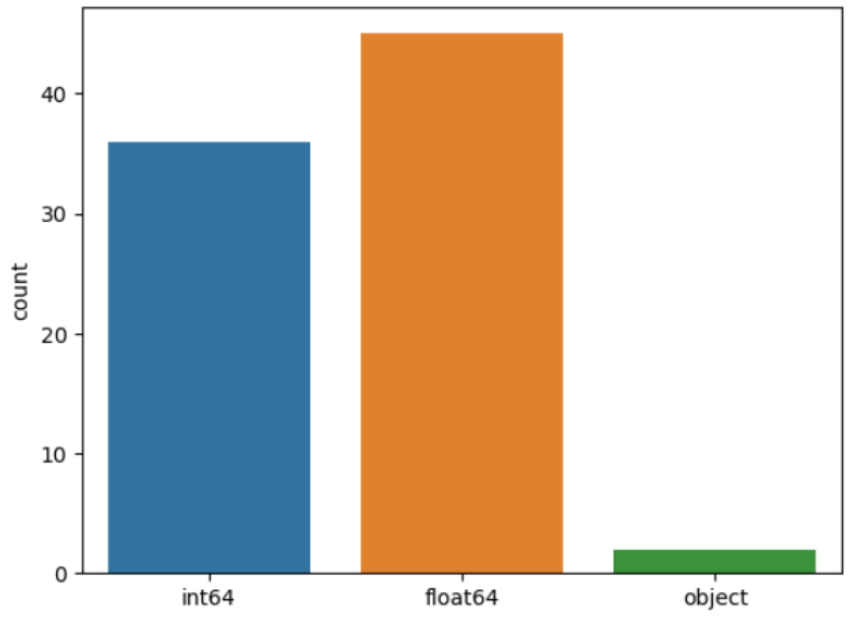


Figure 4.2: Datatypes in the dataset CICDDoS2019.

For the feature ‘SimillarHTTP’, the data being object type indicates potential trouble while encoding them with One-Hot Encoding (OHE). In OHE, only 1 element of a finite set has an index set to “1” and all other elements are assigned indices within the range of 0 to 1 less than the number of elements. It is a technique for encoding categorical data to numeric data as it is a required format for machine learning models.

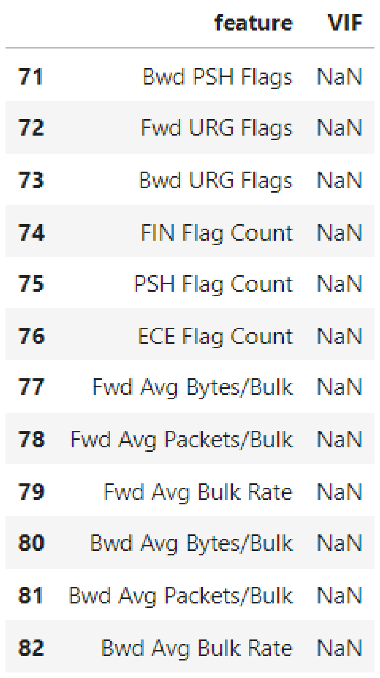
4.2.3 Collinearity

For a particular dataset when 2 variables are highly correlated and provide similar information on the variance of that dataset, it is known as collinearity. Multicollinearity involves 3 or more independent variables that are highly correlated. As the seemingly independent variables here are influencing each other, it affects the overall output of the model. As the variables are measuring the same thing, it can be seen as the same measurement is being taken multiple times which can make the understanding of dependent and independent variables a complicated one. Perfect multicollinearity can be thought of as an extreme case here which occurs when the relationship between the independent variables is exactly linear.

Detection of multicollinearity in data is important as it significantly reduces the statistical significance of the independent variables, even though the explanatory power of the model remains somewhat unreduced. Variance Inflation Factor (VIF) is a tool to measure multicollinearity. The formula for calculating VIF is,

$$VIF_i = \frac{1}{1 - {R_i}^2} \qquad \textbf{4.1}$$

; where ${R_i}^2$ refers to the unadjusted coefficient of determination for regressing the independent variable (i) on the remaining variables.



	feature	VIF
71	Bwd PSH Flags	NaN
72	Fwd URG Flags	NaN
73	Bwd URG Flags	NaN
74	FIN Flag Count	NaN
75	PSH Flag Count	NaN
76	ECE Flag Count	NaN
77	Fwd Avg Bytes/Bulk	NaN
78	Fwd Avg Packets/Bulk	NaN
79	Fwd Avg Bulk Rate	NaN
80	Bwd Avg Bytes/Bulk	NaN
81	Bwd Avg Packets/Bulk	NaN
82	Bwd Avg Bulk Rate	NaN

Figure 4.3: Features with perfect collinearity.

Variables are considered to be not correlated for VIF has the value of 1, moderately correlated for VIF is in the range of 1 to 5 and highly correlated if VIF is greater than 5. If VIF has the value of NaN (Not a Number) or undefined, it shows perfect collinearity for the independent variables.

For the dataset used here, measuring VIF gave NaN value for a few features, as shown in Figure 4.3.

4.2.4 Statistical Data Type

There are 3 types of statistical data – numerical, categorical, and ordinal.

- Numerical data has meaning as measurement and are also known as quantitative data.
- Categorical data represents characteristics of data and is qualitative. Categorical data can be assigned mathematical values, but they don't have any mathematical meaning.
- Ordinal data mixes numerical and categorical data. The numbers assigned to categories are not mere numbers, they have meanings.

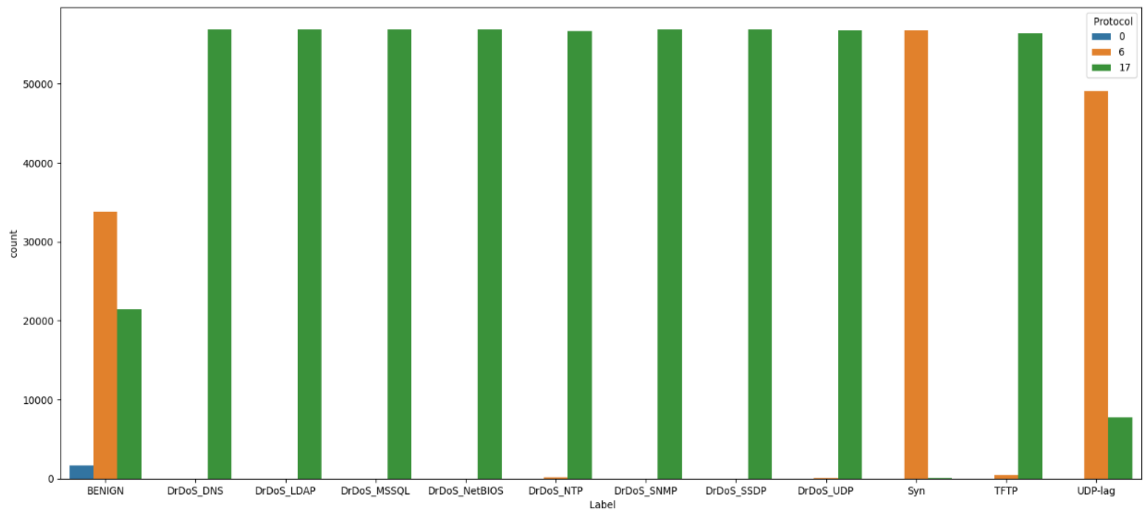


Figure 4.4: Countplot for the feature 'Protocol'.

The dataset, as shown in figure 4.2, very less data was of object data type. These can be counted as categorical features from the initial point of view. However, there is a lot of data in the dataset that are ordinal – which had the potential to play a vital role in model performance.

For the feature ‘Protocol’, from figure 4.4, it can be seen that there are only 3 protocols for the whole dataset classes. These 3 protocols, although are denoted by numbers, these numbers denote to the corresponding protocol categories like 6 for TCP, 17 for UDP and 0 for none.

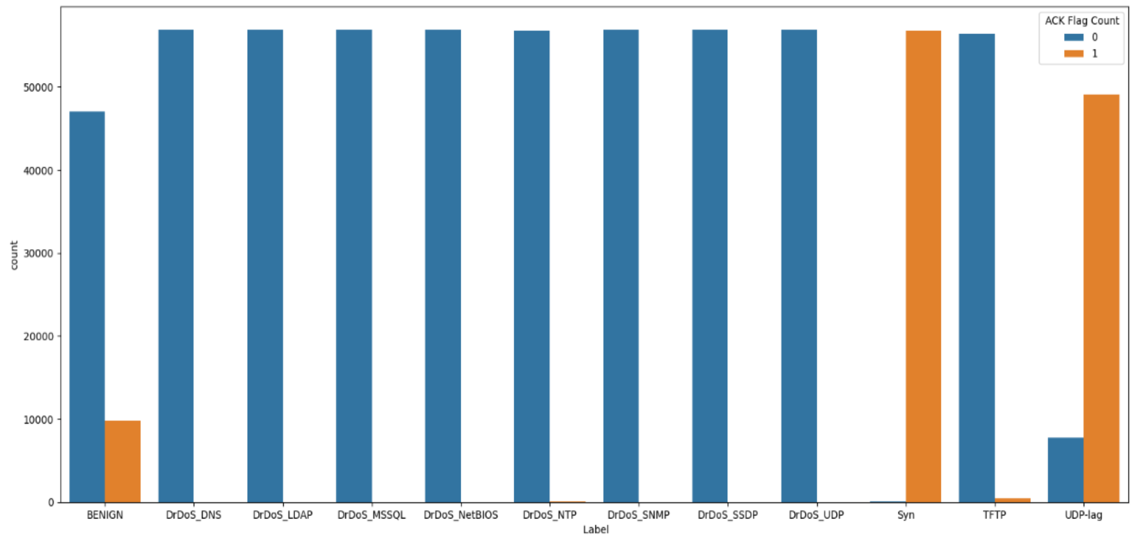


Figure 4.5: Countplot for the feature ‘ACK Flag Count’.

This feature, thus, is an ordinal feature which, for the sake of its meaning, can be considered as a categorical feature for training the model.

Similarly, for the feature ‘ACK Flag Count’, there are 2 categories as shown in Figure 4.5 – it will be either affirmative (1) or negative (0). This can also be considered as a categorical feature while training the model.

This idea of seeing the potential categories hidden in the guise of numbers was applicable for the some other features like, 'Fwd PSH Flags', ' RST Flag Count', 'URG Flag Count', ' CWE Flag Count', ' Inbound', ' Source Port', ' Destination Port'.

' Source IP', ' Destination IP' were treated as categorical features. As the IPv4 addresses are represented by a series of numbers separated by periods in the dataset, they could not be counted as numerical. As the numbers separated by periods were not referring to discrete classes like the ordinal features, they were considered to be categorical and moving forward, categorical embeddings were created from them.

4.2.5 Data with No Useful Interpretation

Not all data can be usable regardless of the problem statement. There were 2 columns in the dataset with data that didn't indicate any meaning for the detection of the attack.

- 'Unnamed: 0': This feature from the original dataset represents IDs for the captured packets. An ID is just a mere number for logging record, which logically, cannot really play a role as an attack characteristic.
- 'Flow ID': This feature is a combination of 'Source IP', 'Destination IP', 'Source Port', 'Destination Port', and 'Protocol'. All these features have individual columns as well; hence the series contains no information that is not otherwise present in the data.

4.3 Input Dataset Preparation

The benchmark dataset originally was an imbalanced dataset having 12 classes including the Benign class, as shown in figure 4.1. For feeding the model in the training stage, the dataset was first prepared and converted into an optimum data frame in the following way-

- **Creating a unified balanced data frame:** The data was imbalanced data, hence, to prepare unbiased learning material for the model, it was converted into a balanced data frame. Benign samples were first extracted from the 11 attack data files prepared from the recorded training day data of the dataset, merging of which produced a tabular data file containing 56,863 benign samples. Since the aim was to prepare a balanced dataset, 56,863 samples were randomly taken from each of these attack classes – 'DrDoS_DNS', 'DrDoS_LDAP', 'DrDoS_MSSQL', 'DrDoS_NetBIOS', 'DrDoS_NTP', 'DrDoS_SSDP', 'DrDoS_SNMP', 'Syn', ‘TFTP’, ‘UDPLag’ and ‘Portmap’. The data frames prepared from the random sampling [51] was then merged to prepare a unified balanced data frame containing 56,863 samples from each of 11 attack classes mentioned above and 1 benign class, making a total of 682356 samples having 88 features in total

- **Dropping redundant columns:** As per the exploratory data analysis mentioned in sections 4.2.3 and 4.2.5, the redundant features - 'Unnamed: 0', 'Flow ID', 'SimillarHTTP', 'Fwd Avg Bytes/Bulk', ' Fwd Avg Packets/Bulk', ' Fwd Avg Bulk Rate', ' Bwd Avg Bytes/Bulk', ' Bwd Avg Packets/Bulk', 'Bwd Avg Bulk Rate', ' Bwd PSH Flags', ' Fwd URG Flags', ' Bwd URG Flags', 'FIN Flag Count', ' PSH Flag Count', and ' ECE Flag Count' were dropped from the unified data frame as uncleaned data can lead to poor model performance [52].

  The feature ' Timestamp' indicates the packet capture time. As the data was recorded over an interval of time, the dataset could be considered as time-series data. For this type of data, gradual change of record is related to the passage of time. However, since the objective was to use the data to classify DDoS attack sub-types, timestamps were considered to hold little to no value in the prediction as the attacks can happen anytime of the day. Hence, this feature was also removed for the experiment.

- **Normalizing data:** Normalization [53] is a data preparation technique to fit numeric data in a common scale without losing any bit of it, with a view to improve the training stability and performance. The most commonly used normalization techniques are min-max scaling, z-score scaling, and constant factor normalization. From the prepared data frame, numerical data was normalized using min-max scaler, which transformed the numerical data into range 0 to 1. The mathematical formula for min-max scaling is,

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad \mathbf{4.2}$$

- **Replacing infinite values:** As per section 4.2.1 the infinite values in the features 'Flow Bytes/s' and ' Flow Packets/s' were replaced with the mode of the respective feature column.

- **Train- test- validation split:** Total data was split optimally [54] into 3 parts- 70% data for training, 20% data for testing and 10% data for validation.
- **Imputing for missing values:** Data imputation [55] refers to substituting missing values of a feature with a statistical value of the feature. After the train-test split, in the training data, the missing values in the features 'Flow Bytes/s' and ' Flow Packets/s' were imputed with the statistical mode of their respective feature column. This was done only on the training data to avoid data leakage.
- **Label encoding:** The class labels in the data frame had object data type. Before training the model, the labels were encoded into 12 different numeric values using Label Encoder.

4.4 Feature Engineering

Creating new features from the existing ones with a view to enhancing model performance is a feature engineering technique. This technique was used for the experiments of this thesis. Since there were no mentionable outliers, finding range was a reasonable decision. Statistical range indicates the spread of data distribution and is a measure of variability which determines how well the results can be generalized. Backed up by this logic, 8 new features were created here from the existing features - ' Fwd Packet Length Range', ' Bwd Packet Length Range', ' Flow IAT Range', ' Fwd IAT Range', ' Bwd IAT Range', ' Packet Length Range', ' Active Range', and ' Idle Range', where $Range\ (x) = Max\ (x) - Min\ (x)$. For example, the feature 'Idle Range' was created by subtracting the two features, 'Idle Max' and 'Idle Min'. Creating these 8 new features provided more attributes to the model for distinguishing more accurately amongst the classes.

