DEVELOPMENT OF EFFICIENT PROTOCOLS FOR INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

R. LALDUHSAKA

MZU REGISTRATION NO.: 2001033

Ph.D. REGISTRATION NO.: MZU/Ph.D./1654 of 06/11/2020

DEPARTMENT OF COMPUTER ENGINEERING

SCHOOL OF ENGINEERING AND TECHNOLOGY

JUNE, 2025

DEVELOPMENT OF EFFICIENT PROTOCOLS FOR INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

BY

R. LALDUHSAKA

Department of Computer Engineering

Supervisor

Prof. AJOY KUMAR KHAN

Joint Supervisor

Dr. R. CHAWNGSANGPUII

Submitted

In partial fulfillment of the requirement of the Degree of Doctor of Philosophy in Computer Engineering of Mizoram University, Aizawl.

CERTIFICATE

This is to certify that the thesis entitled "*Development of Efficient Protocols for Intrusion Detection System using Machine Learning*" is a record of the research that R. Lalduhsaka completed under our supervision and guidance, and it is submitted to the Mizoram University in the Department of Computer Engineering under the School of Engineering and Technology, in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Engineering.

All help received by him from various sources has been duly acknowledged. No part of this thesis has been reproduced elsewhere for the award of any other degree.

Prof. Ajoy Kumar Khan
Supervisor & Professor
Dept. of Computer Engineering
Mizoram University
Aizawl-796001

Place:
Date:

Dr. R. Chawngsangpuii
Joint Supervisor & Associate Professor
Dept. of Information Technology
Mizoram University
Aizawl-796001

Place:
Date:

DECLARATION

MIZORAM UNIVERSITY

JUNE 2025

I **R. LALDUHSAKA**, hereby declare that the subject matter of this thesis is the record of work done by me, that the contents of this thesis did not form basis of the award of any previous degree to me or to do the best of my knowledge to anybody else, and that the thesis has not been submitted by me for any research degree in any other University/Institute.

This is being submitted to the Mizoram University for the **Degree of Doctor of Philosophy in Computer Engineering.**

(R. Lalduhsaka)

(Prof. AJOY KUMAR KHAN)
Supervisor

(Dr. VD AMBETH KUMAR)
Head

(Dr. R. CHAWNGSANGPUII)
Joint Supervisor

Acknowledgement

This Thesis marks the pinnacle of years of hard work, persistence, and the steadfast support of many individuals who have played significant role in my academic journey. It is with deep gratitude that I acknowledge the guidance, encouragement, and belief in my potential extended by each of them.

First and foremost, I express my sincere appreciation to my Ph.D. supervisor, **Prof. Dr. Ajoy Kumar Khan**, whose insightful guidance, critical feedback, motivation and unwavering support have been instrumental in shaping this research. His mentorship has been a cornerstone of my academic growth, and I am deeply indebted to him. I am equally thankful to my joint supervisor, **Dr. R Chawngsangpuii**, for her constant support, expert advice, and encouragement throughout this journey.

My heartfelt thanks go to **Dr. V.D. Ambeth Kumar**, Head of the Department of Computer Engineering, Mizoram University. His leadership and continued support have fostered an environment conducive to research and academic excellence.

I am profoundly grateful to my mother, **Mrs. C. Lalrinliani** and my father **Mr. R. Lalriniana**, whose unwavering support, love, and sacrifices have been my pillars of strength. Their endless patience and encouragement have been indispensable throughout this journey.

I wish to express my deepest appreciation to my wife, **Mrs. K. Lalnunpuii**, for her boundless support, understanding, and love. Her constant encouragement and belief in me have been my greatest sources of motivation.

And lastly, I would like to thank the Almighty God for His abundant blessings, guidance, and strength throughout this research journey. Without His grace, this achievement would not have been possible.

This Thesis would not have been possible without the contributions and support of these remarkable individuals. I extend my heartfelt thanks to each one of them for their unwavering faith in me and their significant roles in this achievement.

(R. LALDUHSAKA)

Contents

List of Figures

List of Tables

List of Abbreviations

ADASYN - Adaptive Synthetic (Sampling)

ANN – Artificial Neural Network

ANOVA – Analysis of Variance

APT – Advanced Persistent Threats

BC - Bagging Classifier

CIC - Canadian Institute for Cybersecurity

CNN – Convolutional Neural Network

DDoS – Distributed Denial of Service

DL – Deep Learning

DNN – Deep Neural Network

DoS – Denial of Service

DT – Decision Tree

ELM - Extreme Learning Machine

FAR – False Alarm Rate

FNR – False Negative Rate

FPR – False Positive Rate

FS – Feature Selection

GA – Genetic Algorithm

GAN – Generative Adversarial Networks

GI – Gini Impurity

GRU – Gated Recurrent Unit

HEFS - Heterogeneous Ensemble Feature Selection

IDS – Intrusion Detection System

IoT – Internet of Things

LOAO – Leave One Attack Out

LR – Logistic Regression

IPS – Intrusion Prevention System

KNN – K-Nearest Neighbour

LGBM - Light Gradient Boosting Machine

LTSM - Long Term Short Memory

MCC - Matthews Correlation Coefficient

MITM – Man-in-the-Middle

ML – Machine Learning

MLP – Multilayer Perceptron

NB – Naïve Bayes / Gaussian Naïve Bayes

PCA – Principal Component Analysis

R2L – Remote to Local User

RENN - Repeated Edited Nearest Neighbour

RF – Random Forest

RL – Reinforce Learning

RNN – Recurrent Neural Network

RO – Random Oversampling

RU – Random Undersampling

SBS - Sequential Backward Selection

SDN – Software Defined Networking

SFS - Sequential Forward Selection

SMOTE – Synthetic Minority Over-sampling Technique

SVM – Support Vector Machine

U2R – User to Root

QDA – Quadratic Discriminant Analysis

CHAPTER 1

INTRODUCTION

The utilization of the internet and its associated features has grown exponentially in the past few decades, becoming an indispensable part of everyday life. With billions of users accessing the internet daily through smartphones, computers, tablets, and Internet of Things (IoT) components, digital connectivity has reached exceptional levels. However, this rapid growth has also attracted malicious actors such as hacktivists, crackers (black hat hackers), and politically motivated organizations; who are continually developing sophisticated network attacks. These attacks aim to compromise sensitive government information, exploit personal data to steal banking credentials, deploy ransomware for financial extortion, and more . Alarmingly, many of these malicious activities continue despite strict cybercrime laws in numerous countries. One of the major challenges in combating these threats is that malicious traffic often mimics legitimate or benign network behaviour, making detection difficult. Hence, cybersecurity is increasingly recognized as a fundamental component of national security. Both governments and private organizations are investing heavily in advanced cybersecurity measures to defend their digital infrastructure. The explosive growths of the interconnected network with its adoption of interconnected systems have dramatically expanded the cyber threat landscape. As dependence on digital technologies increases, so does the risk of cyberattacks such as Ransomware, Phishing, Brute Force attack, zero-day exploits, and more.

Antivirus and firewalls serve as essential security layers but are not sufficient against sophisticated and evolving cyber threats. Intrusion Detection Systems (IDS) compliments antivirus and firewalls by providing deeper, behavioural-based analysis of network or host activities IDS are widely recognized as effective tools for identifying malicious activities within a network. They have evolved into a vital part of network security infrastructure, especially as the frequency and complexity of cyberattacks continue to rise in modern networking environments. The significance of IDS has grown in response to this trend. The concept of intrusion detection dates

back to 1980, when James P. Anderson introduced the foundational ideas of computer security surveillance and threat monitoring in his report, highlighting the critical role such systems play in safeguarding information systems [1]. An IDS monitors all the incoming packet as well as the outgoing packet within a network to determine if it exhibits an abnormal behaviour or any sign of breach [2]. An IDS equipped with artificial intelligence must be capable of finding out different kinds of attacks and inform the systems in the network.

1.1 Machine Learning Overview

Machine Learning (ML) techniques are able to identify patterns and behaviours from data, enabling them to make generalized decisions by adjusting their internal parameters, without the need for explicitly defined rules. The development of an IDS using ML typically follows a structured, multistage approach. The ML step includes data preprocessing, selection of model, training and testing of the selected algorithm [3]. The general ML pipeline is illustrated in Figure 1.1. There can be various addition steps beyond this general pipeline that can enhance the performance and improve the outcome.

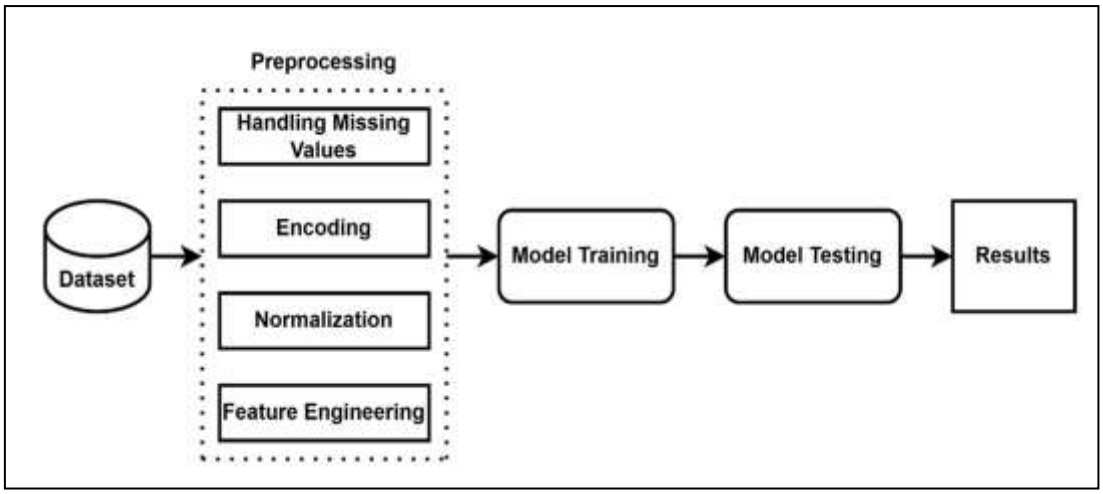

Figure 1.1 General ML pipeline

In the general ML pipeline shown in Figure 1.1, a dataset is first preprocessed so that it will be suitable for training ML model(s). The dataset is then divided into two parts, the training part and the testing part. The training part of the dataset is used to train ML model (Model Training) and the trained model is subsequently tested using

the testing part (Model Testing). The results or the performance of the model indicates how effectively the ML model operates on the dataset, especially the testing part. There can be multiple steps that can be added in between to optimize the performance of the ML models.

1.1.1 Data Preprocessing

It is a crucial step that prepares the dataset to make it suitable and more effective ML application. It involves several operations to clean, transform, and structure the dataset for analysis. Data in this context can broadly be classified into numerical and categorical types [4]. Numerical data consists of measurable quantities that can either be discrete or continuous. Categorical data, on the other hand, consists of labels or names representing different categories. It can be further divided into nominal data (without inherent order) and ordinal data (where they have a logical sequence or meaningful order). Recognizing these data types is crucial because they influence the choice of preprocessing techniques applied during model development. There are different steps in data preprocessing depending on the quality and composition of the dataset:

1. **Handling Missing Values**: In any datasets, incompletely data is often encountered and they can degrade the effectiveness of the ML models if not addressed properly. Addressing the missing value is a critical preprocessing step aimed at improving data quality and ensuring robust model training. Techniques for dealing with missing values include deletion which remove records with missing fields, imputation which fills the missing entries with statistical estimates like mean, median, or mode, and the use of model based approaches for handling incomplete data [5]. How to address the missing values depends on the number of rows with missing data and characteristics of the dataset. Simple strategies like mean imputations are often effective for small gaps, whereas complex methods are preferred when missingness is substantial. Careful handling of missing values is necessary to avoid introducing biases and to preserve the integrity of the dataset. In large datasets like intrusion detection dataset, removing of rows with missing value is a common method.

2. **Encoding:** Encoding transforms the categorical data into machine readable format i.e. numerical formats, enabling its use in ML tasks. Since most ML models require numerical input, categorical features must be encoded without losing their inherent meaning. One of the common encoding techniques is the Label Encoding which maps each category to a distinct numerical value. Another common method is the One-Hot Encoding where a unique binary value is created for each category by creating binary column [6]. Addressing the categorical data using encoding is usually needed for ML algorithms as they can only operate on numerical data. One-Hot Encoding is usually used when there is lack of order, whereas Label Encoding is more suitable where the sequence or ranking holds significance. Researchers always have to be careful with encoding data as higher numerical data can often mean to more significant value than the one with lower values.
3. **Normalization:** Normalization method is employed to rescale numerical values within the range of 0 and 1. This process is important as it guarantees that every feature in the dataset is converted and put under the same scale. It is particularly effective if different features measured different scales. It is also especially important for ML algorithms that are performing poorly when features have large value disparities, such KNN and neural networks [7]. By applying normalization, the influences of features with larger ranges are minimized, allowing the ML algorithms to achieve better learning outcomes as well as performances.
4. **Feature Engineering:** It is the process of creating, transforming, or selecting relevant more informative attributes from raw data to optimize model performance and accuracy. Effective feature engineering can expose hidden patterns and relationships that make the learning process more efficient and accurate. This process may involve generating new features through mathematical transformations, combining existing features, or encoding domain-specific knowledge into the dataset [6]. Good feature engineering can enhances the ability of algorithms to detect meaningful patterns in the dataset, this helps the ML model to understand the dataset better.

1.1.2 Feature Selection

FS is a selection of relevant features among all the features of the dataset that reduces the number of attributes of a dataset. FS is the process of reducing the dimensionality of the dataset by identifying and selecting the relevant and important subset of features from the original features. It is an important approach to reduce computational overhead, and more importantly enhance model's generalization and improves detection accuracy. FS can be divided into 3 methods:

1. **Filter method:** The filter method of FS assesses the importance of features by examining the inherent characteristics of the data, independently of any ML algorithm. Features are prioritized and ranks based on various statistical criteria, including correlation, mutual information, chi-square statistics, information gain, and variance. These methods help identify and eliminate irrelevant or redundant features before the model training phase, thereby reducing computational cost and improving generalization. Since filter methods are model-agnostic, they are typically faster and more scalable to large datasets compared to wrapper and embedded methods. However, because they do not consider interactions with a specific learning algorithm, their selected features may not always yield optimal model performance. According to Chandrashekar & Sahin [8], filter methods are especially useful in high dimensional data scenarios, where computational efficiency is crucial, and can act as a preprocessing step.

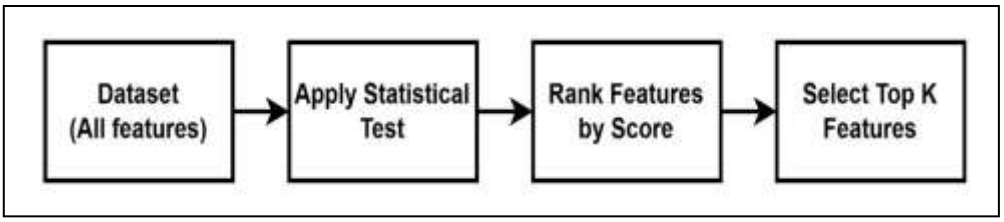

Figure 1.2 Filter Method of Feature Selection

Figure 1.2 shows the general steps of FS using filter method. All features are given values using statistical test and the features are ranked by their score. Top K features are selected as a subset of features for that dataset. This assists in removing features that contribute less to the model's performance, thereby lowering its computational overhead

2. **Wrapper Method:** Wrapper methods perform FS by evaluating subsets of features through the process of model training and testing, aiming to identify the most effective combinations. It assesses feature usefulness based on model performance. This process involves searching through various feature subsets, using techniques like forward selection, backward elimination, or exhaustive search to identify the subset that provides optimal performance. Although wrapper methods often produce higher predictive performance than filter methods, they lead to elevated computational complexity, especially with large datasets or feature spaces. They are also tightly coupled with the learning algorithm used, meaning results may vary with different models. Wrapper methods tends to be computationally demanding, but can enhance the predictive performance because they consider feature dependencies and model specific interactions [8].

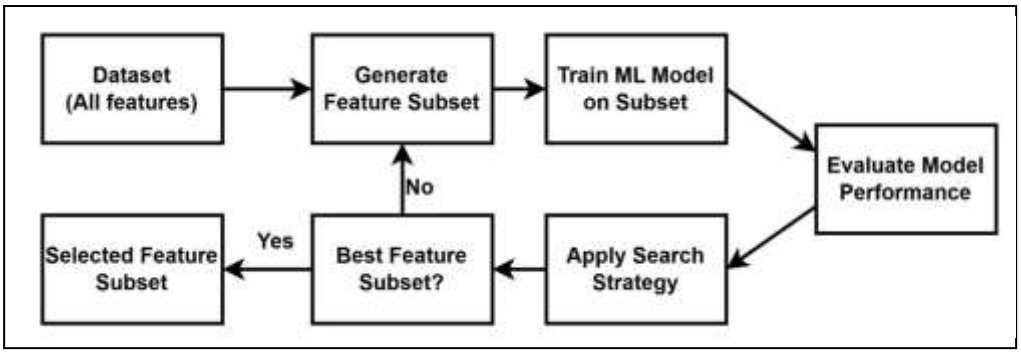

Figure 1.3 Wrapper Method for Feature Selection

Figure 1.3 shows the flow of Wrapper method of FS. Firstly, a number of subsets of features are generated. The goal is to search the best feature subset from the generated feature subsets. A predictive model is used for each feature subset. The result is used as the feature subset score. Both the total number of features included in the subset and the performance score it achieves during evaluation are important considerations. These two factors together help in identifying a subset can be used to select the feature subset.

3. **Embedded Method:** It integrates FS directly into the model training process, allowing the algorithm to automatically detect and keep only the most significant features during learning. Unlike filter and wrapper methods,

embedded techniques benefit from both computational efficiency and model awareness, as they exploit the internal workings of specific ML models to evaluate the significance of the features. Common examples include Lasso (L1 regularization) in linear models, DT-based methods like RF and XGBoost, which naturally rank features according to their contribution on the splits. These methods achieve a compromise between computational efficiency and performance by selecting features as part of the model optimization. [9] highlighted that embedded methods provide a compromise between the generality of filters and the accuracy of wrappers, making them a strong choice for large-scale and domain-specific applications.

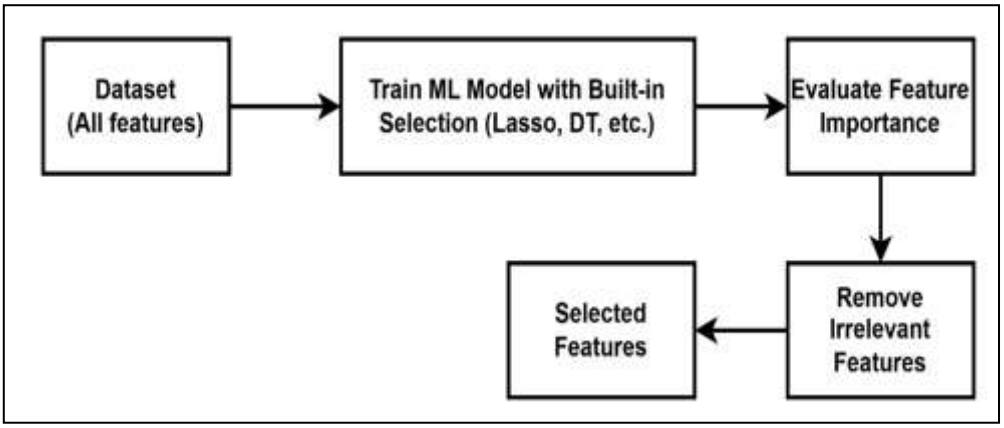

Figure 1.4 Embedded Method for Feature Selection

The general pipeline of Embedded method of FS is shown in Figure 1.4. The dataset is trained using the model with built-in FS method where feature importance is evaluated and irrelevant features are removed within the training process.

1.1.3 Resampling

To address the skewed distribution of classes frequently observed in intrusion detection data, resampling techniques are applied during preprocessing, especially when one class, such as benign traffic or attacks, vastly exceeds the other in quantity. This unequal class distribution can cause the ML models to be biased favouring the classes with high occurrences, which can again result in poor detection performance on lesser occurrence classes. Resampling modifies the occurrences of classes in the

dataset to improve representation for each class. This helps the ability of ML model on learning the patterns especially from underrepresented data. Resampling can be divided into two parts: oversampling and undersampling. Oversampling aims to balance the class distributions of the dataset by augmenting the class with lesser occurrences, commonly using methods such as SMOTE, which synthetically generates new data points [10]. Undersampling conversely decreases the instances belonging to the majority class by removing records, potentially using methods like NearMiss or Tomek Links. Resampling is crucial for IDS because failing to account for imbalanced data may result in overall high accuracy while yielding poor detection rate for rare and critical attacks. By balancing the dataset, resampling enables ML models to generalize better and improves evaluation metrics for the minority class [11].

1. **Random Undersampling and Random Oversampling:** Random Undersampling (RU) and Random Oversampling (RO) are fundamental resampling techniques used for addressing the unequal distribution of classes present in the datasets, especially in IDS where normal traffic often dominates attack instances. RO works by randomly duplicating the minority class so that the minority class can have a good representation in the dataset. While this method is simple and effective in improving the model's sensitivity to minority class, it may lead to overfitting, as it replicates existing patterns without adding new information [12]. RU, conversely, remove instances from the majority class randomly so that the majority class will not overwhelm the dataset. This method cuts down the training time and thus is computationally efficient, but it risks discarding potentially informative data, which might hurt the model's ability to generalize [13]. Both techniques are useful in different contexts and may be used as per requirements. RO is preferred when retaining all original data is crucial, while RU is used when model training time and dataset size are major concerns.
2. **SMOTE Oversampling:** SMOTE (Synthetic Minority Over-sampling Technique) is a common resampling technique intended to handle class imbalance by generating synthetic samples that is similar to the minority

class, rather than creating duplicate of the minority class. Proposed by Chawla et al. [10], SMOTE create a new artificial sample by first selecting a sample from the minority class, and find its nearest k minority class neighbour, creating a line between them and create new point (instances) within the space that lies between the selected samples. This interpolation improves the representation of the minority class in a more informative and diverse manner than random oversampling. The use of SMOTE can be beneficial as it helps mitigating the risk of overfitting that often occurs with RO, as it introduces novel instances instead of merely replicating existing ones. By enriching the feature space of the under-represented class samples, SMOTE helps the ML model distinguish between classes more effectively and increases its sensitivity to the minority classes. However, SMOTE can sometimes generate ambiguous samples in overlapping regions of the classes, potentially increasing misclassification if not combined with noise-filtering methods or used carefully with high-dimensional data.

3. **Near Miss Undersampling:** NearMiss is a family of undersampling techniques used to address class imbalanced by selectively removing instances from the majority class samples according to how close they are to the minority class samples. Unlike RU, NearMiss methods aim to preserve the most informative majority class instances, especially those that are hardest to distinguish from the minority class. There are three common variants of NearMiss: Firstly, NearMiss-1 selects majority class samples that have the lowest average distance to their k nearest minority class neighbours. Secondly, NearMiss-2 which chooses majority class samples that have the smallest average distances to the k farthest minority class instances. And Lastly, NearMiss-3 which retains majority samples that are closest to each minority class instance, helping preserve local structure [14]. By focusing on majority samples near the decision boundary, NearMiss helps ML models learn more effective discriminative patterns, which is critical in applications like intrusion detection where detecting subtle attack patterns is vital. However, NearMiss can lead to the removal of a large number of majority

instances, potentially causing information loss, especially when the dataset is highly imbalanced.

1.1.4 Machine Learning Model

In this sub-section, brief introductions of the traditional ML techniques which are used in this thesis are discussed:

1. **Decision Tree (DT)**: A DT is a tree-structured ML model that makes predictions by repeatedly dividing the data into smaller parts based on feature values. Each internal node of the tree represents a condition or rule derived from the input data, guiding the flow toward the final output at the leaf nodes, with each leaf node representing a specific class label. The splitting process is typically guided by impurity evaluation methods, viz. Gini index, entropy, or information gain [15]. A key strength of DT lies in their ease of interpretability. The model's structure is intuitive, making it accessible to both technical and non-technical users. It also requires minimal data preprocessing, handling both numerical and categorical features, feature scaling does not have positive as well as negative impact on DT. However, they are prone to overfitting, especially when grown deep without pruning. They can be highly responsive to minor changes in the data, potentially resulting in inconsistent or unstable model structures.
2. **XGBoost**: Extreme Gradient Boosting (XGBoost) is a ML algorithm widely used for classification due to its high accuracy and scalability. This method constructs a sequence of DTs, with each subsequent tree aiming to reduce the prediction errors made by the previous ones by minimizing a loss function using gradient descent. XGBoost demonstrate strong performance in intrusion detection tasks, especially when labelled data is available, as it learns directly from known attack and normal instances. It effectively handles large, high dimensional datasets and missing values. However, it may require careful parameter tuning and is less effective in environments lacking labelled data. Despite these challenges, XGBoost remains a robust and powerful choice for supervised anomaly detection in complex network environments [16].

3. **K-Nearest Neighbour (KNN)**: KNN is a simple ML algorithm used for both classification and regression tasks. The core idea of KNN is to classify a data point based on the majority class among its k-nearest neighbours in the feature space, using distance metrics such as Euclidean, Manhattan, or Minkowski distance [17]. One of the main advantages of KNN is its non-parametric nature, indicating that it does not rely on predefined distributional properties of the data. It is also easy to implement and interpret. KNN can effectively capture nonlinear relationships in the data and works well with multi-class classification problems. However, KNN suffers from several limitations. It is computationally expensive, especially with large datasets, since it stores all training samples and performs calculations during the prediction phase. Moreover, KNN is sensitive to irrelevant features and feature scaling, requiring proper preprocessing for good performance. It also struggles with imbalanced datasets, where the majority class may dominate predictions.
4. **Logistic Regression (LR)**: LR is a ML algorithm for binary classification problems. It predicts the probability that a data instance belongs to a particular class using the logistic (sigmoid) function, which outputs values between 0 and 1. The model estimates coefficients by maximizing the likelihood of the observed data using methods like gradient descent [18]. LR is particularly interpretable, and has low computation overhead, works well on linearly separable datasets, and can be regularized to prevent overfitting. However, it has several limitations. LR relies on a linear association between the independent variables and the logarithm of the odds for the target class, making it less effective for capturing complex, nonlinear patterns. It also struggles with multicollinearity among input variables and is sensitive to outliers. In imbalanced datasets, its performance may skew toward the majority class, requiring techniques like resampling or class weighting.
5. **Naïve Bayes (NB)**: NB is a probabilistic classification algorithm based on Bayes' Theorem, which calculates the posterior probability of a class given the input features. The model assumes conditional independence among features when the class label is known, an assumption that rarely holds true in

practice, hence the term "naïve" [19]. Despite this strong independence assumption, NB performs surprisingly well in many applications, including those with high dimensional data. It has low computation overhead, and is easy to implement, and requires a relatively small training data. The main advantage of NB lies in its speed and scalability. It works well with large datasets and can be trained quickly. Moreover, it is robust to irrelevant features and handles missing data reasonably well. However, its key disadvantage is the assumption of feature independence, which can lead to lower accuracy if the features are highly correlated. It also assumes that features follow specific distributions (e.g., Gaussian for continuous data), which may not always be true.

6. **Quadratic Discriminant Analysis (QDA)**: QDA is a classification algorithm based on the Bayesian decision theory. It assumes that each class in the data follows a Gaussian distribution, QDA allows for different covariance matrices for each class. This flexibility enables it to model more complex, non-linear decision boundaries, making QDA more powerful when class distributions differ significantly in shape and spread [20]. QDA is particularly useful when the assumption of equal co-variances does not hold. Its non-linear boundaries make it suitable for datasets where classes are not linearly separable. Additionally, QDA works well with moderate sized datasets and performs better than linear models when the data distribution supports it. However, QDA also comes with limitations. The need to compute a distinct covariance matrix for every class makes the process computationally intensive and unreliable when only limited data samples is there or with very large dataset. This can lead to overfitting and poor generalization if the number of features is large compared to the number of observations.
7. **Random Forest (RF)**: RF is an ensemble learning algorithm that builds a large number of DT during training and outputs the mode of the classes (for classification) or the mean prediction (for regression) of the individual trees [21]. It introduces randomness in two ways: by bootstrapping (sampling with replacement) from the training data and by selecting a random subset of features for each tree split. This helps in reducing overfitting and improving

generalization performance. The advantage of RF is its robustness and accuracy on a wide range of datasets, including high-dimensional and noisy data. It also provides feature importance scores, which are valuable for FS. Moreover, it is less sensitive to missing data and outliers compared to individual DT. However, RF models can be computationally intensive, especially with a large number of trees or when applied to massive datasets. Also, the model's interpretability is lower compared to a single DT, making it harder to explain individual predictions.

1.1.5 Performance Metrics

In evaluating the effectiveness of ML models for IDS, selecting appropriate performance metrics is crucial. These metrics help quantify how well a model distinguishes between malicious and legitimate network activities. Unlike traditional classification tasks, intrusion detection often deals with highly imbalanced datasets, where accuracy alone is insufficient to capture a model's reliability. Therefore, a combination of metrics is usually utilized to ensure a thorough assessment of the model's effectiveness [22]. Performance metrics are used to assess the quality of the predictions made by a model. The performance metrics can be calculated from the confusion matrix shown in Table 1.1.

Table 1.1 Confusion Matrix

Confusion Matrix		**Predicted**	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

- Accuracy: This metric quantifies the accuracy of the model by evaluating the proportion of correct predictions.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

- Precision: This metric measures the accuracy of positive predictions by evaluating the proportion of true positives

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

- Recall: This metric evaluates the model's ability to correctly identify true positive instances.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

- False Positive Rate (FPR): It refers to the ratio of actual negative instances that are wrongly classified as positive by the model.

$$\text{FPR} = \frac{FP}{FP+TN} \quad (4)$$

- False Negative Rate (FNR): The FNR measures the ratio of actual positive instances that are wrongly classified as negative by the model.

$$\text{FNR} = \frac{FN}{FN+TP} \quad (5)$$

- F1-Score: It is a balanced measure of the model's accuracy in terms of false positive and false negative rates. It is also called the harmonic mean of precision and recall.

$$\text{F1-Score} = 2 \text{ x } \frac{Precision*Recall}{Precision+Recall} \quad (6)$$

- F-measure: The F-measure mixes the properties of the previous two measures as the harmonic mean of precision and recall.

$$\text{F} - \text{measure} = \frac{2}{\frac{1}{\text{Precision}}+\frac{1}{\text{Recall}}} \quad (7)$$

The chosen metrics provide a comprehensive and sufficient evaluation for this study. Accuracy reflects overall correctness, while precision indicates the impact of false positives. Recall indicates detection of all intrusions or anomaly, prioritizing security. FNR and FPR are the rate of incorrect classifications.

1.2 Network Security

Network security has become a critical concern in the digital era, with IDS playing a pivotal role in safeguarding computer networks against unauthorized access and malicious activities. As cyber threats evolve in complexity and frequency, the development and implementation of effective IDS have garnered significant attention in both academic research and industry applications. Network security encompasses policies, procedures, and technologies designed to protect the

underlying network infrastructure from unauthorized access, misuse, or theft [23]. Traditional perimeter based defence mechanisms such as firewalls and antivirus software remain foundational; however, they are increasingly inadequate in combating sophisticated and stealthy attacks like Advanced Persistent Threats (APTs) and zero-day exploits [24]. In response to these limitations, security researchers and practitioners advocate for IDS, which monitor network traffic for suspicious behaviour and policy violations, often leveraging ML for anomaly detection [25].

Network security is a critical aspect of modern digital infrastructures, ensuring the confidentiality, integrity, and availability of data as it traverses communication networks. The shift from the conventional network architectures to Software-Defined Networking (SDN) has introduced both advancements and challenges in the realm of network security. In conventional networks, security mechanisms are typically distributed across various hardware devices such as routers, switches, and firewalls. These devices operate autonomously, enforcing security policies and managing traffic based on predefined rules. However, this decentralized approach often leads to complexities in policy management, inconsistencies in security enforcement, and challenges in responding swiftly to emerging threats. Moreover, the static nature of traditional networks makes them less adaptable to dynamic security requirements [26]. SDN introduces a paradigm shift by decoupling the control plane from the data plane, centralizing network intelligence, and enabling programmability of network behaviour. This architectural transformation offers enhanced flexibility and simplified management. However, it also brings forth new security considerations across different planes of the SDN architecture:

1. **Control Plane**: The SDN controller, acting as the network's brain, becomes a high value target for attackers. Compromising the controller can grant adversaries control over the entire network, leading to potential data breaches or service disruptions.
2. **Data Plane**: While SDN enhances network agility, it also introduces vulnerabilities in the data plane. For instance, the separation of control and data planes can expose communication channels to interception or

manipulation. Implementing robust encryption and authentication mechanisms is essential to safeguard data plane communications.