4.5 Feature Selection

Reducing the dimension of input data to a model can play a role in optimization of performance. Feature selection refers to selecting relevant data leaving the noise data. It reduces overfitting and makes the model more interpretable.

As per the taxonomy, the feature selection techniques are classified into wrapper, filter, embedded, and hybrid methods. Figure 4.6 shows the appropriate feature selection method choices based on the types of input and output variable. The data used in this work had both numerical and categorical input and categorical output – which directed to the feature selection methods – Analysis of Variance (ANOVA), chi-2, and Mutual Information (MI). These 3 fall into the category of filter methods.

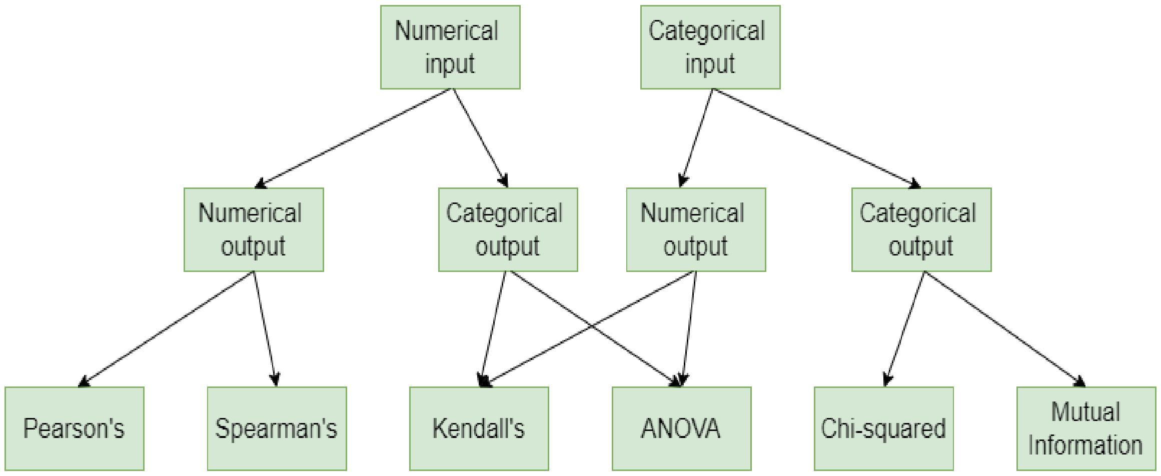


Figure 4.6: Appropriate feature selection methods depending on variable types.

4.5.1 Mutual Information

MI [43] between two random variables measures the non-linear relationship between the variables indicating how much information can be obtained from one random variable from observing the other. If the random variables are independent, MI will be 0. When the MI is higher, the uncertainty is less. MI helps to select the comparatively relevant features from a huge dataset like CICDDoS2019.

MI scores were calculated for each of the features of the prepared data frame and sorted to filter out the top relevant features. The feature selection method from the scikit-learn library was used for the purpose.

4.5.2 Analysis of Variance

ANOVA[42] is a tool to determine the influence of independent variables on the dependent variable. This statistical technique indicates if 2 or more groups have means significantly different from each other. The result obtained from the ANOVA formula is known as the F statistic or F-ratio which allows the analysis of multiple data groups with a view to determining the variability existing within the data samples and between the data samples. This F-ratio will be almost 1 if no true variance exists between the data groups. The formula for ANOVA is,

$$F = \frac{Mean\ sum\ of\ squares\ between\ the\ groups}{Mean\ squares\ of\ errors} \qquad \textbf{4.3}$$

; where F is the ANOVA coefficient.

ANOVA scores were calculated for each of the features of the prepared data frame and sorted to filter out the top relevant features. The feature selection method from the scikit-learn library was used for the purpose.

4.5.3 Chi-squared

It is a statistical test used to examine the differences between categorical variables from a random sample. It says whether categorical variables are independent in influencing the test results. The formula for chi-2[41] is,

$$x_c{}^2 = \sum \frac{(O_i - E_i)^2}{E_i} \qquad \textbf{4.4}$$

; where, c = degrees of freedom, O = observed value(s), E = Expected value(s)

Chi-2 scores were calculated for each of the features of the prepared data frame and sorted to filter out the top relevant features. The feature selection method from the scikit-learn library was used for the purpose.

4.5.4 Ensemble Feature Selection

From the selected and sorted features from section 4.3.1-4.3.3, an ensemble feature selection technique was created. The formula for scaling each feature score was,

$$E_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad \mathbf{4.5}$$

On the sorted ANOVA, chi-2, and MI scores, min-max scaling was used. The ensemble score is the mean of these 3 scaled scores obtained – which, if represented by a mathematical equation, will be -

$$E_{ensemble} = \frac{E_{scaled(ANOVA)} + E_{scaled(chi-2)} + E_{scaled(MI)}}{3} \quad \mathbf{4.6}$$

The features considered for the training of the model and their respecting ensembled feature scores are shown in table 4.1.

Table 4.1: Ensemble feature scores for the top 58 features.

Feature	Ensemble Score
Min Packet Length	1
Fwd Packet Length Mean	0.9907251427839014
Avg Fwd Segment Size	0.9907251427839014
Average Packet Size	0.9834499082520399
Fwd Packet Length Min	0.9830263155868882
Packet Length Mean	0.9433361460239803
Fwd Packet Length Max	0.6878712657601843
Inbound	0.48760390825909616
ACK Flag Count	0.4086757924559665
Source Port	0.3873849548447954
Protocol	0.36801084259675126

Flow Bytes/s	0.2352942948431845
Max Packet Length	0.21079465104691222
Down/Up Ratio	0.0945562998088002
act_data_pkt_fwd	0.09300034441243475
URG Flag Count	0.08152834511345072
Flow Packets/s	0.06831411492958293
Fwd Packets/s	0.05894352792215577
Bwd Packet Length Min	0.031809598700893754
Init_Win_bytes_forward	0.03094544073181942
CWE Flag Count	0.02647575021996576
Total Length of Fwd Packets	0.024389535981127877
Subflow Fwd Bytes	0.024389535981127877
Bwd Packet Length Mean	0.018941856758371175
Avg Bwd Segment Size	0.018941856758371175
Fwd PSH Flags	0.01729783777609373
RST Flag Count	0.01729783777609373
Packet Length Std	0.015223092435409862
Destination Port	0.01413707791134781
Fwd Packet Length Std	0.013062816676077603
Init_Win_bytes_backward	0.012106842619165734
Fwd Packet Length Range	0.012007960842104363
Bwd Packet Length Std	0.01014794822594065
Flow IAT Mean	0.009815512288695174
Flow IAT Std	0.009626448694317816

Fwd IAT Mean	0.00945310984681248
Packet Length Range	0.009201498179249546
Flow Duration	0.009048521323008873
Fwd IAT Std	0.008981288588803777
Bwd Packet Length Max	0.008917014629826903
Fwd IAT Total	0.008720229687208108
Flow IAT Max	0.008055900446993522
Flow IAT Range	0.008020156520650482
Bwd IAT Total	0.007709172594791566
Fwd IAT Max	0.0076229117108505455
Bwd IAT Min	0.007609813613209473
Fwd IAT Range	0.007593784972966081
Idle Max	0.007089049560989116
Idle Mean	0.006972811180315966
Bwd Packet Length Range	0.006794851560529678
Idle Min	0.006630827069863016
Bwd IAT Max	0.0055734844090985294
Bwd IAT Range	0.005573475806935316
Idle Range	0.005408933011942262
Idle Std	0.003925933960304446
Bwd IAT Std	0.0038414573518576298
min_seg_size_forward	0.002742690283320585
Bwd IAT Mean	0.0025586779141427286

The 58 features mentioned in the table were considered for training the model. ‘Source IP’ and ‘Destination IP’ were also added to the list making a total of 60 features to train the model. The reason behind not considering these 2 features for calculation the ensemble feature score is, that MI, chi-2 and ANOVA feature selection methods cannot deal with categorical data.

Chapter 5

5 Model Development and Evaluation

This chapter records the required hardware configuration for model development, describes the architecture of the developed models and their performance on the input dataset prepared in section 4.3. A comparison of the obtained results with the existing works is also presented in the concluding section of the chapter.

5.1 Development Resources

Server: A dedicated server with high configuration was employed to aid the model development and training process. The hardware configuration of the server used is presented in table 5.1.

Table 5.1: Hardware configuration of the server.

Processor	12th Gen Intel(R) Core (TM) i9-12900K, 3.20 GHz
RAM	128 GB
Operating System	64-bit, x64-based processor
Windows version	Windows 11 Pro
GPU	NVIDIA GeForce RTX 3090
Dedicated GPU Memory	24 GB
CUDA Version	12.0

Personal computer (PC): For carrying out the data analyzing and preprocessing tasks, and developing the classic machine learning models, a PC with the hardware configuration mentioned in table 5.2 was used.

Table 5.2: Hardware configuration of the PC.

Processor	Intel(R) Core (TM) i7-10510U CPU @ 1.80GHz 2.30 GHz
RAM	32 GB
Operating System	64-bit, x64-based processor
Windows version	Windows 10
GPU	NVIDIA GeForce MX250
Dedicated GPU Memory	4 GB
CUDA Version	12.0

5.2 Deep Learning Algorithms

The tabular data that is being dealt with here is heterogenous data as it has data of multiple data types. Even though DL algorithms work best for homogenous data, for the processed data frame made for training the model in this thesis, DL algorithms [56] handling the deep tabular components like TabMLP, TabResnet, TabPerceiver and the transformers for tabular data like TabTransformer, TabFastFormer, FT- Transformer showed promising results.

The sub-sections will focus on the development of these deep learning models and highlight the performance of each model.

5.2.1 TabMLP

Multi-layer perceptron (MLP) is a fully connected multi-layer neural network algorithm for supervised learning in which the mapping between input(s) and output is not linear. It is an example of Artificial Neural Network (ANN) that has 1 input layer, 1 output layer, and 1 or many hidden layers. The neurons stacked together in the hidden layers can use any arbitrary activation function unlike a perceptron which generates outputs with real values in the range of 0 to 1 or -1 to 1. In TabMLP architecture [56], as shown in figure 5.1, embeddings created to represent the categorical features and batch normalized continuous features are concatenated and then passed through MLP.

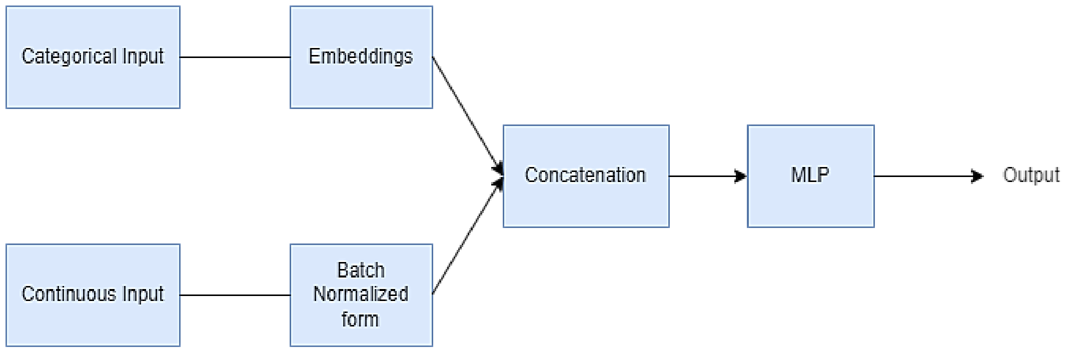


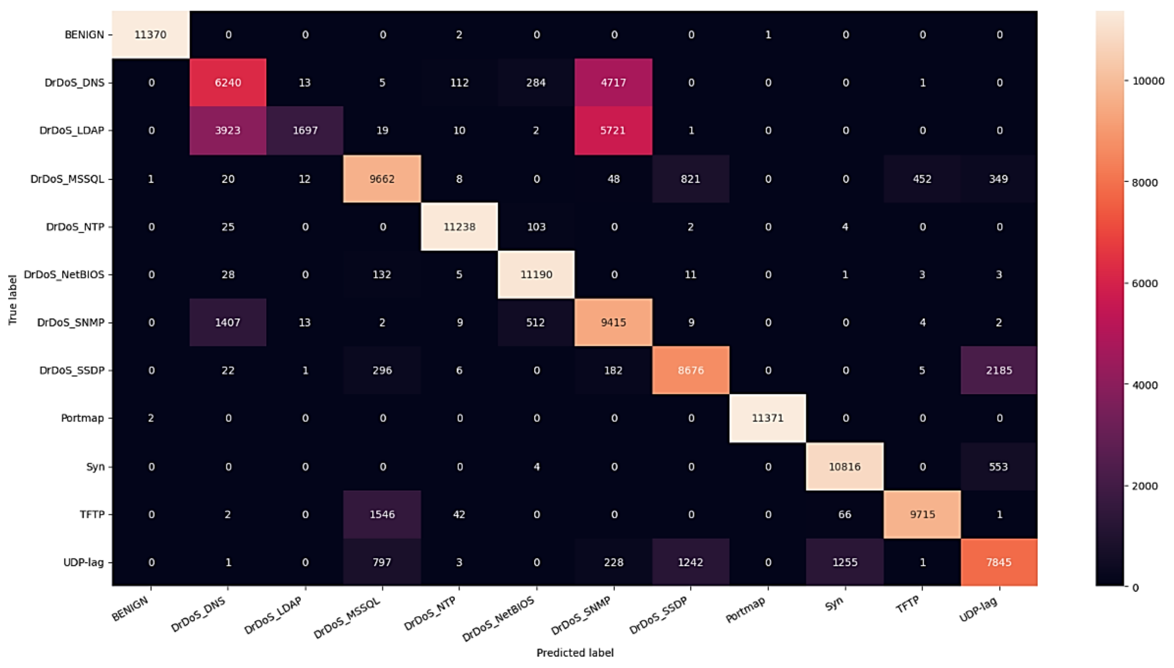
Figure 5.1: TabMLP model architecture.

In the TabMLP model developed as a part of this thesis, embeddings of different sizes were created for each feature representing categories as mentioned in section 4.2.4, and the continuous features or the numerical ones were batch normalized to be concatenated with the embeddings created. For the model parameters, the hidden layer dimensions or the number of dense layer neurons were taken to be 200 and 100 and 50% neurons were randomly dropped in every iteration. Activation functions for the dense layers of MLP was chosen to be ‘Leaky ReLU’ having negative slope of 0.01.

The evaluation metrics obtained are recorded in table 5.3 and figure 5.2 shows the confusion matrix for the developed model.

Table 5.3: Performance metrics for TabMLP.