3. **Application Plane**: SDN applications, which dictate network behaviour, can be exploited if not properly secured. Malicious or vulnerable applications may introduce unauthorized network configurations or expose the network to further attacks.

While traditional networks rely on distributed security mechanisms, SDN offers centralized control. However, this centralization also means that any compromise of the SDN controller can have widespread implications. Therefore, transitioning to SDN requires not only leveraging its benefits but also addressing its unique security challenges through comprehensive strategies, including controller hardening, secure communication protocols, and vigilant application management [27].

1.3 Intrusion Detection System

An IDS plays a vital role in cybersecurity by monitoring system or network activities for signs of malicious behaviour. Its main objective is to identify unauthorized access, suspicious actions, or abnormal patterns within the network. IDS can be categorised into Information Source which can further be divided into Host based IDS and Network based IDS, and Detection Method which is further divided into Signature based IDS and Anomaly based IDS [28] as shown in Fig 1.5.

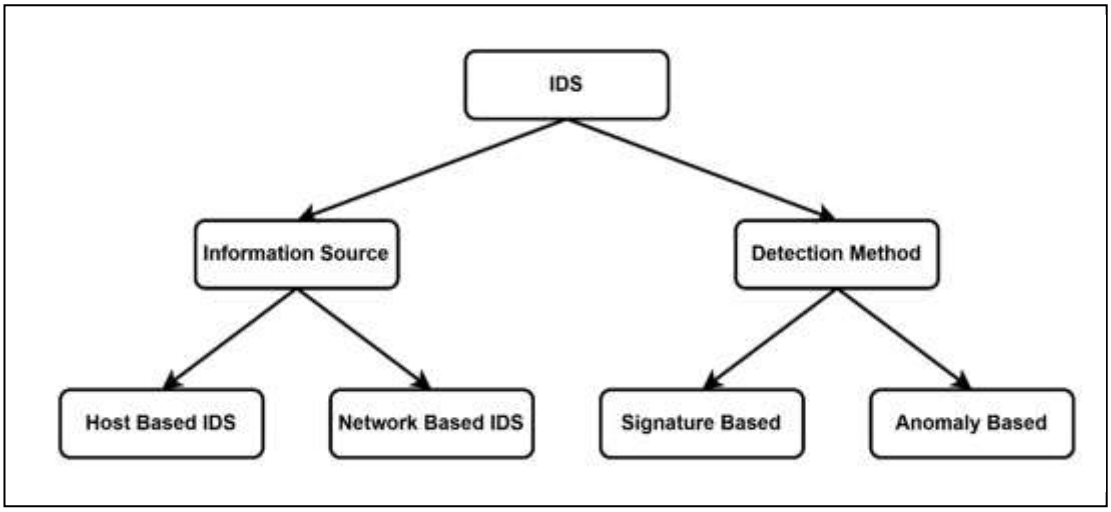

Figure 1.5 Basic IDS taxonomy

1. **Host Based IDS**: A Host-Based IDS is installed directly on an individual device or system, such as a server or computer. It monitors system-level activities, including file access, system calls, application logs, and user behaviour. Because it operates at the host level, it can detect internal threats or unauthorized modifications that might not be visible at the network level. HIDS is particularly useful for identifying suspicious behaviour within a specific machine [29].
2. **Network Based IDS**: A Network-Based IDS is deployed at strategic points within a network to analyse traffic passing through the entire network. It inspects data packets for signs of malicious activity, such as unusual traffic patterns, port scanning, or suspicious payloads. Unlike Host Based IDS, Network Based IDS does not focus on individual systems but rather monitors network communications as a whole, making it effective at detecting attacks such as DDoS or Man-in-the-Middle (MITM) attacks [30].
3. **Signature Based IDS**: Signature-Based IDS operates by comparing network or system activity to a predefined database of known attack signatures or patterns. When a match is found, the IDS generates an alert. This method is highly accurate for detecting previously identified threats, however, it struggles to identify novel or previously unseen threats, such as zero-day exploits, since they do not match any existing signature [31].
4. **Anomaly Based IDS**: An Anomaly-Based IDS builds a baseline model of what is considered normal behaviour within a system or network. It then monitors activity and flags deviations from this baseline as potential threats. This approach is capable of identifying previously unseen attacks, but it may also produce more false positives because unusual but legitimate activities can be mistaken for malicious behaviour [32].

An IDS acts as a security alarm within a network, analysing traffic and system behaviour to detect anomalies or known attack patterns, and promptly notifying administrators or connected systems when suspicious activity is identified. While IDS are important tools in monitoring and detecting potential threats, they are not without their challenges. IDS often struggle with false positives, where benign activity is mistakenly flagged as malicious. While this can cause unnecessary alerts

and waste resources, a more serious issue is false negatives, where actual threats go undetected. In such cases, malicious actors can infiltrate the organization's network without being noticed, leaving IT and security personnel unaware of the breach. While IDS detect threats and alert the system, Intrusion Prevention System (IPS) offers real-time threat blocking thereby preventing threats in real time before they cause harm to the system. But IPS come with the risk of false positives which could possibly block important communication or even shut down services.

1.4 Machine Learning in Network Security

Machine learning (ML) plays a critical role in extracting insights from large datasets, detecting anomalies, and automating decision-making. It includes various paradigms such as supervised, unsupervised, and reinforcement learning. The increasing availability of data and computational power has driven significant progress in ML applications across domains such as healthcare, finance, and cybersecurity [33]. Unsupervised learning [34] involves training models on data without labelled responses. The goal is to uncover hidden patterns or groupings in the data. This technique is widely used for clustering, anomaly detection, and dimensionality reduction. It is especially useful in exploratory data analysis where labels are not available. Algorithms such as K-Means, DBSCAN, and Principal Component Analysis (PCA) are commonly used. Reinforcement learning (RL) is a learning paradigm where an agent interacts with an environment by taking actions and receiving rewards. It learns to make sequences of decisions that maximize cumulative rewards. RL is widely used in robotics, game playing, and autonomous systems. Key concepts include exploration, exploitation, states, and policies. Algorithms like Q-learning and Deep Q-Networks are popular RL methods [35].

Supervised learning is a type of ML where the algorithm is trained on a labelled dataset, meaning the input data is paired with correct output labels. It aims to learn a mapping function from input to output to predict unseen data accurately. Common applications include spam detection, image classification, and fraud detection. Algorithms such as Decision Trees (DT), Support Vector Machines, and Logistic Regression (LR) fall under this category [36].

ML has evolved as a crucial element in enhancing network security, particularly through the development of IDS. Traditional IDS methods, which rely heavily on predefined signatures, often struggle to detect novel and sophisticated cyber threats. In contrast, ML-based IDS can analyse patterns in network traffic to identify both known and unknown attacks, offering a more dynamic defence mechanism. Recent studies have introduced innovative ML models to improve IDS performance. For instance, a study published in Scientific Reports [37] combined ML and deep learning (DL) techniques to enhance intrusion detection capabilities. Another research by Kaushik et al. [38] presented robust IDS using a unique feature selection algorithm based on basic statistical methods, resulting in improved detection accuracy and reduced training time.

1.5 Motivation

As digital technology continues to advance, the number of devices and users connected to the internet is growing rapidly. Alongside these advancements, there has been a corresponding rise in the complexity and frequency of cyber threats. From large-scale DDoS attacks to stealthy, targeted intrusions, network environments today face a diverse array of security challenges. These attacks can lead to financial loss, data breaches, service downtime, and even damage to national infrastructure. Consequently, ensuring the safety of network systems has become a vital concern for both public and private organizations.

Traditional security tools such as firewalls, antivirus programs, and static access control mechanisms have long been used to defend networks. While these technologies offer a basic level of protection, they are often ineffective against modern attacks that are adaptive, fast-changing, and sometimes unknown to the system. In particular, signature-based IDS, which rely on predefined attack signatures, can fail to detect new or evolving threats. On the other hand, anomaly-based IDS, which identify deviations from normal behaviour, often suffer from high false alarm rates (FAR), reducing their reliability and increasing the workload for security analysts. These limitations highlight the need for more intelligent and flexible security mechanisms. One promising direction is the use of ML in the design of IDS. ML techniques can analyse patterns in network traffic and learn to

distinguish between legitimate and malicious behaviour. Unlike traditional rule-based systems, ML models can continuously improve as they are exposed to new data. This adaptability is crucial for dealing with the dynamic nature of cyber threats.

The integration of ML into intrusion detection has shown significant potential, but it also brings new challenges. One of the major concerns is the false positive rate (FPR)—the frequency with which legitimate traffic is misclassified as malicious. High FPR can lead to unnecessary alerts, wasted resources, and loss of confidence in the system. Reducing false alarms while maintaining high detection accuracy is therefore an essential goal in the development of ML-based IDS. This need forms a central part of the motivation behind this research. Another motivation stems from the fact that most existing IDS solutions attempt to create one-size-fits-all models. These models aim to detect various types of attacks using the same algorithm and feature set. However, different attack types often exhibit distinct characteristics and patterns. For example, DoS attacks typically involve high traffic volume, while user-to-root (U2R) attacks may be more subtle and occur over a longer period. A model trained to detect one type of attack may not perform well on others. Therefore, creating attack-specific models or protocols could lead to more accurate and reliable detection systems.

The rise of SDN offers a new opportunity to enhance the efficiency of IDS. SDN separates the control plane from the data plane, allowing for centralized control and real-time programmability of the network. This centralized view makes it easier to monitor traffic flows, apply security policies, and respond to threats quickly. An SDN-enabled environment also supports dynamic and automated responses, which are difficult to implement in traditional networks. Designing lightweight, SDN-based IDS protocols that can integrate ML algorithms effectively is a promising area of exploration. Additionally, another challenge lies in the high dimensionality of network traffic data, which includes numerous features such as port numbers, IP addresses, packet sizes, and protocol types. Not all of these features contribute equally to the detection of intrusions, and using too many can lead to increased computational costs and overfitting. Thus, applying feature selection techniques to

identify the most relevant attributes is critical for building efficient and scalable IDS solutions.

While several datasets like NSL-KDD, UNSW-NB15, and CIC-IDS are available for training and testing ML-based IDS, many studies still focus heavily on optimizing performance metrics such as accuracy, with less attention given to real-world issues like deployment, interpretability, and resource constraints. Moreover, some approaches that show high accuracy in lab environments fail to generalize well in live network conditions. There is a need for practical, adaptive IDS solutions that strike a balance between accuracy, speed, robustness, and scalability. In light of these considerations, this research is motivated by the desire to develop an efficient, intelligent, and practical intrusion detection protocol that addresses these ongoing challenges. The proposed study aims to investigate how different ML algorithms perform on specific types of network attacks, minimize false positives without sacrificing detection accuracy, apply feature selection (FS) techniques to reduce complexity, and explore the use of SDN to deploy an effective, real-time IDS framework.

1.6 Objectives

Considering the security concerns and research gap in IDS, the objectives are broken down into the following points:

1. **Development of Efficient Protocol to Detect Malicious Nodes for Intrusion Detection System Using ML**: To develop an efficient protocol for Network-IDS using ML/DL techniques to get the optimum results, overcoming the existing models. Apart from classification accuracy, the focus will be on FPR. FPR is one of the big concerns in IDS. In this objective, the IDS dataset containing different kinds of attack will be classified with ML/DL algorithms into normal and malicious node regardless of what kind of attack it may contain.
2. **Development of Efficient Protocol for Detection of Attack**: To develop an efficient protocol for detection of particular types of networking attack using ML/DL techniques. A Network-IDS cannot have a single model that is efficient for all kinds of networking attacks, so, developing a protocol for

particular type of attack will give the information for what algorithms is best suitable for such particular attack. Some common attack found in publicly available datasets are DoS, Probe, U2R, R2L, web attack, etc. In this objective, classification will be done for a particular type of attack at a time.

3. **Development of Efficient SDN Based Network Intrusion Detection System**: To develop an efficient SDN Based Network-IDS using ML/DL algorithms. In view of IDS, SDN is an additional monitoring mechanism. To develop a FS method with classifiers which reduces the dimensions of the dataset as well as reducing the computational cost and time, this is an on-going challenge as extra-large dataset are problematic for researchers in various ways.

1.7 Contributions

The main contributions are:

1. A comprehensive analysis and review of recent research articles focused on IDS. It examines the most commonly utilized benchmark datasets, explores the range of ML algorithms applied in IDS development, and identifies the types of cyber threats that these systems are designed to detect. The review also outlines the existing limitations in current IDS solutions and brings attention to key research gaps. To enhance understanding of how well various IDS address different categories of cyber threats, the study introduces a flexible and extendable taxonomy for classifying cyberattacks.
2. An anomaly-based IDS is developed using an ensemble approach. Three ML algorithms viz. naïve bayes (NB), Quadratic Discriminant Analysis (QDA), and DT (ID3) are combined for their low computational cost. Two intrusion detection datasets viz. CIC-IDS2017 dataset and CSE-CIC-IDS2018 dataset were used for evaluation of the proposed methods. Random forest regressor is used for FS. The ensemble algorithm results are compared with the standalone algorithms and significant improvements had been achieved with the ensemble method especially with classification accuracy and FAR in both the datasets.
3. Two staged feature selection method: A novel two-stage FS method (GIGA) to optimize and enhance IDS is developed to reduce high dimensionality of

intrusion detection datasets while also improving detection accuracy. The first stage employs Gini impurity (GI) to filter out features with less importance, followed by a Genetic Algorithm (GA) with a DT-based fitness function to identify the most relevant subset of features. Experiments on the CIC-IDS2017, CSE-CIC-IDS2018, and CIC-DDoS2019 datasets demonstrate notable improvements: especially on test accuracy FPR. The number of features is significantly reduced from 71 to 8, 70 to 4, and 69 to 8 for the datasets, respectively. The proposed method improves detection accuracy across ML models like Random Forest (RF) and Decision Tree (DT). By addressing the key challenges in dimensionality and performance, GIGA offers a scalable and robust solution for enhancing ID systems.

4. Hybrid sampling: A hybrid sampling approach designed to enhance IDS for IoT environments by addressing data imbalance. The hybrid sampling method combines the Near Miss-1 undersampling technique with Synthetic Minority Over-sampling Technique (SMOTE) to create a more balanced dataset without significant loss of information from the data. The Bot-IoT dataset is used to evaluate the proposed approach across various ML algorithms, assessing their performance using key metrics including accuracy, FPR, and False Negative Rate (FNR). The imbalance Bot-IoT dataset, which was originally 99.99:0.01 ratio has been resampled to 80:20 ratio. The experimental results demonstrate that RF and K-Nearest Neighbour (KNN) excel across all metrics including accuracy and FPR. The performance of the proposed approach is compared with other state-of-the-art methods, focusing on binary classification tasks and the ability to detect attacks while minimizing FPR. This study provides valuable insights into the application of hybrid sampling techniques for improving the detection capabilities of IDS in IoT networks.
5. Zero-day attack detection: A model to detect zero-day attack is developed using ML. The model is evaluated using InSDN dataset which is an intrusion detection dataset particularly developed for SDN network. The InSDN dataset contain different kinds of attacks. Zero-day attack detection is performed by separating the dataset into training set and test set, where the

test set contain a particular type of attack (in addition to attacks that are present in the training set) that is not present in the training set. This step is repeated for each kind of attack in the dataset. This experiment shows which attack type is vulnerable to IDS and which attack type is robust against the IDS.

1.8 Thesis Organization

This thesis consists of 7 chapters, which are organised as follows;

Chapter 1 gives introduction on ML, Network Security, taxonomy and definitions. IDS and the uses of ML in IDS. Research motivations, thesis objectives and contributions of the thesis are highlighted in this chapter. Chapter 2 presents a comprehensive review of recent research related to the topic. This chapter serves as the groundwork for the current study by identifying existing research gaps and potential areas for improvement. Through this review, the chapter helps to shape the research objectives outlined in the following chapters.

Chapter 3 provides a detailed description and analysis of the intrusion detection datasets utilized in this study. It also presents the description of each attack present in the intrusion detection dataset. Chapter 4 presents the proposed the ML Ensemble Method for classification of attack and normal data in two intrusion detection datasets; CIC-IDS2017 dataset and CSE-CIC-IDS2018 dataset. The blending ensemble method is a combination of NB, QDA and DT which are selected because of their difference classification method and low computation cost. Different performance metrics like Accuracy, Precision, Recall, FPR, FNR, and also computation time (in second) were used.

Chapter 5 address the data imbalance in the intrusion detection dataset by introducing hybrid sampling method. The hybrid sampling method is a combination of SMOTE oversampling and Near Miss Undersampling. Bot-IoT dataset is used in this chapter, the majority class is undersampled using Near Miss method without altering the number of instances of minority class, then the minority class is oversampled using SMOTE without changing the number of the majority class (which is already undersampled).

Chapter 6 introduces a Two-Staged feature selection method by utilizing a Filter method of FS as 1st stage and wrapper method of FS and 2nd stage. Gini Impurity is used for filter method and GA is used for Wrapper method. Chapter 7 presents a comprehensive experiment on InSDN dataset that is deign specifically for SDN environment. Leave One Attack Out (LOAO) technique is employed to simulate zero-day attack detection. And Lastly, Chapter 8 holds the Conclusion of the thesis and Future Works in the domain.

CHAPTER 2

LITERATURE REVIEW

With the rapid expansion of digital infrastructure, protecting systems from unauthorized access and malicious activities has become increasingly vital. IDS are crucial tools in this domain. Detection strategies in IDS are mainly divided into signature-based methods and anomaly-based methods, which often using ML or statistical models. As threats evolve, the utilization of ML and artificial intelligence has shown promise in enhancing detection efficiency and minimizing false alerts. This literature review presents a detailed examination of existing IDS approaches, tracking their development from conventional techniques to more advanced, intelligent solutions, thereby setting the stage for the research focus of this study.

2.1 Machine Learning in IDS

IDS can be categorised into two types, signature-based IDS and anomaly-based IDS. Anomaly based IDS can further be categorised into Statistical methods which use mathematical models to establish a baseline of normal system behaviour, Heuristic methods which rely on expert defined rules or threshold to detect abnormal activities and lastly ML-based methods which use algorithms that learn patterns from historical data to classify normal or malicious behaviour. In this thesis, we are limiting the focus to ML techniques and an overview of some recent researches are highlighted. A comparison of ML techniques viz. KNN, LR, and perceptron neural networks on NSL-KDD is performed for intrusion detection. Neural networks demonstrate exceptional performance metrics viz. accuracy, precision and recall, in detecting network threats, with potential for hybrid models to enhance IDS solutions [39]. Taylor, et.al [40] provides an overview of various IDS types, essential evaluation metrics, and the evolving field of ML techniques, while also highlighting potential future directions, including the creation of updated datasets and the effective utilization of cloud-based resources.

A novel IDS leveraging ML techniques to enhance network security is proposed to address the challenges in real-time threat detection [41]. The paper designs an IDS using ML techniques like LR, Random Forest (RF), and XGBoost for

real-time threat detection, classifying attacks into DDoS, Remote to Local (R2L), U2R, or Probing. The working model classifies incoming requests as normal or intrusions, categorizing the latter into various types of attacks. Using the CSE-CIC-IDS dataset, Singh et al., compares signature-based systems and ML-based systems. 10 different ML algorithms were tested and is found that the ML-based systems outperform the signature-based systems. Top models which achieve high performances include KNN, XGBoost and Adaboost algorithm [42]. An efficient network intrusion detection and classification system was developed using ML [43]. Leveraging NSL-KDD dataset and UNSW-NB15 dataset, and using ML algorithms like SVM, RF, and Neural Networks to improve threat detection capabilities and lower the rate of false alarms, improving network security and computational efficiency.

The use of ML in IDS has also extend to IoT environment. A study by Iqbal et al., [44] explores ML and DL algorithms for secure IoT and Smart City IDS, examining architectures, methods, and techniques such as federated learning, edge computing, and explainable AI to improve security and resource usage. A comprehensive survey of ML-driven IDS strategies is carried by Hemraj et al., [45]. Encompassing diverse classification algorithms, FS methodologies, and anomaly detection techniques, encompassing singular, amalgamated, and ensemble classifiers is furnished. It is also found that Innovative methodologies enhance intrusion detection accuracy. Likewise, there are numerous researches on IDS using ML that contributes greatly to the area of cyber and network security.

2.2 Ensemble Methods for IDS

An ensemble method combines predictions from multiple ML models to enhance accuracy, resilience, and the ability to generalize beyond what a single model can achieve. Instead of relying on one algorithm, ensemble methods integrate the strengths of several algorithms to reduce errors like bias, variance, or overfitting. Ensemble methods are widely used in IDS, especially in IoT and cybersecurity domains, for their ability to handle imbalanced datasets and detect complex patterns. Ensemble methods can be broadly divided into Bagging, Boosting and Stacking.

2.2.1 Bagging

Bagging is short for Bootstrap Aggregating, it constitutes a collective learning strategy aimed at augmenting the predictive dependability of models, especially those vulnerable to fluctuations in output when exposed to variations in training data [46]. This methodology operates by constructing numerous distinct learners, each trained on a separate, randomly generated subset of the available data, obtained through sampling with replacement. These independent learners yield individual predictions, which are subsequently amalgamated, commonly through a majority consensus in classification tasks or by averaging in regression contexts. Such aggregation diminishes the susceptibility of the overall model to random noise or anomalies present within the training set, thereby enhancing its resilience and generalisation capability. Bagging is frequently employed with inherently unstable models, like decision trees, where its ability to curtail overfitting proves particularly advantageous.

Rosy et al., [47] proposed a bagging ensemble approach to increase detection rate accuracy for all attack types of UNSW-NB15 dataset. They aim to enhance the detection rates of various types of attacks, as well as for specific type of attacks by using the ensemble of NB and RF. K-fold cross-validation is used to evaluate the proposed approach, ensuring robust assessment of its performance. The ensemble method outperforms the standalone method in accuracy, precision, and recall. Pathak et al., [48] also focuses on the development of an ensemble model for IDS that combines the strengths of different ML classifiers, utilizing the CICIDS-2017 intrusion detection dataset for analysing and classifying various cyber-attack types. Their proposed bagging ensemble model integrates RF and Bagging Classifier (BC) using a voting scheme ensemble technique. Their proposed model achieve a remarkable accuracy of 99.85%. The proposed method provides a robust and accurate detection method for dynamic intrusion detection using ensemble of ML-based techniques.

Ntayagabiri et al., [49] introduces Optimized Multiclass Intrusion Classifier (OMIC), a novel and resource efficient ensemble learning framework designed specifically for detecting intrusions in large scale IoT networks. OMIC addresses the

limitations of traditional IDS through an optimized ensemble model that combines the strengths of LightGBM and XGBoost within a bagging based architecture. In this work, both classifiers apply randomized subsampling strategies, such as row and column sampling, and generate multiple learners whose outputs are aggregated using probability averaging. This ensemble mechanism enhances robustness against overfitting and improves generalization across a wide spectrum of attack types. Supporting this core methodology is a memory-efficient processing pipeline designed for high-throughput environments. It includes adaptive chunk based data handling, type conversion for memory savings, and frugal learning to address the class distribution disparities. The framework is rigorously evaluated on the CICIoT2023 dataset, which contains over a million records and 33 attack types. OMIC achieves a classification accuracy of 99.26%, with high precision and recall, particularly excelling in the detection of volumetric threats such as DDoS and DoS. In comparative analysis, OMIC consistently outperforms conventional methods and DL models, including RF, CNN, LSTM, and GRU, in different performance metrics, especially in stability across diverse attack categories. Preprocessing techniques such as label encoding, robust scaling, and different conversion contribute to the system's operational efficiency. Despite its strong performance, OMIC shows limitations in identifying certain low prevalence or stealthy attacks, such as web-based and reconnaissance intrusions highlighting a potential area for future refinement. Nonetheless, by integrating a sophisticated bagging strategy with high-performing gradient boosting models and efficient resource management, OMIC offers a powerful and scalable solution for real-time IoT network security, demonstrating practical value for deployment in real-world systems.

Zewdu et al., [50] introduces a hybrid intrusion detection approach that merges ensemble learning with expert system reasoning to improve both detection rate and interpretability in network security. Recognizing the limitations of individual classifiers in handling complex and evolving threats, the authors evaluate three ensemble strategies; bagging, boosting (AdaBoostM1), and random subspace. Each ensemble methods are applied to three base classifiers: RF, Bayes Net, and SMO (Sequential Minimal Optimization). Among these, the bagging method plays a

critical role by leveraging bootstrap sampling to train multiple models on varied subsets of the data, thus reducing variance and improving generalization. In particular, the bagging ensemble combined with RF achieved perfect precision, recall, and F1-score (100%), demonstrating its strength in consistent classification without overfitting. Bagging was also shown to enhance the performance of SMO and Bayes Net, albeit with slightly less effectiveness than in RD. The evaluation was conducted using the NSL-KDD dataset with 10-fold cross-validation, ensuring robustness in the results. While boosting showed marginally higher performance overall, the bagging based models delivered stable and accurate results with minimal false positives, making them especially useful in high-stakes environments where reliability is critical. To address the knowledge gap often associated with ML based IDS, the proposed framework incorporates an expert system that interprets and augments the output of ensemble classifiers using rule-based reasoning derived from domain expertise. This integration allows not only for high-accuracy intrusion detection but also for actionable insights and countermeasures. Though highly effective, the study acknowledges certain limitations, including the use of all dataset attributes without FS and the challenge of interpreting complex ensemble outputs. Overall, the research demonstrates that combining ensemble methods particularly bagging with expert systems can yield a robust and interpretable solution for intrusion detection and network forensics.

A bagging ensemble framework aimed at improving anomaly detection in IoT sensor environments is proposed by Reddy et al., [51]. The proposed method is vulnerable to irregular and unexpected behaviours due to their distributed and resource constrained nature. The authors propose leveraging the ensemble learning paradigm specifically, the bagging technique to enhance the classification performance of base models while mitigating variance and overfitting, both of which are common issues in standalone learners. This is achieved by bagging method by generating multiple versions of the training dataset through random sampling with replacement and training each individual classifier on these diverse subsets. In the proposed model, the outputs of multiple classifiers are aggregated through majority voting, which contributes to more robust and generalized predictions. The study uses

datasets collected from IoT sensors in a smart environment and evaluates the system's performance across various base classifiers such as DT and SVM. The results indicate that the bagging ensemble approach significantly outperforms individual classifiers in terms of accuracy, precision, recall, and F1-score, with notable improvements in detecting anomalies that would otherwise be missed by single models. In addition, the model demonstrate flexibility and scalability across various IoT settings, demonstrating strong potential for real world deployment in domains like smart cities and industrial monitoring. The researchers also highlight the inherent strength of bagging in reducing prediction instability caused by noise and sensor level variability. While the approach proves effective, the paper acknowledges limitations such as the computational overhead associated with maintaining multiple learners and the need for further investigation into online or incremental ensemble learning methods for real time applications.

Raghad et al., [52] presents an ensemble based intrusion detection framework designed to identify GPS spoofing attacks targeting Unmanned Aerial Vehicles (UAVs), with a particular focus on balancing high detection accuracy and real time processing efficiency. The framework integrates a hybrid FS method that combines Gradient Boosting-based Recursive Feature Elimination (GBM-RFE) with Spearman Correlation Analysis (SCA) to reduce redundant or irrelevant features, thus minimizing overfitting and computational overhead. Ensemble classifiers, specifically bagging, boosting, stacking, and soft voting—are applied to enhance robustness and performance across multiple attack scenarios. Among these, the bagging classifier emerges as the most effective, delivering the highest classification accuracy (99.54%), the lowest misdetection rate (0.45%), and a minimal processing time of 0.003 seconds on benchmark GPS spoofing datasets. Bagging achieves this by training multiple base learners on bootstrapped samples and aggregating their predictions through majority voting, thereby reducing variance and improving generalization. Experimental results across two datasets, the GPS spoofing dataset and the UAV location spoofing dataset demonstrate the superior performance of bagging over other ensemble and standalone models, especially under constraints typical of UAV systems, such as limited processing time and class imbalance.

Moreover, the hybrid RFE technique consistently outperforms conventional FS approaches like mutual information gain and standalone correlation analysis. While DL methods are explored for comparative analysis, the paper highlights the suitability of lightweight ensemble methods like bagging for real time UAV cybersecurity due to their low latency and high interpretability.

Kanade et al., [53] presents a comprehensive security framework for NoSQL databases that integrates a data masking mechanism with an ensemble ML technique to enhance protection and validate predictive accuracy. The proposed method combines Fernet cryptography for symmetric encryption with a novel ensemble learning model termed EBC-LR (Ensemble Bagging Classifier with Logistic Regression), designed specifically to secure NoSQL data and classify its integrity. Central to the approach is the use of the bagging ensemble technique, which constructs multiple bootstrap samples of the training data and applies LR as the base classifier. This method reduces variance, improves generalization, and minimizes overfitting, particularly in binary classification scenarios relevant to NoSQL threat detection. Experimental evaluation using a diabetes dataset reveals that the proposed EBC-LR algorithm outperforms individual classifiers and other ensemble variants in all key performance metrics, achieving an accuracy of 85.21%, precision of 81.8%, recall of 85.21%, and F1-score of 80.99%. Notably, EBC-LR demonstrates superior performance compared to traditional methods due to its ability to leverage the explainability of LR and robustness of bagging. Despite its strong results, the framework acknowledges limitations such as computational overhead and challenges in integrating real time dynamic data. Nonetheless, the study establishes a novel and efficient pipeline for securing and classifying NoSQL databases.

Zhang et al., [54] also introduces a bagging based ensemble learning framework for network intrusion detection that effectively addresses key challenges such as data imbalance, feature redundancy, and instability in classifier performance. The proposed model leverages bootstrap aggregating to build diverse base learners by sampling training data with replacement and integrates their outputs via hard voting. The authors employ Extreme Random Trees (ERT) to rank and select informative features, reducing feature space and enhancing generalization. KNN is

identified as the most suitable base estimator among various tested algorithms, and its parameters are fine-tuned using Bayesian Optimization (BO) to enhance detection capability. Using the NSL-KDD dataset, the optimized BO-KNN-Bagging model achieves superior results, reaching an accuracy of 82.48%, F1-score of 82.58%, and an out-of-bag (OOB) error as low as 0.0053. Compared to individual models and other ensemble strategies, this method demonstrates notable improvements in classification stability and robustness, particularly in handling skewed class distributions and maintaining high recall. The paper further emphasizes that the bagging approach reduces variance and mitigates overfitting by enabling base classifiers to learn different patterns from resampled data, while the inclusion of OOB evaluation enhances its real world applicability. Despite slight computational overhead, the model outperforms conventional ML and DL baselines such as SVM, RF, AdaBoost, MLP, and LSTM, making it a practical and effective solution for real time intrusion detection in complex network environments.

Gaikwad et al., [55] proposes an IDS framework that utilizes a bagging ensemble approach with Partial Decision Trees (PART) as base classifiers to improve detection accuracy and reduce false positives in identifying network based threats. The authors address the common challenges associated with single classifiers, particularly their limited generalization ability and sensitivity to data variability, by employing bootstrap aggregating to generate diverse training subsets from the original dataset and train multiple PART models. The predictions from these individual models are then combined using majority voting, which enhances overall robustness and mitigates overfitting. Evaluations were conducted on the KDD Cup 99 dataset. Results demonstrate that the proposed bagging-PART ensemble outperforms standalone DT classifiers, achieving higher detection rates and reduced error margins, particularly for DoS and Probe attacks. The framework was also observed to improve classifier stability across multiple runs and reduce false alarms, which are critical concerns in real world IDS applications. While the method benefits from improved predictive performance and interpretability of rule based classifiers, the computational complexity may increase due to the ensemble structure. The predictive performance is always a priority when computational cost not limited.

2.2.2 Boosting

The Boosting ensemble approach is a widely used ML technique tailored to enhance the predictive performance of weak learners by combining them sequentially into a stronger overall model. In this strategy, each learner is trained in succession, with particular emphasis placed on instances that were misclassified by previous models, allowing the ensemble to progressively correct errors and enhance accuracy [56]. Weak learners, often shallow DT, are typically employed as base models due to their simplicity and ability to capture fundamental patterns. The overall prediction is derived by aggregating the weighted outputs of the models, where better performing models contribute more significantly to the overall decision. Prominent examples of boosting algorithms include AdaBoost, Gradient Boosting Machine (GBM), and Extreme Gradient Boosting (XGBoost). Due to its ability to effectively handle complex learning problems, boosting has been successfully applied in diverse domains, including IDS.