Model	Precision	Recall	Accuracy	F1
TabMLP	0.836	0.800	0.800	0.788



True label \ Predicted label	BENIGN	DrDoS_DNS	DrDoS_LDAP	DrDoS_MSSQL	DrDoS_NTP	DrDoS_NetBIOS	DrDoS_SNMP	DrDoS_SSDP	Portmap	Syn	TFTP	UDP-lag
BENIGN	11370	0	0	0	2	0	0	0	1	0	0	0
DrDoS_DNS	0	6240	13	5	112	284	4717	0	0	0	1	0
DrDoS_LDAP	0	3923	1697	19	10	2	5721	1	0	0	0	0
DrDoS_MSSQL	1	20	12	9662	8	0	48	821	0	0	452	349
DrDoS_NTP	0	25	0	0	11238	103	0	2	0	4	0	0
DrDoS_NetBIOS	0	28	0	132	5	11190	0	11	0	1	3	3
DrDoS_SNMP	0	1407	13	2	9	512	9415	9	0	0	4	2
DrDoS_SSDP	0	22	1	296	6	0	182	8676	0	0	5	2185
Portmap	2	0	0	0	0	0	0	0	11371	0	0	0
Syn	0	0	0	0	0	4	0	0	0	10816	0	553
TFTP	0	2	0	1546	42	0	0	0	0	66	9715	1
UDP-lag	0	1	0	797	3	0	228	1242	0	1255	1	7845

Figure 5.2: Confusion Matrix for TabMLP.

5.2.2 TabResnet

TabResnet has a similar architecture to TabMLP, the mentionable difference is the additional ResNet blocks in the TabResnet architecture [56]. As shown in figure 5.3, embeddings are created from the categorical inputs and concatenated with the batch normalized continuous inputs. Batch normalization of the continuous inputs is optional here, hence the dotted lines. If batch normalization is not chosen, the continuous inputs are concatenated with the embeddings coming out of the Resnet blocks. The MLP block here is optional- the final output can be the output from the MLP layer or the output from the Resnet Block or the output from the last concatenation layer.

In the TabResnet model developed as a part of this thesis, embeddings of different sizes were created for each feature representing categories as mentioned in section 4.2.4, and the continuous features or the numerical ones were batch normalized to be concatenated with the embeddings created. Batch normalization was chosen here in the final TabResnet model developed as it generated better outcome than when it was not chosen.

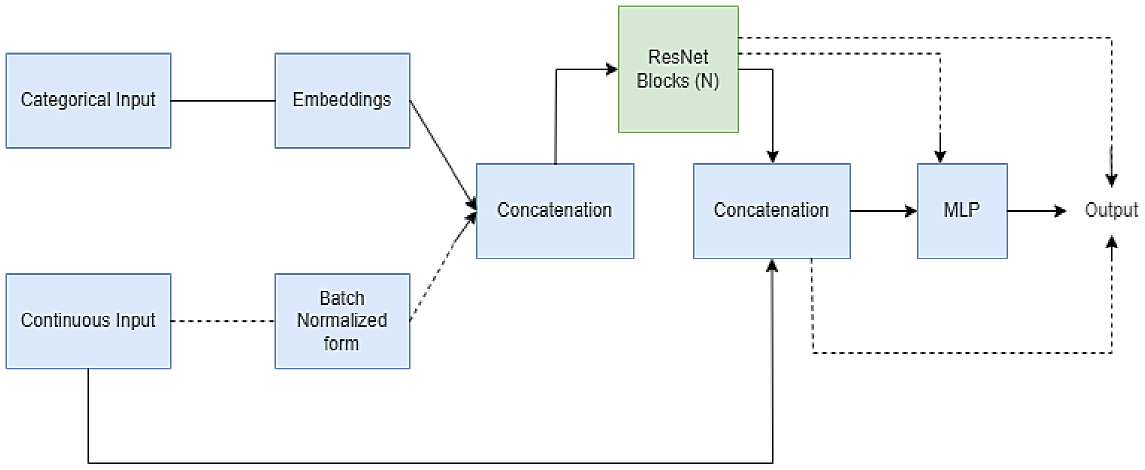
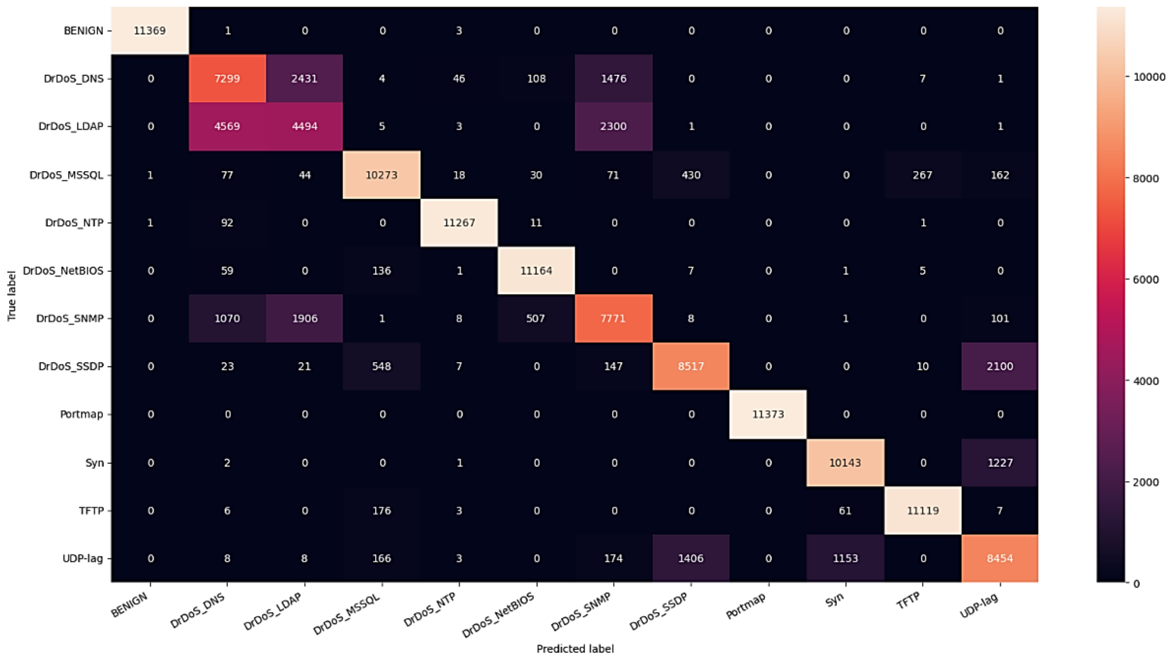


Figure 5.3: TabResnet model architecture.

For the model parameters, the hidden layer dimensions or the number of dense layer neurons were taken to be 100 and 50 and block dimensions for the Resnet block



	BENIGN	DrDoS_DNS	DrDoS_LDAP	DrDoS_MSSQL	DrDoS_NTP	DrDoS_NetBIOS	DrDoS_SNMP	DrDoS_SSDP	Portmap	Syn	TFTP	UDP-lag
BENIGN	11369	1	0	0	3	0	0	0	0	0	0	0
DrDoS_DNS	0	7299	2431	4	46	108	1476	0	0	0	7	1
DrDoS_LDAP	0	4569	4494	5	3	0	2300	1	0	0	0	1
DrDoS_MSSQL	1	77	44	10273	18	30	71	430	0	0	267	162
DrDoS_NTP	1	92	0	0	11267	11	0	0	0	0	1	0
DrDoS_NetBIOS	0	59	0	136	1	11164	0	7	0	1	5	0
DrDoS_SNMP	0	1070	1906	1	8	507	7771	8	0	1	0	101
DrDoS_SSDP	0	23	21	548	7	0	147	8517	0	0	10	2100
Portmap	0	0	0	0	0	0	0	0	11373	0	0	0
Syn	0	2	0	0	1	0	0	0	0	10143	0	1227
TFTP	0	6	0	176	3	0	0	0	0	61	11119	7
UDP-lag	0	8	8	166	3	0	174	1406	0	1153	0	8454

Figure 5.4: Confusion Matrix for TabResnet.

were chosen to be 200, 100 and 100 which gave two blocks with input/output of 200/100 and 100/100 respectively. Activation functions for the Resnet block was chosen to be

‘Leaky ReLU’ having negative slope of 0.01 and for the MLP layer it was chosen to be ‘ReLU’.

The evaluation metrics obtained are recorded in table 5.4 and figure 5.4 shows the confusion matrix for the developed model.

Table 5.4: Performance metrics for TabResnet.

Model	Precision	Recall	Accuracy	F1
TabResnet	0.856	0.846	0.846	0.833

5.2.3 TabTransformer

TabTransformer[6] is one of the strongly recommended models for tabular data. The architecture is similar to TabMLP, the addition in a TabTransformer is the Transformer[57] block. Transformer is a neural network that learns context in sequential data.

It uses the concept of self-attention and consists of a multi-head self-attention layer that is followed by feed-forward layer and after each layer element-wise addition

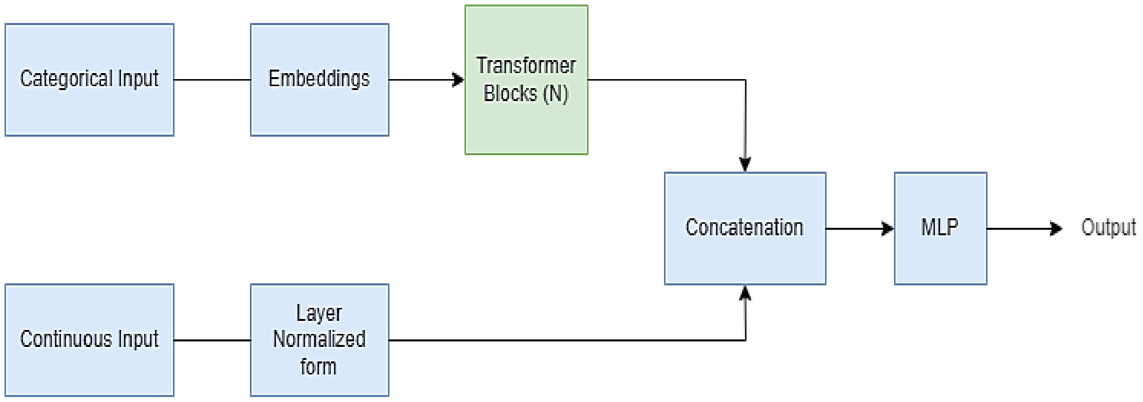
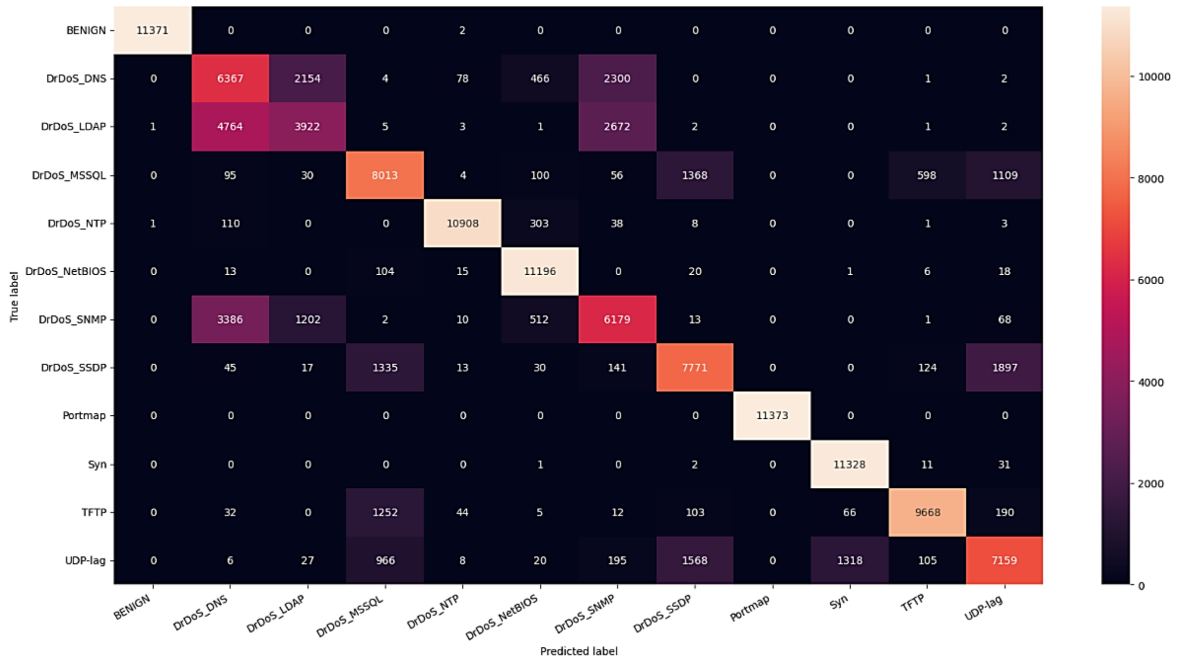


Figure 5.5: TabTranformer model architecture.

and layer-normalization takes place. As shown in figure 5.5, embeddings are created and stacked from the categorical inputs that go through a series of transformer blocks.

Embeddings here are of the class, shared embeddings- the list of the embedding columns share the same weights. The output of the transformer blocks is then concatenated with the layer normalized continuous inputs and sent through MLP layer, the output of which is the final output.

In the TabTransformer model developed as a part of this thesis, embeddings of different sizes were created for each feature representing categories as mentioned in section 4.2.4, and the continuous features or the numerical ones were layer normalized to be concatenated with the embeddings created. For the model parameters, the number of transformer blocks was 4, the number of attention heads per transformer block was 8. For the transformer, multi-head attention dropout had a value of 0.2, feed forward network dropout had a value of 0.1 and the activation function was 'GeLU'. The activation function used for MLP was 'ReLU' and the dropout value for the MLP layer was 0.1.



True label \ Predicted label	BENIGN	DrDoS_DNS	DrDoS_LDAP	DrDoS_MSSQL	DrDoS_NTP	DrDoS_NetBIOS	DrDoS_SNMP	DrDoS_SSDP	Portmap	Syn	TFTP	UDP-lag
BENIGN	11371	0	0	0	2	0	0	0	0	0	0	0
DrDoS_DNS	0	6367	2154	4	78	466	2300	0	0	0	1	2
DrDoS_LDAP	1	4764	3922	5	3	1	2672	2	0	0	1	2
DrDoS_MSSQL	0	95	30	8013	4	100	56	1368	0	0	598	1109
DrDoS_NTP	1	110	0	0	10908	303	38	8	0	0	1	3
DrDoS_NetBIOS	0	13	0	104	15	11196	0	20	0	1	6	18
DrDoS_SNMP	0	3386	1202	2	10	512	6179	13	0	0	1	68
DrDoS_SSDP	0	45	17	1335	13	30	141	7771	0	0	124	1897
Portmap	0	0	0	0	0	0	0	0	11373	0	0	0
Syn	0	0	0	0	0	1	0	2	0	11328	11	31
TFTP	0	32	0	1252	44	5	12	103	0	66	9668	190
UDP-lag	0	6	27	966	8	20	195	1568	0	1318	105	7159

Figure 5.6: Confusion Matrix for TabTransformer.

Using the basic TabTransformer model, generated the evaluation metrics presented in table 5.5. Although it did not exceed the performance of the previous 2 architectures, this led to experimenting with the other transformer models that showed promising results, as mentioned in the next section.

Table 5.5: Performance metrics for TabTransformer.

Model	Precision	Recall	Accuracy	F1
TabTansformer	0.772	0.771	0.771	0.768

5.2.4 FT- Transformer

Like TabTransformers, FT-Transformers also use Transformers and the differences between these two are,

- To pass the continuous features as well to the Transformer block along with the categorical embeddings, embeddings of continuous or numeric features are created. These embeddings are linear, and each numeric feature has a separate embedding layer.
- A special classification token (CLS) is used for the output in an FT-Transformer. The first token of every sequence is always a CLS -The final hidden state corresponding to this token is used as the aggregate sequence representation for classification tasks[58]. Embeddings are also created for the CLS token.

The numerical, categorical and CLS embeddings are augmented in the FT-Transformer, as shown in figure 5.7.

In the FT-Transformer model developed as a part of this thesis, embeddings were created for each feature representing categories as mentioned in section 4.2.4, and for the continuous features. For the model parameters, the number of FT-Transformer blocks used was 6, the hidden layers had a dropout of 0.5 and batch normalization was used in the dense layers. The activation function used was ‘ReGLU’. The size of the output tensor containing the predictions of the model was 12.

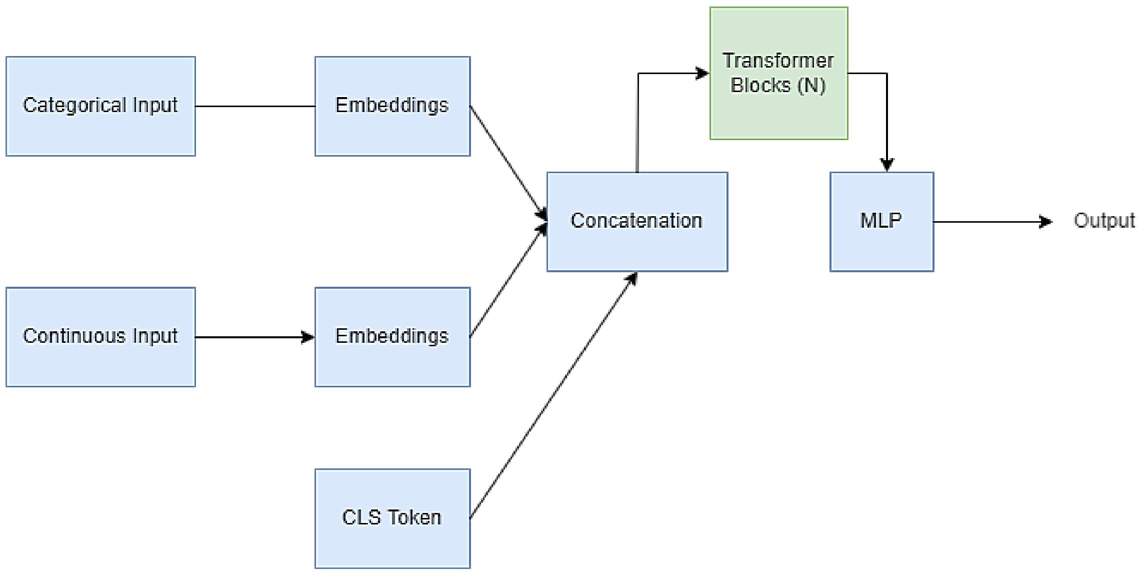
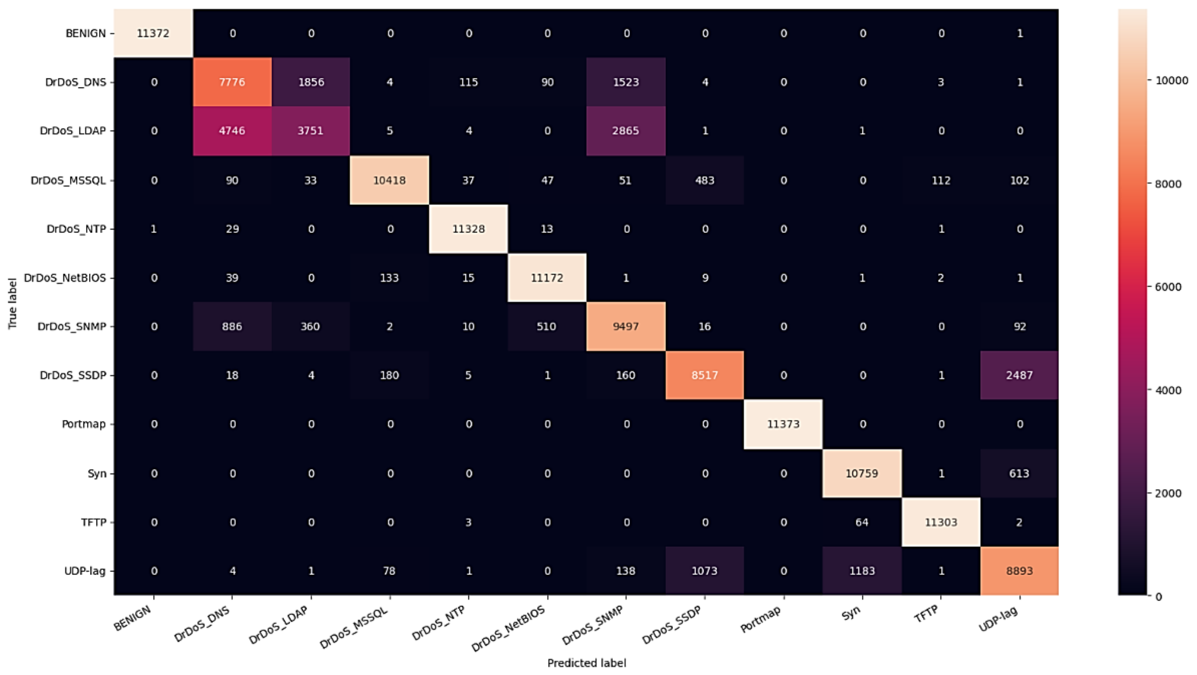


Figure 5.7: FT-Tranformer model architecture.



	BENIGN	DrDoS_DNS	DrDoS_LDAP	DrDoS_MSSQL	DrDoS_NTP	DrDoS_NetBIOS	DrDoS_SNMP	DrDoS_SSDP	Portmap	Syn	TFTP	UDP-lag
BENIGN	11372	0	0	0	0	0	0	0	0	0	0	1
DrDoS_DNS	0	7776	1856	4	115	90	1523	4	0	0	3	1
DrDoS_LDAP	0	4746	3751	5	4	0	2865	1	0	1	0	0
DrDoS_MSSQL	0	90	33	10418	37	47	51	483	0	0	112	102
DrDoS_NTP	1	29	0	0	11328	13	0	0	0	0	1	0
DrDoS_NetBIOS	0	39	0	133	15	11172	1	9	0	1	2	1
DrDoS_SNMP	0	886	360	2	10	510	9497	16	0	0	0	92
DrDoS_SSDP	0	18	4	180	5	1	160	8517	0	0	1	2487
Portmap	0	0	0	0	0	0	0	0	11373	0	0	0
Syn	0	0	0	0	0	0	0	0	0	10759	1	613
TFTP	0	0	0	0	3	0	0	0	0	64	11303	2
UDP-lag	0	4	1	78	1	0	138	1073	0	1183	1	8893

Figure 5.8: Confusion Matrix for FT-Transformer.

The evaluation metrics obtained are recorded in table 5.6 and figure 5.8 shows the confusion matrix for the developed model.

Table 5.6: Performance metrics for FT-Transformer.

Model	Precision	Recall	Accuracy	F1
FT-Transformer	0.851	0.851	0.851	0.846

5.2.5 FastFormer

FastFormer[7] is also another transformer based model that is comparatively computationally efficient. In the FastFormer model developed as a part of this thesis, embeddings were created for each feature representing categories as mentioned in section 4.2.4. For the model parameters, the number of FastFormer blocks used was 2 and the number of attention heads per FastFormer block was 4. The activation function used was ‘ReLU’.

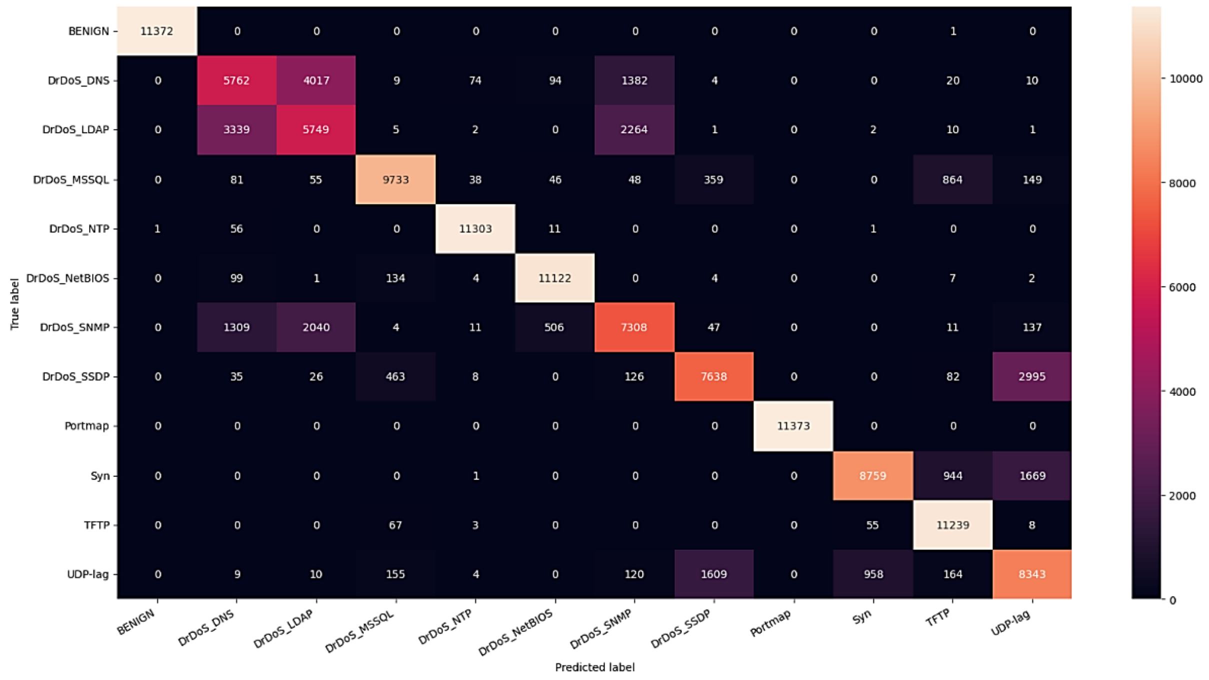


Figure 5.9: Confusion Matrix for FastFormer.

The evaluation metrics obtained are recorded in table 5.7 and figure 5.9 shows the confusion matrix for the developed model.

Table 5.7: Performance metrics for FastFormer.

Model	Precision	Recall	Accuracy	F1
FastFormer	0.830	0.833	0.833	0.828

5.2.6 Perceiver

Perceiver[8] is also another transformer based model that is comparatively computationally efficient. The model uses an asymmetric attention mechanism to iteratively densify inputs to allow it to scale to handle very large inputs [8]. In the Perceiver model developed as a part of this thesis, embeddings were created for the categorical and continuous inputs. For the model parameters, the number of Perceiver blocks used was 1, number of transformer encoder blocks per latent

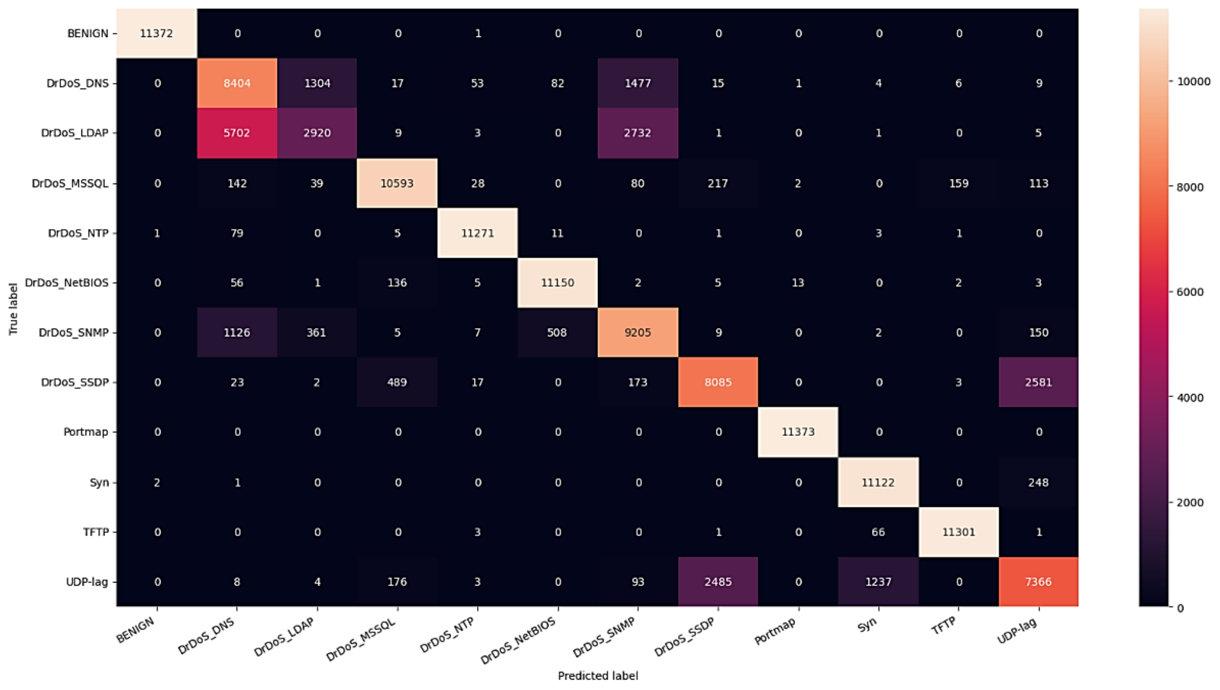


Figure 5.10: Confusion Matrix for Perceiver.

transformer was 3, number of attention heads per latent transformer was 2, and latent dimension was 32. The activation function used was ‘GeGLU’.

The evaluation metrics obtained are recorded in table 5.8 and figure 5.10 shows the confusion matrix for the developed model.

Table 5.8: Performance metrics for Perceiver.

Model	Precision	Recall	Accuracy	F1
Perceiver	0.836	0.837	0.837	0.828

5.3 Bagging Algorithms

Section 2.3.3 describes the architecture and the working mechanism of bagging algorithms in general and an example of bagging algorithms is Random Forest (RF). RF is an ensemble of a large number of individual decision trees that provide a class prediction and the class getting the majority vote becomes the final model prediction.