Akinrotimi et al., [57] presents a comparative evaluation of two boosting-based ensemble algorithms (AdaBoost and XGBoost) for enhancing the performance of IDS in the context of network security. Using the NSL-KDD dataset, a well-known benchmark in intrusion detection research, the authors adopt the CRISP-DM methodology to guide data preprocessing, normalization, and FS using the chi-square based K-Best technique. The boosting methods are then applied to a multiclass classification problem encompassing various attack categories including DoS, Probe, U2R, and R2L. AdaBoost and XGBoost are examined based on multiple evaluation metrics, such as accuracy, precision, recall, F1-score, sensitivity, specificity, and error rate. Both algorithms demonstrate high classification capability; however, XGBoost clearly outperforms AdaBoost with a classification accuracy of 99.93%, a precision of 1.0, and a significantly lower error rate of 0.0007. These results are attributed to XGBoost's ability to reduce overfitting through regularization and to better manage high dimensional data, making it more robust for real time detection. AdaBoost, while slightly less accurate, also yields competitive performance, particularly when dealing with moderately imbalanced data. The study underscores the practical value of boosting ensembles in detecting complex and evolving cyber

threats, offering a balance between speed and predictive power. Although the findings are promising, the authors acknowledge limitations such as the use of a single benchmark dataset and the lack of real time implementation testing. Nonetheless, this comparative analysis reinforces the suitability of boosting techniques, especially XGBoost for developing scalable and efficient IDS models that are adaptable to modern cybersecurity demands.

Ismanto et al., [58] investigates the effectiveness of enhanced ensemble learning techniques in detecting cyberattacks on IoT networks, with a strong emphasis on boosting based algorithms. They evaluate four ensemble models such as RF, AdaBoost, Gradient Boosting Machine (GBM), and Extreme Gradient Boosting (XGBoost) on the UNSW-NB15 dataset using both binary and multiclass classification schemes. The boosting based methods, particularly XGBoost, demonstrate superior accuracy, robustness, and generalization capability across both evaluation modes. XGBoost consistently outperforms other models, achieving an accuracy of 98.56% in binary classification and 97.47% in multiclass classification after hyperparameter optimization via Grid Search and Random Search. The strength of boosting lies in its sequential learning mechanism, where each model in the ensemble focuses on correcting the errors made by its predecessors, and XGBoost enhances this further through advanced regularization techniques and parallel computation. In addition, the study highlights how boosting methods effectively handle imbalanced datasets and adapt to complex patterns inherent in IoT traffic. Preprocessing steps such as feature scaling, label encoding, and normalization ensure data consistency, while performance is assessed using standard metrics. The findings validate the practical viability of optimized boosting ensembles in building intelligent, scalable, and accurate IDS tailored for IoT environments.

An in depth evaluation of various ensemble learning techniques for enhancing the detection of DoS attacks in network IDS, with a specific focus on boosting based methods is presented by Thanh et al., [59]. Using the UNSW-NB15 dataset, the authors examine six ensemble techniques, Bagging, AdaBoost, Stacking, Decorate, RF, and Voting, across multiple base classifiers such as DT, NB, LR, SVM, KNN, and Random Trees. Among the tested methods, AdaBoost, representing

the boosting category achieves particularly strong performance when paired with a DT base classifier, yielding an F-measure of 0.9923 and outperforming several other combinations in terms of classification accuracy, precision, recall, and AUC. The study highlights how AdaBoost's focus on challenging instances enables it to deliver high precision and recall even in the presence of class imbalance and complex attack signatures. Despite its superior performance, the boosting approach comes with increased computational demands. Nevertheless, the analysis confirms that AdaBoost remains a highly effective method for strengthening weak learners and producing robust, high-accuracy IDS. The authors also point out that while Stacking and Decorate ensembles yield competitive results, they are less efficient for large scale datasets due to their complexity. Ultimately, the research establishes that boosting ensembles particularly AdaBoost offer a practical and scalable solution for enhancing DoS detection, and recommends future exploration into combining boosting with FS to further improve computational efficiency and model performance in real world network IDS applications.

Al-Sharif and Bushnag [60] introduce a refined intrusion detection architecture for cloud computing, underscored by the strategic deployment of ensemble learning, with a particular inclination toward Boosting methodologies. Given the convoluted, high throughput, and often imbalanced nature of cloud network environments, traditional IDS implementations frequently falter in sustaining precision and responsiveness. The authors interrogate this challenge by leveraging the CICIDS2017 dataset, integrating meticulous preprocessing steps such as feature filtration via Neighbourhood Component Analysis and scaling adjustments to reduce dimensional complexity. Their investigation accentuates an evaluative comparison between ensemble learning frameworks, with special attention to Boosting constructs, notably AdaBoost, LPBoost, and RUSBoost. RUSBoost, an augmentation of conventional Boosting that intertwines undersampling mechanisms, exhibited notable resilience in navigating the prevalent class imbalance dilemma. Empirical findings reveal that this approach attained a superior classification efficacy of 99.82%, surpassing its ensemble counterparts, including RF, especially in delineating infrequent or subtle attack profiles. Despite its merits, the framework's

computational intensiveness, particularly during large-scale deployments, remains a consideration for real-world applicability. Nonetheless, the findings underscore the practicality of Boosting ensembles, specifically those augmented for data imbalance, in fortifying cloud-based IDS, offering a substantive contribution to safeguarding complex networked infrastructures.

Hossain and Islam [61] propose a sophisticated ensemble ML framework tailored for botnet detection, with particular emphasis on enhancing detection accuracy and robustness through hybrid FS and ensemble learning paradigms. Acknowledging the limitations of conventional single classifier models and the dynamic nature of botnet attacks, the authors incorporate a comprehensive methodology that integrates Categorical Analysis, Mutual Information, and Principal Component Analysis to extract the most salient features from high-dimensional network traffic data. A pivotal aspect of their ensemble strategy is the incorporation of Boosting, particularly through the utilization of XGBoost algorithm. XGBoost operates by sequentially training an ensemble of DT, where every following model aims to fix the misclassifications of its predecessors. Empirical validation, conducted using diverse benchmark datasets such as N-BaIoT, Bot-IoT, CTU-13, and CICIDS, demonstrates the superiority of the ensemble approach. Among the techniques explored, XGBoost, as a Boosting method, achieves exceptionally high detection performance, with accuracy, precision, recall, and F1-scores exceeding 99% across multiple datasets. The findings underscore Boosting's effectiveness in handling data imbalance and improving the identification of subtle attack patterns that traditional classifiers often overlook.

Ogobuchi et al. [62] present BoostedEnML, an efficient intrusion detection framework specifically designed to address the challenges of detecting cyberattacks in IoT environments. Recognizing the resource constraints and high data imbalance characteristic of IoT networks, the authors adopt a Boosting oriented ensemble methodology to enhance detection performance while minimizing computational complexity. The proposed approach integrates two leading Boosting algorithms, LightGBM and XGBoost, whose lightweight design and superior learning capabilities make them particularly suited for real time intrusion detection tasks. The

study employs comprehensive data preprocessing steps, including FS via RF importance measures and rigorous data balancing using SMOTE and ADASYN techniques, to mitigate the inherent class imbalance present in widely used benchmark datasets such as CICIDS2017 and CSE-CICIDS2018. Through stratified K-fold cross-validation, the model ensures robustness and generalization. A notable contribution of this work lies in its comparative evaluation of multiple ensemble learning strategies, including Bagging, Stacking, and Boosting based configurations. Empirical results consistently demonstrate the superior performance of the Boosting driven BoostedEnML model, achieving 100% accuracy, precision, recall, and F1-score across both datasets for multiclass classification tasks. The authors attribute this to the iterative, error correcting nature of Boosting, which effectively amplifies the learning focus on rare and difficult instances, thereby reducing false positives and false negatives.

Bhati et al. [63] propose an improved ensemble intrusion detection framework that takes advantage the strength of XGBoost to address the escalating complexity of cyberattacks in network environments. The study highlights the limitations of conventional IDS, particularly in detecting unknown and evolving attacks, and presents Boosting as a viable solution to enhance detection accuracy and robustness. Their proposed approach employs XGBoost, which applies a sequential learning strategy wherein each subsequent model focuses on minimizing the errors of its predecessors. This iterative refinement enables the model to achieve an improved bias variance trade-off, making it particularly effective for complex intrusion detection scenarios. The framework is evaluated using the KDDCup99 dataset. Extensive experiments demonstrate that the XGBoost based IDS significantly outperforms traditional methods, including AdaBoost, another Boosting technique. Specifically, the proposed system achieves a detection accuracy of 99.95%, notably higher than the 91.51% accuracy attained by the AdaBoost based counterpart. The authors attribute this performance gain to XGBoost's ability to efficiently handle feature weighting, perform regularization, and construct optimized DT that enhance generalization while mitigating overfitting. In conclusion, the stacking ensemble method outperforms the conventional methods.

2.2.3 Stacking

Stacking or stacked generalization, is an advanced ensemble learning technique that integrates multiple classification or regression models to enhance predictive performance. Unlike Bagging or Boosting, where identical or similar models are combined through parallel or sequential strategies, Stacking employs a diverse set of base learners, often of different algorithmic types, to capture varied data patterns. The predictions generated by these base models serve as input features for a higher level model, known as a meta learner, which synthesizes these outputs to produce the final prediction. This hierarchical structure enables stacking to leverage the complementary strengths of different models, reducing both bias and variance in the overall prediction. By facilitating the combination of heterogeneous learners, stacking is particularly effective in complex, real world tasks where no single model is sufficient to capture the underlying data distribution [64].

An IDS that utilizes ensemble ML to enhance detection performance against modern cyber threats is proposed by combining multiple classifiers [65]. The authors address the limitations of traditional single classifier based IDS approaches by introducing an ensemble method that adapts to the dynamic nature of cyber threats. Pansare et al., [66] also proposed an ensemble learning approach for IDS, combining RF, Gradient Boosting and Multilayer Perceptron (MLP). The authors prioritized high recall and low false negatives which are achieved using the ensemble method. Through this, they provide a robust and generalizable solution for improving IDS performance and mitigating cyber threats, emphasizing the importance of detecting even a single missed malicious packet. The proposed model achieved overall accuracy of 0.9998, demonstrating excellent precision and recall. A stacking ensemble method for IDS in IoT systems is proposed by Lala et al., [67], utilizing the NSL-KDD dataset. The proposed ensemble method combines multiple ML techniques, specifically KNN, DT, RF and LR. This ensemble approach aims to enhance the accuracy and performance metrics of IDS compared to traditional ML and DL models. The contribution of the paper lies in demonstrating that the ensemble model achieves an accuracy of 98.59, along with superior scores in metrics such as Matthews Correlation Coefficient (MCC) and Cohen's Kappa Score. The

paper suggests that further performance improvements can be achieved by incorporating advanced algorithms and techniques like K-fold cross-validation.

Thokchom et al. [68] introduce an intrusion detection framework based on ensemble learning, designed to outperform conventional single classifier models. The ensemble is constructed by combining lightweight ML algorithms, namely NB, LR, and DT, serving as base learners, while Stochastic Gradient Descent functions as the meta classifier to aggregate their predictions. To ensure optimal model performance, the Chi-square statistical method is employed for FS, isolating the most significant attributes from the dataset. The proposed model is rigorously assessed on three widely used benchmark datasets: KDD Cup 1999, UNSW-NB15, and CIC-IDS2017. The evaluation covers both binary and multiclass classification tasks, demonstrating the model's flexibility across various intrusion detection scenarios. Experimental findings reveal that the ensemble method offers significant improvements in detection performance, particularly in addressing class imbalance, where the consequences of incorrect classification are substantial. Ammar et al, [69] presents an IDS that utilizes an ensemble based approach, specifically stacking ensemble learning, which combines multiple ML algorithms to enhance detection performance. The ensemble method incorporates three base models such as RF, DT, and KNN, along with a meta model represented by the LR algorithm. The proposed system employs preprocessing techniques to improve classification efficiency and integrates a total of seven ML algorithms. The dataset used for evaluation is the UNSW-NB15 dataset, which is commonly utilized for testing IDS. Through the proposed model, the authors achieves an accuracy of 96.16% during the training phase and 97.95% during the testing phase, with precision rates of 97.78% and 98.40% for training and testing, respectively, demonstrating significant improvements in various measurement criteria.

Urmi et al. [70] present a stacked ensemble based intrusion detection framework aimed at enhancing cyberattack detection performance through efficient FS and classifier integration. The study investigates the impact of three prominent FS techniques such as RFE, Mutual Information (MI), and Lasso on the classification effectiveness of Cyber-Attack Detection Systems (CADS). They propose a

multilayered ensemble model that integrates RF, XGBoost, and Extra Trees as base learners, with LR serving as the meta-learner. The approach is evaluated on two benchmark datasets, CICIDS2017 and NSL-KDD, addressing both binary and multiclass classification scenarios. Results demonstrate that the combination of FS and stacking significantly improves detection accuracy, with RFE contributing to superior performance, achieving up to 99.95% accuracy on NSL-KDD and 99.90% on CICIDS2017. The research highlights the efficacy of ensemble learning, particularly stacking, in mitigating overfitting and enhancing robustness against evolving cyber threats, making the approach suitable for real world practices.

Ghazi and Raghava [71] propose an advanced intrusion detection approach for cloud environments by integrating stacked ensemble learning with big data technologies. Recognizing the limitations of single classifier models in managing the vast and complex cloud network data, the authors introduce a scalable solution that combines diverse ML classifiers with FS techniques. Their framework employs six heterogeneous classifiers viz. J48, RF, XGBoost, OneR, Multi-Layer Perceptron, and SVM, while leveraging a ranking based selection strategy to identify the most effective subset for stacking. The use of Apache Spark and HDFS enables the system to efficiently process large scale intrusion datasets, specifically NSL-KDD and UNSW-NB15. Through extensive experimentation, the proposed stacked ensemble model, with XGBoost as the meta classifier, demonstrates superior detection performance compared to individual classifiers and other ensemble methods. FS based on Gain Ratio effectively reduces dimensionality, enhancing both model simplicity and accuracy. This research highlights the potential of combining stacking with big data platforms to address the scalability and performance challenges of modern intrusion detection in cloud environments.

Oriola [72] proposes an enhanced IDS utilizing a 2-tier stacked generalization ensemble approach to improve classification accuracy in network security. Acknowledging the limitations of traditional ML and ensemble models in addressing sophisticated cyberattacks, the study introduces a refined stacking framework that combines both classical stacking and multi-feature based stacking, with ANN serving as meta-learners at both levels. The approach leverages a FS process using Chi-

square testing to identify the most influential attributes from the NSL-KDD dataset, ensuring model efficiency and reducing complexity. Four base classifiers viz. SVM, LR, NB, and MLP are employed to generate metafeatures, which are subsequently optimized through GridSearch for the second tier stacking. Experimental evaluation demonstrates that the proposed method surpasses conventional stacking, bagging, voting ensembles, and DL models in detection accuracy, precision, recall, and F-score, achieving 97% accuracy and 98% in other metrics. The study underscores the potential of optimized stacking ensembles in building effective and computationally efficient IDS for real world network environments.

Rajadurai and Gandhi [73] present a stacked ensemble learning framework to enhance intrusion detection performance in wireless networks. The study addresses the limitations of single classifier models, which often exhibit poor generalization and struggle with complex network attack patterns. The proposed system integrates RF and Gradient Boosting as base learners, whose outputs are combined and passed to a meta classifier for final prediction. This hierarchical learning structure aims to leverage the complementary strengths of the base models while mitigating individual weaknesses. The approach is evaluated using the NSL-KDD dataset, incorporating diverse attack categories such as DoS, Probe, R2L, and U2R. Experimental results demonstrate that the stacked ensemble method achieves superior detection rates, recall, and overall accuracy compared to conventional ML models and neural networks. The study highlights the effectiveness of combining bagging and boosting techniques within a stacking framework for robust intrusion detection, offering improved resilience against both known and unknown network threats.

2.3 Resampling methods for IDS

Most intrusion detection datasets suffer from significant class imbalance. This imbalance can often lead to overfitting, making it essential to address to improve the quality of data fed into the ML model. Resampling refers to a collection of statistical and ML techniques where new samples are generated from an existing dataset to improve model training, evaluation, or to address data distribution challenges. It involves creating modified versions of the original dataset by either systematically drawing samples with or without replacement. In the context of

classification tasks, particularly with imbalanced datasets, resampling is widely adopted to mitigate the dominance of majority classes and enhance the model's ability to recognize minority class instances [74]. Common resampling strategies include oversampling, where additional instances of the minority class are synthetically generated, and undersampling, where instances from the majority class are reduced.

2.3.1 Undersampling

There are various techniques for undersampling, and there are numerous researches based on each technique. Some of the researches are discussed here. Zuech et al. [75] conducted a comprehensive investigation into the impact of Random Undersampling (RU) as a strategy to alleviate extreme class imbalance in web attack detection tasks. Using the CSE-CIC-IDS2018 dataset, they simulated highly skewed data distributions by combining ten days of normal traffic with web attack instances, producing an imbalance ratio as severe as 14,429:1. The study systematically applied RUS at varying degrees, exploring eight different majority-to-minority class ratios, such as 1:1, 3:1, and 99:1, to assess how selective reduction of majority class samples influences detection performance. A suite of classifiers, including ensemble models like RF, XGBoost, CatBoost, and LightGBM, alongside traditional learners such as NB and LR, were employed for evaluation. The findings revealed that appropriate undersampling significantly enhanced classification metrics, with ensemble based models, particularly LightGBM, demonstrating superior robustness under optimal RU configurations. This work underscores the critical role of well calibrated undersampling in conjunction with ensemble learning for improving detection rates in highly imbalanced intrusion detection environments. Rafrastara et al. [76] also proposed a malware detection approach that addresses the class imbalance problem using RU in conjunction with the RF classifier. The study utilized a real world dataset derived from the UCI Machine Learning Repository, comprising goodware and malware instances with an initial imbalance ratio of approximately 1:9.5. To alleviate this imbalance, the majority class was selectively reduced through RU, resulting in a balanced dataset with equal representation of both classes. Following this, RF, known for its ensemble structure and robustness, was

applied to the balanced data for classification. The model's performance was compared against KNN, NB, and LR classifiers using accuracy, recall, and specificity as evaluation metrics. The experimental results demonstrated that RF, when combined with RU, achieved superior performance, recording 98.3% across all three metrics. These findings affirm the effectiveness of RU as a straightforward yet impactful technique for enhancing classification reliability in imbalanced malware detection tasks.

Hartono et al. [77] explored the use of Tomek Links as a standalone undersampling strategy to address the challenges of class imbalance in binary classification tasks. Their study focused on the highly imbalanced Kaggle credit card fraud dataset, where fraudulent transactions represent a minority of just 0.17% of the total instances. Tomek Links is utilized exclusively to eliminate borderline majority class samples that contribute to class overlap. The cleaned dataset was subsequently used to train a Backpropagation Neural Network (BPNN) model. The experimental results demonstrated notable performance improvements following undersampling, with the model's accuracy increasing from 74.68% to 80.42%, accompanied by a reduction in mean squared error and an enhancement in F1-score. These findings affirm that Tomek Links alone can effectively refine decision boundaries and mitigate the negative effects of class imbalance without introducing synthetic samples. The study provides valuable insight into the role of simple yet effective undersampling techniques in improving classifier performance, particularly in domains where overgeneration of minority class samples may not be desirable.

A.More [78] presents a comprehensive survey of resampling methodologies designed to mitigate the challenges associated with imbalanced datasets in classification tasks. The study systematically categorizes existing techniques into oversampling, undersampling, hybrid approaches, and algorithm level modifications. A notable emphasis is placed on undersampling strategies, particularly those that selectively prune the majority class to achieve a more balanced representation without excessively compromising informative data. Among these, the Edited Nearest Neighbours (ENN) method is highlighted as a boundary refinement technique that effectively reduces class overlap. ENN operates by eliminating

majority class instances whose labels disagree with the majority of their nearest neighbours, thereby improving class separability and reducing noise around decision boundaries. Unlike purely RU, ENN preserves the structural integrity of the dataset by targeting ambiguous or mislabeled samples rather than indiscriminately discarding majority class instances. A.More underscores ENN's role in enhancing model generalization, especially when dealing with complex, real world imbalanced data where minority classes are sparse and critical to accurate classification. The survey concludes by acknowledging that, although ENN reduces majority class size and boundary ambiguity, its performance may depend on the choice of nearest neighbour parameters and the underlying data distribution, suggesting the need for careful tuning in practical applications. Vuttipittayamongkol and Elyan [79] explored advanced undersampling techniques to address the persistent challenges posed by class imbalance and overlapping instances in binary classification tasks. Their work introduced a neighbourhood driven strategy that extends the principles of the Neighbourhood Cleaning Rule (NCR) to selectively prune majority class samples contributing to class ambiguity. The method employs KNN to detect majority instances that lie within regions of class overlap or are misaligned with their surrounding neighbourhood, thus refining the decision boundary without excessive loss of informative data. Unlike RU, which indiscriminately discards data, NCR based undersampling prioritizes preserving the structural characteristics of the dataset while mitigating the influence of noisy or misleading samples. Through extensive experimentation on both synthetic and real world datasets, the study demonstrated that NCR based undersampling substantially improved classifier sensitivity and overall predictive performance.

Aziz and Ahmad [80] introduced a cluster-centroid based undersampling approach to mitigate class imbalance in IDS. Their method employs k-means clustering to partition the majority class, after which representative samples from cluster centroids are retained while peripheral or sparse data points are removed. This strategy reduces the volume of majority class instances while preserving their structural diversity, thereby maintaining critical decision boundary information. The undersampled dataset, combined with all minority class samples, is then used to train

various ML models, including NB, KNN, SVM, MLP, and RF, evaluated on NSL-KDD and UNSW-NB15 datasets. The results indicate that cluster-centroid undersampling enhances detection rates and accuracy across models, particularly when dealing with high imbalance ratios. However, the approach exhibits certain limitations, such as sensitivity to the initial cluster count (k) and potential removal of informative samples if clusters are not well formed. Consequently, its effectiveness relies on appropriate parameter tuning and dataset specific clustering behaviour. Silva et al. [81] conducted a detailed comparative study of undersampling methods for mitigating class imbalance in the context of network NIDS development. The research utilized the NSL-KDD dataset to evaluate the performance of various undersampling strategies, including RU, Tomek Links, Cluster Centroid, and the distance based NearMiss technique. Among these, NearMiss demonstrated notable effectiveness by selectively retaining majority class samples that are located near minority class instances, thereby reinforcing the representation of critical boundary regions. This targeted reduction of majority data helped enhance minority class detection, particularly in highly imbalanced scenarios where conventional models tend to overlook rare attack instances. The study highlighted that NearMiss, when applied appropriately, contributes to improved classification balance and increased sensitivity towards minority attacks. However, the authors also acknowledged that excessive application of NearMiss may lead to over reduction of majority class samples, potentially discarding informative data and compromising overall model robustness. Consequently, the effectiveness of NearMiss is closely tied to careful parameter selection and dataset characteristics, underscoring the importance of tuning undersampling techniques in practical IDS deployment.

2.3.2 Oversampling

There are multiple techniques for oversampling, which creates synthetic data based on requirements. Some of the recent researches for oversampling are discussed here. Bagui and Li [82] conducted a comprehensive study on the impact of various resampling strategies to address the significant class imbalance problem in network intrusion detection datasets. Their work focused on six widely used cybersecurity datasets, including KDD99, NSL-KDD, UNSW-NB15, CICIDS2017, CICIDS2018,

and BoT-IoT, all of which exhibit highly skewed class distributions, with minority classes representing rare but critical attack types. The study evaluated several resampling methods, namely RU, RO, and hybrid approaches such as RU-RO, RU-SMOTE, and RU-ADASYN. Among these, RO was applied as a straightforward technique to duplicate minority class instances to achieve class balance. The resampled data were subsequently used to train ANN classifiers implemented within a Spark MLlib environment to accommodate the scale of the datasets. The results indicated that while RO effectively increased detection rates for minority classes, it also introduced risks of overfitting due to repetitive data. In contrast, more sophisticated methods such as SMOTE and ADASYN demonstrated improved generalization by generating synthetic samples rather than simple duplication. The authors emphasized that while RO offers a computationally inexpensive solution, it may not be suitable for complex intrusion scenarios where class boundaries are not well defined. Additionally, the study highlights the importance of selecting resampling methods based on dataset characteristics and application requirements, as no single technique universally guarantees optimal performance across all IDS contexts.

Khedkar et al. [83] developed an intrusion detection framework that integrates spatial and temporal feature learning with oversampling to address class imbalance in network traffic data. The authors employ SMOTE as the only oversampling strategy to generate synthetic minority class samples, with the goal of enhancing the identification of minority class attack types without altering the majority class. The proposed system combines a CNN for extracting spatial characteristics and a bidirectional long short term memory (Bi-LSTM) model for capturing temporal patterns within network traffic. The methodology was evaluated on benchmark datasets, including CIC-IDS2017, NSL-KDD, and UNSW-NB15, all of which exhibit highly skewed class distributions. The results demonstrate that the application of SMOTE enhances the model's ability to detect rare attack instances, contributing to improved accuracy, precision, and recall across minority classes. However, the study also acknowledges inherent drawbacks associated with SMOTE. The synthetic samples are generated through interpolation, which may not fully

capture the complex variations present in real world intrusion patterns, potentially limiting the model's generalization capability in highly dynamic environments. Moreover, excessive reliance on SMOTE can lead to the introduction of noisy or redundant synthetic instances, affecting overall model robustness. Despite these challenges, the research highlights the practical benefits of applying SMOTE alone to address imbalance, particularly when combined with deep learning architectures capable of capturing intricate data representations.

Liu et al. [84] introduce a network IDS which focuses on countering data imbalance using ADASYN oversampling, paired with a LightGBM classifier to accelerate learning without sacrificing accuracy. The methodology involves standardizing and encoding features, applying ADASYN to generate synthetic samples in regions of sparsity, then training LightGBM on the rebalanced NSL-KDD, UNSW-NB15, and CICIDS2017 datasets. The combined pipeline significantly improved detection rates for minority attacks, achieving accuracy scores of 92.57%, 89.56%, and 99.91%, while reducing runtime overhead. Despite its strengths, the approach exhibits limitations such as the reliance on interpolation, which may produce synthetic samples that oversimplify intricate attack patterns, potentially leading to suboptimal generalization. Additionally, ADASYN can amplify noise when majority class samples are erroneously treated as minority neighbours, reducing effectiveness in highly complex or noisy data environments.

Yang and Cha [85] introduced GMOTE, a Gaussian based minority oversampling method designed to address the drawbacks of traditional interpolation based techniques in imbalanced classification tasks. Unlike methods such as SMOTE, which generate synthetic samples along linear paths between minority instances, GMOTE leverages Gaussian Mixture Models to learn the underlying distribution of the minority class. By incorporating Mahalanobis distance as a measure of local density and outlier presence, the approach selectively generates new synthetic instances in regions of the feature space where minority samples are sparse, while avoiding excessive oversampling in dense or noisy regions. Through experiments on multiple benchmark datasets, the study demonstrates that GMOTE outperforms conventional oversampling techniques in improving both classification

accuracy and F1-score. However, the method is not without limitations. Its reliance on Gaussian assumptions may restrict its ability to model complex, non-Gaussian minority class distributions effectively. Moreover, the computational cost associated with fitting Gaussian Mixture Models and calculating Mahalanobis distances may limit scalability, particularly for large, high dimensional datasets. Despite these considerations, the work presents a meaningful advancement in oversampling strategies, offering a statistically grounded alternative for improving minority class representation.

2.3.3 Hybrid Sampling

Hybrid sampling technique employs both the oversampling and undersampling technique and is useful especially when the dataset is highly imbalanced. Abdelkhalek and Mashaly [86] proposed a resampling method for IDS that combines Adaptive Synthetic (ADASYN) oversampling and Tomek Links undersampling techniques to address class imbalance, enhancing detection rates for minority classes while maintaining high accuracy in the NSL-KDD dataset. In order to balance the dataset, classes with less instances are oversampled with ADASYN and classes with overwhelm instances are undersampled with Tomek Links. The proposed resampling techniques increase the detection rate of DL algorithms like CNN, MLP, DNN, etc. Chen et al. [87] employs RU and SMOTE oversampling to address imbalanced datasets in intrusion detection viz. NSL-KDD dataset and the CIC-IDS2017 dataset. This combination of resampling enhances the identification of minority attack samples while maintaining overall detection accuracy, crucial for effective IDS. The main objective of the paper is to reduce false positives and false negatives of the classification which is performed using a CNN model. After applying the resampling method, significant increase in the classification results has been achieved when tested on different kinds of attacks.

The hybrid sampling method for the IDS that utilizing the ADRDB algorithm is proposed in 2022 [88]. ADRDB is a combination of ADASYN and Repeated Edited Nearest Neighbour (RENN) to balance the dataset. Addressing class imbalance, and enhancing feature extraction for improved accuracy in detecting

intrusions. Three datasets viz. NSL-KDD, UNSW-NB15 and CIC-IDS2017 dataset were used for evaluation. The authors compare the performance of their proposed ADRDB with different sampling methods like RO, RU, SMOTE, ADASYN, RENN, etc. and proves that their proposed method outperforms all other methods in all the three tested datasets. Aldallal in 2022 [90] also utilise SMOTE oversampling and RU to balance the CSE-CIC-IDS 2018 dataset since the dataset has 95% normal class and 5% attack class. They perform the resampling to avoid bias in the training dataset. They proposed a current unit Long Term Short Memory (LTSM) gated recurrent unit to improve the efficiency in detecting abnormal network traffic in IDS. Different performance metrics were used to evaluate the results and the proposed model significantly outperforms state of art model by up to 12%. MLIDS22 model [90] is proposed to address the class imbalance issue through hybrid sampling techniques that combine ADASYN and RU. The proposed model is a hybrid of CNN and LTSM. They evaluate the proposed model on CIC-IDS 2017 dataset, CSE-CIC-IDS 2017 dataset, and a mixture of CIC-IDS 2017 dataset and CSE-CIC-IDS 2018 dataset. By effectively addressing the class imbalance, they reduce overfitting risks and achieve high accuracy and AUC in both intra- and inter-dataset evaluations. The hybrid sampling also significantly improved the generalization ability of the model when evaluated across inter-dataset scenarios.

2.4 Feature Selection for IDS

FS is the process of identifying and selecting a subset of the most relevant features from a larger set of available data that contribute the most to a predictive model's performance. FS plays an important role like reducing computational complexity, enhance model's generalization, improves interpretability and most importantly improves detection accuracy. FS can be divided into 3 methods:

- Filter methods: It use statistical techniques (e.g., correlation, mutual information) to rank the features. Features with high rank will be selected.
- Wrapper methods: Evaluate subsets of features using a predictive model (e.g., Recursive Feature Elimination). This method gives the feature subset with highest predictive value.

- Embedded methods: Feature selection is integrated into the model itself (e.g., LASSO, tree-based feature importance). Thus, typically more efficient than Filter and Wrapper method. Especially for large datasets.

2.4.1 Filter-based method

Damtew et al. [91] proposes a Heterogeneous Ensemble Feature Selection (HEFS) method using five filter methods: Probabilistic Significance, Symmetrical Uncertainty, Gain Ratio, Classifier Attribute Evaluator, and ReliefF, to enhance FS and improve prediction performance in IDS. The proposed method is evaluated on NSL-KDD dataset which has 41 features. The proposed HEFS approach demonstrates superior predictive capability when compared to individual FS techniques and alternative frameworks. Performance evaluation metrics, such as accuracy and detection rate, indicate that the method enhances the effectiveness of the network IDS, outperforming other existing approaches. Siddiqi and Pak [92] introduces a new filter-based FS process for IDS, emphasizing the importance of normalization before FS. It evaluates five transformations and demonstrates improved efficiency and accuracy in identifying relevant features compared to existing methods. The proposed process flow can locate a more relevant set of features with high efficiency and accuracy. The dataset used for their experiments are ISCX-2012 and CIC-IDS 2017 dataset, and improvements is shown in the proposed method in accuracy, precision, recall, and F1 Score. The experimentation highlights that filter based FS can be further improved using statistical methods. A comprehensive survey of filter based FS techniques for cyberattacks is presented, and the authors also discuss the search algorithms and relevance measures commonly used in the filter based technique [93]. The paper focuses on the filter based FS technique. It provides an overview and classification of existing FS techniques. Selecting appropriate FS techniques is challenging and this comprehensive study on filter-based FS method give status of FS method from different angle and different intrusion detection dataset.

2.4.2. Wrapper-based method

Almaghthawi et al. [94] investigated the usage of multiple wrapper FS methods in intrusion detection, such as GA, sequential forward selection (SFS), and sequential backward selection (SBS). The efficiency of the three wrapper based FS techniques is tested with SVM and MLP classification using the CIC-IDS2017, CSE-CIC-IDS218, and NSL-KDD datasets. Different number of feature subset (selected features) are considered for each dataset using k-fold cross validation. For NSL-KDD, CIC-IDS 2017, and CSE-CIC-IDS 2018 dataset, 29, 35 and 47 features are selected using GA for all the dataset. With SVM, MLP and MLP as the best classifier respectively for each dataset. The experimental results were compared with similar state of the art techniques and the proposed method proves to be superior especially in detection accuracy and FPR. Umar et al. [95] proposed a hybrid IDS model which include wrapper based FS as part of the process in building IDS. The wrapper based FS use DT as the feature evaluator. The UNSW-NB15 dataset is used for evaluation. The features have been reduced from 42 features to 19 features. ML algorithms like ANN, SVM, KNN, RF and NB are compared using accuracy, detection rate and FPR with RF achieving highest performances before and after FS. However, when compared with other state of art work, the proposed model is not performing well especially in accuracy and FAR. Fatima and Ali [96] also proposed an effective FS approach for multiclass intrusion detection using a wrapper based GA. Their method focuses on selecting optimal subsets of features to reduce dataset dimensionality and enhance the performance of IDS. GA was combined with NB as the fitness function to evaluate feature subsets, and ensemble classifiers (stacking and bagging) were used for classification. The approach was evaluated on UNSW-NB15 and CICDDoS2019 datasets. The GA-based FS significantly improved detection accuracy, reduced FAR, and yielded lightweight datasets suitable for real-time IDS applications. On the UNSW-NB15 dataset, the stacking classifier achieved an accuracy of 85.6% and a FAR of 13.6%, while on CICDDoS2019, it reached 99.7% accuracy with a FAR of just 0.18%. The study demonstrates that integrating GA-based FS with ensemble methods enhances the overall effectiveness and efficiency of IDS, outperforming several existing approaches.

2.4.3 Embedded method

Mokbal et al. [97] introduces a robust IDS that leverages embedded FS combined with XGBoost based ensemble learning for high performance network attack detection using the CICIDS2017 dataset. This method operates in two key stages: First, it employs tree based feature importance scores derived from gradient boosting trees to assess the relevance of each feature. Importance is computed based on the frequency and quality (gain) of a feature being used in tree splits. This step reduces the impact of noise and lowers computational complexity. Second, these scores are used in a model based selection process, where subsets of features are evaluated iteratively against the model's performance to find the optimal set that maintains high detection accuracy while minimizing redundancy and overfitting. This approach ensures a uniform and attack-agnostic selection of features, meaning that the selected subset effectively represents all types of attacks rather than optimizing for individual ones. Such a strategy enables the IDS to generalize better across various threat types. The framework was evaluated on both binary and multi-class classification tasks, achieving exceptional results: up to 99.86% accuracy in binary classification and 99.90% in multi-class classification, with extremely low FPR and FNR. Zhang et al. [98] proposes an ensemble based FS framework to enhance intrusion detection performance across large scale cybersecurity datasets (UNSW-NB15, CIC-IDS2017, and CSE-CIC-IDS2018). The study focuses on embedded FS, integrating it with filter methods to design an automatic FS technique called EAFS (Ensemble Automatic Feature Selection). The authors employ five individual FS strategies; Pearson Correlation Coefficient, Mutual Information, mRMR, Extra Trees (ET), and XGBoost. Embedded methods like ET and XGBoost are particularly emphasized due to their ability to evaluate features using internal model mechanisms, such as split importance scores and gain values. EAFS dynamically selects features without requiring fixed thresholds. It ranks features by importance, progressively builds feature subsets, and evaluates them using a newly designed metric called NSOM (Normalized Score of Mixed). NSOM balances classification accuracy, the number of features, and training time, thus optimizing both model performance and efficiency. 30% of original features are selected for

UNSW-NB15 and CIC-IDS2017 dataset and 10% of original features for CSE-CIC-IDS 2018. Experimental results show good performance in comparison to other similar works.

As part of efforts to enhance intrusion detection in IoT environments, Liu et al. [99] proposed a hybrid IDS framework that effectively combines embedded FS techniques with DL models for efficient attack detection and classification. Their work introduces two variants, RCNN and XCNN that integrate traditional ensemble learning methods, namely RF and XGBoost, with a CNN classifier. The key innovation lies in the use of embedded FS, wherein RF and XGBoost are leveraged to identify the most significant input features during model training. In their experiments on the CCD-INID-V1 dataset, an IoT-specific dataset developed by the authors. RF reduced the feature set from 83 to 41, while XGBoost further minimized it to just 7 features, demonstrating the ability to manage high dimensional IoT data effectively. After feature reduction, the refined data is processed through a CNN, enabling robust binary and multiclass intrusion detection. Evaluations across multiple datasets (CCD-INID-V1, BaIoT, and DoH20 dataset) show that this hybrid approach maintains high detection performance, with AUC values reaching 0.999. Notably, the computational efficiency of the models improved significantly compared to traditional algorithms like KNN, achieving over 85% runtime reduction which is particularly advantageous for deployment in resource constrained IoT environments. This study highlights the potential of embedded FS in reducing model complexity and enhancing detection capabilities, making it a valuable direction for developing lightweight and scalable IDS solutions in modern IoT networks.

Recent researches on FS for ID is highlighted on Table 2.1. This highlight includes different FS method using different intrusion detection dataset, results and remarks of the researches.

Table 2.1 Recent researches on Feature Selection for IDS

Dataset and references	Feature Selection method	Results	Remarks/Limitations
NSL-KDD [91]	HEFS (ensemble of five filter method)	Features reduced from 41 features to 10 features (using HEFS)	HEFS improves prediction performance in network IDS. Merit-based evaluation reduces redundancy in FS.
ISCX-2012 and CIC-IDS 2017 [92]	Optimized process flow for filter-based FS	From 82-37 features for ISCX-2012 dataset and 79 to 35 features for CIC-IDS2017	Normalization before FS enhances feature relevance significantly.
NSL-KDD, CIC-IDS 2017 and CSE-CIC-IDS 2018 [94]	GA, SBS and SFS	GA proves to have the best outcomes if detection accuracy is priority though it may not always give the least number of features	The processing time of wrapper method are usually high and high FPR needs to be address.
UNSW-NB15 [95]	Wrapper based FS with DT as feature evaluator	Features reduced from 42 to 19 features. Among compared ML algorithms, RF achieves highest accuracy.	The proposed method has extremely high FAR which needs to be addressed.
CIC-IDS 2017 [97]	Embedded FS with XGBoost	Achieved accuracy of 99.86% with 50 features. Outperform existing methods on multiple metrics.	The number of selected features is still very large in comparison to other FS method.
UNSW-NB15, CIC-IDS2017 and CSE-CIC-IDS2018 [98]	Ensemble Automatic FS	The method outperforms other recent methods. The experimental results show the effectiveness of the method.	Fixed thresholds require a priori knowledge for feature selection.

2.5 IDS in Software Defined Networking

Software Defined Networking (SDN) is a modern networking approach that decouples the control plane from the data plane, enabling centralized management through a programmable controller. Unlike traditional networks where each device independently manages forwarding and control logic, SDN centralizes decision making, allowing dynamic configuration and real time policy enforcement across the entire network. The main distinction lies in architecture; traditional networks are device centric with static configurations, whereas SDN provides a centralized, programmable, and flexible network model. This separation enhances scalability, simplifies network management, and allows rapid adaptation to changing conditions.

In SDN environments, IDS plays an important role in securing the network. Leveraging the centralized controller, IDS in SDN benefits from global visibility and can detect and respond to threats more effectively than in traditional networks. It enables real time traffic monitoring, dynamic rule updates, and rapid mitigation of anomalies or attacks. IDS can be rule based or anomaly based and can be integrated with the SDN controller to automate security responses. Combining SDN with IDS enhances network resilience, making it well suited for dynamic, large scale infrastructures like IoT, cloud, and data centre networks. In this section, we will look into recent researches of IDS in SDN.

Zhao et al. [100] try to enhance IDS efficiency in SDN by optimizing packet sampling using GA. The IDS model is deployed on the SDN control plane. It uses GA to dynamically determine the optimal sampling probability for each switch to balance detection accuracy and processing overhead. The model improves detection rates and reduces network overhead. Simulations confirm better efficiency than traditional fixed rate sampling techniques. The proposed method enhanced detection accuracy while minimizing resource usage through optimal sampling strategies. Their objective is to address class imbalance and enhance multiclass classification performance in SDN based IDS. 92. Chouikik et al. [101] explore DDoS attacks in SDN environments and evaluate the role of IDS in mitigation. They implement IDS in a Mininet and OpenDaylight based SDN testbed, focusing on throughput and

delay metrics. The proposed improves network stability and mitigates the impact of DDoS attacks. During the DDoS attack, the presence of IDS reduces throughput fluctuations and enhances resilience under attack, on the other hand when the IDS is absent, there is a significant increase in oscillations, which indicate the success of the proposed IDS which stabilize the network during DDoS attack. With the objective to develop a scalable IDS for IoT systems integrated with SDN, using ML methods.

Alshammari and Alserhani [102] proposed and implements five ML algorithms (KNN, SVM, LR, DT, RF) within SDN controllers using the ToN-IoT dataset. With the integration of different ML algorithms in the SDN, RF outperformed other ML techniques in detecting IoT threats and demonstrated high detection rates and robustness for diverse IoT traffic patterns. Radhi and Mohammed [103] proposed a DL based network IDS on the controller side of SDN using binary classification. They use multiple DL model such as CNN, DNN, RNN, LSTM, and Gated Recurrent Unit (GRU) are trained on NSL-KDD with only 12 features selected from 41 total features using the FS method. Among all the tested DL model, CNN achieved the highest accuracy and other evaluation metrics. DL models were generally effective. The proposed DL models were compared with other state of art method and the proposed method achieve success in identifying attacks with great efficiency.

Abdallah et al. [104] develops a hybrid IDS by combining CNN and LTSM, to improve zero-day attack detection in SDN using DL methods. The proposed model is capable of capturing the spatial and temporal features of the network traffic. L1 and L2 regularization techniques are used to address overfitting problem in the InSDN dataset. The proposed method outperforms standalone CNN or LSTM models by achieving 96.32% accuracy. Zero-day attack are detected using the proposed method and proves robust performance across all metrics. Latah and Toker [105] proposed a multilevel hybrid classification system based on flow statistics. KNN is employed in the first level, followed by Extreme Learning Machine (ELM) and Hierarchy ELM for the remaining levels. This is developed in order to create a high accuracy IDS for SDN using a multilevel approach. NSL-KDD dataset is used for evaluation and an accuracy of 84.29% is achieved which is higher than traditional

ML models. With the experiments, it is found that the proposed model is efficient and scalable, and is ideal for SDN environments. Abbas et al. [106] also introduced an innovative anomaly based IDS that leverages SDN's capability to extract statistical flow level features from network traffic. They make use of ensemble ML techniques to improve SDN IDS performance. They use multiple classifiers (voting system) on NSL-KDD and KDDCup99 datasets, integrated into the SDN controller.

Table 2.2 Recent researches on IDS for Software Defined Networking

Dataset and references	**Contribution**	**Methods**	**Remarks**
Custom simulated traffic [100]	GA-based sampling for IDS in SDN	Genetic Algorithm, Traffic Sampling, and Custom simulation	Improved sampling efficiency and accuracy by utilizing the control plane for installation of IDS.
Mininet and OpenDaylight (simulated) [101]	Mitigation of DDoS attacks using IDS in SDN	Throughput analysis	IDS stabilize performance during DDoS attacks
ToN-IoT [102]	ML-based IDS for SDN-IoT environments	Integration of ML IDS in SDN. ML algorithms like RF, KNN, DT, LR and SVM are considered	Random Forest outperformed other ML algorithms and proves to be most scalable and robust.
NSL-KDD [103]	Deep learning classifiers for binary SDN intrusion detection	CNN, DNN, RNN, LSTM, GRU	CNN achieved highest accuracy, effective binary classification.
InSDN [104]	Hybrid CNN-LSTM model with regularization for anomaly or zero-day attack detection	CNN, LTSM and CNN+LTSM for classification. L1 and L2 regularization	FPR and FNR are quite high and needs to be addressed.
NSL-KDD [105]	5-level hybrid classification with flow-based features	KNN, ELM, H-ELM,	Accuracy of 84.29%, low-complexity with few flow features. FAR can further be reduced.

CHAPTER 3

DATASET DESCRIPTION AND ANALYSIS

Intrusion detection datasets are structured collection of packets in the network that includes both benign (normal) and malicious (attack) activities. These datasets are used to train, test, and evaluate IDS, especially those using ML and DL. There are various publicly available datasets for research. In this section, we are going to discuss the datasets which are used in this thesis.

3.1 CIC-IDS2017 Dataset

The CICIDS2017 dataset [107] was developed by the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick in 2017. It is designed to provide a comprehensive and realistic evaluation environment for IDS, including both benign and malicious traffic. The dataset reflects modern network traffic with legitimate user behaviour and diverse attack scenarios, captured over a period of five working days (July 3–7, 2017). The dataset is available in PCAP file and CSV files. It has over 80 features. The dataset contains 14 attack types such as DoS Hulk, DoS GoldenEye, Dos Slowloris, DoS SlowHTTPTest, DDoS, FTP-Patator, SSH-Patator, Web Attack-Brute Force, Web Attack-XSS, Web Attack-SQL Injection, Bot, Infiltration, PortScan and Heartbleed. It has a total of around 2.8 million records out of which around 2.3 million records are Benign and around 470,000 records are malicious (Attack). Table 3.1 shows the list of the features present in the CIC-IDS2017 dataset.

Table 3.1 List of Features of CIC-IDS2017 Dataset

Sl. No	**Feature Name**	**Sl. No**	**Feature Name**
1	Flow ID	2	Source Port
3	Source IP	4	Destination IP
5	Destination Port	6	Protocol
7	Timestamp	8	Flow Duration
9	Total Fwd Packets	10	Total Backward Packets
11	Total Length of Fwd Packets	12	Total Length of Bwd Packets
13	Fwd Packet Length Max	14	Fwd Packet Length Min

15	Fwd Packet Length Mean	16	Fwd Packet Length Std
17	Bwd Packet Length Max	18	Bwd Packet Length Min
19	Bwd Packet Length Mean	20	Bwd Packet Length Std
21	Flow Bytes/s	22	Flow Packets/s
23	Flow IAT Mean	24	Flow IAT Std
25	Flow IAT Max	26	Flow IAT Min
27	Fwd IAT Total	28	Fwd IAT Mean
29	Fwd IAT Std	30	Fwd IAT Max
31	Fwd IAT Min	32	Bwd IAT Total
33	Bwd IAT Mean	34	Bwd IAT Std
35	Bwd IAT Max	36	Bwd IAT Min
37	Fwd PSH Flags	38	Bwd PSH Flags
39	Fwd URG Flags	40	Bwd URG Flags
41	Fwd Header Length	42	Bwd Header Length
43	Fwd Packets/s	44	Bwd Packets/s
45	Min Packet Length	46	Max Packet Length
47	Packet Length Mean	48	Packet Length Std
49	Packet Length Variance	50	FIN Flag Count
51	SYN Flag Count	52	RST Flag Count
53	PSH Flag Count	54	ACK Flag Count
55	URG Flag Count	56	CWE Flag Count
57	ECE Flag Count	58	Down/Up Ratio
59	Average Packet Size	60	Avg Fwd Segment Size
61	Avg Bwd Segment Size	62	Fwd Header Length
63	Fwd Avg Bytes/Bulk	64	Fwd Avg Packets/Bulk
65	Fwd Avg Bulk Rate	66	Bwd Avg Bytes/Bulk
67	Bwd Avg Packets/Bulk	68	Bwd Avg Bulk Rate
69	Subflow Fwd Packets	70	Subflow Fwd Bytes
71	Subflow Bwd Packets	72	Subflow Bwd Bytes
73	Init Win bytes forward	74	Init Win bytes backward
75	act data pkt fwd	76	min seg size forward
77	Active Mean	78	Active Std
79	Active Max	80	Active Min
81	Idle Mean	82	Idle Std
83	Idle Max	84	Idle Min

Abbreviations: ID=Identifier; IP=Internet Protocol; Fwd=Forward; Bwd=Backward; Max=Maximum; Min=Minimum; Std=Standard Deviation; IAT=Inter Arrival Time; Fin=Finish; SYN=Synchronize; PSH=Push;

ACK=Acknowledgement; URG=Urgent; CWE= Congestion Window Reduced; ECE=Explicit Congestion Notification Echo; Avg=Average; Init=Initial; Win=Window; seg=Segment; act=Active; pkt=Packet;

The Class distribution of CIC-IDS2017 dataset is shown in different parts as the data distribution range is high ranging from 11 to 2,359,289 records.

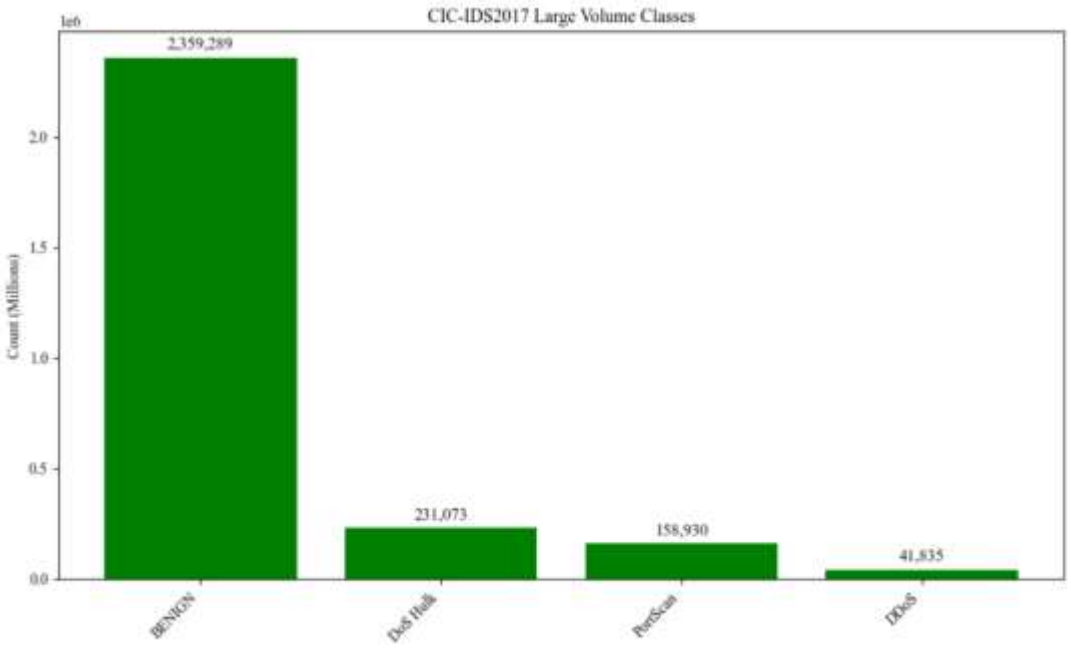

Figure 3.1 Large Volume Class of CIC-IDS2017 Dataset

Figure 3.1 shows the class distribution of CIC-IDS2017 dataset which has higher volume of records. This includes Benign, DoS Hulk, PortScan and DDoS class.

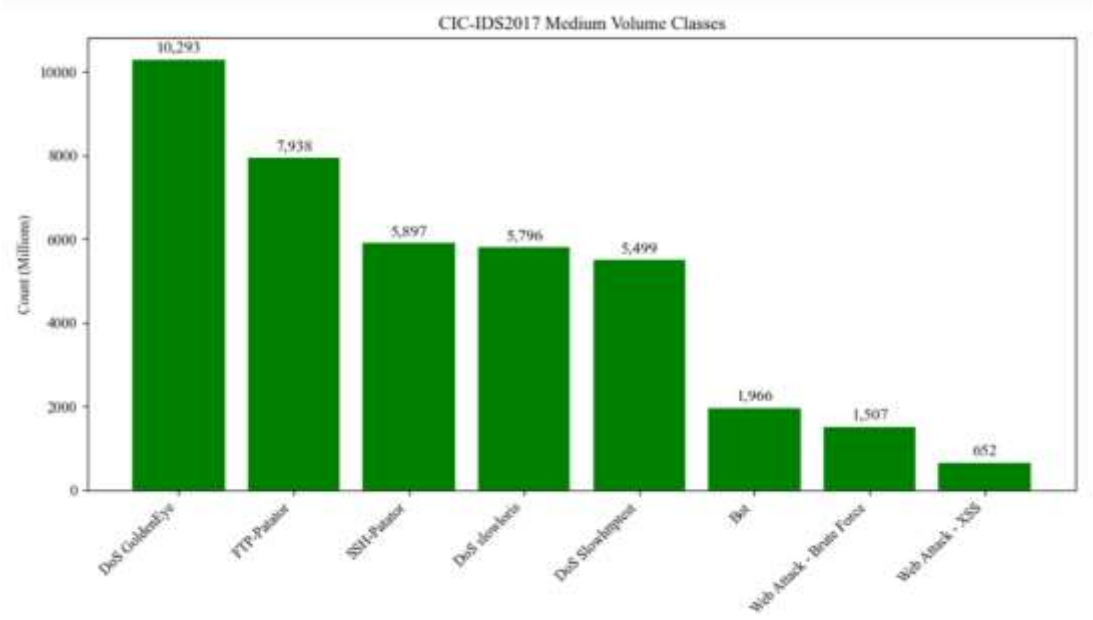

Figure 3.2 Medium Volume Class of CIC-IDS2017 Dataset

Figure 3.2 shows the class distribution of CIC-IDS2017 dataset which has medium number of records in the dataset. This includes DoS GoldenEye, FTP-Patator, SSH-Patator, DoS Slowloris, DoS Slowhttptest, Bot, Web Attack - Brute Force, and Web Attack - XSS class. Medium volume class ranges from 652 records to 10,293 records.

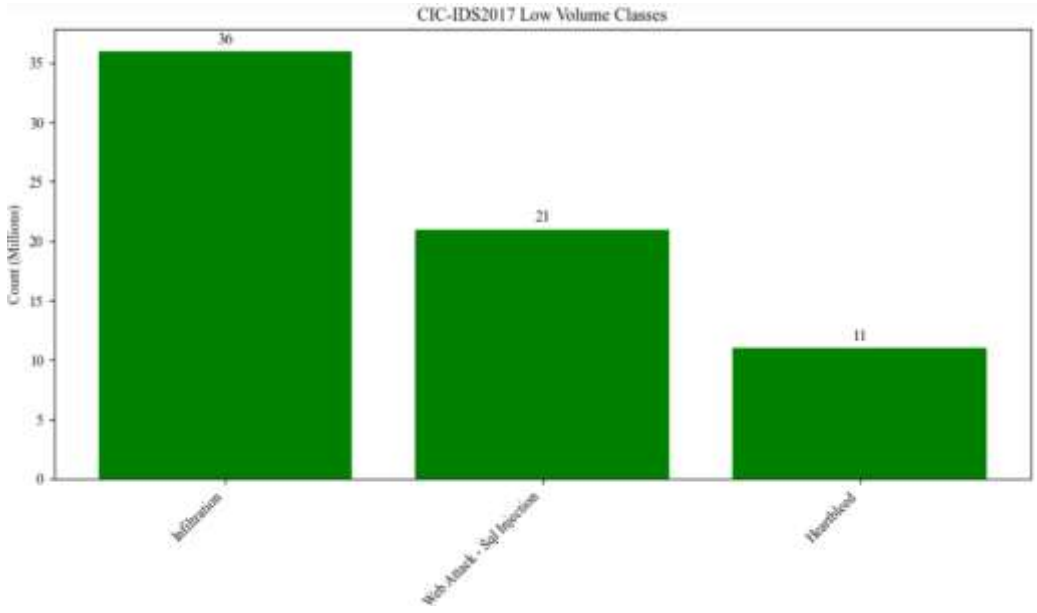

Figure 3.3 Low Volume Class of CIC-IDS2017 Dataset

Figure 3.3 shows the class distribution of CIC-IDS2017 dataset which has low volume of records. This includes Infiltration, Web Attack - SQL Injection, and Heartbleed class. Low volume class ranges from 11 records to 36 records.

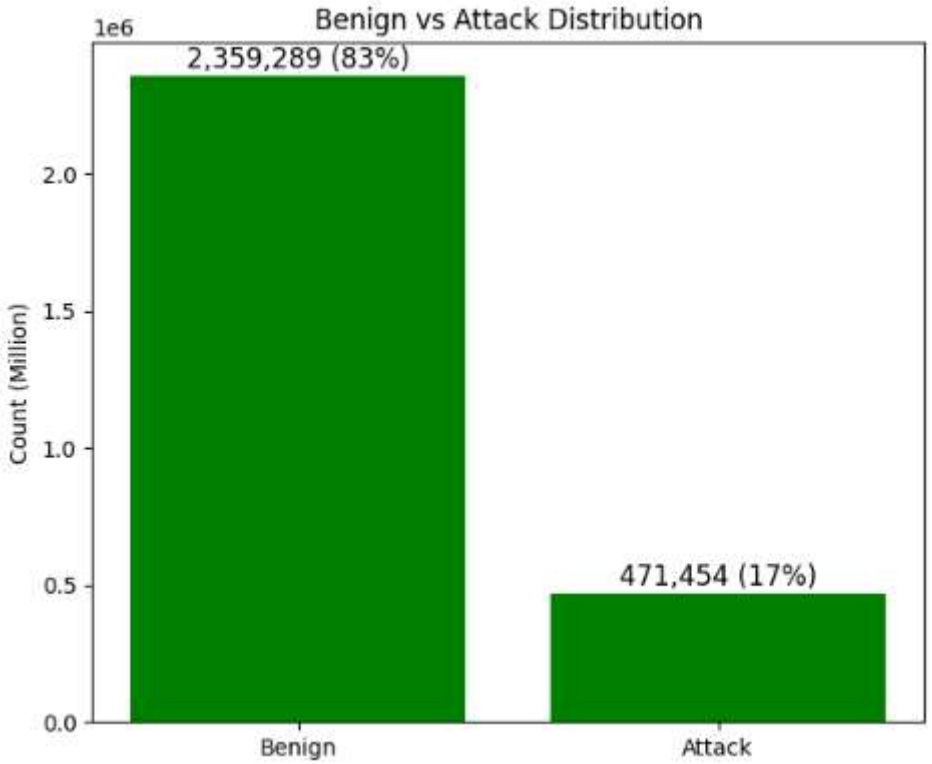

Figure 3.4 Benign and Attack Distribution of CIC-IDS2017 Dataset

Figure 3.4 shows the class distribution of CIC-IDS2017 dataset in binary form (Benign and Attack). It can be seen that the Benign records overwhelm the Attack records constituting 83% of the dataset.

3.2 CSE-CIC-IDS2018 Dataset

The CSE-CIC-IDS2018 dataset [107] was developed collaboratively by the CIC and Communications Security Establishment (CSE) in 2018. It is an extension of the CICIDS2017 dataset and addresses its limitations by introducing more attack types, more diversity in traffic, and a larger volume of realistic network activity. The dataset aims to facilitate the development and evaluation of robust IDS capable of handling modern and complex threats. The network traffic was captured over 10 consecutive days. The dataset is available in PCAP (raw

files) and CSV files (for each day). It has around 80 features which are presented in Table 3.2. This dataset also contains 14 attack types such as Bot, Brute Force Attack, DDoS, DoS GoldenEye, DoS Hulk, DoS SlowHTTPTest, DoS Slowloris, FTP-BruteForce, Heartbleed, Infiltration, PortScan, SQL Injection, SSH-BruteForce, and Web Attack. It has around 16 million records out of which around 13 million records are Benign and around 3 million records are attacks.

Table 3.2 List of features of CSE-CIC-IDS2018 dataset

Sl. No	**Feature Name**	**Sl. No**	**Feature Name**
1	Dst Port	2	Protocol
3	Flow Duration	4	Tot Fwd Pkts
5	Tot Bwd Pkts	6	TotLen Fwd Pkts
7	TotLen Bwd Pkts	8	Fwd Pkt Len Max
9	Fwd Pkt Len Min	10	Fwd Pkt Len Mean
11	Fwd Pkt Len Std	12	Bwd Pkt Len Max
13	Bwd Pkt Len Min	14	Bwd Pkt Len Mean
15	Bwd Pkt Len Std	16	Flow Byts/s
17	Flow Pkts/s	18	Flow IAT Mean
19	Flow IAT Std	20	Flow IAT Max
21	Flow IAT Min	22	Fwd IAT Tot
23	Fwd IAT Mean	24	Fwd IAT Std
25	Fwd IAT Max	26	Fwd IAT Min
27	Bwd IAT Tot	28	Bwd IAT Mean
29	Bwd IAT Std	30	Bwd IAT Max
31	Bwd IAT Min	32	Fwd PSH Flags
33	Bwd PSH Flags	34	Fwd URG Flags
35	Bwd URG Flags	36	Fwd Header Len
37	Bwd Header Len	38	Fwd Pkts/s
39	Bwd Pkts/s	40	Pkt Len Min
41	Pkt Len Max	42	Pkt Len Mean
43	Pkt Len Std	44	Pkt Len Var
45	FIN Flag Cnt	46	SYN Flag Cnt
47	RST Flag Cnt	48	PSH Flag Cnt
49	ACK Flag Cnt	50	URG Flag Cnt
51	CWE Flag Count	52	ECE Flag Cnt
53	Down/Up Ratio	54	Pkt Size Avg
55	Fwd Seg Size Avg	56	Bwd Seg Size Avg
57	Fwd Byts/b Avg	58	Fwd Pkts/b Avg

59	Fwd Blk Rate Avg	60	Bwd Byts/b Avg
61	Bwd Pkts/b Avg	62	Bwd Blk Rate Avg
63	Subflow Fwd Pkts	64	Subflow Fwd Byts
65	Subflow Bwd Pkts	66	Subflow Bwd Byts
67	Init Fwd Win Byts	68	Init Bwd Win Byts
69	Fwd Act Data Pkts	70	Fwd Seg Size Min
71	Active Mean	72	Active Std
73	Active Max	74	Active Min
75	Idle Mean	76	Idle Std
77	Idle Max	78	Idle Min

Footnote: Dst=Destination; Tot=Total; Fwd=Forward; Pkts=Packets; Bwd=Backward; TotLen=Total Length; Len=Length; Max=Maximum; Min=Minimum; Std=Standard Deviation; Byts/s=Bytes per second; Pkts/s=Packets per second; IAT=Inter Arrival Time; Fin=Finish; Cnt=Count; SYN=Synchronize; PSH=Push; ACK=Acknowledgement; URG=Urgent; CWE= Congestion Window Reduced; ECE=Explicit Congestion Notification Echo; Byts/b=Bytes per Bulk; Avg=Average; Init=Initial; Win=Window; seg=Segment; Act=Active;

The Class distribution of CSE-CIC-IDS2018 dataset is also shown in different parts due to its vast range of number of records for different classes.