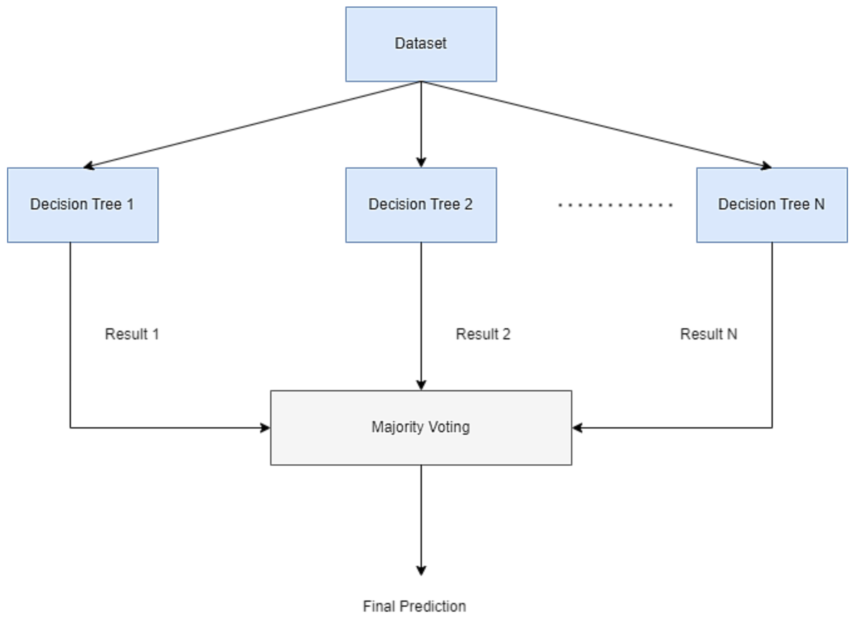


Figure 5.11: Schematic illustration of Random Forest.

The ensemble prediction from uncorrelated models is more accurate than the individual predictions as the individual trees protect each other from their errors.

In the RF model developed as a part of this thesis, for the model parameters, the number of trees in the forest was 100, criterion was ‘gini’, minimum number of samples required to split an internal node was 2, number of features to consider when looking for the best split was the ‘square root’ of the number of total features. For training the model, as per the section 4.2.3, all the ordinal features representing categories were considered before selecting features as per the ensemble feature selection described in section 4.3, and the categorical features ‘Source IP’ and ‘Destination IP’ were dropped for this model.

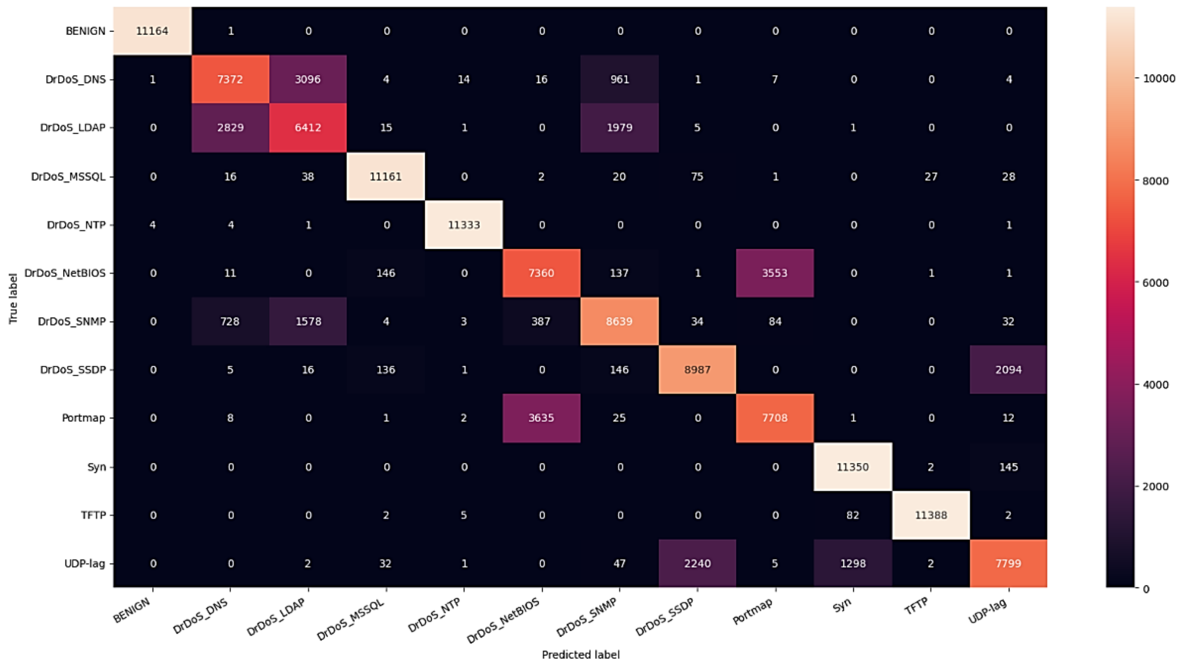


Figure 5.12: Confusion Matrix for RF.

The evaluation metrics obtained are recorded in table 5.9 and figure 5.12 shows the confusion matrix for the developed model.

Table 5.9: Performance metrics for RF.

Model	Precision	Recall	Accuracy	F1
RF	0.809	0.811	0.811	0.810

5.4 Boosting Algorithms

As mentioned in section 2.4, boosting is an ensemble technique when a series of decision trees participates in voting classification. Unlike bagging approach, each tree in boosting is developed considering the output from the previous trees. In this thesis, XGBoost, LGBM, CatBoost models were developed and trained with the processed data as the tree ensemble models (i.e., XGBoost) are usually recommended for solving classification and regression problems with tabular data[59].

5.4.1 XGBoost

XGBoost (eXtreme Gradient Boosting) is the more powerful version of the gradient boost with a significantly higher predictive power.It possesses both a linear model and the tree learning algorithm which makes it almost 10 times faster than the other gradient booster algorithms [11]. This boosting technique, being a regularized one, reduces model over-fitting. XGBoost supports both- stochastic and regularized gradient boosting. Sparse awareness, block structure, and continued training are some of the key implementation features of the algorithm. One of features that makes XGBoost stand out is its scalability. It is due to several algorithmic optimizations like handling sparse data as it is a theoretically justified weighted quantile sketch procedure that enables handling instance weights in approximate tree learning and working for parallel and distributed computing makes learning faster which enables quicker model exploration [11].

In the XGBoost model[60] developed in this thesis, after tuning the hyper-parameters using random search[61], the learning rate was taken to be 0.3, value of maximum depth was taken to be 6, L2 regularization value was taken to be 1. The grow policy for the model was ‘depthwise’, and the learning objective was chosen to be ‘softprob’. For training the model, like the RF model, ‘Source IP’ and ‘Destination IP’ were dropped from the preprocessed data.

The evaluation metrics obtained are recorded in table 5.10 and figure 5.13 shows the confusion matrix for the developed model.

Table 5.10: Performance metrics for XGBoost.

Model	Precision	Recall	Accuracy	F1
XGBoost	0.841	0.838	0.838	0.836

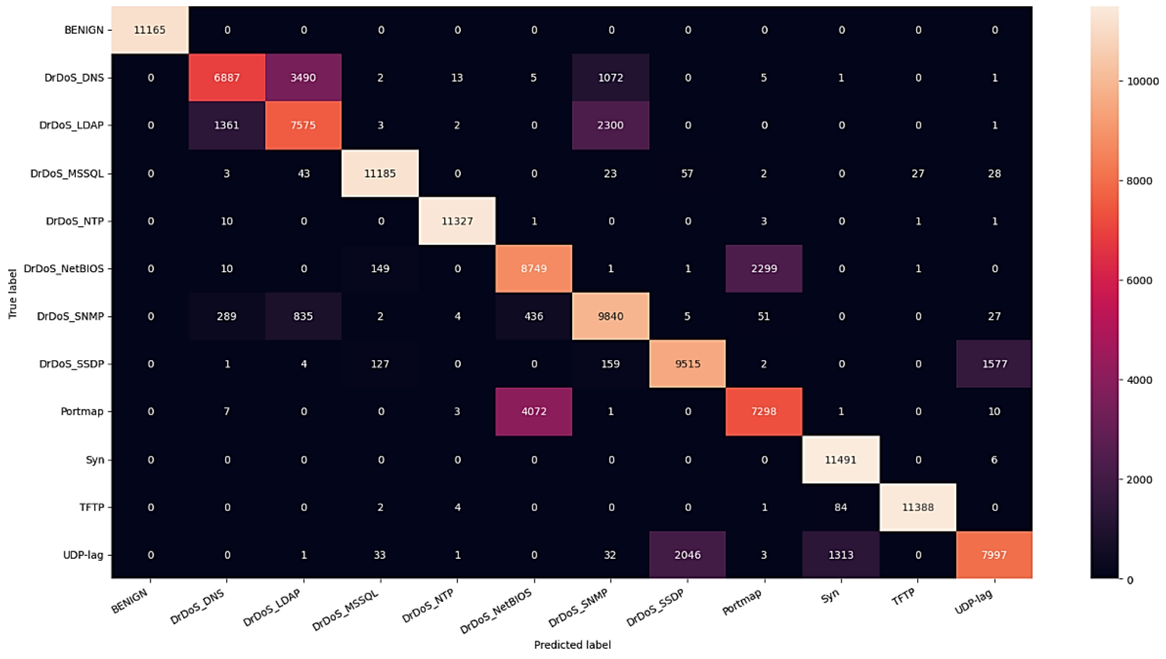


Figure 5.13: Confusion Matrix for XGBoost.

5.4.2 LightGBM

For larger datasets with higher feature dimensions, the performance of XGBoost is not satisfactory as all the data instances of each feature needs scanning for the estimation of the information gain of all possible split points which can be very time consuming. To solve this problem, Guolin et al. [10] proposed an integration of two novel techniques - Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) with the existing gradient boosting algorithm. GOSS keeps all the instances with large gradients and performs random sampling on the instances with small gradients and EFB bundles the mutually exclusive features using a greedy algorithms to reduce the number of features [10]. LightGBM uses histogram-based algorithms and supports applications like regression, binary classification, multi-classification, cross-entropy, and LambdaRank.

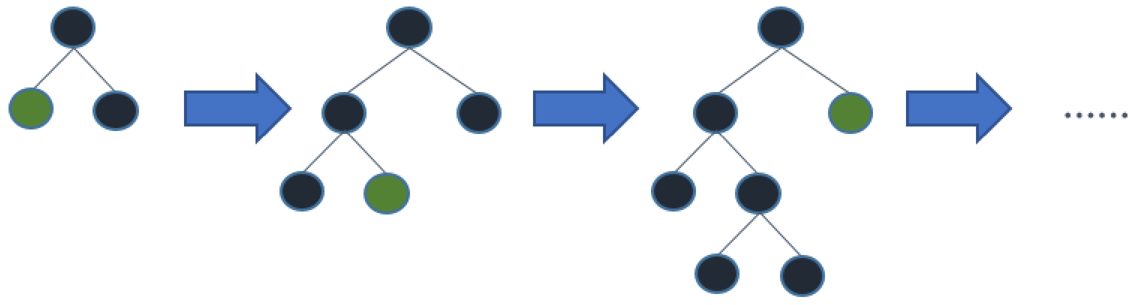


Figure 5.14: Tree expansion in LightGBM [62]**.**

LightGBM grows trees leaf-wise (best-first) which means will choose the leaf with maximum delta loss to grow; such leaf-wise algorithms result in lower loss than level-wise algorithms that most of the decision tree algorithms use [62]. In the case of small datasets, growing leaf-wise may cause overfitting but this problem can be handled by limiting the tree depth. This algorithm has more than 100 control parameters to work with.

In the LGBM model developed in this thesis, after tuning the hyper-parameters using Optuna[63], number of trees was taken to be 100, learning rate was taken to be 0.1 and number of leaves was taken to be 31. The objective was ‘multiclass’ and traditional gradient boosting tree was used for the model. The features, ‘Source IP’ and ‘Destination IP’ were dropped from the preprocessed data.

Table 5.11: Performance metrics for LGBM.

Model	Precision	Recall	Accuracy	F1
LGBM	0.845	0.842	0.842	0.840

The evaluation metrics obtained are recorded in table 5.11 and figure 5.14 shows the confusion matrix for the developed model.

5.4.3 CatBoost

Unlike the rest of the gradient boosting algorithms that convert categorical features to numbers during pre-processing and before training, CatBoost can handle categorical

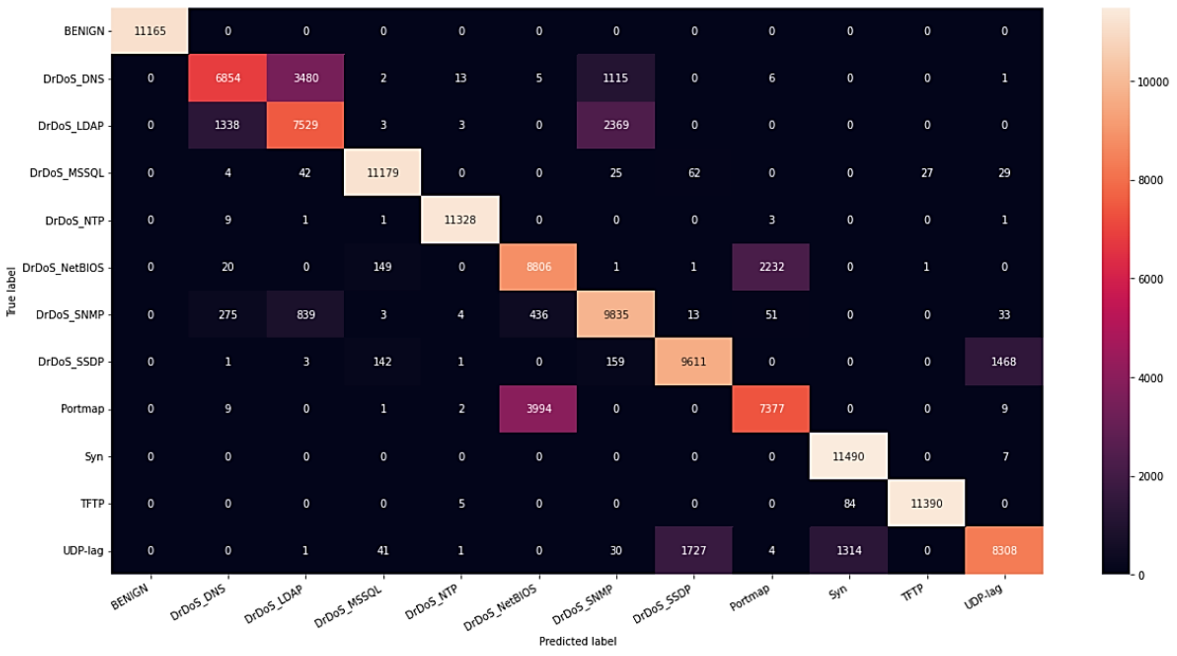


Figure 5.15: Confusion Matrix for LGBM.

features successfully - hence the name. It supports implementations to use both CPU and GPU. Some of the key features of Catboost-

- CatBoost builds symmetric (balanced) trees by default, meaning that each level of the tree is fully expanded before moving to the next level, and all the leaves are at the same depth. This helps to avoid overfitting by ensuring that each split is equally important in the tree, and it can improve generalization performance.
- CatBoost provides native feature support saving pre-processing time.
- CatBoost uses ordered boosting, which is a variant of gradient boosting that is designed to improve the convergence speed and generalization performance of the model. In ordered boosting, the trees are trained in a specific order, where each tree tries to correct the errors made by the previous trees.

CatBoost handles the numerical features like the rest of the gradient boosting algorithms but handles categorical features using strategies like – one-hot encoding for binary features, target encoding with random permutation, and greedy search combinations.