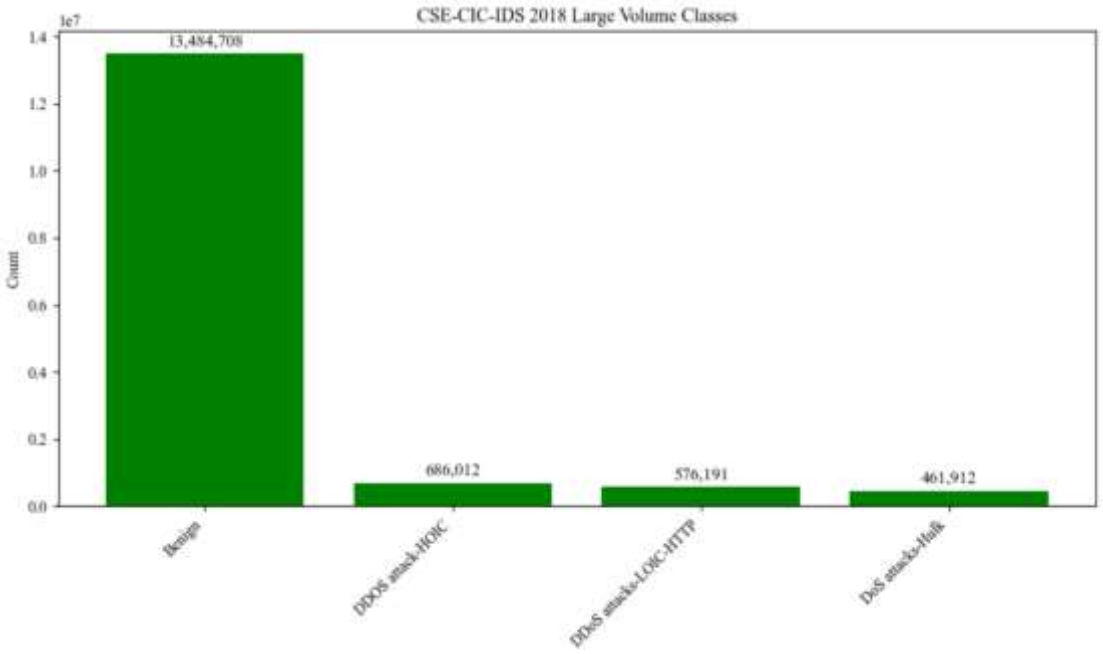

Figure 3.5 Large Volume Class of CSE-CIC-IDS2018 Dataset

Figure 3.5 shows the class distribution of CSE-CIC-IDS2018 dataset which has higher volume of records ranging from 461,912 records to 13,484,708 records. This include Benign, DDoS attack-HOIC, DDoS attack-LOIC-HTTP, and DoS attack-Hulk.

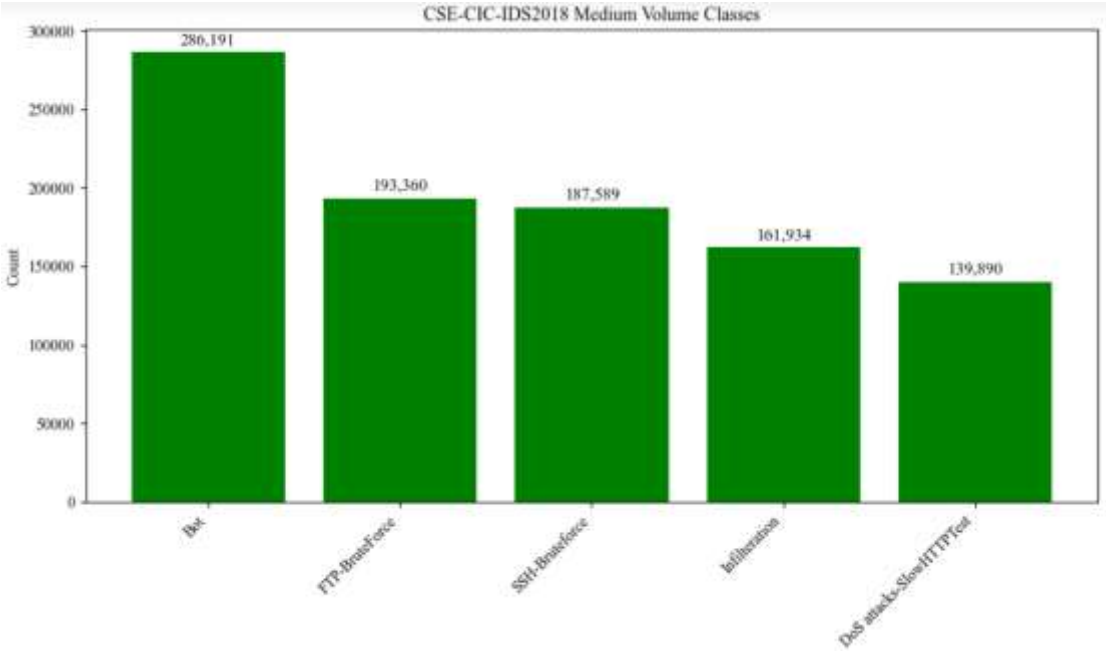

Figure 3.6 Medium Volume Class of CSE-CIC-IDS2018 Dataset

Figure 3.6 shows the class distribution of CSE-CIC-IDS2018 dataset which has medium number of records in the dataset ranging from 139,890 records to 286,191 records. This includes Bot, FTP-Brute Force, SSH-Brute Force, Infiltration, and DoS attack-SlowHTTPTest.

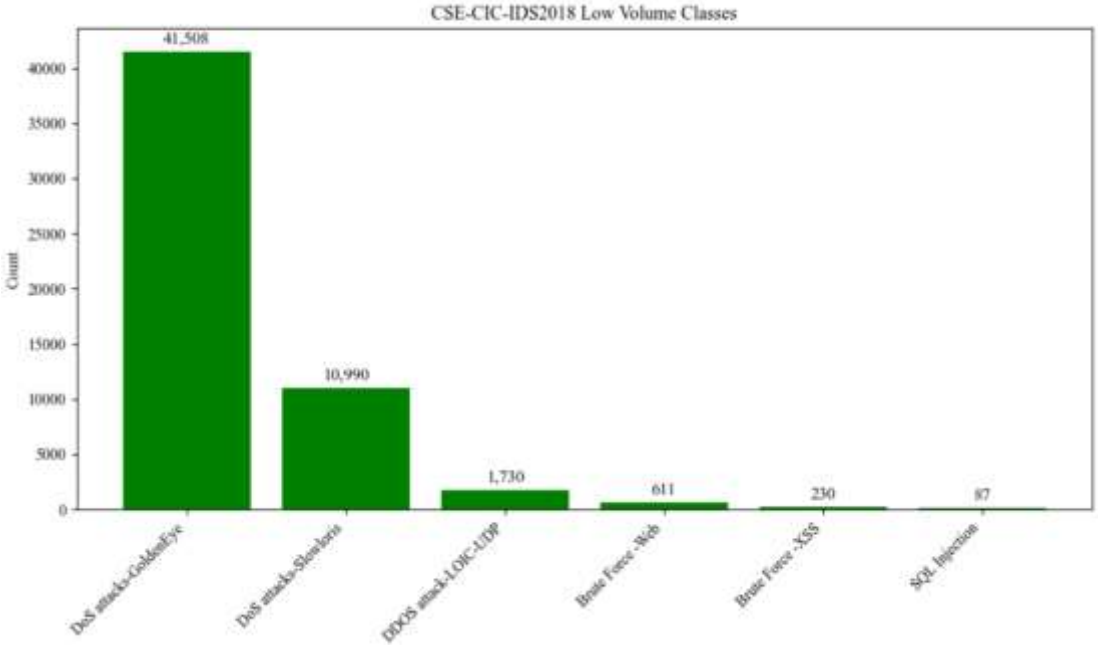

Figure 3.7 Low Volume Class of CSE-CIC-IDS2018 Dataset

Figure 3.7 shows the class distribution of CSE-CIC-IDS2018 dataset which has low volume of records ranging from 87 records to 41,508 records. This includes DoS attacks-GoldenEye, DoS attacks-Slowloris, DDOS attack-LOIC-UDP, Brute Force-Web, Brute Force-XSS, and SQL Injection.

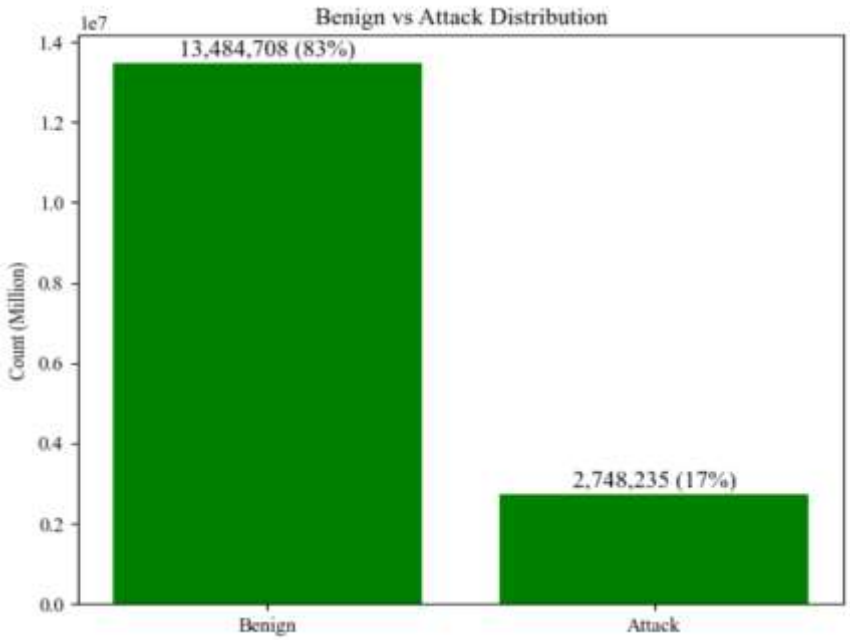

Figure 3.8 Benign and Attack Distribution of CSE-CIC-IDS2018 Dataset

Figure 3.8 shows the class distribution of CSE-CIC-IDS2018 dataset in binary form (Benign and Attack). It can be seen that the Benign records overwhelm the Attack records constituting 83% of the dataset.

3.3 CIC-DDoS2019 Dataset

The CIC-DDoS2019 dataset [108], developed by the CIC was released in 2019, and is specifically designed to support research in detecting and mitigating DDoS attacks. This dataset addresses the limitations of previous datasets by providing a diverse range of DDoS attack scenarios executed in a realistic network environment, accompanied by both raw traffic captures and extracted flow-based features. This dataset is available in PCAP format and CSV files. There are around 80 features in this dataset which is presented in Table 3.3. There are approximately 70 million records in this dataset out of which only around 100,000 records are Benign, which is less than 1% of the total records. The rest (more than 99%) comprises different kinds of DDoS attack. This indicates that the CIC-DDoS2019 dataset is severely imbalanced and thus need to carefully considered and resampled when feeding into ML models. The different kind of attack present in this dataset are DDoS_DNS, DDoS_LDAP,

DDoS_MSSQL, DDoS_NTP, DDoS_NetBIOS, DDoS_SSDP, DDoS_SNMP, DDoS_UDP, Syn, TFTP, UDP_lag, WebDDoS, and Portmap.

Table 3.3 List of features of CIC-DDoS2019 Dataset

Sl. No	**Feature Name**	**Sl. No**	**Feature Name**
1	Source Port	2	Destination Port
3	Protocol	4	Flow Duration
5	Total Fwd Packets	6	Total Backward Packets
7	Total Length of Fwd Packets	8	Total Length of Bwd Packets
9	Fwd Packet Length Max	10	Fwd Packet Length Min
11	Fwd Packet Length Mean	12	Fwd Packet Length Std
13	Bwd Packet Length Max	14	Bwd Packet Length Min
15	Bwd Packet Length Mean	16	Bwd Packet Length Std
17	Flow Bytes/s	18	Flow Packets/s
19	Flow IAT Mean	20	Flow IAT Std
21	Flow IAT Max	22	Flow IAT Min
23	Fwd IAT Total	24	Fwd IAT Mean
25	Fwd IAT Std	26	Fwd IAT Max
27	Fwd IAT Min	28	Bwd IAT Total
29	Bwd IAT Mean	30	Bwd IAT Std
31	Bwd IAT Max	32	Bwd IAT Min
33	Fwd PSH Flags	34	Bwd PSH Flags
35	Fwd URG Flags	36	Bwd URG Flags
37	Fwd Header Length	38	Bwd Header Length
39	Fwd Packets/s	40	Bwd Packets/s
41	Min Packet Length	42	Max Packet Length
43	Packet Length Mean	44	Packet Length Std
45	Packet Length Variance	46	FIN Flag Count
47	SYN Flag Count	48	RST Flag Count
49	PSH Flag Count	50	ACK Flag Count
51	CWE Flag Count	52	ECE Flag Count
53	Down/Up Ratio	54	Average Packet Size
55	Avg Fwd Segment Size	56	Avg Bwd Segment Size
57	Fwd Header Length.1	58	Fwd Avg Bytes/Bulk
59	Fwd Avg Packets/Bulk	60	Fwd Avg Bulk Rate
61	Bwd Avg Bytes/Bulk	62	Bwd Avg Packets/Bulk
63	Bwd Avg Bulk Rate	64	Subflow Fwd Packets
65	Subflow Fwd Bytes	66	Subflow Bwd Packets
67	Subflow Bwd Bytes	68	Init_Win_bytes_forward

69	Init_Win_bytes_backward	70	act_data_pkt_fwd
71	min_seg_size_forward	72	Active Mean
73	Active Std	74	Active Max
75	Active Min	76	Idle Mean
77	Idle Std	78	Idle Max
79	Idle Min	80	Inbound

Footnote: Fwd=Forward; Bwd=Backward; Max=Maximum; Min=Minimum; Std=Standard Deviation; Byts/s=Bytes per second; Pkts/s=Packets per second; IAT=Inter Arrival Time; PSH=Push; URG=Urgent; FIN=Finish; SYN=Synchronize; RST=Reset; ACK=Acknowledgement; CWE= Congestion Window Reduced; ECE=Explicit Congestion Notification Echo; Avg=Average; Init=Initial; Win=Window; seg=Segment; Act=Active.

3.4 Bot-IoT Dataset

The Bot-IoT dataset [109] was developed by the Cyber Range Lab at UNSW Canberra to address the need for a comprehensive dataset that captures realistic IoT network traffic, including both benign and malicious activities. Released in 2018, this dataset is designed to support research in intrusion detection, network forensics, and ML applications within IoT environments. The dataset was generated using a realistic testbed that emulated a typical IoT network environment. The BoT-IoT dataset encompasses a diverse range of attack types such as DoS, DDoS, Information Theft, Probing attack and its sub-attack. There are 35 features on this dataset which are presented in Table 3.4. There are more than 73 million records out of which only 9534 records are Benign and the rest are different kinds of attacks. This dataset also has a severe class imbalance which needs to be carefully pre-processed before feeding into ML/DL model.

Table 3.4 List of features and its description of BOT-IoT dataset

Features	**Descriptions**	**Features**	**Descriptions**
pkSeqID	Packet sequence Identity	stime	Record state time
flgs	Flow state flag in transaction	proto	Transaction protocol in network flow

saddr	Source address	sport	Source port number
daddr	Destination address	dport	Destination port number
pkts	Total count of packets	bytes	Total number of bytes
state	Transaction state	ltime	Record last time
seq	Argus sequence number	dur	Record Total duration
mean	Average duration of records	stddev	Standard deviation of duration of records
smac	Source MAC address	dmac	Destination MAC address
sum	Total duration of records	min	Minimum duration of records
max	Maximum duration of records	soui	Source organizational unique identity
doui	Destination organizational unique identity	sco	Source IP country code
dco	Destination IP Country code	spkts	Source-destination packet count
dpkts	Destination-source packet count	sbytes	Source-destination transaction bytes
dbytes	Destination-source transaction bytes	rate	Total packet per second
srate	Source-destination packet per second	drate	Destination-source packet per second
attack	Class label	category	Traffic category
subcategory	Traffic subcategory		

The Class distribution of Bot-IoT dataset is shown in parts due to its high range of number of records in the dataset.

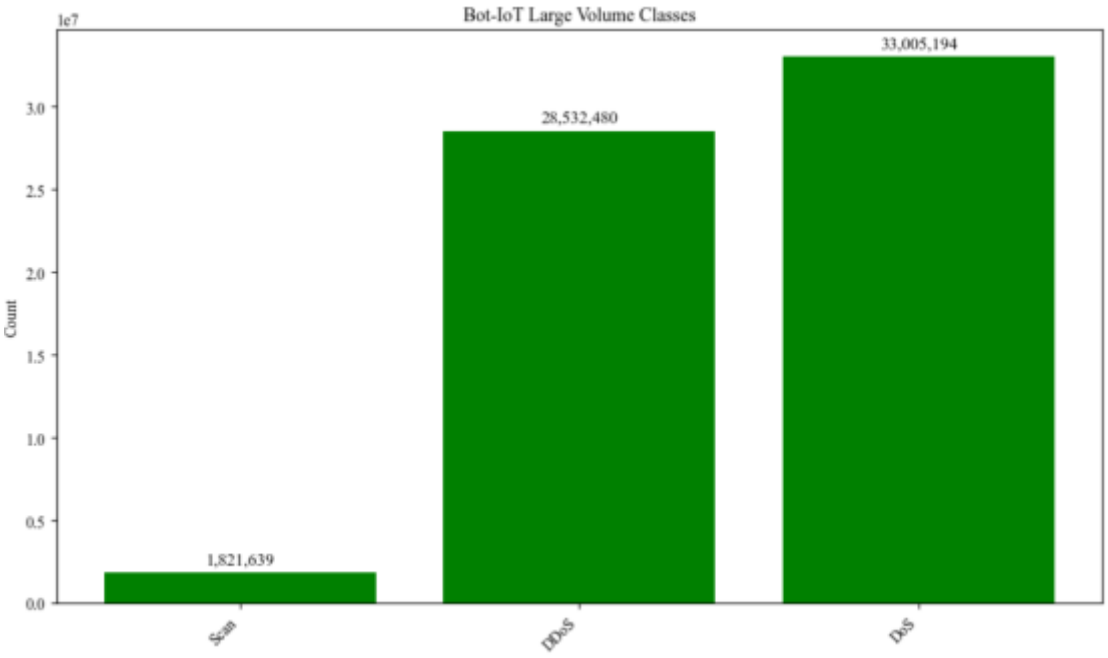

Figure 3.9 Large Volume Class of Bot-IoT Dataset

Figure 3.9 shows the class distribution of Bot-IoT dataset which has high volume of records ranging from 1,821,639 records to 33,005,194 records. This includes Scan, DDoS, and DoS attack class.

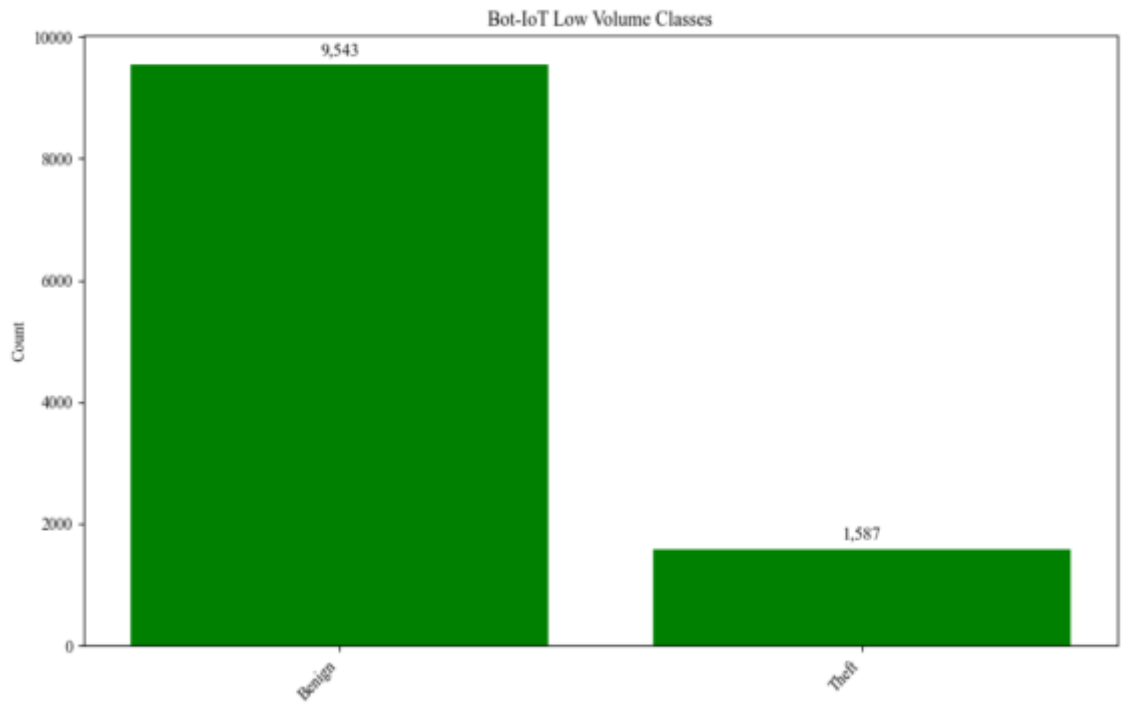

Figure 3.10 Low Volume Class of Bot-IoT Dataset

Figure 3.10 shows the class distribution of Bot-IoT dataset which has low volume of records ranging from 1,587 records to 9,534 records. This includes Theft and Benign class.

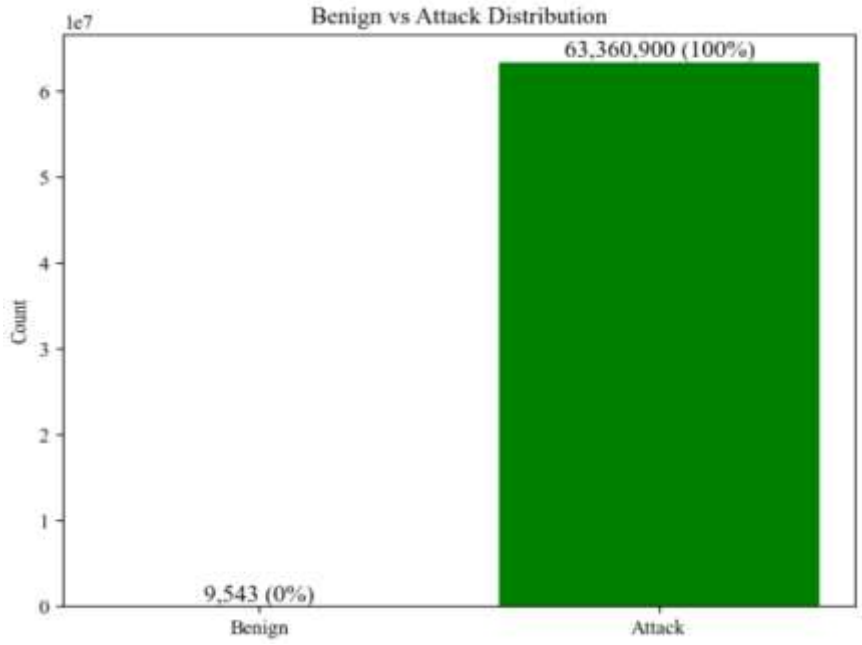

Figure 3.11 Benign and Attack Distribution of Bot-IoT Dataset

Figure 3.11 shows the class distribution of Bot-IoT dataset in binary form (Benign and Attack). It can be seen that the Attack records overwhelm the Benign records constituting 99.99% of the dataset.

3.5 **InSDN Dataset**

The InSDN dataset [110] is a specialized intrusion detection dataset designed for SDN environments. Developed to address the unique security challenges inherent in SDN architectures, it provides a rich set of labelled data suitable for training and evaluating ML models in SDN-based networks. The dataset is divided into three groups (csv) based on the traffic types and the target machines. The first group (Normal.csv) includes the normal traffic only. The second group (Metasploitable2.csv)contains the attack traffics that target the mealsplotable 2 server. Lastly, the third group (OVS.csv) includes the attacks on OVS machine. There are 84 features on this dataset which is presented in Table 3.5, it includes Normal instances and attack instances like DoS, DDoS, Probe, BFA (brute force

attack), Web-Attack, Botnet, and U2R. There are around 340,000 records in this dataset, out of which around 68,000 are Normal and the rest are distributed between the said attacks.

Table 3.5 List of features of InSDN Dataset

Sl. No	**Feature Name**	**Sl. No**	**Feature Name**
1	Flow ID	2	Src IP
3	Src Port	4	Dst IP
5	Dst Port	6	Protocol
7	Timestamp	8	Flow Duration
9	Tot Fwd Pkts	10	Tot Bwd Pkts
11	TotLen Fwd Pkts	12	TotLen Bwd Pkts
13	Fwd Pkt Len Max	14	Fwd Pkt Len Min
15	Fwd Pkt Len Mean	16	Fwd Pkt Len Std
17	Bwd Pkt Len Max	18	Bwd Pkt Len Min
19	Bwd Pkt Len Mean	20	Bwd Pkt Len Std
21	Flow Byts/s	22	Flow Pkts/s
23	Flow IAT Mean	24	Flow IAT Std
25	Flow IAT Max	26	Flow IAT Min
27	Fwd IAT Tot	28	Fwd IAT Mean
29	Fwd IAT Std	30	Fwd IAT Max
31	Fwd IAT Min	32	Bwd IAT Tot
33	Bwd IAT Mean	34	Bwd IAT Std
35	Bwd IAT Max	36	Bwd IAT Min
37	Fwd PSH Flags	38	Bwd PSH Flags
39	Fwd URG Flags	40	Bwd URG Flags
41	Fwd Header Len	42	Bwd Header Len
43	Fwd Pkts/s	44	Bwd Pkts/s
45	Pkt Len Min	46	Pkt Len Max
47	Pkt Len Mean	48	Pkt Len Std
49	Pkt Len Var	50	FIN Flag Cnt
51	SYN Flag Cnt	52	RST Flag Cnt
53	PSH Flag Cnt	54	ACK Flag Cnt
55	URG Flag Cnt	56	CWE Flag Count
57	ECE Flag Cnt	58	Down/Up Ratio
59	Pkt Size Avg	60	Fwd Seg Size Avg
61	Bwd Seg Size Avg	62	Fwd Byts/b Avg
63	Fwd Pkts/b Avg	64	Fwd Blk Rate Avg
65	Bwd Byts/b Avg	66	Bwd Pkts/b Avg
67	Bwd Blk Rate Avg	68	Subflow Fwd Pkts
69	Subflow Fwd Byts	70	Subflow Bwd Pkts
71	Subflow Bwd Byts	72	Init Fwd Win Byts
73	Init Bwd Win Byts	74	Fwd Act Data Pkts
75	Fwd Seg Size Min	76	Active Mean

77	Active Std	78	Active Max
79	Active Min	80	Idle Mean
81	Idle Std	82	Idle Min
83	Idle Max		

Footnote: ID=Identifier; Src=Source; Dst=Destination; Tot=Total; Fwd=Forward; Pkts=Packets; Bwd=Backward; TotLen=Total Length; Max=Maximum; Min=Minimum; Std=Standard Deviation; Byts/s=Bytes per second; Pkts/s=Packets per second; Len=Length; IAT=Inter Arrival Time; FIN=Finish; SYN=Synchronize; RST=Reset; PSH=Push; ACK=Acknowledgement; URG=Urgent; CWE= Congestion Window Reduced; ECE=Explicit Congestion Notification Echo; Avg=Average; Init=Initial; Win=Window; seg=Segment; Act=Active;

The Class distribution of InSDN dataset is also shown in two parts due to its high range of number of records in the dataset.

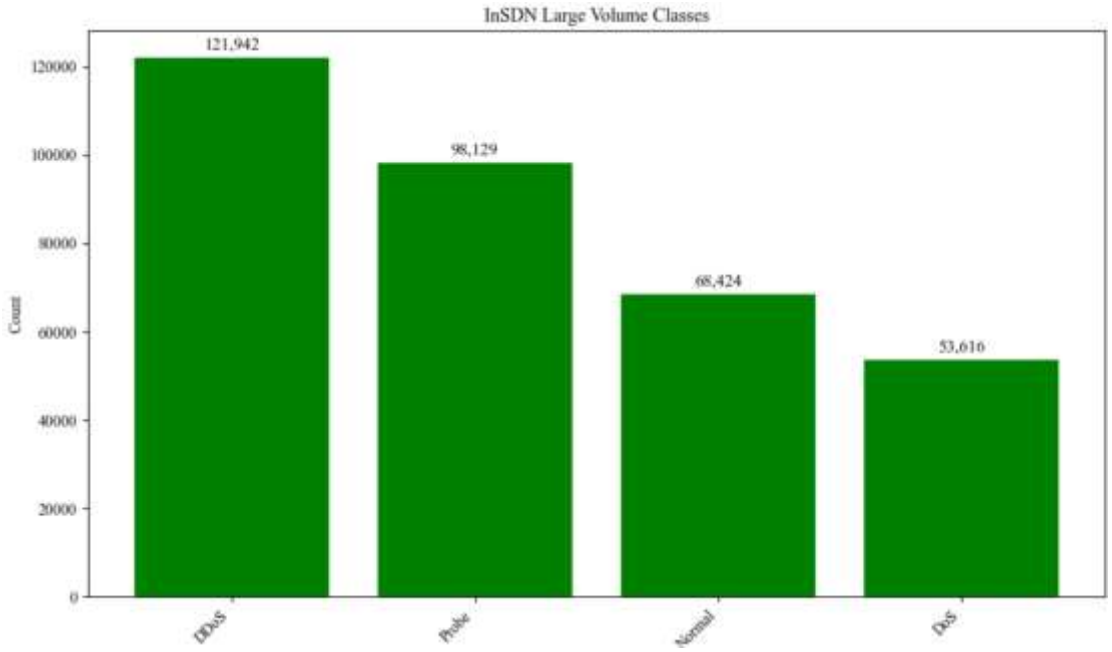

Figure 3.12 Large Volume Class of InSDN Dataset

Figure 3.12 shows the class distribution of InSDN dataset which has Large volume of records ranging from 53,616 records to 121,942 records. This includes DoS, Normal, Probe, and DDoS class.

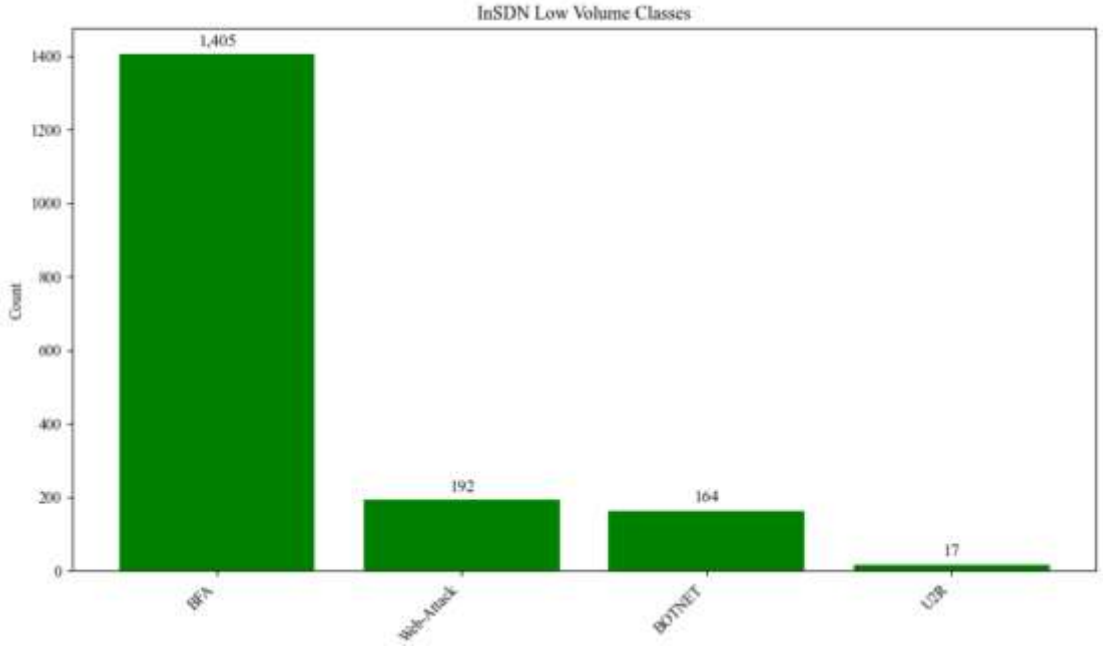

Figure 3.13 Low Volume Class of InSDN Dataset

Figure 3.13 shows the class distribution of InSDN dataset which has Low volume of records ranging from 17 records to 1,405 records. This includes U2R, BOTNET, Web-Attack, and BFA class.

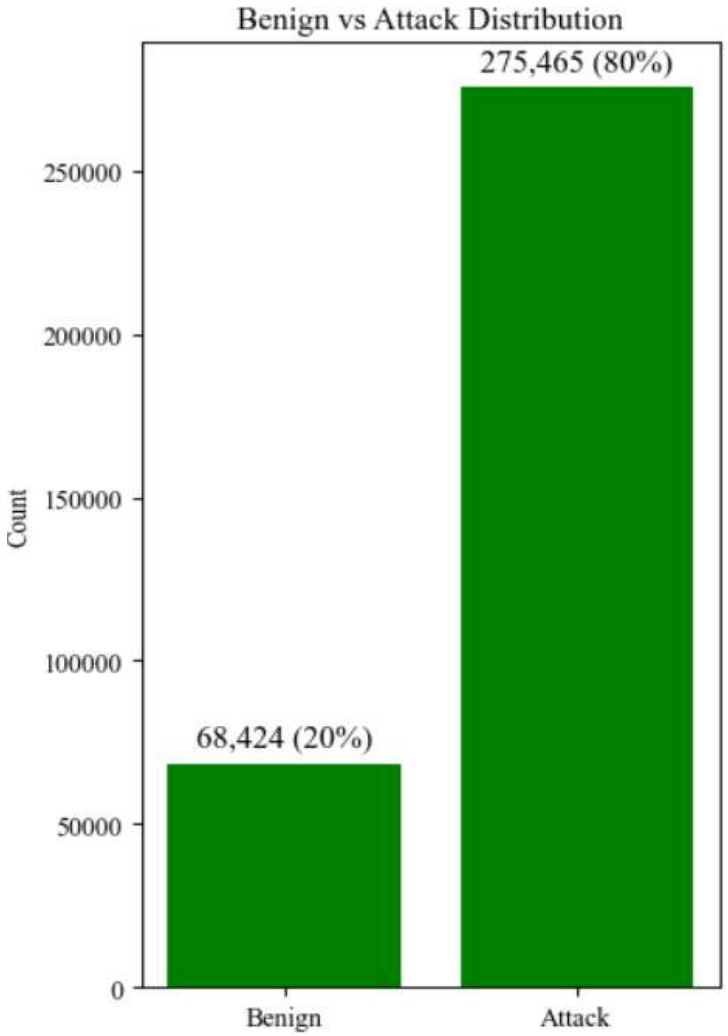

Figure 3.14 Benign and Attack Distribution of InSDN Dataset

Figure 3.14 shows the class distribution of Bot-IoT dataset in binary form (Benign and Attack). It can be seen that the Attack records overwhelm the Benign records constituting 80% of the dataset.

3.6 Networking Attacks present in the Datasets

Networking attacks are malicious activities aimed at disrupting, stealing, or compromising the flow of data across networks. These attacks can target individuals, organizations, or infrastructure. There are numerous numbers of networking attacks which can broadly be divided into passive attack and active attack. There also exist the third category: Social Engineering Attacks which are manipulative techniques used by attackers to trick individuals into revealing

sensitive information or performing actions that compromise security. This is beyond the scope of intrusion detection and can only be mitigated by awareness and carefulness of individuals. In this section, we are discussing the network attacks which are present in the datasets used for evaluation on this study.

3.6.1 Denial of Service (DoS)

A DoS attack is a type of cyber threat that aims to interrupt the normal functioning of a system, server, or network by overwhelming it with an excessive volume of traffic or deliberately exploiting its vulnerabilities. The primary goal is to exhaust the target's available resources such as bandwidth, memory, or processing power so that legitimate users are unable to access the service. These attacks often originate from a single malicious source and can involve techniques such as packet flooding or malformed requests. Although simpler than Distributed DoS (DDoS) attacks, DoS incidents can still cause significant disruption and financial loss [111]. Different DoS variant that are present in this study are:

- DoS Hulk: The DoS Hulk (HTTP Unbearable Load King) attack is a type of Denial of Service (DoS) that targets web servers by generating a massive number of unique HTTP requests. Developed as a proof-of-concept tool, Hulk aims to overwhelm server resources by bypassing caching mechanisms and forcing the server to process each request individually. It achieves this by randomizing elements such as URLs, headers, and user-agent strings, making each request appear distinct and preventing efficient caching or filtering. This method leads to rapid consumption of server resources like CPU and memory, potentially causing service degradation or complete unavailability. The Hulk attack is particularly effective against servers lacking robust rate-limiting or traffic analysis defences [112].
- DoS Goldeneye: The DoS GoldenEye attack is a type of Denial of Service (DoS) that targets web servers by generating a massive number of unique HTTP requests. It achieves this by randomizing elements such as

URLs, headers, and user-agent strings, making each request appear distinct and preventing efficient caching or filtering. This method leads to rapid consumption of server resources like CPU and memory, potentially causing service degradation or complete unavailability. The GoldenEye attack is particularly effective against servers lacking robust rate-limiting or traffic analysis defenses [111].

- DoS Slowloris: The DoS Slowloris attack is a low-bandwidth, application-layer DoS technique that targets web servers by opening numerous simultaneous connections and keeping them open as long as possible. It achieves this by sending partial HTTP requests and periodically adding to them without completing the requests, thereby consuming server resources and preventing legitimate connections. This method is particularly effective against servers that allocate resources per connection, such as Apache 1.x and 2.x, leading to resource exhaustion and service unavailability. Due to its stealthy nature and minimal bandwidth usage, Slowloris attacks can be challenging to detect and mitigate. [111].
- DoS SlowHTTPTest: The DoS SlowHTTPTest attack is a type of DoS that targets web servers by exploiting the way they handle HTTP connections. It operates by initiating numerous HTTP connections to the target server and sending partial HTTP requests at a very slow rate. This behaviour causes the server to keep these connections open, waiting for the completion of the requests, thereby consuming server resources such as memory and connection slots. Over time, this can lead to resource exhaustion, preventing legitimate users from establishing connections and effectively rendering the service unavailable. The SlowHTTPTest tool is capable of simulating various slow HTTP attacks, including Slowloris, Slow POST, and Slow Read attacks, making it a versatile tool for testing server resilience against such threats [111].
- DoS TCP: TCP-based DoS attacks, notably SYN floods, exploit the TCP three-way handshake by inundating a server with SYN requests without

completing the handshake. This leads to resource exhaustion, rendering the server incapable of handling legitimate connections. Recent studies highlight vulnerabilities in Network Address Translation (NAT) devices, where attackers can remotely terminate TCP connections by exploiting deficiencies in the Path MTU Discovery mechanism, causing legitimate sessions to drop unexpectedly [113].

- DoS UDP: UDP-based DoS attacks leverage the protocol's connectionless nature to flood targets with unsolicited traffic. Attackers often use amplification techniques, sending small requests to misconfigured servers that reply with large responses to the victim, overwhelming their network. A recent study introduced a mitigation method using a Snort UDP module in fog computing environments to detect and prevent such Distributed Reflection DoS (DRDoS) attacks, enhancing security in resource-constrained settings [114].
- DoS HTTP: HTTP-based DoS attacks, including Slow HTTP attacks, target web servers by initiating numerous HTTP requests and keeping them open, exhausting server resources. These attacks are challenging to detect due to their low bandwidth usage and mimicry of legitimate traffic. Research has focused on enhancing detection mechanisms by analysing TCP/IP packet behaviours to identify such slow attacks effectively.

3.6.2 Distributed Denial of Service (DDoS)

A DDoS attack overwhelms a target system or network by flooding it with traffic from multiple compromised sources, typically coordinated through botnets. Unlike basic DoS attacks, DDoS attacks are harder to mitigate due to their distributed nature and large-scale traffic. Common techniques include amplification and reflection, often exploiting vulnerable services to magnify the impact. These attacks can severely degrade service availability, causing downtime and economic loss. The rise of IoT devices has further expanded the attack surface, making systems more vulnerable [115]. DDoS attack is present in all the datasets used for evaluation in this study.

3.6.3 Brute Force Attack (BFA)

A BFA is a method used to gain unauthorized access by systematically attempting all possible credential combinations until the correct one is found. Unlike attacks that exploit software flaws, BFA rely on weak passwords and lack of rate-limiting or account lockout policies [116]. These attacks commonly target services like SSH and FTP, which are frequently exposed in networked environments.

- FTP-Patator: It is a tool that performs brute force attacks on FTP servers by trying multiple username-password pairs until access is granted. Once compromised, attackers can upload malicious files or exfiltrate sensitive data, especially on poorly configured servers [117].
- SSH-Patator: I targets SSH services similarly, attempting to guess login credentials to gain shell access. This can lead to full system compromise if adequate security measures, such as multi-factor authentication or IP blocking are not in place [116].

3.6.4 Web-Attack

Web-based attacks exploit vulnerabilities in web applications to compromise system integrity, steal data, or gain unauthorized access. These attacks often target login forms, input fields, and poorly validated user inputs.

- Web Attack - Brute Force: It involves systematically guessing login credentials through repeated attempts on web interfaces. Attackers automate this process using tools to bypass account protections and gain access to user accounts.
- Web Attack - XSS (Cross-Site Scripting): It allows attackers to inject malicious scripts into trusted web applications. When executed in a user's browser, these scripts can steal cookies, session tokens, or redirect users to malicious sites [118].
- Web Attack - SQL Injection targets web applications that fail to properly sanitize database queries. By inserting malicious SQL code into input fields, attackers can retrieve, modify, or delete backend database records [119].

3.6.5 Botnet

A botnet or bot is a group of internet-connected devices infected with malware and controlled remotely by an attacker. These compromised devices, called bots, are often used to carry out large-scale attacks such as DDoS, data theft, or spamming without the users' knowledge. Botnets exploit weakly secured systems, especially in IoT environments, to silently perform malicious tasks. Their distributed nature makes detection difficult, as the traffic they generate can resemble legitimate behaviour [120].

3.6.6 Infiltration

Infiltration attacks involve unauthorized access to internal systems by bypassing external defences, often through compromised hosts, backdoors, or remote access tools. Once inside, attackers can monitor activity, extract sensitive data, or prepare for larger attacks. These attacks are stealthy and can go undetected for long periods, making them particularly dangerous in enterprise and IoT networks. Infiltration is typically part of an advanced persistent threat (APT) strategy, where the intruder maintains long-term access for data exfiltration or sabotage [121].

3.6.7 Heartbleed

The Heartbleed attack was caused by a bug in a popular security software called OpenSSL, which is used to protect internet communication. This bug allowed attackers to trick a server into sending small chunks of its memory without needing a password or permission [122]. These memory leaks could include very private data like usernames, passwords, or even the keys used to secure websites.

3.6.8 Portscan

A PortScan attack is like a digital "door check" on a computer or network. Attackers use it to scan multiple ports-communication points on a device-to find out which ones are open and what services are running [123]. Each open port can give clues about weaknesses that might be used to break into the system later.

Although some port scanning can be used for legal network testing, attackers often use it as a first step before launching more harmful attacks. It helps them map out the network and choose the easiest targets [124]. Detecting and blocking unusual scanning behaviour is key to stopping attacks early.

3.6.9 Information Gathering

Information gathering is the early stage of a cyberattack where an attacker collects details about a target system to identify possible weaknesses. Two common methods are service scanning and OS fingerprinting [125].

- Service scanning involves checking which services (like web or mail servers) are running on open ports, helping attackers understand what software is in use.
- OS fingerprinting tries to identify the operating system (Windows, Linux, etc.) based on the system's responses to specific network requests. This information can guide attackers in choosing the most effective exploits [126].

3.6.10 Information Theft

Information theft is a serious type of cyberattack where the main goal is to steal private or sensitive data without the user's knowledge. This stolen information may include login credentials, financial records, business documents, or personal identity details. Two of the most common techniques used in such attacks are keylogging and data theft [127].

- Keylogging is a method where malicious software secretly records everything a user types on their keyboard. This includes usernames, passwords, credit card numbers, and even personal messages. The attacker can then retrieve this information and use it for unauthorized access or financial fraud. Keyloggers are often installed through phishing emails or infected software and can operate in the background without being noticed.
- Data theft, on the other hand, involves the unauthorized copying or transfer of valuable files and information stored on a system or network.

Attackers may gain access through security loopholes, malware, or insider threats, and extract sensitive data such as customer databases, confidential documents, or intellectual property [128]. This can have severe consequences, including identity theft, loss of privacy, legal issues, and damage to an organization's reputation.

3.6.11 User to Root (U2R) attack:

An U2R attack happens when an attacker who already has limited access to a system tries to gain full administrative control, known as "root" access. In simple terms, the attacker starts as a normal user and looks for system vulnerabilities to escalate their privileges and become a superuser with complete authority over the system [129]. In U2R attacks, attackers often exploit software bugs, misconfigurations, or weak access control mechanisms. Techniques include buffer overflow attacks, exploiting hidden commands, or taking advantage of poorly secured system files. Once root access is achieved, attackers can modify system settings, install malware, steal sensitive data, or even take down the system entirely.

U2R attacks are particularly dangerous because they usually leave few signs of intrusion during the early stages. Detecting such attacks requires strong user activity monitoring, behaviour analysis, and regularly updating and patching software to close security gaps.

3.6.12 Probe

A Probe attack is a type of cyberattack where an attacker scans a network or system to discover weak points without immediately causing damage. It is like a burglar checking all the doors and windows before deciding how to break into a house. The goal is to gather information about active machines, open ports, or running services that could later be exploited for a larger attack [130]. Probing usually involves sending specially crafted network requests to different devices and analysing the responses. Common probing tools can detect system types, software versions, and even security flaws. Although some network scanning is legitimate for system maintenance, when done by malicious actors, probe attacks can be the first step in a planned intrusion [131].

CHAPTER 4

MACHINE LEARNING ENSEMBLE METHOD FOR INTRUSION DETECTION SYSTEM

4.1 Problem Statement and Background

Despite the growing deployment of IDS in network security, existing solutions face significant challenges in accurately identifying malicious traffic, especially in the presence of emerging and zero-day attacks. Signature-based IDS are limited to known attack patterns and fail to detect novel threats, while anomaly based IDS, though capable of identifying unknown behaviours, suffer from high FAR and computational inefficiencies. Traditional ML models applied to intrusion detection have shown promise but often fall short due to limitations such as long training times, reduced scalability with high-dimensional data, and poor generalization to diverse attack types. Moreover, many existing models struggle to strike a balance between detection accuracy, FAR, and computational cost. In this context, there is a pressing need for a more effective and lightweight approach that can maintain high detection performance while minimizing resource consumption.

Applying ML to a field requires relevant datasets for effective learning. Among the publicly available intrusion detection datasets, NSL-KDD dataset is considered the benchmark dataset for ID in the last decade [132]. A study focusing on the importance of feature reduction for training an ID model was published in 2012 [133]. Feature-Vitality Based Reduction Method (FVBRM) was used to reduce the NSL-KDD datasets from 41 features to 24. Naïve Bayes Classifiers' mean accuracy was improved from 95.11% to 97.78% when feature reduction was used in contrast to the selection of all attributes of the dataset. However, the U2R attack detection accuracy was low at 64%. The study implies that the selection of reduced features gives a better overall performance in designing an ID system in terms of efficiency and effectiveness.

Latifur et.al, [134] used Clustering Trees based on SVM to reduce the training time of SVM on the DARPA98 dataset. The accuracy rate for normal and Probe was observed as 98% and 88% respectively, but for DoS, U2R, and R2L, the

accuracies were very low at 39%, 23%, and 15% respectively. The Clustering Trees-based SVM however, improve the pure SVM average accuracy from 57.6% to 69.8% as well as training time was reduced from 17.34 hours to 13.18 hours. Yassin et.al, [135] proposed an integrated ML algorithm using the K-Means classifier and Naïve Bayes on KDD CUP '99 dataset. The proposed method is to address the high FAR and to surpass the detection accuracies of the existing ID system. This model significantly improves the accuracy, DR up to 99% and 98.8%, respectively, while also decreasing the FAR to 0.5%.

Kostas [136] used seven ML methods; NB, RF, KNN, MLP, Adaboost, ID3, and Quadratic Discriminant Analysis (QDA) independent to each other in order to compare their performance. For feature reduction, Random Forest Regressor was used on CICIDS 2017 dataset and achieved an F-score as NB: 0.86, QDA: 0.86, RF: 0.94, ID3: 0.95, AdaBoost: 0.94, MLP: 0.83, and KNN: 0.97. The study also shows that the computation time of AdaBoost, MLP, and KNN are extremely long in comparison to NB, QDA, and ID3. Gautam and Amit [137] used Information Gain to select important features on the KDDCUP99 dataset, and an ensemble of NB, Adaptive Boost, and PART (Partial Decision Tree) for classification. The ensemble method improved the classification results of the imbalanced data and thus surpasses all the lone classifiers (NB, Adaptive Boost and PART).

Tama, Comuzzi, & Rhee [138] proposed a two-level ensemble with hybrid FS using three evolutionary algorithms viz. Particle Swarm Optimization, Genetic Algorithm, and Ant Colony Algorithm. A Two-level ensemble using rotation forest and Bagging on NSL-KDD and UNSW-NB15 dataset achieving classification accuracy of 85.8% and 91.27% respectively which surpass the state of art individual and meta-classifier. Yang, Sheng & Wang [139] proposed a parallel Quadratic ensemble method on CICIDS 2017. The authors use Gradient Boosting DT to deal with the spatial data and Gated Recurrent Unit to deal with the temporal data. Combining both the algorithm to make a quadratic ensemble method. The proposed model achieves classification accuracies of 99.9% for different kinds of attacks on the CICIDS 2017 dataset.

4.2 Methodology

The proposed ensemble method is the combination of three ML classification algorithms with the purpose of improving the existing solutions in detection accuracy as well as reducing FAR. Random Forest Regressor is used for FS and then the ensemble ML technique is used for classification. The ensemble technique uses predictions from different models and learns how to best combine these models using some other ML model. In this model, particularly, the blending ensemble technique is used to combine three different ML models to make the ID system classify between attack and benign network traffic data. In this method, CICIDS 2017 and CICIDS 2018 datasets are used for evaluation. The dataset consisting of around 3 million and 6 million traffic data records respectively have been analysed for discrepancies, cleaned, and preprocessed. This preprocessing is done to remove any inconsistent or incomplete records. The architecture of the proposed ID System is as shown in Figure 4.1. The traffic data is initially preprocessed and cleaned, then the entire records labels are converted into 1 (attack) and 0 (benign) labels for classification. Random forest regressor is used to calculate the feature importance and an appropriate number of features were selected based on the weight of the features. This data was used for implementing ML algorithms, a blending ensemble model; NB, QDA, ID3 are chosen for level-1 i.e., base models (Level-1) and RF classifier as Meta-Model (level-2) for final predictions. These algorithms were selected from various traditional ML algorithms based on their respective performance and time taken to train the models. In blending ensemble method, the training data is used for training different types of model (ML algorithms).

Intrusion Detection models especially anomaly-based requires an abundant of data for training the predictive model, consisting of both normal traffic and abnormal traffic (attacks). For this purpose, different datasets have been produced to simulate a real-world scenario of a network, there are multiple available datasets for intrusion detection. For this ensemble method, CIC-IDS2017 dataset and CSE-CIC-IDS2018 dataset were used for training and evaluation as they are the most recent and most realistic datasets for intrusion detection which is available publicly [140]. They also contain newer types of attacks that are not available in the older intrusion detection

datasets. Dataset distribution are shown in Table 4.1. CSE-CIC-IDS2018 dataset is much bigger than CIC-IDS2017 dataset. The traffic flow capture duration is 10 days and 5 days respectively. The number of attack infrastructure and victim infrastructure of the network where the traffic is captured is also much larger in CSE-CIC-IDS2018 dataset. The CICIDS datasets were selected in this study due to their wide variety of attacks and protocol range in comparison to the older and benchmark intrusion detection dataset.

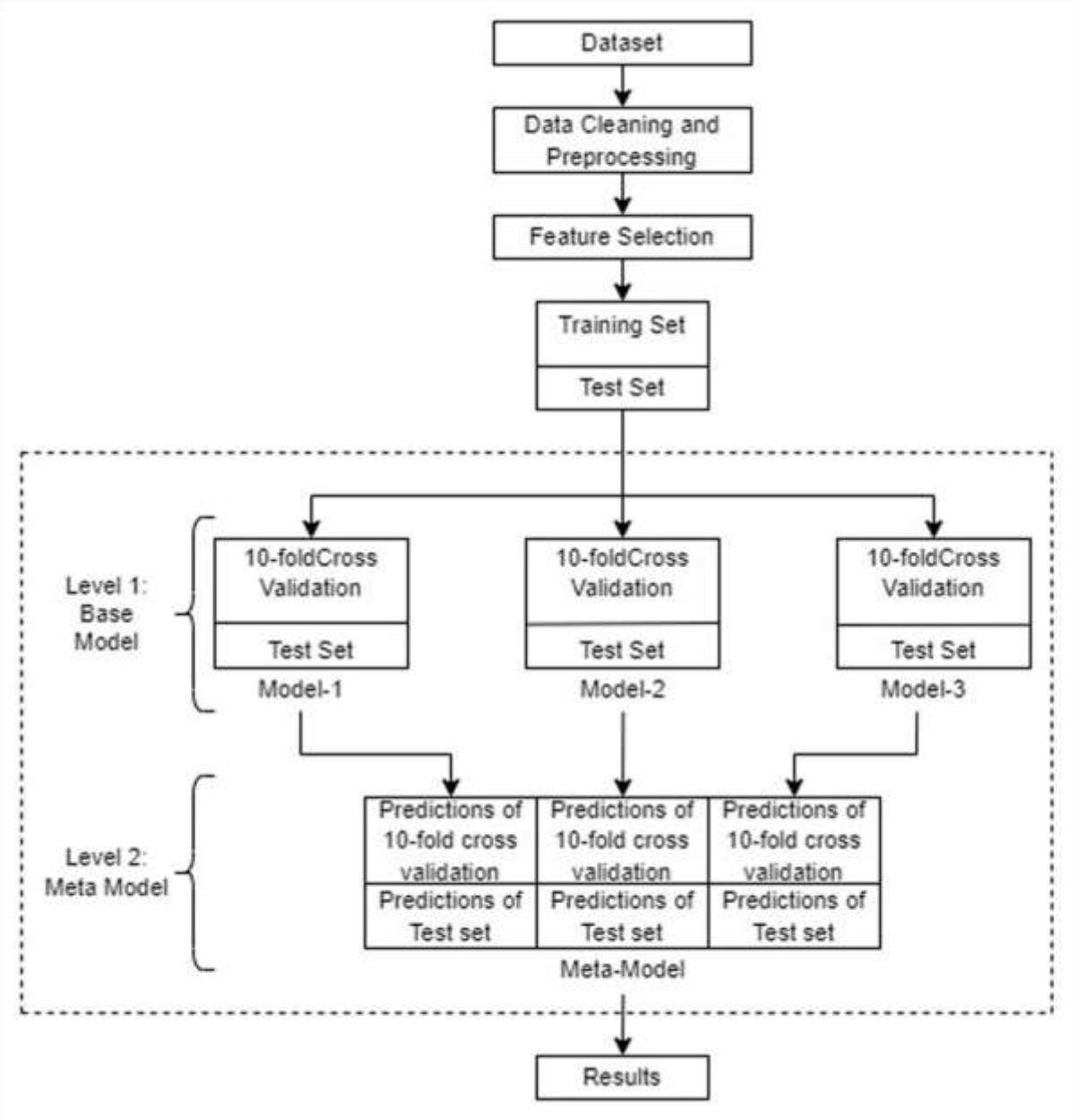

Figure 4.1 Proposed Ensemble Method

Table 4.1 Data Distribution of CIC-IDS2017 and CSE-CIC-IDS2018 dataset

Dataset	**CICIDS 2017**	**CICIDS 2018**
Capture duration	5 days	10 Days
Number of Classes	15	18
Number of features	80	83
Number of records	2,830,743	16,233,022
Number of attacks	471,454(17%)	2,759,364 (17%)
Number of Benign	2,359,289 (83%)	13,473,658 (83%)

4.3 Dataset Preprocessing

The CIC-IDS 2017 dataset consists of 8 CSV files, containing 5 days network traffic stream as shown in Table 4.1. These CSV files were combined, forming a single CSV consisting of 28,30,743 records. On experimentation, there were few incomplete records seen in the dataset, which were removed from the file. It was observed "Fwd Header Length" attribute of the dataset existed twice in the set, which was corrected by removing one copy of the attribute. "Flow Bytes/s", "Flow Packets/s" contained infinity and NaN values (Missing values) which were converted into integer 1 and 0 respectively. It was also seen that multiple attributes contained string values, including the Label column, which was handled by converting all the benign stream records into integer 0 and the remaining attack records into integer 1, to be able to make these records fit for training.

In order to evaluate the performance of the model, the CIC-IDS 2017 dataset is fed to the model to compute the performance of the learning done by the model. The dataset is hence split into two partitions: a training set and a test set. The ML algorithms try to learn from the training set and verified on the test set. However, the CIC-IDS 2017 dataset does not have these pre-defined splits, hence the data is divided into two parts manually using python libraries. Sklearn uses the train_test_split [141] function to split the dataset randomly such that 80% of the data is used for training the model i.e., training set and remaining 20% of data as the test

set. The results obtained over test data is the performance of that model, which is used to calculate the performance metrics for the ML model.

The same method of preparation of dataset which includes data cleaning, converting character data into machine-readable data, and conversion of labelled data into 0 (benign) and 1 (attack) is performed on CSE-CIC-IDS2018 dataset. This dataset consists of 10 CSV files of 10 days network traffic stream, out of which 7 CSV files were combined forming 1 CSV file; 3 CSV files were excluded due to its size and no presence of attack in the file. The combined CSV file contains 6.1 million records which are around 38% of the total records in the CSE-CIC-IDS2018 dataset. It also contains a greater number of sub-types of attacks. The same ratio of train_test_split is used as in CIC-IDS2017 dataset.

4.4 Feature Selection

CIC-IDS2017 dataset consisted of 80 features with 28,30,743 records; CSE-CIC-IDS2018 dataset consisted of 83 features with 162,33,022 records out of which 61,87,054 (38%) records were used for implementation. Using the dataset with all the features to train the ML model is computationally costly and also results in extended training times. FS is hence applied to the dataset in order to determine which features are important to define a traffic record as an attack or benign.

An embedded method of FS: Random Forest Regressor is used in this study. Random Forest Regressor fits plenty of DT on different sub-samples of the dataset and calculate the importance (weight) of each feature in the dataset. Random Forest Regressor uses averaging to control overfitting and also to improve the accuracy. The dataset contains only attack (1) and benign (0) labels; hence the feature importance will be focused only on the importance of classifying between attack or benign data. The features list (top 12) along with their weight importance is tabulated in Table 4.2.

Table 4.2 Top weighted Features and their weight

CICIDS 2017		CICIDS 2018	
Features	**Weight**	**Features**	**Weight**
Bwd Packet Length Std	0.24662	Dst Port	0.275073
Flow bytes/s	0.17877	Flow IAT Max	0.066496
Total Length of Fwd Packets	0.10241	Flow Byts/s	0.048871
Fwd Packet Length Std	0.06388	Flow IAT Mean	0.012347
Flow IAT Std	0.00989	Bwd Pkt Len Min	0.013189
Flow IAT Min	0.00694	Fwd Pkt Len Max	0.006626
Flow IAT Total	0.00512	Bwd Pkt Len Mean	0.005703
Flow Duration	0.00415	Bwd Pkt Len Std	0.003946
Bwd Packet Length Max	0.00400	Fwd Pkt Len Mean	0.003443
Flow IAT Max	0.00357	Tot Bwd Pkts	0.002907
Flow IAT Mean	0.00326	Bwd Pkt Len Max	0.000898
Total Length of Bwd Packets	0.00130	TotLen Bwd Pkts	0.000376

The algorithm for the proposed method is shown in Algorithm 4.1 where the summarization of all the steps of the proposed ensemble method can be pictured.

Algorithm 4.1. Proposed Ensemble Method

Input: Preprocessed dataset *D*

Output: Predicted class labels *Y*

1: Data Preprocessing:

- Clean dataset(s) by removing missing values, duplicates and infinite values
- Encode categorical variables numerically.
- Normalize feature values
- Convert class label to 0 and 1.

2: Feature Selection:

- Apply Random Forest Regressor to compute feature importance scores.
- Retain top-N features.
- Construct a reduced dataset *D`* using the selected features.

3: Dataset splitting:

- Split *D′* into training and testing sets using a desired ratio (80:20)

4. Base Learner Training (Level-1):

- Train Naïve Bayes on the training set and generate predictions P_1.
- Train QDA on the training set and generate predictions P_2.
- Train ID3 on the training set and generate predictions P_3.

5. Meta-Level Data Construction:

- Stack the outputs P_1, P_2, P_3 to form a new training set X_{meta}.
- Use the original training labels Y_{train} as the target variable for meta classifier training.

6. Meta-Classifier Training (Level-2):

- Train a Random Forest classifier using X_{meta} and Y_{train}.

7. Testing Phase:

- Generate predictions Q_1, Q_2, Q_3 on the test set using the trained based classifiers.
- Stack Q_1, Q_2, Q_3 to form test input X_{meta_test}.
- Use the trained Random Forest meta-classifier to produce final predictions Y on X_{meta_test}

8: Performance Evaluation:

- Compute performance metrics such as accuracy, precision, recall, F1-score, and false alarm rate based on Y and the actual test labels.

The proposed method adopts a layered ensemble learning strategy to strengthen intrusion detection by leveraging classifier diversity. After preprocessing and sanitizing the dataset, a RF based importance evaluation is performed to isolate the most relevant features, effectively streamlining the input space. Three base level models; NB, QDA, and ID3 are individually trained using this optimized feature subset. Instead of relying on a single model's judgment, their outputs are merged to form a secondary feature space, which encapsulates multiple decision perspectives. A RF meta learner is then trained on this aggregated output to make the final classification. This hierarchical design enables the model to capture complementary strengths from distinct classifiers, improving its sensitivity to subtle attack patterns while maintaining high overall accuracy. During inference, the same structure is followed to classify unseen traffic. The approach enhances resilience, minimizes both false positives and missed detections, and is evaluated using a comprehensive set of performance metrics.

4.5 Implementation and Experimental Results

The Anomaly-based IDS devised uses the ensemble technique of ML, which is a technique that uses predictions from different models and learns how to best combine these models using some other ML model. This results in improving upon the effectiveness that can be achieved by any of these models independently. In particular, for this study, a blending ensemble technique where different ML models were used to train the training data is used to combine these different ML models to classify between attack and benign network traffic data.

The ML algorithms used in the level-1 are selected on the basis of experimentation performed on various ML classification algorithms; Model-1 is NB, Model-2 is QDA and Model-3 is ID3 (DT). These 3 algorithms are selected based on the results (accuracy & training time) achieved when applying them on the given dataset: NB (F-Measure: 0.86, Time: 1.825s), QDA (F-Measure: 0.86, Time: 2.37s) and ID3 (F-Measure: 0.95, Time: 9.51s) [142].

These three base models are trained through 10-fold cross-validation, and examined over their respective test sets, results of which are stacked to create new training and test set for the level-2 meta-model as shown in Figure 4.1. The meta-model is trained using the RF classifier, and prediction of this meta-model determines the final performance metrics of the system. The set of attributes to be used is selected on an experimental basis, where a suitable set of features is selected for implementation based on the computed feature importance values presented in Table 4.2, along with the variation in F1 scores corresponding to different feature counts, as illustrated in Figure 4.2.

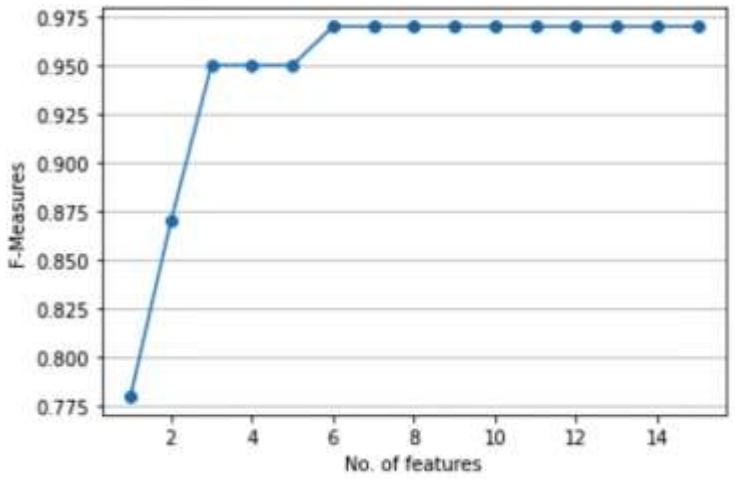

Figure 4.2 Change in F-measure with number of features used

It is evident from Figure 4.2 that there is little to no change in the F-scores after the selection of the 6 most important features from the feature importance shown in Table 4.2. Hence, the top 7 and top 8 features were selected which makes up to roughly 95% of feature importance to get optimal results for the ML implementation of the proposed IDS model. This method of FS removes around 70 features in these CICIDS datasets. The effectiveness of the proposed model is realized through experiments applying on the CIC-IDS2017 and CSE-CIC-IDS2018 datasets via normal and attack classification. The computation time of the models are also recorded as an additional measure. The whole implementation of the proposed method was carried out on a PC with Intel core i5-10th Gen, 8Gb RAM and GPU of MX-350 running windows 10.