Catboost also has some brilliant techniques for ranking the features as per importance which have been presented by Anna et al. [12]. Some of these techniques are prediction value change, loss function change, internal feature importance, SHapley Additive exPlanations (SHAP). on similar sizes of ensembles CatBoost can be scored around 25 times faster than XGBoost and around 60 times faster than LightGBM [12].

In the CatBoost model developed in this thesis, after tuning the hyper-parameters using Optuna[63], the number of trees was taken to be 1000, learning rate was taken to be 0.1, random strength was taken to be 0.1, and depth was taken to be 8. The loss function was 'multiclass' and the leaf estimation method was 'Newton'. The features, ' Protocol', 'Fwd PSH Flags', ' RST Flag Count',' ACK Flag Count', ' URG Flag Count', ' CWE Flag Count', ' Inbound', 'Source Port', ' Destination Port', ' Source IP', ' Destination IP', were considered as categorical features.

The evaluation metrics obtained are recorded in table 5.12 and figure 5.15 shows the confusion matrix for the developed model.

Table 5.12: Performance metrics for CatBoost.

Model	Precision	Recall	Accuracy	F1
CatBoost	0.894	0.895	0.895	0.893

5.5 Comparative Analysis

This section presents a comparative analysis of the performance metrices obtained in this thesis with the related works using the same dataset, with the same objective.

Table 5.13 shows the performance comparison of the deep learning algorithms on the CICDDoS2019 dataset, for 12 classes. Deep learning models for tabular data, TabResnet and transformer based deep learning algorithms, TabPerceiver, FastFormer, and FT-Transformer have outperformed the existing works that were conducted using deep learning.

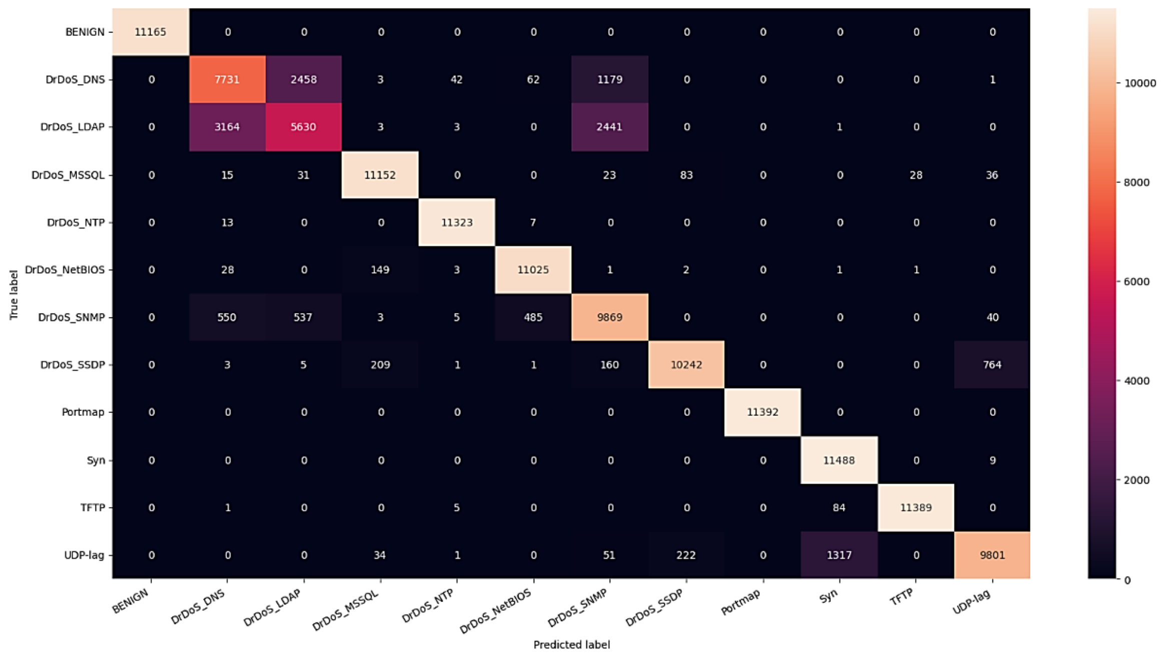


Figure 5.16: Confusion Matrix for CatBoost.

In the following tables, under the column author, self refers to the experiments done in this thesis and all the evaluation metric values scoring higher that the existing related works are showed in bold numbers.

Table 5.13: Performance comparison for deep learning algorithms.

Approach	Model	Author	Result			
			Precision (%)	Recall (%)	Accuracy (%)	F1
Deep Learning models	CNN	Ferrag et al. [34]	0.850	0.600	-	0.550

	RNN	Ferrag et al. [34]	0.720	0.620	-	0.570
	DNN	Ferrag et al. [34]	0.660	0.620	-	0.560
		Chartuni et al. [35]	0.831	0.799	0.818	0.815
	Ensemble DNN	Sayed et al. [38]	0.848	0.834	0.822	0.815
	TabTransformer	Self	0.772	0.771	0.771	0.768
	TabMLP		0.836	0.800	0.800	0.788
	TabPerceiver		0.836	**0.837**	**0.837**	**0.828**
	FastFormer		0.830	**0.833**	**0.833**	**0.828**
	TabResnet		**0.856**	**0.846**	**0.846**	**0.833**
	FT-Transformer		**0.851**	**0.851**	**0.851**	**0.846**

Table 5.14 shows the significant improvement in performance using the gradient boosting algorithms in comparison with the deep learning algorithm metrices.

Table 5.14: Performance comparison for the boosting algorithms.

Approach	Model	Author	Result			
			Precision (%)	Recall (%)	Accuracy (%)	F1
Bagging algorithm	RF	Sharafaldin et al. [22]	0.770	0.560	-	0.620
		Self	0.809	0.811	0.811	0.810
Boosting algorithm	**XGB**	Self	0.841	**0.838**	**0.838**	**0.836**
	LGBM		0.845	**0.842**	**0.842**	**0.840**
	CatBoost		**0.894**	**0.895**	**0.895**	**0.893**

The best scores from the existing works have already been crossed here for all 3 models developed in this thesis utilizing the boosting technique.

In comparison with all the model performances- of the similar research works and the developed models of this thesis- the model **CatBoost** shows a mentionable improvement outperforming all the models by a significant rise in the performance metrices. The reason could be the way it handles categorical data, compared to LGBM and XGBoost. CatBoost uses an embedding creation technique called "Ordered Boosting", which sorts categorical variables by their target statistics within each subset of data during the boosting process. This approach enables it to effectively capture relationships between categorical variables and the target variable. In contrast, LGBM incorporates categorical variables directly into the tree construction process, treating them as separate branches in the decision tree. It uses techniques like Gradient-based One-Side Sampling (GOSS) to focus more on the categories that are helping the most and Exclusive Feature Bundling (EFB) to group similar categories together to efficiently handle the categorical features. On the other hand, XGBoost uses a preprocessing step to convert categorical variables into numerical values through techniques like label encoding or OHE.

However, as proved by this research, CatBoost's categorical encoding approach reduces the risk of information loss and allows it to better retain the intrinsic characteristics of categorical data.

The findings from the research can be leveraged in practical scenarios. The benefits of this research are two-fold: firstly, it can enhance a business's ability to detect and mitigate specific attack vectors quickly, reducing service downtime and minimizing financial losses. Secondly, it can contribute to the larger cybersecurity community by advancing the state of knowledge in sub-classifying DDoS attacks.

Chapter 6

6 Explainable Artificial Intelligence

Chapter 5 highlights the evaluation scores obtained from the models developed in this thesis and provides a comparison of evaluation metrices obtained from the related works which brings CatBoost algorithms at the top of the leaderboard. Since CatBoost outperformed all the models developed for solving the multi-classification problem, this chapter will focus on explaining the interpretability of the model behavior.

After the preprocessing of the large dataset used, feature selection was done using an ensemble feature selection method mentioned in section 4.3.4. This feature selection method was used to aid an efficient training process by reducing the dimension of the data to be fed to the model. The pre-notion of which features could be comparatively more important was a pre-requisite for filtering the features. Feature importance plot, as shown in figure 6.1, could also be used to make an assumption of the important features to keep for training the model in the most efficient way. None of these could provide numeric values denoting the importance of each feature on the prediction- which, keeps the behavior of the model a mystery. Traditionally, machine learning models so far have been treated as a ‘black box’. Depending on assumptions, the processed data would be filtered and fed, hoping for an extraordinary outcome. The concept of SHapley Additive exPlanations (SHAP)[46] is a tool to look beyond the walls of the ‘black box’ to understand model behaviour. SHAP values explain how each feature impacts the output or the prediction of the model.

The confusion matrix as shown in figure 6.2 was created to visualize the performance of CatBoost shows where the model showed confusion or failed to predict with precision.

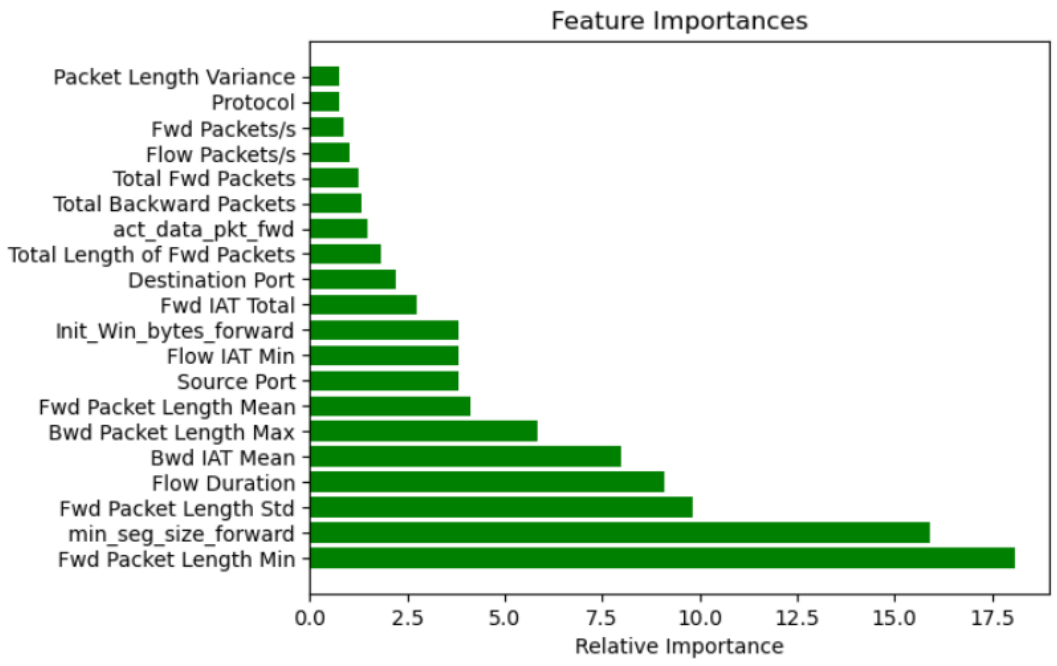
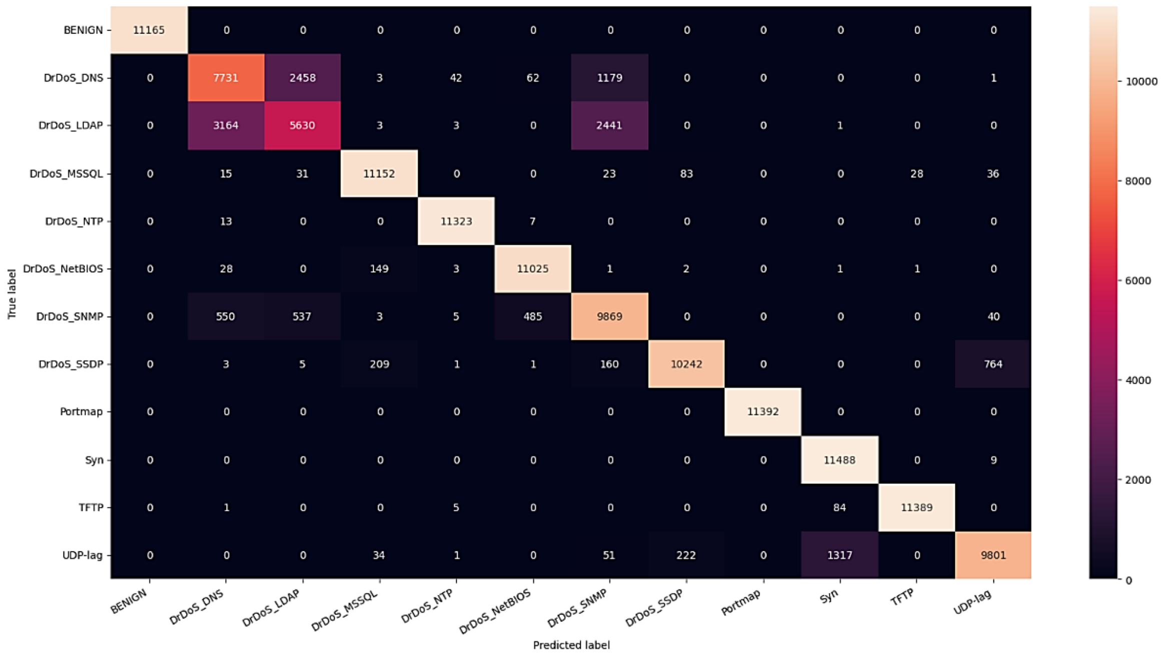


Figure 6.1: Relative importance of features.



	BENIGN	DrDoS_DNS	DrDoS_LDAP	DrDoS_MSSQL	DrDoS_NTP	DrDoS_NetBIOS	DrDoS_SNMP	DrDoS_SSDP	Portmap	Syn	TFTP	UDP-lag
BENIGN	11165	0	0	0	0	0	0	0	0	0	0	0
DrDoS_DNS	0	7731	2458	3	42	62	1179	0	0	0	0	1
DrDoS_LDAP	0	3164	5630	3	3	0	2441	0	0	1	0	0
DrDoS_MSSQL	0	15	31	11152	0	0	23	83	0	0	28	36
DrDoS_NTP	0	13	0	0	11323	7	0	0	0	0	0	0
DrDoS_NetBIOS	0	28	0	149	3	11025	1	2	0	1	1	0
DrDoS_SNMP	0	550	537	3	5	485	9869	0	0	0	0	40
DrDoS_SSDP	0	3	5	209	1	1	160	10242	0	0	0	764
Portmap	0	0	0	0	0	0	0	0	11392	0	0	0
Syn	0	0	0	0	0	0	0	0	0	11488	0	9
TFTP	0	1	0	0	5	0	0	0	0	84	11389	0
UDP-lag	0	0	0	34	1	0	51	222	0	1317	0	9801

Figure 6.2: Confusion Matrix for the CatBoost model performance.

The model seemed confused between ‘DDoS LDAP’ and ‘DDoS DNS’ attack types. The reason behind cannot be inferred from the confusion matrix.

The SHAP summary plot in figure 6.3 explains the average impact of the features on making the model prediction output. For the clarity of representation, only the top 20 features are depicted in the plot. The y-axis of the plot represents the features, and the x-axis of the plot shows the magnitude of the impact of the features on model prediction. If the plot is interpreted for 1 feature as an example, the ‘Min Packet Length’, as it can be seen, affects the ‘DDoS LDAP’, ‘DDoS NetBios’ classes the most and ‘DDoS SNMP’, ‘UDP- Lag’ classes the least. It is evident from the plot that the classes ‘DDoS LDAP’ and ‘DDoS DNS’

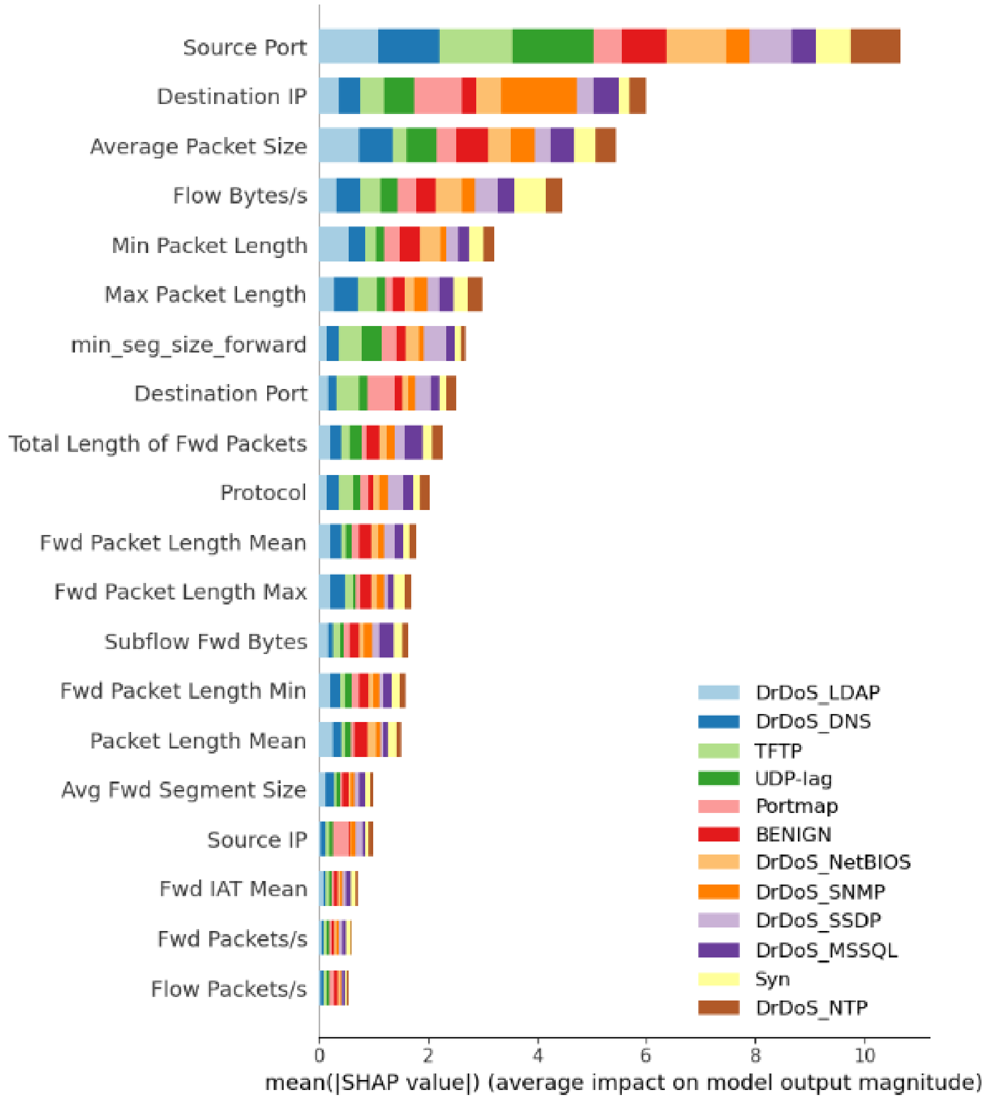


Figure 6.3: SHAP summary plots.

used the majority of features equally. This explains the model being confused between these two classes.

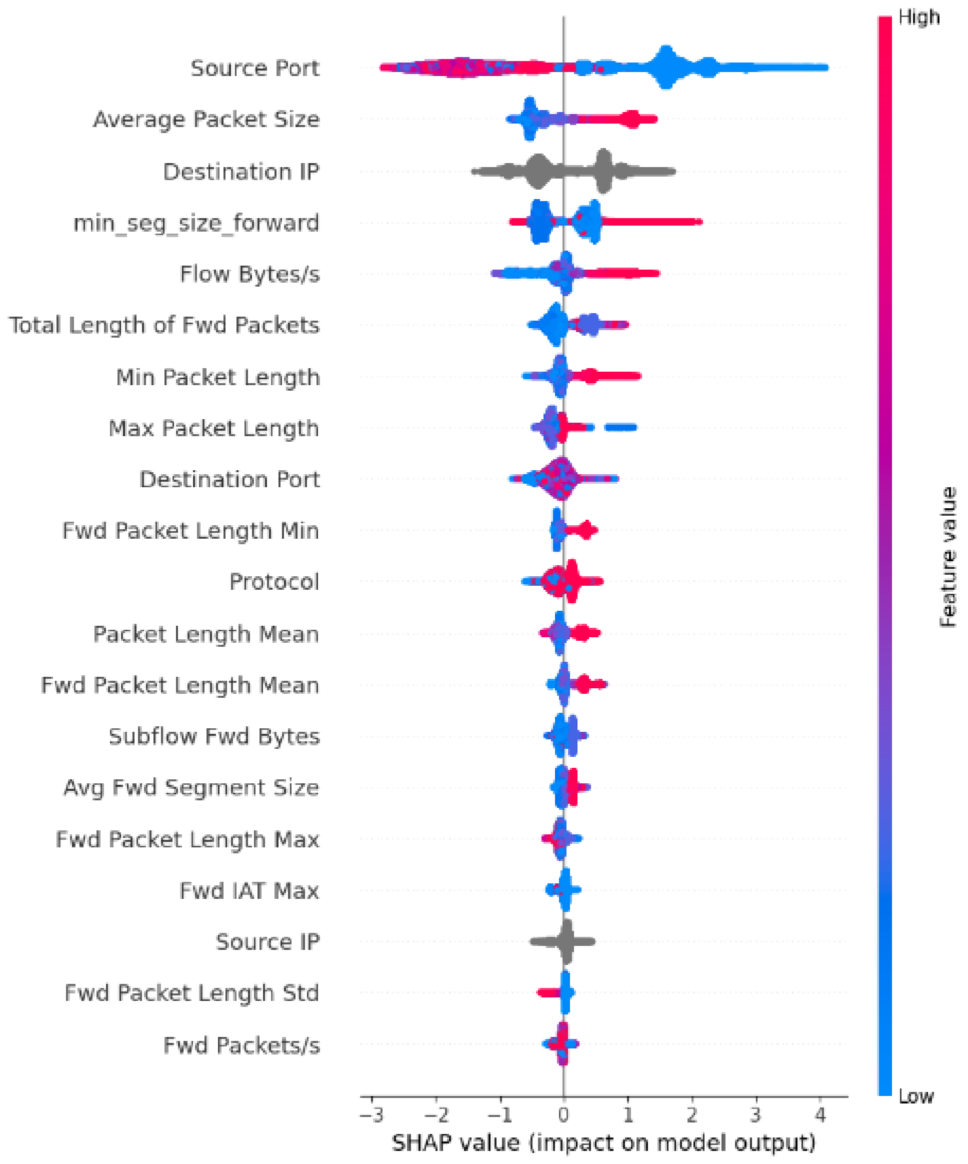


Figure 6.4: SHAP summary plot with the feature values.

Figure 6.4 also is a SHAP summary plot. Each point on this plot represents the Shapley value for a feature. The y-axis here is for the features and the x-axis here is for Shapley values. The features are shown in order or importance. The colour code represents the feature values from high to low. The plot efficiently gives a sense of the distribution of the Shapley values per feature.

If the plot is interpreted for 1 feature as an example, for the ‘Average Packet Size’, one of the top features affecting the model output, it can be seen that for when the SHAP value is higher, the model prediction is affected positively. For the feature ‘Fwd Packet Length Max’, it can be seen that the lower feature value of the feature affects the model output positively.

Figure 6.5 is the SHAP waterfall plot that is another analysis plot for the prediction for a single instance. *f(x)* here is the model prediction probability value: -5.265 and *E[f(x)]* is the base value, which here is -3.494. The features are on the left and the arrows provide an idea of the direction of contribution of the features. Negative values of the units on the x-axis refer to probabilities of less than 0.5 as these are log-odds units. The values of the arrow headed quadrilaterals represent what role each feature plays in moving the model output from the base value or expected model output. For examples, the feature ‘min_seg_size_value’

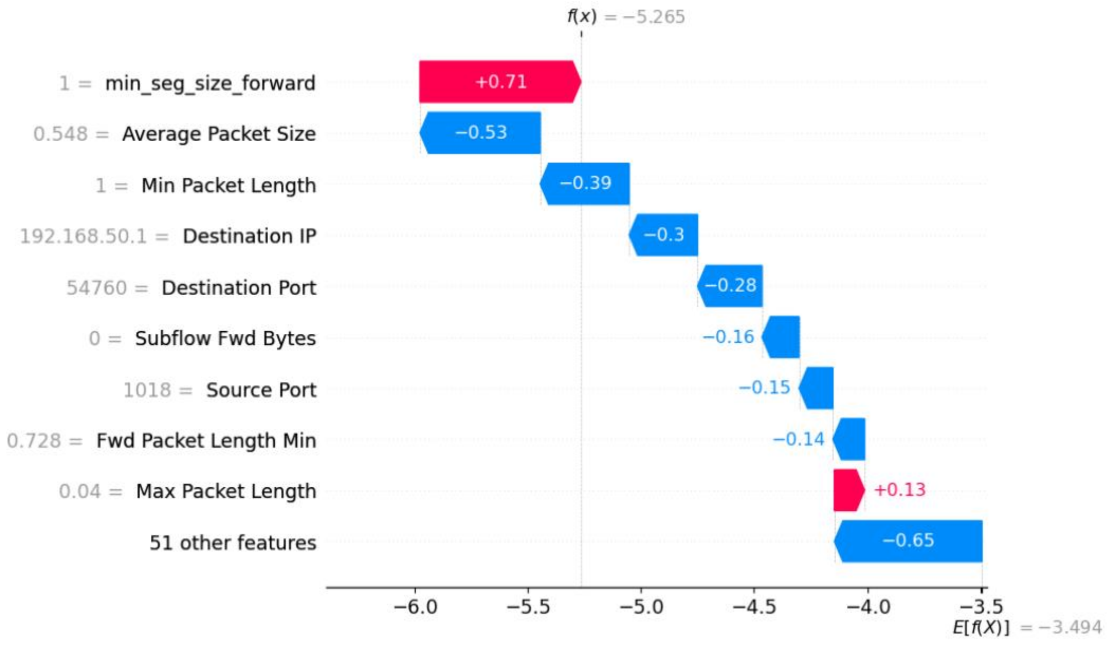


Figure 6.5: SHAP summary plot with the feature values.

having positive value affects the model prediction and moves the prediction line towards the base value. The cumulative effect of all the features moving towards or away from the base value results in the model output prediction.

Chapter 7

7 Conclusion

This thesis aimed at filling up a portion of the research gap from existing research by looking at the data from a different point of view. It looked beyond the numerical data in the chosen network traffic dataset CICDDoS2019 and explored data ordinality. Unlike the similar research works mentioned in Chapter 3, this thesis developed and investigated transformer-based deep learning models and boosting models aiming at sub-classifying DDoS attacks precisely, as accurate detection is necessary for coming up with the solution to the widely growing problem. It proved the importance of feature engineering and feature selection for optimization of performance and showed stellar performance scores obtained from the Catboost model that outperformed the previous models developed.

This chapter is to represent a summary of the thesis and future research directions to keep the flow going.

7.1 Contributions

The contributions of the thesis can be summed up as the following –

- The ensemble feature selection method introduced played a positive role in improving model performance. A model giving the same output prediction with lesser dimension of training data is computationally efficient and here, with this feature selection [44] technique the model generated a higher accuracy score on top of being computationally efficient.
- In addition to the original features mentioned in the CICDDoS2019 dataset, a bunch of new features were created as a part of feature engineering which influenced the model performance positively.
- Majority of the existing works for solving the addressed problem [22][33][34][35][37][38] were based on deep learning algorithms. The research developed some recently introduced transformer [9] based deep learning models

like TabTransformer, FTTransformer for solving the problem and exceeded the performances of the existing deep learning-based models.

- The research proved the developed CatBoost model to be the best fit for the CICDDoS-2019 [32] network traffic dataset. One key discovery is considering the ordinal and categorical features as categorical data and creating embeddings for them to handle the categorical features differently than the other bagging and boosting algorithms that use OHE. The specialty of CatBoost [12] is handling the categorical data in a more intuitive way, that played a significant role in developing the model in this thesis that obtained a stellar accuracy score of 89.5%, outperforming all the other related existing works.
- The thesis also goes one step forward, beyond achieving a good score. It explains the model behavior and influence of each feature on the model prediction with quantitative data, using the tool SHAP [46]. This solves the mystery of why the model behaved the way it behaved; this leads to a smarter perspective to look at the machine learning models.

7.2 Limitations

The research explores a portion of the spectrum of scopes in classifying cyber-attacks, more specifically, DDoS Attacks. It is not a study with every corner explored an aced, like most other existing research. The limitations are -

- The confusion matrix even for the best model, showed a confusion differentiating the ‘DDoS LDAP’ and ‘DDoS DNS’ classes, regardless of the model used.
- The training data used for this research is a sampled dataset having a subset of the whole data. Future research can uncover the potential of using the whole dataset for training/testing or other benchmark datasets containing network traffic data.
- The scalability of the developed models for larger datasets or multiple simultaneous attacks were not investigated. While the findings of this research provide valuable insights, the unexplored aspect of scalability could potentially impact the applicability of the findings in broader contexts.

7.3 Future Research Direction

The research had mentionable contributions but like any research, it is an ongoing process the shows a range of potential future scopes to work on.

Future research in this field can focus on the following aspects –

- More research can be done to create engineered features that will differentiate the ‘DDoS LDAP’ and ‘DDoS DNS’ classes better for the model as the dataset does not provide enough differentiable features for differentiating between these two classes.
- The existing works on multiclass prediction problems did not use the recent state-of-the-art architectures as developed in this thesis like CatBoost or the transformer based deep learning models. The thesis proves the capability of these algorithms for network traffic data. These can be explored with other benchmark datasets containing similar data.
- The feature ‘Timestamp’ from the CICDDoS2019 dataset can be leveraged for time-series analysis in DDoS Classification.
- Addressing scalability can lead to broader and more practical applications the research.

Bibliography

[1] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments," *Energy Reports*, vol. 7, pp. 8176–8186, Nov. 2021, doi: 10.1016/J.EGYR.2021.08.126.

[2] "History of Cybercrime | Arctic Wolf." https://arcticwolf.com/resources/blog/decade-of-cybercrime/ (accessed Feb. 14, 2023).