The analysis is carried out in accordance with different performance metrics. CIC-IDS2017 dataset as well as CSE-CIC-IDS2018 dataset (but not simultaneously) after data cleaning and preprocessing is split into a training set and test set; the training set is to train the ML model and the test set is to test the performance of the trained model. The confusion matrix for the ensemble model given in Figure 4.3, illustrates the True Positives (TP), False-negatives (FN), False-positives (FP), and True negatives (TN) predicted over the CICIDS 2017 dataset (a and b) and CICIDS 2018 dataset (c and d).

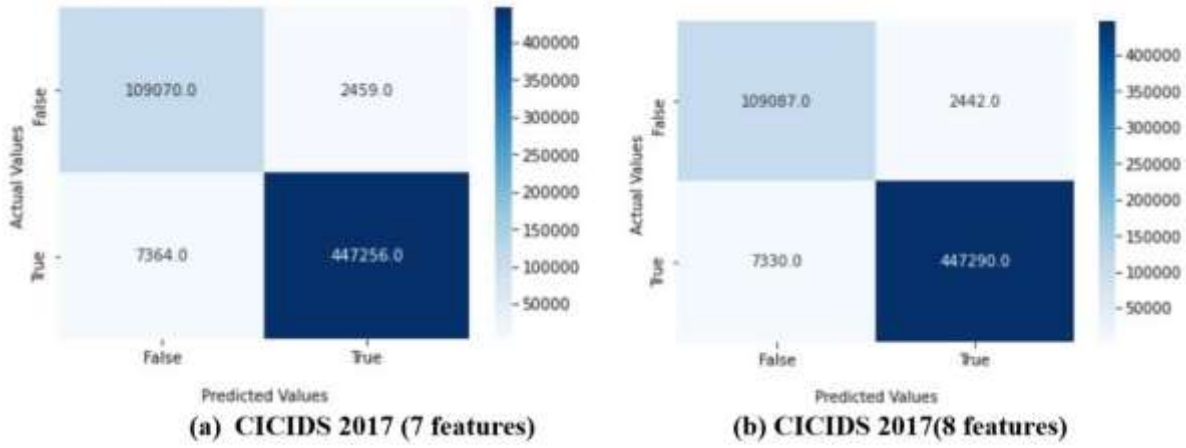

Figure 4.3: Confusion Matrices of CIC-IDS2017 dataset

The confusion matrices shown in Figure 4.3 are CICIDS2017 with 7 features and CICIDS2017 with 8 features. There is little to no change in the results with CICIDS2017 with 8 features slightly better than the other.

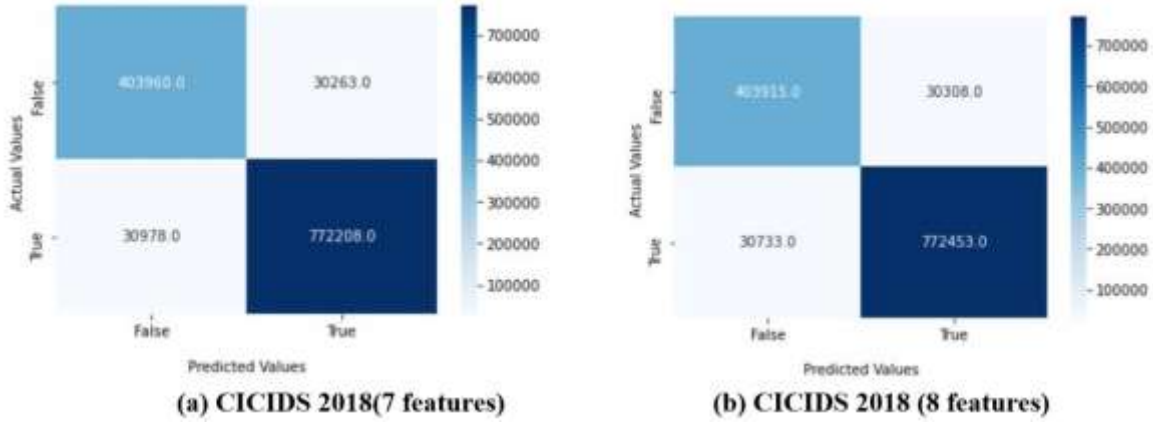

Figure 4.4: Confusion Matrices of CSE-CIC-IDS2018 dataset

It is observed from Table 4.3 and Table 4.4 that the ensemble technique significantly improves the classification results upon the selected individual ML algorithms. It is also evident that addition of one feature i.e. from 7 features to 8 features, does not have a huge impact on the classification results. There is little to no change in the outcome and may not have high positive impact on increasing the number of features. In Table 4.3, the proposed ensemble model has lower computation time than the ID3 while having better results. The overall performance of the ensemble method surpasses the individual ML algorithms, especially in the

FAR. However, in Table 4.4, the FAR for NB and QDA are lower than FAR of the ensemble method even though their overall performances are not good. The proposed model gives an accuracy of 98.3% and 95.1% with low false alarm rate of 2% and 6.9% for CIC-IDS2017 and CSE-CIC-IDS2018 dataset respectively. The training time of the ensemble algorithm is higher than that of the individual algorithms, but this is very less and negligible in comparison to neural networks and other DL models.

Table 4.3 Performance of the proposed ensemble method (CIC-IDS2017)

No of features	**Algorithms**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**False Alarm Rate**	**Training Time (Sec)**
With 7 Features	**Ensemble Method**	**0.983**	**0.966**	**0.981**	**0.973**	**0.022**	**10.15**
	Naïve Bayes	0.820	0.712	0.681	0.693	0.549	1.50
	QDA	0.847	0.779	0.695	0.721	0.555	2.04
	ID3	0.958	0.971	0.896	0.928	0.206	13.13
With 8 features	**Ensemble Method**	**0.983**	**0.966**	**0.981**	**0.973**	**0.021**	**11.21**
	Naïve Bayes	0.817	0.707	0.681	0.692	0.544	1.57
	QDA	0.842	0.759	0.692	0.716	0.556	2.18
	ID3	0.959	0.968	0.901	0.930	0.193	15.86

Table 4.4 Performance of the proposed ensemble method (CIC-IDS2018)

No of features	**Algorithms**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**False Alarm Rate**	**Training Time (Sec)**
With 7 Features	**Ensemble Method**	**0.951**	**0.946**	**0.946**	**0.946**	**0.069**	**37.2**
	Naïve Bayes	0.683	0.745	0.748	0.683	0.036	3.55
	QDA	0.684	0.746	0.748	0.684	0.037	4.33
	ID3	0.944	0.940	0.937	0.938	0.088	28.4
With 8 features	**Ensemble Method**	**0.951**	**0.946**	**0.946**	**0.946**	**0.069**	**37.87**
	Naïve Bayes	0.687	0.747	0.750	0.687	0.037	3.54
	QDA	0.682	0.745	0.747	0.682	0.037	5.16
	ID3	0.948	0.942	0.943	0.943	0.071	31.08

The graphical presentation of the performance evaluation of the proposed method and other individual algorithms can be visualized in Figure 4.5 and Figure 4.6. The proposed ensemble method has the best result in every aspect in CIC-IDS2017 dataset. However, in CIC-IDS2018 dataset, though the proposed method has the best result in overall classification accuracy, it has higher FAR than that of NB and QDA. This is because the Naïve Bayes and QDA algorithm has very less false positive and very high false negative.

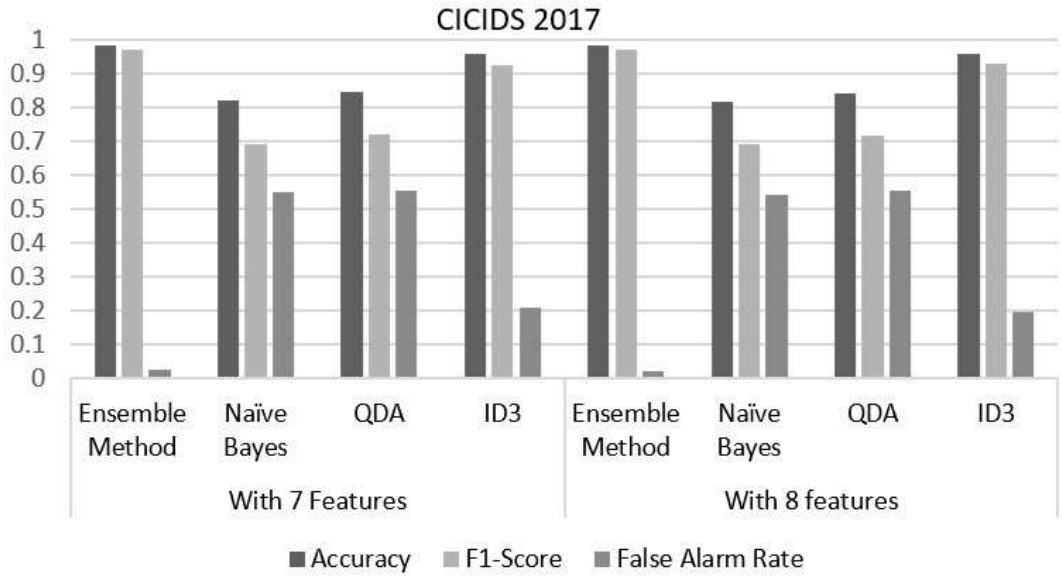

Figure 4.5 Performance evaluations of different algorithms on CIC-IDS2017

Figure 4.5 shows the accuracy, F1-Score and FAR of the proposed ensemble method, NB, QDA and ID3 (DT) for CIC-IDS2017 with 7 features (left) and 8 features (right). It can be seen that the proposed ensemble method out performs the individual classifiers in all the metrics. Similar behavior can seen in Figure 4.6 where evaluation is done using CICIDS2018 dataset. In Figure 4.6, there is a close competition between the ensemble method and the IDS classifier.

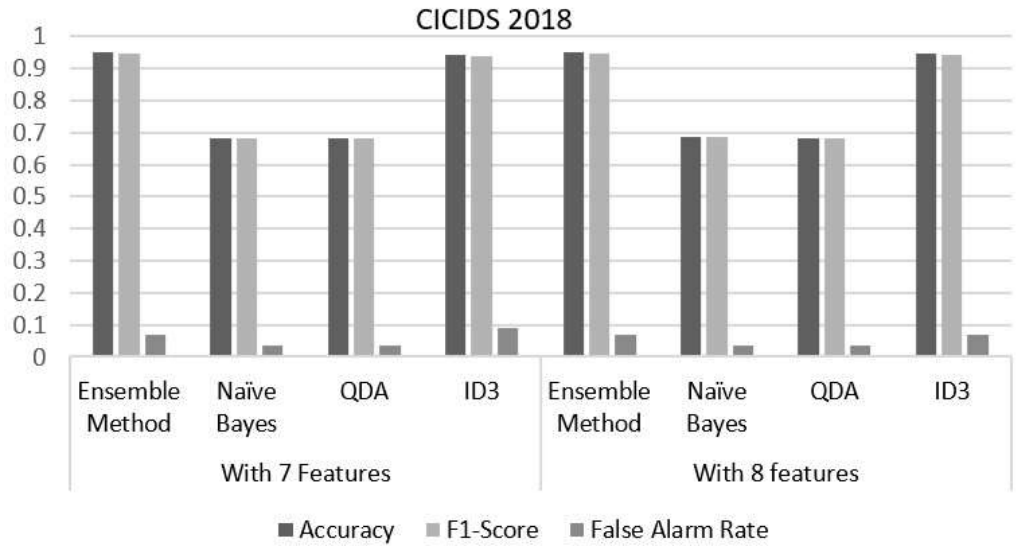

Figure 4.6 Performance evaluation of different algorithms on CIC-IDS2018

The ROC curve plots the TPR and the FPR at different classification threshold. The more the curve bends towards the top left corner, the better is the curve. The ROC curve of the ensemble method on CIC-IDS2017 and CIC-IDS2018 is depicted in Figure 4.7.

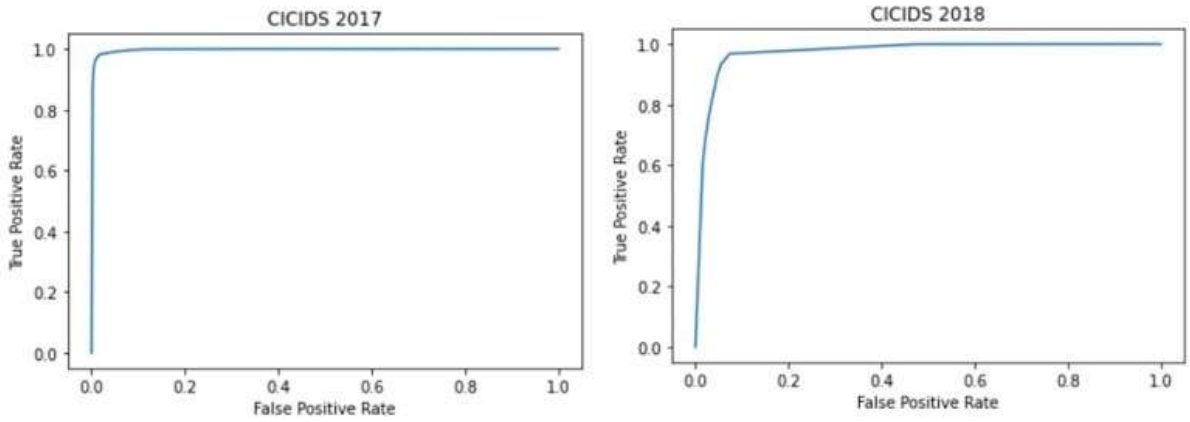

Figure 4.7 ROC curve of ensemble method in CICIDS 2017 (left) and CICIDS 2018 (right)

CIC-IDS2018 consisted of a new and wider range of protocols and attacks. The proposed model is trained and designed for CIC-IDS2017 dataset and CIC-IDS2018 dataset is used for validating the model as CIC-IDS2017 and CIC-IDS2018 dataset have the same source of network traffic. The ability to ensemble multiple ML

algorithms produces more efficient models even with algorithms with weak performance matrices. The comparison of the proposed work with other related works is shown in Table 4.5. Some related works with DL models is not added in this comparative analysis as the computation costs are very high in comparison to traditional ML Algorithms, and one of the objectives of this study is to develop a model with efficient computation cost.

Table 4.5 Comparative analysis of the proposed work with other related works.

Authors and Year	**Research Aspects/Model**	**Datasets used**	**Results**
Latifur Khan, Mamoun, and Bhavani (2007) [134]	Clustering Trees based on SVM	DARPA 98	Average Acc=69.8%
S. Mukherjee and N. Sharma (2012) [133]	Feature-Vitality Based Reduction Method (FVBRM) And Naïve Bayes	NSL-KDD	Acc=97.78%
Yassin, Udzir, Muda, Sulaiman (2013) [135]	K-Means and NB	KDDCUP 99	K Means DR=99% NB DR=98.8%
Gautam and Amit (2018) [137]	Ensemble of Naïve Bayes, Adaptive Boost, and PART	KDDCUP 99	Acc=99.97%
Kostas (2018) [136]	Different traditional Machine Learning Algorithms	CICIDS 2017	KNN Acc=97% (highest)
Tama, Comuzzi, & Rhee (2019) [138]	Ensemble of rotation forest and Bagging	NSL-KDD and UNSW-NB15	NSL-KDD Acc=85.8% UNSW-NB15 Acc=91.27%
Beulah and Punithavathani (2020) [143]	Clustering-based outlier detection (CBOD)	NSL-KDD	Acc=97.96% FAR=1.88%
Our Proposed method	Ensemble of NB, QDA and ID3using Random Forest	CICIDS 2017 and CICIDS 2018	CICIDS 2017 Acc=98.3% CICIDS 2018 Acc=95.1%

4.6 Security Achievements

The proposed blending ensemble based IDS introduces notable security advancements by integrating the predictive strengths of NB, QDA, and ID3 as base classifiers and RF as the meta classifier. This architecture enhances the model's ability to detect a broad spectrum of threats in the CIC-IDS2017 and CSE-CIC-IDS2018 datasets while maintaining high accuracy and low false alarm rates. The system's ability to balance detection effectiveness with model simplicity is the key contribution. By selecting only the most relevant features using RF importance scores, the model reduces complexity and training overhead, yet achieves impressive results of 98% accuracy on CIC-IDS2017 with just a 2% false alarm rate, and similarly strong performance on CIC-IDS2018 despite its increased data diversity. These metrics reflect the model's capability to detect both prevalent and subtle attack patterns, minimizing both false negatives and false positives which is critical for any reliable IDS.

The model's evaluation across different attack types shows it performs consistently across categories, rather than favouring high frequency attacks only. This indicates its utility in environments where attack behaviour is dynamic and unbalanced. Unlike complex deep learning models, the system maintains a manageable training time and can be deployed in constrained environments such as edge-based or small enterprise networks.

4.7 Summary

In this study, it was aimed to detect anomalies on a network by creating an IDS model using ML techniques, addressing the high FAR, computation time and improving the classification accuracy. For this, an anomaly-based network IDS model was developed using ensemble techniques of ML. For the implementation, three ML algorithms were used as base models, Naïve Bayes, Quadratic Discriminant Analysis (QDA) and Iterative Dichotomiser 3 (ID3) which were combined using another ML algorithm; Random Forest Classifier. The training and testing of the model have been carried out using the CIC-IDS 2017 dataset. Feature selection has been done using Random Forest Regressor; this is to reduce the

computation cost and also removal of features with less importance. The same proposed method was implemented on CIC-IDS2018 dataset for validation and the results proves to be positive.

The proposed ensemble model outperforms the individual ML algorithms in classification accuracy and FAR, with little to no increase in computational cost. Individually, NB and QDA algorithm does not have high classification results on the CIC-IDS datasets, but when these algorithms are blended with ID3, the outcome of the combined algorithms surpass all the individual classification results. FS is done using the weight importance of the features evaluated using Random Forest Regressor. Out of around 80 features, 7 features with highest weights are selected for implementation as their weight consist of around 95% of the total weight. This reduction in features reduces the computational cost to around one tenth of the computational cost with full feature.

The future work may include feature subset selection, which will search the best subset of features, this will result in higher overall classification accuracy, which will also have a significant decrease in computational cost. The detection of particular attack by training the model for only one type of attack at a time will give more detailed information for that particular attack type.

How can ensemble learning models be made more explainable and transparent for real time intrusion detection in critical systems such as healthcare and banking systems? Such systems demand high detection accuracy as well as transparency and interpretability in their security mechanisms. In such domains, false positives may disrupt vital services. To address this requirement, the proposed ensemble based IDS can be made more explainable and transparent through a combination of model, feature, and decision level interpretability strategies. Feature importance visualization (Table 4.2) offers a direct and readable explanation of which traffic attributes contribute most to the detection decision. This provides domain experts with insight into the behavioural indicators of malicious traffic. Methods such as SHAP can also be integrated to describe individual predictions in real time. It assign contribution scores to each feature for a given alert, helping

analysts in verifying if the detection aligns with legitimate network behaviour. The ensemble structure itself enhances interpretability by combining relatively simple and transparent base models with a RF meta learner, each decision path remains traceable. The ID3 component provides explicit decision rules, while NB and QDA contribute probabilistic reasoning. To maintain transparency in real time deployment, every prediction made by the system can be stored with its confidence score, the key features that influenced it, and a timestamp. These stored records make it possible to review past alerts, check whether the system made the right decisions, and ensure it continues to perform reliably over time.

CHAPTER 5

HYBRID SAMPLING METHOD FOR INTRUSION DETECTION SYSTEM

5.1 Problem Statement and Background

The rapid growth of the IoT has introduced an unprecedented level of interconnectivity between physical devices, enabling real-time data exchange and intelligent automation across diverse domains such as healthcare, transportation, manufacturing, and smart cities. While this technological evolution has delivered transformative benefits, it has also significantly expanded the surface area for potential cyberattacks. Due to their typically lightweight architecture and resource constraints, IoT devices often lack robust built-in security mechanisms, making them attractive targets for malicious actors. Consequently, securing IoT networks has emerged as a critical challenge, and IDS play an essential role in identifying and mitigating security breaches within such environments.

ML-based IDS have gained widespread adoption for their ability to analyse patterns in network traffic and detect anomalies without the need for explicit attack signatures. However, one of the most significant obstacles to the effective deployment of ML-based IDS in IoT networks is the severe class imbalance inherent in real-world intrusion detection datasets. In many intrusion detection datasets, benign (normal) traffic constitutes very less portion of the total records. This disproportionate distribution causes traditional ML algorithms to become biased towards the majority class, typically various forms of attack traffic, leading to poor performance in detecting minority class instances. As a result, false negatives (undetected attacks) become alarmingly high, undermining the reliability of the IDS.

Existing data balancing methods, including oversampling techniques such as the SMOTE and undersampling approaches like random undersampling or Near Miss, have shown potential in addressing this problem. Nevertheless, these techniques come with notable limitations. Oversampling can cause overfitting by generating synthetic examples that closely resemble existing minority instances, while undersampling may discard valuable information from the majority class, thereby reducing the classifier's ability to generalize across different types of attacks.

Given these challenges, there is a compelling need for a robust and scalable data balancing strategy that can effectively handle extremely skewed class distributions without compromising the integrity of the dataset. Such a strategy should preserve essential patterns in both benign and malicious traffic, ensure sufficient variability for model learning, and facilitate the development of IDS models that perform well across multiple attack scenarios.

The application of ML in the context of IDS within the realm of the IoT has been ongoing for several decades, with a plethora of research endeavours dedicated to this domain. Baich, et al. [144] present a state-of-the-art study on IoT-NIDS utilizing ML techniques with the objective of identifying the most effective and commonly used ML approaches. Experiments were conducted to compare ML techniques using the NSL-KDD dataset. Following preprocessing, different FS techniques were applied. The models employed in this study include DT, RF, NB, and SVM. RF algorithm demonstrated the highest efficiency, with 99.13% accuracy. Mahmoud et al. [145] introduce a novel ML approach named "AE-LSTM" for intrusion detection, employing a 6-layer Autoencoder (AE) with LSTM, proving to be highly effective in identifying anomalies. AE-LSTM optimally utilizes the finest reconstruction function, one of the main areas where the differentiation between normal network traffic and other abnormal activity is essential is based on the use of the "NSL-KDD test dataset" for evaluation, in which the performance of the proposed AE-LSTM is significantly higher compared to the use of the traditional methods and achieving the micro and weighted F1-score of 98.69% and 98.70% respectively. Liu and Du [146] discuss FS using the GA applied to detect botnet using the Bot-IoT dataset. This approach is able to achieve the intended aim of reducing the number of features from 40 to 6 in order to achieve a detection accuracy of 99.98% as well as an F1-score of 99.63%, the method outperforms alternatives in training time and accuracy. However, limitations include potential accuracy variations due to diverse factors and the impact of different FS processes and classification models on final performance. Even though FS reduces dataset complexity and improves the performance of the ML model, the risk of overfitting remains high due to extreme class imbalance. Hnamte et al. [147] proposed a DNN

based method for detection and mitigation of DDoS using the InSDN dataset and CICIDS2018 dataset achieving 99.98% and 100% accuracy with low loss rate, their results highlight the potential of DNN particularly for the detection of DDoS attack.

Fernando et al. [148] evaluates the performance of unsupervised ML algorithms for unbalanced data using the BoT-IoT dataset. The authors analyze K-means++, DBSCAN, Local Outlier Factor (LOF), and Isolation Forest (I-forest) based on metrics like purity, homogeneity, and adjusted mutual information. K-means++ achieves 95% purity, while I-forest demonstrates best efficiency, using only 10% of CPU resources compared to 16% for other algorithms. The study highlights the potential of unsupervised learning for intrusion detection in resource-constrained environments, advocating for future research in hybrid approaches and real-world validation. Tsogbaatar et al. [149] have presented DeL-IoT, a reliable deep ensemble learning model implemented for IoT networks based on SDN for the detection of anomalous behaviors and the prediction of occurrences. The DeL-IoT framework is composed of three main components: it includes anomalies detection, an intelligent and efficient traffic flow management, and identifying the future condition of devices or Systems. The experiments on testbed and benchmark dataset manifest that this approach gets around 3% more accuracy as compared to isolated models, even within a 1% imbalanced dataset. Notably, their model exhibits superior performance compared to relevant competing models showcased in the existing literature. Other datasets that are commonly used for IDS solution include NSL-KDD, UNSW-NB15, CICIDS2017 [150]. Though these datasets do not represent a pure IoT environment, they share similarities in data types and patterns with datasets specifically designed for IDS in IoT networks.

Alosaimi and Saad [151] proposes a new approach involving DL with three levels has been developed to quickly identify attacks on IoT networks. It also shows detections performance improvement by using the proposed method on the Bot-IoT dataset in comparison to prior approaches, with potential extensions to enhance security across various IoT applications. The research focuses on discovering IoT network attacks using ML techniques, leveraging the Bot-IoT dataset for its attack diversity and network protocols. Two main approaches are employed: data size

reduction and addressing data imbalance using SMOTE. 100% accuracy is achieved using the Ensemble bag algorithm with the DT algorithm closely behind at 99.9%.

Numerous researchers have employed various Intrusion Detection Datasets to evaluate IDS in IoT networks. While these datasets encompass a wide array of attacks, including those pertinent to IoT networks, utilizing datasets specifically tailored to IoT environments would enhance the reliability of evaluations. Leevy et al. [152] applied elementary ML techniques to the Bot-IoT dataset, while also Ibrahimi and Hafsa [153] employed conventional ML techniques on the same dataset, employing both binary and multi-class classifications. Both studies assert enhancements in detection accuracy and other performance metrics as a result of their methodologies. Aruna and Karthik [154] introduced a hybrid sampling and anomaly detection method for predicting diabetes. The combination of SMOTE oversampling and ENN undersampling has proven effective in enhancing prediction accuracy.

5.2 Methodology

An IDS model utilizing ML specifically designed for IoT environments is introduced. This model incorporates a hybrid sampling method and binary classification techniques. Figure 5.1 shows the flow of the proposed method.

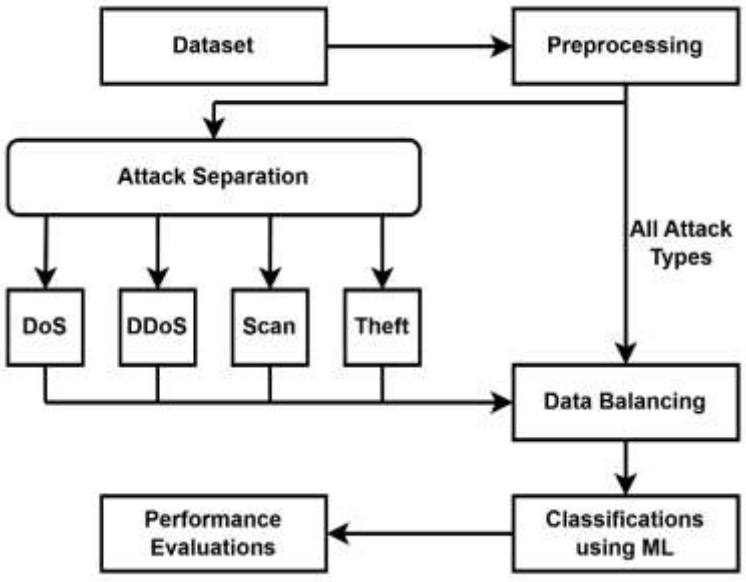

Figure 5.1 Proposed Method

In this proposed method, we utilize the Bot-IoT dataset for evaluation due to its representation nature of IoT environments. However, this dataset presents a significant challenge: an extreme class imbalance, where majority of the class vastly outnumbers the minority class which is likely to lead the decisions made by the classification algorithm unreliable. To address this, we implement a hybrid sampling method using Near Miss-1 for undersampling the majority class and SMOTE for oversampling the minority class, thereby balancing the dataset to approximately an 80:20 ratio, where 80% represents attacks and 20% represents benign instances. The Bot-IoT dataset contains four different types of attacks, for each of which a subset of the dataset is created as shown in Figure 5.1. These subsets of dataset also undergo the data balancing process with the same ratio. Binary classification is then performed on each subset of data as well as the full dataset using selected ML algorithms. The 80:20 ratio was chosen for preserving the majority of attack instances while significantly increasing the representation of normal instances. This ratio is sufficient for effective learning of both classes without overfitting to the minority class or losing important information from the majority class. Alternative ratios were explored, but 80:20 provided the best trade-off between class balance and maintaining dataset integrity.

5.3 Dataset Preprocessing

The Bot-IoT dataset is specifically designed to simulate attacks on IoT devices. It includes wide range of attacks and more realistic anomalies that are specific to IoT. The original PCAP file is around 70GB in size. In the Bot-IoT dataset repository, there are 74 CSV files containing all the records and attributes, totalling approximately 17GB in size. There are more than 73 million records with 35 attributes in the CSV files. Additionally, there is a 5% dataset comprising 5% of each class label. This 5% dataset is mostly used by researchers due to its smaller size. Table 5.1 shows the numerical detail of the Bot-IoT dataset.

Table 5.1 Numerical Analysis of Bot-IoT dataset

Category	**Subcategory**	**No. of records**
Benign	-	9543
Information Gathering (Scan)	Service Scanning OS Fingerprint	1463364 358275
DDoS	DDoS TCP DDoS UDP DDoS HTTP	19547603 8965106 19771
DoS	DoS TCP DoS UDP DoS HTTP	12315997 20659491 29706
Information Theft	Key logging Data Theft	1469 118

The number of "Benign" instances is extremely small compared to the number of "Attack" instances. Benign instances comprise only around 0.013% of the entire dataset, indicating a significant imbalance. The 35 features of the Bot-IoT dataset contributes uniquely to the identification of specific attack types. For instance, features such as "pkts" and "bytes" are particularly useful for detecting volumetric attacks like DDoS and DoS, as they reflect the volume of traffic. Similarly, the "proto" feature, which indicates the protocol type (e.g., TCP, UDP), helps distinguish between different attack vectors. Features like "state" and "dur" provide insights into the connection state and duration, which are critical for identifying anomalies in network behavior. By leveraging these features, the proposed IDS can effectively differentiate between normal traffic and various attack types. Either "attack", "category" or "subcategory" can be used as the class label, and in this study, we are using "attack" as the class label as it contains the information of the records being an "attack" or "benign". After studying the features and their corresponding descriptions, it is evident that some features are derived from others. Additionally, certain features consist of sequential numbers and addresses, which are unlikely to contribute to the classification results. The 74 CSV files of Bot-IoT dataset is combined into a single CSV file to make it more suitable for further processing. Data preprocessing is done as follows:

- *Handling Missing Values*: The features “sport” and “dport” where feature “proto” (Transaction protocol in network flow) is “arp” are all empty. These empty values are filled with ’0’. There are also a few missing values that are removed from the dataset.
- *Data Cleaning*: Some instances exhibit peculiar characteristics compared to others, such as the presence of the feature “proto” being “ipv6-icmp”; or the feature state being “CLO”, “RSP”, “PAR” and so forth. These instances have low occurrences but have the potential to negatively impact the classification results of ML algorithms and are consequently removed from the dataset. In addition, three instances with characteristic data type are identified within the numerical type feature “dport”. These instances are also removed from the dataset.
- *Encoding and Normalization*: Features like “proto”, “state”, and “flgs” are in nominal categorical format. Onehot encoding [155] is applied to the instances of these features to ensure suitability for ML classifiers. The entire dataset is then normalized [156] to a common scale so that the ML model can evaluate the data effectively, without being biased by differences in feature scales.
- *Data separation*: The dataset comprises 4 types of attacks which can further be categorized into 10 different types of sub-attacks. A subset of the dataset is created for each type of attack along with the normal (benign) instances in the ratio 80:20. To achieve this ratio, the proposed hybrid sampling method is used. In addition to these subsets, the main (full) dataset is also resampled in the ratio 80:20.
- *Feature cleaning*: The features “smac”, “dmac”, “soui”, “doui”, “sco”, and “dco” are all empty features and thus are removed. The features “pkSeqID” and “seq” represent the sequence numbers of the rows and flows, the features “saddr” and “daddr” represent the address of the machines, and lastly the features “category” and “subcategory” represent the attack types. It is imperative not to include these features in the model as they may inadvertently reveal class label information due to the sequential arrangement of classes in the dataset

5.4 Hybrid Sampling Method

The Bot-loT dataset initially contains 35 attributes and 73,370,443 records, with 9,543 records labeled as normal and the remainder as attacks. Following pre-processing, the dataset comprises 37 attributes and 73,359,273 records, with 9,385 normal records. This step ensures that redundant and irrelevant features are removed while preserving relevant information. Denote the initial dataset as $D = \{(x_i, y_i)\}_{i=1}^{n}$, where x_i the feature vector, and $y_i \in \{0,1\}$ represents the class label (0 for normal, 1 for attack).