[3] "Five Most Famous DDoS Attacks and Then Some | A10 Networks." https://www.a10networks.com/blog/5-most-famous-ddos-attacks/ (accessed Feb. 14, 2023).

[4] "The Beat Goes On | NETSCOUT." https://www.netscout.com/blog/asert/beat-goes (accessed Feb. 15, 2023).

[5] "Cybersecurity Trends 2023 - Leaseweb Blog." https://blog.leaseweb.com/2023/01/03/cybersecurity-trends-2023/ (accessed Feb. 15, 2023).

[6] X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "TabTransformer: Tabular Data Modeling Using Contextual Embeddings," Dec. 2020, Accessed: Apr. 05, 2023. [Online]. Available: https://arxiv.org/abs/2012.06678v1

[7] Y. Jin Kim and H. Hassan Awadalla, "FastFormers: Highly Efficient Transformer Models for Natural Language Understanding," p. 149, Accessed: Apr. 06, 2023. [Online]. Available: https://huggingface.co/

[8] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira, "Perceiver: General Perception with Iterative Attention," 2021, [Online]. Available: http://arxiv.org/abs/2103.03206

[9] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, no. Nips, p. 30. doi: 10.1007/978-3-319-29409-4_3.

[10] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 3147–3155, 2017.

[11] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, doi: 10.1145/2939672.

[12] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: gradient boosting with categorical features support," pp. 1–7, 2018, [Online]. Available: http://arxiv.org/abs/1810.11363

[13] "2023 Security Report: Cyberattacks reach an all-time high in response to geo-political conflict, and the rise of 'disruption and destruction' malware - Check Point

Research.” https://research.checkpoint.com/2023/2023-security-report-cyberattacks-reach-an-all-time-high-in-response-to-geo-political-conflict-and-the-rise-of-disruption-and-destruction-malware/ (accessed Feb. 17, 2023).

[14] “National Cyber Threat Assessment 2023-2024 - Canadian Centre for Cyber Security.” https://cyber.gc.ca/en/guidance/national-cyber-threat-assessment-2023-2024 (accessed Feb. 17, 2023).

[15] “Top 10 Data Breaches So Far in 2022 - Cybersecurity | Digital Forensics | Crypto Investigations.” https://ermprotect.com/blog/top-10-data-breaches-so-far-in-2022/ (accessed Feb. 17, 2023).

[16] M. Kjaerland, “A taxonomy and comparison of computer security incidents from the commercial and government sectors,” *Comput. Secur.*, vol. 25, no. 7, pp. 522–538, Oct. 2006, doi: 10.1016/J.COSE.2006.08.004.

[17] S. Hansman and R. Hunt, “A taxonomy of network and computer attacks,” *Comput. Secur.*, vol. 24, no. 1, pp. 31–43, Feb. 2005, doi: 10.1016/J.COSE.2004.06.011.

[18] C. Simmons, C. Ellis, S. Shiva, D. Dasgupta, and Q. Wu, “AVOIDIT: A cyber attack taxonomy, tech. Report,” *Univ. Memphis, USA*, pp. 1–9, 2009.

[19] J. Mirkovic and P. Reiher, “A taxonomy of DDoS attack and DDoS defense mechanisms,” *Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53, Apr. 2004, doi: 10.1145/997150.997156.

[20] A. Asosheh and N. Ramezani, “A comprehensive taxonomy of DDoS attacks and defense mechanism applying in a smart classification,” *WSEAS Trans. Comput.*, vol. 7, no. 4, pp. 281–290, 2008.

[21] A. Bhardwaj, G. V. B. Subrahmanyam, V. Avasthi, H. Sastry, and S. Goundar, “DDoS attacks, new DDoS taxonomy and mitigation solutions - A survey,” *Int. Conf. Signal Process. Commun. Power Embed. Syst. SCOPES 2016 - Proc.*, pp. 793–798, Jun. 2017, doi: 10.1109/SCOPES.2016.7955549.

[22] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, “Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy,” *Proc. - Int. Carnahan Conf. Secur. Technol.*, vol. 2019-Octob, Oct. 2019, doi: 10.1109/CCST.2019.8888419.

[23] M. Zounemat-Kermani, O. Batelaan, M. Fadaee, and R. Hinkelmann, “Ensemble machine learning paradigms in hydrology: A review,” *J. Hydrol.*, vol. 598, no. December 2020, p. 126266, 2021, doi: 10.1016/j.jhydrol.2021.126266.

[24] M. Zahid Hasan, K. M. Zubair Hasan, and A. Sattar, “Burst Header Packet Flood Detection in Optical Burst Switching Network Using Deep Learning Model,” *Procedia Comput. Sci.*, vol. 143, pp. 970–977, Jan. 2018, doi: 10.1016/J.PROCS.2018.10.337.

[25] N. G. Bhuvaneswari Amma and S. Subramanian, “VCDeepFL: Vector Convolutional Deep Feature Learning Approach for Identification of Known and Unknown Denial of Service Attacks,” *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, vol. 2018-Octob, pp. 640–645, Feb. 2019, doi: 10.1109/TENCON.2018.8650225.

[26] K. B. Virupakshar, M. Asundi, K. Channal, P. Shettar, S. Patil, and D. G. Narayan, “Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based Private Cloud,” *Procedia Comput. Sci.*, vol. 167, pp. 2297–2307, Jan. 2020, doi: 10.1016/J.PROCS.2020.03.282.

[27] J. Chen, Y. tao Yang, K. ke Hu, H. bin Zheng, and Z. Wang, “DAD-MCNN: DDoS attack detection via multi-channel CNN,” *ACM Int. Conf. Proceeding Ser.*, vol. Part F1481, pp. 484–488, 2019, doi: 10.1145/3318299.3318329.

[28] A. R. Shaaban, E. Abd-Elwanis, and M. Hussein, “DDoS attack detection and classification via Convolutional Neural Network (CNN),” *Proc. - 2019 IEEE 9th Int. Conf. Intell. Comput. Inf. Syst. ICICIS 2019*, pp. 233–238, Dec. 2019, doi: 10.1109/ICICIS46948.2019.9014826.

[29] L. Wang and Y. Liu, “A DDoS Attack Detection Method Based on Information Entropy and Deep Learning in SDN,” *Proc. 2020 IEEE 4th Inf. Technol. Networking, Electron. Autom. Control Conf. ITNEC 2020*, pp. 1084–1088, Jun. 2020, doi: 10.1109/ITNEC48623.2020.9085007.

[30] U. Sabeel, S. S. Heydari, H. Mohanka, Y. Bendhaou, K. Elgazzar, and K. El-Khatib, “Evaluation of Deep Learning in Detecting Unknown Network Attacks,” *2019 Int. Conf. Smart Appl. Commun. Networking, SmartNets 2019*, 2019, doi: 10.1109/SmartNets48225.2019.9069788.

[31] M. V. O. de Assis, L. F. Carvalho, J. J. P. C. Rodrigues, J. Lloret, and M. L. Proença, “Near real-time security system applied to SDN environments in IoT networks using convolutional neural network,” *Comput. Electr. Eng.*, vol. 86, p. 106738, 2020, doi: 10.1016/j.compeleceng.2020.106738.

[32] “DDoS 2019 | Datasets | Research | Canadian Institute for Cybersecurity | UNB.” https://www.unb.ca/cic/datasets/ddos-2019.html (accessed Jan. 17, 2023).

[33] D. C. Can, H. Q. Le, and Q. T. Ha, “Detection of Distributed Denial of Service Attacks Using Automatic Feature Selection with Enhancement for Imbalance Dataset,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12672 LNAI, pp. 386–398, 2021, doi: 10.1007/978-3-030-73280-6_31/FIGURES/5.

[34] M. A. Ferrag, L. Shu, H. Djallel, and K. K. R. Choo, “Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0,” *Electron. 2021, Vol. 10, Page 1257*, vol. 10, no. 11, p. 1257, May 2021, doi: 10.3390/ELECTRONICS10111257.

[35] A. Chartuni and J. Márquez, "Multi-Classifier of DDoS Attacks in Computer Networks Built on Neural Networks," *Appl. Sci. 2021, Vol. 11, Page 10609*, vol. 11, no. 22, p. 10609, Nov. 2021, doi: 10.3390/APP112210609.

[36] M. Shurman, R. Khrais, and A. Yateem, "DoS and DDoS attack detection using deep learning and IDS," *Int. Arab J. Inf. Technol.*, vol. 17, no. 4A Special Issue, pp. 655–661, 2020, doi: 10.34028/iajit/17/4A/10.

[37] A. E. Cil, K. Yildiz, and A. Buldu, "Detection of DDoS attacks with feed forward based deep neural network model," *Expert Syst. Appl.*, vol. 169, no. April 2020, p. 114520, 2021, doi: 10.1016/j.eswa.2020.114520.

[38] M. I. Sayed, I. M. Sayem, S. Saha, and A. Haque, "A Multi-Classifier for DDoS Attacks Using Stacking Ensemble Deep Neural Network," *2022 Int. Wirel. Commun. Mob. Comput. IWCMC 2022*, pp. 1125–1130, 2022, doi: 10.1109/IWCMC55113.2022.9824189.

[39] S. Arık and T. Pfister, "TabNet: Attentive Interpretable Tabular Learning," *35th AAAI Conf. Artif. Intell. AAAI 2021*, vol. 8A, pp. 6679–6687, Aug. 2019, doi: 10.1609/aaai.v35i8.16826.

[40] J. Heaton, "An empirical analysis of feature engineering for predictive modeling," *Conf. Proc. - IEEE SOUTHEASTCON*, vol. 2016-July, Jul. 2016, doi: 10.1109/SECON.2016.7506650.

[41] H. Liu and R. Setiono, "Chi2: feature selection and discretization of numeric attributes," *Proc. Int. Conf. Tools with Artif. Intell.*, pp. 388–391, 1995, doi: 10.1109/tai.1995.479783.

[42] L. Sthle and S. Wold, "Analysis of variance (ANOVA)," *Chemom. Intell. Lab. Syst.*, vol. 6, no. 4, pp. 259–272, 1989, doi: 10.1016/0169-7439(89)80095-4.

[43] "Mutual information - Scholarpedia." http://www.scholarpedia.org/article/Mutual_information?em_x=22 (accessed Apr. 04, 2023).

[44] J. Tang, S. Alelyani, H. L. A. and applications, and undefined 2014, "Feature selection for classification: A review," *cvs.edu.in*, Accessed: Apr. 29, 2023. [Online]. Available: https://www.cvs.edu.in/upload/feature_selection_for_classification.pdf

[45] N. Agarwal and S. Das, "Interpretable Machine Learning Tools: A Survey," *2020 IEEE Symp. Ser. Comput. Intell. SSCI 2020*, pp. 1528–1534, 2020, doi: 10.1109/SSCI47803.2020.9308260.

[46] S. M. Lundberg and S. I. Lee, "A Unified Approach to Interpreting Model Predictions," *Adv. Neural Inf. Process. Syst.*, vol. 2017-December, pp. 4766–4775, May 2017, Accessed: Apr. 07, 2023. [Online]. Available:

https://arxiv.org/abs/1705.07874v2

[47] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?' Explaining the Predictions of Any Classifier," *NAACL-HLT 2016 - 2016 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Proc. Demonstr. Sess.*, pp. 97–101, 2016, doi: 10.18653/v1/n16-3020.

[48] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," *32nd AAAI Conf. Artif. Intell. AAAI 2018*, pp. 1527–1535, 2018, doi: 10.1609/aaai.v32i1.11491.

[49] "Applications | Research | Canadian Institute for Cybersecurity | UNB." https://www.unb.ca/cic/research/applications.html (accessed Feb. 19, 2023).

[50] "GitHub - ahlashkari/CICFlowMeter: CICFlowmeter-V4.0 (formerly known as ISCXFlowMeter) is an Ethernet traffic Bi-flow generator and analyzer for anomaly detection that has been used in many Cybersecurity datsets such as Android Adware-General Malware dataset (CICAAGM2017), IPS/IDS dataset (CICIDS2017), Android Malware dataset (CICAndMal2017) and Distributed Denial of Service (CICDDoS2019)." https://github.com/ahlashkari/CICFlowMeter (accessed Feb. 19, 2023).

[51] W. A. Shewhart, "Random Sampling," *Am. Math. Mon.*, vol. 38, no. 5, pp. 245–270, 1931, doi: 10.1080/00029890.1931.11987187.

[52] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data cleaning: Overview and emerging challenges," *Proc. ACM SIGMOD Int. Conf. Manag. Data*, vol. 26-June-2016, pp. 2201–2206, 2016, doi: 10.1145/2882903.2912574.

[53] D. Borkin, A. Némethová, G. Michaľčonok, and K. Maiorov, "Impact of Data Normalization on Classification Model Accuracy," *Res. Pap. Fac. Mater. Sci. Technol. Slovak Univ. Technol.*, vol. 27, no. 45, pp. 79–84, 2019, doi: 10.2478/rput-2019-0029.

[54] V. R. Joseph, "Optimal ratio for data splitting," *Stat. Anal. Data Min.*, vol. 15, no. 4, pp. 531–538, 2022, doi: 10.1002/sam.11583.

[55] S. I. Survey, "Missing-data imputation".

[56] "GitHub - jrzaurin/pytorch-widedeep: A flexible package for multimodal-deep-learning to combine tabular data with text and images using Wide and Deep models in Pytorch." https://github.com/jrzaurin/pytorch-widedeep (accessed Apr. 06, 2023).

[57] A. M. Rush, "The Annotated Transformer," pp. 52–60, Jun. 2018, doi: 10.18653/V1/W18-2509.

[58] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep

Bidirectional Transformers for Language Understanding," *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, pp. 4171–4186, Oct. 2018, Accessed: Apr. 06, 2023. [Online]. Available: https://arxiv.org/abs/1810.04805v2

[59] R. Shwartz-Ziv and A. Armon, "TABULAR DATA: DEEP LEARNING IS NOT ALL YOU NEED," 2021.

[60] "XGBoost Parameters — xgboost 1.7.5 documentation." https://xgboost.readthedocs.io/en/stable/parameter.html (accessed Apr. 07, 2023).

[61] "sklearn.model_selection.RandomizedSearchCV — scikit-learn 1.2.2 documentation." https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html (accessed Apr. 07, 2023).

[62] "Features — LightGBM 3.3.5.99 documentation." https://lightgbm.readthedocs.io/en/latest/Features.html (accessed Feb. 18, 2023).

[63] "Optuna: A hyperparameter optimization framework — Optuna 3.1.0 documentation." https://optuna.readthedocs.io/en/stable/ (accessed Apr. 07, 2023).

Curriculum Vitae

Name:	Amreen Anbar
Post-secondary Education and Degrees:	M.Sc. Candidate, Computer Science The University of Western Ontario 2021-2023
	B.Sc., Electrical, Electronic and Communication Engineering Military Institute of Science and Technology, Bangladesh 2016-2019
Honors and Awards	Western Graduate Research Scholarship (WGRS) The University of Western Ontario 2021-2022
	Dean's List of Honor Military Institute of Science and Technology, Bangladesh 2016-2019
	MIST Scholarship Military Institute of Science and Technology, Bangladesh 2016-2019
Related Work Experience	Teaching and Research Assistant The University of Western Ontario 2021-2023