While numerous studies focus on botnet detection models, only a few integrate feature engineering to mitigate issues like duplication and multicollinearity in large datasets. Multicollinearity can be described as a condition where one feature is highly linearly dependent on another. Mathematically, for two features x_j and x_k , multicollinearity exists if:

$$\text{corr}(x_j, x_k) \approx 1 \text{ or } \beta_j x_j + \beta_k x_k + \epsilon = 0 \quad (8)$$

Where $\text{corr}(x_j, x_k)$ is the Pearson correlation coefficient between feature j and k, β_j and β_k are coefficients, and ϵ is an error terms.

Failure to handle such correlations will lead to overfitting [157], where the model performs correct on the training data but not on unseen data. Overfitting is mathematically represented by minimizing the empirical risk R_{emp} on the training data but leading to poor generalization error R_{gen}:

$$R_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n} L(f(x_i), y_i) \text{ but } R_{gen}(f) > R_{emp}(f) \quad (9)$$

Where $L(f(x_i), y_i)$ is the loss function used to measure the error between the predicted value $f(x_i)$ and the true value y_i.

Class Imbalance in the Bot-IoT dataset is another significant challenge, with a skewed distribution between normal and attack records. Researchers have employed various methods to tackle the class imbalance inherent in the Bot-IoT dataset. In this study, we propose a hybrid sampling method that sequentially utilizes

Near Miss-1 undersampling and SMOTE oversampling technique on the dataset, as shown in Figure 5.2.

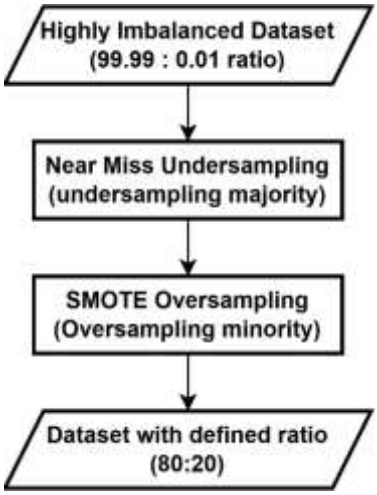

Figure 5.2 Hybrid Sampling Method

In this hybrid sampling method, Near Miss-1 and SMOTE are performed sequentially to achieve a more balanced dataset while also preserving the quality of the data. The Near Miss-1 algorithm basically finds samples belonging to the majority class most similar to the ones in the minority class. The selection is purely based on the extending of majority class examples which are nearest to the minority class in order to retain the informational nature of data. This approach ensures that important patterns and characteristics present in the majority class are not lost during the undersampling process. The algorithm for Near Miss-1 is shown in Algorithm 5.1.

Algorithm 5.1. Near Miss-1 Undersampling

Input: Imbalanced dataset (X,y)

Output: Resampled dataset ($X_{resampled}$, $y_{resampled}$)

1: Identify Minority and Majority Class instances

- Extract Minority class instances $X_{minority}$ and majority class instances $X_{majority}$
- Extract corresponding labels $y_{minority}$ and $y_{majority}$

2: Compute Nearest Neighbors for each minority class instances $x_m \in X_{minority}$

- For each x_m, calculate distance $D(x_m,x_{Mj})=||x_m-x_{Mj}||$ for every $x_{Mj} \in X_{majority}$

3: Identify k-Nearest Neighbor

- For each minority class instance x_m, identify the set of k-Nearest Neighbor $N_k(x_m)$ from majority class based on computed distances
- $N_k(x_m)=\{x_{Mj} \in X_{majority} \mid rank(D(x_m,x_{Mj})) \leq k$

4: Select and Combine

- Initialize $X_{resampled}=X_{minority}$ and $y_{resampled}=y_{minority}$
- For each $x_m \in x_{minority}$, add x_m and its k-nearest neighbor $N_k(x_m)$ to $X_{resampled}$
- And the corresponding labes to $y_{resampled}$

5: Combine all selected instances and their corresponding labels to form the final resampled dataset ($X_{resampled}$,$y_{resampled}$)

After applying Near Miss-1 technique, the overwhelming majority class is greatly reduced and thus the dataset become more balanced than before, but is still highly imbalanced for ML techniques. To further balance the dataset, SMOTE is applied to the minority class which will enhance the presentation of the minority class. The algorithm for SMOTE oversampling is shown in Algorithm 5.2.

Algorithm 5.2. SMOTE Oversampling

Input: Resampled dataset ($X_{resampled}$, $Y_{resampled}$)

Output: Augmented dataset ($X_{augmented}$,$Y_{augmented}$)

1: Initialize empty arrays ($X_{augmented}$,$Y_{augmented}$)

2: Identify minority class instances $X_{minority}$ and their corresponding label $Y_{minority}$

3: For each minority instances (X_i, Y_i) $\in$ ($X_{minority}$,$Y_{minority}$)

Find the "k nearest neighbors" of X_i in the minority class

4: Generate synthetic samples:

For j=1 to N

- Randomly selects one of the neighbors of X_i
- Generate a synthetic instances $X_{synthetic}$ by interpolating between X_i and its selected neighbor
- Append $X_{synthetic}$ to $X_{augmented}$
- Append Y_i to $Y_{augmented}$

5: Concatenate the input dataset ($X_{resampled}$, $Y_{resampled}$) with the generated dataset($X_{augmented}$,$Y_{augmented}$) and store in ($X_{augmented}$,$Y_{augmented}$)

The hybrid approach of applying Near Miss-1 and SMOTE sequentially ensures that the dataset is now more balanced and also that the important patterns in the majority class are not lost during undersampling, while SMOTE enriches the

minority class with synthetic samples. The resulting resampled dataset is then used for training ML classifiers, ensuring more accurate and reliable classification results.

The hybrid sampling method is applied to the full dataset and its subsets. The resulting numbers are presented in Table 5.2. With this hybrid sampling method, the ratio of the dataset, which originally is worse than 99:1 becomes approximately 80:20, and this ratio is applied to all the subsets. Prior to sampling, each subset contains 9385 instances labeled as 'Normal' (denoted as '0'), as the occurrence of normal behavior is relatively low across all subsets compared to attack types, except for the 'Theft' attack.

The 80:20 ratio as a figure may still suggest that one class is significantly more prevalent than the other, which in turn leads to a situation whereby the majority class dominates the minority class. Nevertheless, the ratio of 80:20 is used here that sedulously ensures the quality of the majority class as well as the minority class. The overuse of resampling techniques on the data comes with the risk for the quality of data. Table 5.2 illustrate the figure of the number of records for the dataset and the subsets before and after applying the proposed hybrid sampling method.

Table 5.2 Results of Hybrid Sampling

Dataset and subsets	**Before Resampling**	**After Resampling**
Full Dataset	1=7,33,59,273 0=9,385	1=7,33,592 0=1,83,398
DDoS subset	1=3,85,32,478 0=9,385	1=3,85,324 0=93,641
DoS subset	1=3,30,05,192 0=9,385	1=3,30,051 0=93,641
Scan subset	1=18,20,018 0=9,385	1=1,82,001 0=45,500
Theft subset	1=1,585 0=9,385	1=2,000 0=8,000

5.5 Implementation and Experimental Results

Experiments were conducted on a system with Intel i7 12th gen processor, 32GB of RAM, and NVIDIA GeForce RTX 3060 Ti GPU. The computational tasks were executed on this high-performance hardware configuration to ensure efficient processing and reliable results. The experiments were implemented and executed within the Python Jupyter Notebook environment, utilizing Python libraries such as scikit-learn [141], imbalanced-learn [141], and seaborn (sns) [158].

Hyperparameter tuning was conducted to optimize the performance of the ML algorithms. For RF, the number of trees (n_estimators) was tuned within the range of 50 to 400, and the maximum depth of the trees (max_depth) and minimum split (min_samples_split) are put to default to give maximum flexibility which are 0 and 2 respectively. The optimal configuration was found to be n_estimators=100, which also provided the best balance between accuracy and computational efficiency. There is little to no change with this tuning of parameter. For KNN, the value of k (number of neighbors) was tested in the range of 3 to 13, with k=3 yielding the highest accuracy. Logistic Regression was tuned using L2 regularization, with the regularization strength (C) tested in the range of 0.01 to 10, and the optimal value was found to be C=10. And lastly for NB, parameter tuning was done on var_smoothing parameter which control the amount of variance added to each feature to prevent numerical instabilities. Var_smoothing was explored over log range from 10-10 to 100. The best var_smoothing value is 10-7. All these hyperparameters were selected using a grid search approach with 5-fold cross-validation to ensure robust performance.

The ML classifiers are evaluated using specific performance metrics on the Full Dataset followed by the DoS Subset, DDoS Subset, Scan Subset, and the Theft Subset in coming Tables. These datasets undergo preprocessing and are balanced using the hybrid sampling method to achieve an approximate ratio of 80:20 between the classes. The evaluation results of Full Dataset after resampling is tabulated in Table 5.3.

Table 5.3 Evaluation results on Full dataset

Classifiers	Acc.	FPR	FNR	Precision	Recall
LR	99.48	1.05	0.38	99.73	99.61
NB	91.80	40.04	0.25	90.89	99.74
KNN	99.96	0.06	0.02	99.97	99.96
RF	99.98	0.02	0.01	99.99	99.98

The data distribution, showcased in Table 4, highlights the distribution before and after sampling. All the ML classifiers were evaluated on the dataset after sampling. In the evaluation of Full dataset after resampling, RF outperforms LR, NB and KNN in all the defined performance metrics. Though FPR is mainly focused by researchers, FNR is also equally important as False Negative might lead to the trespassing of networking attacks into the system without being noticed by the IDS or the system.

Table 5.4 Evaluation Results on DoS subset

Classifiers	Acc.	FPR	FNR	Precision	Recall
LR	99.91	0.26	0.03	99.92	99.96
NB	99.89	0.09	1.07	99.97	99.89
KNN	99.98	0.05	0.009	99.98	99.99
RF	99.98	0.03	0.006	99.99	99.99

Table 5.4 also demonstrates the superiority of RF over other selected classifiers in terms of FPR and FNR. While KNN provides strong competition to RF in the DoS subset, LR and NB do not perform well in comparison to KNN and RF.

Table 5.5 Evaluation Results on DDoS subset

Classifiers	Acc.	FPR	FNR	Precision	Recall
LR	99.87	0.54	0.01	99.86	99.98
NB	99.61	0.17	0.44	99.95	99.55
KNN	99.97	0.07	0.006	99.98	99.99
RF	99.98	0.04	0.002	99.98	99.99

Table 5.6 Evaluation results on Scan subset

Classifiers	Acc.	FPR	FNR	Precision	Recall
LR	98.32	1.30	1.77	99.67	98.22
NB	90.67	42.41	1.11	90.37	98.88
KNN	99.83	0.42	0.09	99.89	99.90
RF	99.98	0.04	0.005	99.98	99.99

The evaluation of DDoS subset and Scan subset is shown in Table 5.5 and Table 5.6 respectively. RF gives superior performance in all metrics.

Table 5.7 Evaluation Results on Theft subset

Classifiers	Acc.	FPR	FNR	Precision	Recall
LR	99.90	0.06	0.25	99.74	99.74
NB	99.90	0.06	0.25	99.74	99.74
KNN	99.75	0.06	1.03	99.73	98.96
RF	99.85	0.06	0.51	99.74	99.48

In Table 5.7, on the Theft subset, which is notably smaller than the other subsets and the Full dataset, LR and NB demonstrate significantly higher results compared to RF and KNN, which shows that LR and NB performs well on small dataset.

The superior performance of RF across most subsets can be attributed to its ensemble nature, which allows it to capture complex patterns in the data while being robust to noise. However, the strong performance of LR and NB on the Theft subset suggests that simpler models may be more effective for smaller, more balanced datasets where overfitting is a greater risk. The FPR and FNR chart for the Full dataset and subsets using RF is shown in Figure 5.3. It is evident from Figure 6.3 that the Theft subset has high FNR while values of all other FPR and FNR are significantly low.

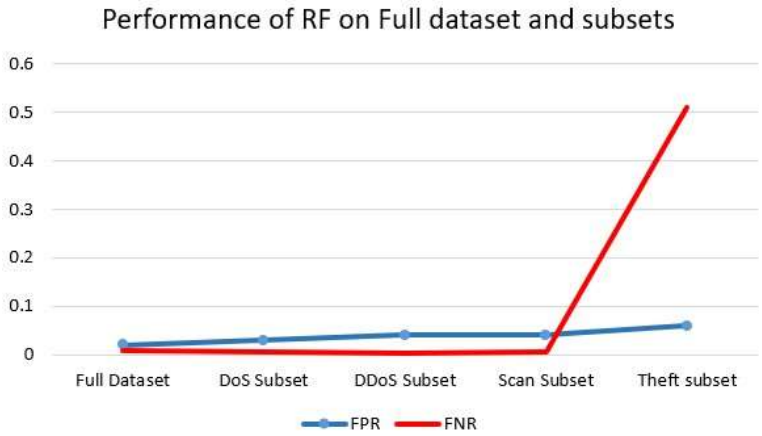

Figure 5.3 RF Performance on all datasets (FPR and FNR)

Figure 5.3 shows the best performing classifier (RF) in FPR and FNR on all the datasets. There is little change of FPR to different subsets while there is a big change on FNR for Theft subset where the FNR is 0.51%.

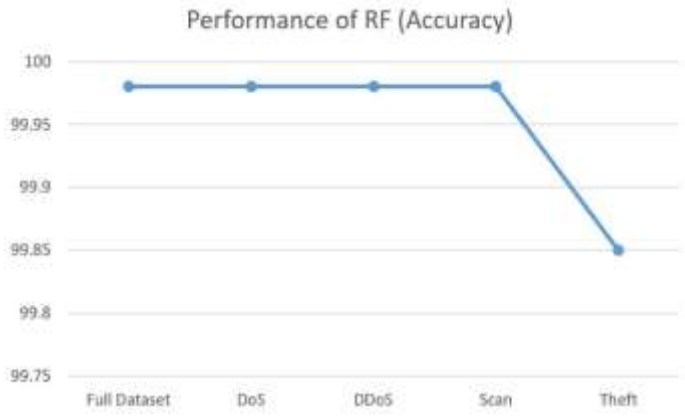

Figure 5.4 . RF performance on all datasets (Accuracy)

Figure 5.4 shows accuracy of the best performing classifier (RF) for all the datasets and subsets. There is little to no change of accuracy for the Full dataset, DoS subset, DDoS subset and Scan subset. There is a big decline on Theft subset where the accuracy is 99.85%.

Figure 5.5 RF performance on all datasets (Precision and Recall)

Figure 5.5 shows the best performing classifier (RF) in Precision and Recall for all the datasets. There is little change of both the Precision and Recall for the Full dataset, DoS subset, DDoS subset and Scan subset. Again, for the Theft subset, there is a great decline in both the Precision and Recall where Precision is 99.73% and Recall is 98.96%.

In our evaluation, RF outperforms LR, NB, and KNN in terms of FPR, FNR, Precision, and Recall. This may be attributed to RF's robustness to imbalanced datasets. While KNN also yields decent results across the entire datasets, LR and NB do not perform as well as RF and KNN, except for the Theft subset. This discrepancy may be due to their lesser effectiveness with large datasets.

To provide a clearer visualization of how the model classifies the data in the test set, a confusion matrix is generated for RF classifier on the Full dataset. The confusion matrix displays the performance of the model by providing the counts of TP, TN, FP, FN values. The confusion matrix (Figure 5.6) shows the actual number of instances of TP, TN, FP and FN along with their percentage. There are 9 instances of FP and 17 instances of FN which is very low compared to the number of total instances on the test set.

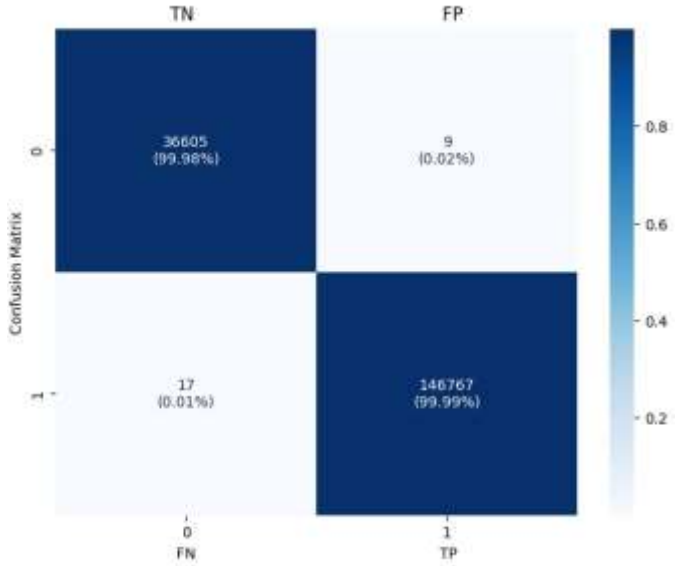

Figure 5.6 Confusion Matrix of Full Dataset using RF

It is observed from the experimental results that there was variation in performance across the different attack subsets. While RF excelled in most scenarios, simpler models like LR and NB showed superior performance on the Theft subset, which was relatively smaller in size and less complex. This finding suggests that dataset characteristics, including size and balance, significantly influence the effectiveness of classification algorithms. Additionally, while KNN demonstrated competitive performance, particularly on subsets like DoS and DDoS, its reliance on local data density made it more susceptible to misclassification in highly imbalanced scenarios. These insights emphasize the importance of tailoring algorithm selection to the specific characteristics of the dataset and attack type to maximize detection efficiency.

When performing resampling techniques to the dataset, most researchers tend to oversample the minority class. The extreme imbalance in the Bot-IoT dataset can lead to adverse outcomes if the minority class is excessively oversampled to balance the majority class. To mitigate these risks, the hybrid sampling is proposed aiming to prevent overfitting while preserving the quality of the data. Table 5.8 highlights that our proposed method performs favourably compared to state-of-the-art methods, particularly in terms of FPR.

Table 5.8 Comparative Analysis with related state-of-art work

Ref.	**Sampling Method**	**Class Ratio**	**Model**	**Acc. (%)**	**FPR (%)**
[157]	SMOTE	50:50	KNN	92.1	NA
[82]	Random Undersampling & SMOTE	94:6	ANN	NA	NA
[87]	SMOTE	50:50	DNN	99.80	NA
[159]	SMOTE & Random Undersampling	92:8	ANN	99.98	23.36
[160]	SMOTE	50:50	SVM	99.99	1.3
Our paper	Near Miss & SMOTE	80:20	RF	99.98	0.02

The proposed hybrid sampling method has important implications for real-world IoT security systems. The approach effectively reduces the likelihood of false positives and negatives, which are critical metrics in intrusion detection, to minimize service disruptions and prevent undetected attacks. Furthermore, the study's findings advocate for the adoption of a multi-model framework in IDS deployments, where different algorithms are optimized for specific attack types. This approach could enhance overall system robustness and adaptability, addressing the diverse and evolving threat landscape of IoT environments. Moreover, the varying performance of algorithms across different attack types suggests that a multi-model approach could be more effective in practice. This approach would involve deploying different algorithms optimized for specific types of attacks, creating a more robust and adaptable defence system. By leveraging the strengths of various classifiers for different attack scenarios, such a system could provide more comprehensive protection against the diverse and evolving threat landscape in IoT networks.

5.6 Security Analysis

The proposed hybrid sampling based intrusion detection framework introduces significant security advancements in the context of IoT environments, where conventional intrusion detection strategies struggle due to the prevalence of class imbalance and high data dimensionality. The implementation of a dual stage

resampling strategy Near Miss-1 undersampling followed by SMOTE oversampling marks a pivotal achievement in preparing the Bot-IoT dataset for effective learning. This approach ensures that essential patterns in the majority class are preserved while simultaneously enhancing the representational adequacy of the minority class. From a security standpoint, this methodological innovation directly contributes to improved detection sensitivity, particularly toward low frequency yet high risk benign events, thereby substantially reducing false negatives.

False negatives in IDS represent a critical weakness as they allow malicious traffic to go undetected, posing serious risks to network infrastructure. The substantial reduction in the FNR achieved by this work which is 0.002% in the DDoS subset using the RF classifier demonstrates the framework's capability in mitigating this vulnerability. Moreover, maintaining an extremely low FPR of 0.02% in the full dataset implies that the IDS is not only precise in identifying intrusions but also efficient in minimizing false alarms, which is vital for real world deployment. This balance between high recall and low false alerting supports proactive threat mitigation and efficient resource allocation for incident response teams.

A noteworthy achievement is the model's performance on the Theft subset, which is particularly challenging due to its relatively small sample size and limited representation. Although overall performance dipped slightly, the model still maintained high precision and recall values, emphasizing its ability to generalize even under constrained conditions. This has practical implications in safeguarding sensitive data and preventing exfiltration in low volume but high impact attack scenarios such as keylogging and data theft.

The security achievements of this hybrid sampling based IDS can be summarized as: enhanced detection accuracy, significant reduction in both false positives and false negatives, adaptability to attack specific contexts, and ethical, interpretable deployment. These contributions collectively address the core vulnerabilities in existing IDS designs for IoT and pave the way for more intelligent, resilient, and responsive intrusion detection architectures.

5.7 Summary

In conclusion, this study demonstrates the efficacy of a novel hybrid sampling approach in addressing the severe class imbalance inherent in the Bot-IoT dataset, a critical challenge in evaluating IDS for IoT environments. By combining Near Miss-1 for undersampling and SMOTE for oversampling, we achieved a balanced dataset without compromising data quality, enabling more reliable evaluation of ML algorithms for intrusion detection. Our findings reveal that among traditional ML models, RF exhibits superior performance across key metrics including accuracy, FPR, FNR, precision, and recall. KNN shows comparable efficacy, while NB and LR demonstrate particular strength in analyzing the Theft subset, a notably smaller dataset. These results underscore the importance of algorithm selection tailored to specific dataset characteristics and attack types in IoT security contexts. The implications of this research extend beyond immediate performance improvements, suggesting the potential for multi-model approaches in comprehensive IoT network protection. As IoT ecosystems continue to expand and evolve, the need for sophisticated, adaptive security measures becomes increasingly critical. This study lays a foundation for developing such systems, offering insights into the nuanced performance of different classification algorithms under various attack conditions and dataset characteristics. Future research in this domain could focus on enhancing the realism of botnet traffic simulations and developing adaptive models to address the dynamic nature of IoT ecosystems. Investigations into robust IDS for zero-day attacks and refined sampling techniques could further improve classification performance. Validating the proposed approach across diverse IoT datasets and in real-world deployments would provide insights into its generalizability and practical efficacy. These directions aim to advance IoT security, addressing the evolving challenges posed by sophisticated cyber threats in increasingly interconnected environments.

How can hybrid sampling methods improve the performance of deep learning based intrusion detection systems in detecting low frequency and rare cyber-attacks? The proposed hybrid sampling method effectively mitigates the extreme class imbalance in intrusion detection datasets. For deep learning based IDS also, this

balance is crucial. Without it, models tend to bias toward frequent attacks and often fail to learn features of rare, low-frequency intrusions like data theft or keylogging. Near Miss-1 selectively retains informative majority samples near minority boundaries, while SMOTE synthetically generates realistic minority instances, enriching data diversity. This balanced distribution stabilizes gradient updates during training, enhances feature representation for minority attacks, and reduces false negatives. Consequently, DL models will achieve higher recall and generalization across both common and rare attack categories. The hybrid approach thus strengthens the sensitivity and reliability of DL based IDS.

CHAPTER 6

TWO STAGED FEATURE SELECTION METHOD FOR INTRUSION DETECTION SYSTEM

6.1 Problem Statement and Background

The exponential expansion of digital infrastructure, fueled by emerging technologies such as the IoT, cloud computing, and SDN, has significantly increased the complexity, scale, and vulnerability of modern network environments. While these technologies enhance operational flexibility and efficiency, they simultaneously introduce a broad and dynamic attack surface. The ever-growing volume and heterogeneity of network traffic make the task of detecting intrusions more challenging than ever before. IDS, a critical layer of cybersecurity defence, are tasked with identifying and mitigating malicious activities in real time. However, the effectiveness of IDS in modern environments is hampered by a number of technical and operational challenges.

One of the foremost issues is the high dimensionality of network traffic data. Modern datasets, particularly those derived from SDN environments, can comprise dozens or even hundreds of features, many of which are redundant, irrelevant, or noisy. The inclusion of such features not only increases computational overhead but also reduces the accuracy and interpretability of ML models due to the “curse of dimensionality.” Traditional IDS solutions that do not implement efficient feature selection often struggle with slow processing times, increased false positives, and reduced scalability. This becomes a severe bottleneck in real-time systems, where timely detection is crucial to prevent the spread or escalation of cyber threats.

Additionally, the lack of privacy-aware mechanisms in existing IDS models poses a critical concern. The data collected and analysed by IDS may include sensitive information, and centralized processing in SDN architectures increases the risk of data leakage or unauthorized access. With growing legal and ethical concerns regarding data privacy, especially in light of regulations such as GDPR and HIPAA, it is imperative that IDS solutions incorporate privacy-preserving methods in their design, particularly during feature extraction and selection processes. Another

pressing challenge is the generalization capability of IDS models. Many ML-based IDS approaches are trained on static datasets and evaluated using known attack vectors. These systems often perform poorly when deployed in real-world scenarios involving zero-day attacks or traffic patterns not seen during training. This limitation severely undermines their practical utility, as adversaries continually develop novel attack strategies to bypass traditional defences. A robust IDS must be capable of adapting to diverse and evolving threats without sacrificing accuracy or computational efficiency.

In light of these challenges, there exists a clear need for an IDS framework that is efficient, accurate, scalable, privacy-aware, and robust to novel attack vectors. Specifically, there is a research gap in the development of intelligent feature selection mechanisms that can reduce data dimensionality while preserving critical information for classification. Furthermore, integrating such feature selection with advanced ML models offers the potential to enhance both performance and security.

Cybersecurity has been an important subject of study and development by many researchers over the past few decades. IDS as a key aspect of cybersecurity have been continuously developed, particularly with the help of ML. Aiming to identify attacks among others in the network, a novel combined model and FS approach is proposed by Jafarian et al.[161]. The proposed model has increased the performance of the model in terms of detection accuracy and FPR. The performance of various ML and FS algorithms for detecting network intrusion and anomalies using the CIC-IDS2017 dataset was analyzed with the WEKA tool [162]. They evaluate accuracy, precision, recall, F-measures, and the time required to construct a model as performance metrics, comparing the performance of algorithms with and without discretization. The simulation outcomes show that combining discretization with FS algorithms lowers the dataset's dimensionality, builds time and false alarms, and produces high-performance outcomes. On the same CIC-IDS2017 dataset, Information Gain has been recognized for its ability to calculate the weight of the features [163]. Information Gain, as well as ranking and grouping of features in accordance with the minimal weights value, is used to choose pertinent features. After that, the reduced feature set was used to build RF, NB, and J48 classifiers. The

findings demonstrate that the pertinent features produced by information gain greatly increase computation speed and detection accuracy. It is also evident that RF performs better than other classifiers in lower feature subsets (15, 22, and 35 features), while J48 outperforms other classifiers in larger feature subsets (52, 57, and 77 features). A survey on ML algorithms on the CIC-IDS2017 dataset was carried out, and it is found that different ML algorithms like ANN, DT, KNN, NB, RF, SVM, CNN, K-Means, etc., are used to conduct tests on the Anomaly-based IDS, and no single ML algorithm is able to detect all types of networking attacks [164]. They conclude that KNN,DT, and NB achieve excellent performance in different metrics such as accuracy, false positive, and false negative.

A CNN model with two convolutional layers and two max-pooling layers was proposed by Kim et al. [165]. The CIC-IDS2018 dataset was used in this experiment. The dataset was converted into images and was trained into the proposed CNN model. The same dataset is also trained using an RNN model. The experimental results show that both the models were able to detect the benign and attack data in the CIC-IDS2018 dataset, but the proposed CNN comes with higher accuracy. They also suggested that if the dataset is preprocessed in consideration of attack and benign labelled data, its accuracy increases. A DL-based autoencoder is used for FS on the CSE-CIC-IDS2018 dataset [166]. They also proposed a Developed Sunflower Optimization with CNN as a classifier model. After FS, the proposed model's performance has increased significantly from 92.77% to 98.85% accuracy.

Recursive Feature Elimination with a Cross-Validation technique is used to reduce the number of features of the CIC-DDoS2019 dataset from 88 features to 59 features [167]. Their proposed method employs DT as an estimator. RF shows superior improvements after FS, increasing from 93.67% accuracy to 99.64%, while the performance of DT and MLP does not improve. Different DDoS attacks were studied to propose a new DDoS taxonomy for the application layer by finding the common limitations of existing datasets [108]. The CIC-DDoS 2019 dataset is generated and used for the evaluation of the ID system using ID3, RF, NB, and LR, where RF and ID3 got the highest performance. The imbalanced CIC-DDoS2019 dataset is rebalanced by removing several numbers of records [168]. The

performance of ensemble learning techniques such as Bagging, Boosting, and Stacking is compared with traditional ML techniques such as KNN, NB, and DT. FS is done using Pearson's Correlation, where features with less than 0.1 value of importance are removed. The experimental results show that Stacking-based ensemble learning gives the best performance in terms of accuracy, FPR, FNR and TPR.

A GA-based FS with LR as a Learning algorithm is proposed by Chaouki et al. [169]. They use the KDD99 dataset and the UNSW-NB15 dataset to implement their proposed model. A feature subset of 18 features and 20 features on the KDD99 and UNSW-NB15 datasets, respectively, were selected. They then apply DT for classification on the selected features, achieving 99.90% accuracy and a 0.105% FPR on the KDD99 dataset. However, they achieve an accuracy of 81.42% and a 6.39% FPR rate on the UNSW-NB15 dataset. A filter-based feature reduction method using the XGBoost algorithm is proposed and applied to the UNSW-NB15 dataset [170]. The resulting 19 features out of 42 features are then used for training and testing using different traditional ML algorithms such as SVM, KNN, LR, ANN, and DT. Comparing the performance before and after the feature reduction, DT has the most significant improvement from 88.13% accuracy to 90.85% accuracy. A wrapper-based method for FS using a hybrid of the Fruit Fly algorithm and the ant lion optimizer is proposed by Samadi Bonab et al. [171]. The KDDCup99, NSL-KDD, and UNSW-NB15 datasets were used in their study. The number of features from 41 to 12, 41 to 16, and 48 to 15 in the KDDCup99, NSL-KDD, and UNSW-NB15 datasets, respectively, was reduced. Their proposed method proves to be superior to other state-of-the-art methods in different metrics.

An IoT IDS that combines the Decisive Red Fox (DRF) optimization and Descriptive Back Propagated Radial Basis Function (DBRF) classification methods was proposed by Rabie et al.[172]. The authors address the challenges of traditional IDS, such as high computational overhead and low detection accuracy. The DRF is used for FS and the DBRF categorizes data into normal and attack types. This approach is validated on five ID datasets, showing superior performance compared to existing methods. It improves training speed and minimizes error rates. In a review

of bio-inspired optimization techniques over 300 methodologies developed in the past decade [173], the paper highlights the superiority of these methods in addressing complex engineering challenges. Comparative analyses using benchmarking metrics validate the performance of these approaches against traditional optimization methods. The survey highlights the practical applicability of algorithms such as GA, Particle Swarm Optimization, and GrayWolf Optimizer.

A novel clustering and optimization methodology using the Perceptual Pigeon Galvanized Optimization for FS and the Likelihood NB for classification of ID was proposed by Shitharth et al. [174]. The proposed system is tested on NSL-KDD, CIC-IDS, and Bot-IoT datasets, showing high classification accuracy and reduced computational cost compared to conventional techniques. [175] proposes an ID system for SCADA networks, integrating genetically seeded flora-based optimization with clustering techniques. Mean-shift clustering is used to organize unlabelled data and proceeds with FS using genetically optimized flora attributes. Classification is performed using the Boltzmann ML algorithm. The proposed system is tested on real-world datasets, and there are significant improvements in detection accuracy and computational efficiency.

6.2 Methodology

In this study, traditional ML algorithms were used to detect intrusions on three different intrusion detection datasets. A two-stage FS method is used to select the best feature subset with the aim of increasing detection accuracy, decreasing the FPR, reducing the FNR, and lowering computation cost simultaneously. Figure 6.1 shows the flowchart of the proposed methodology.

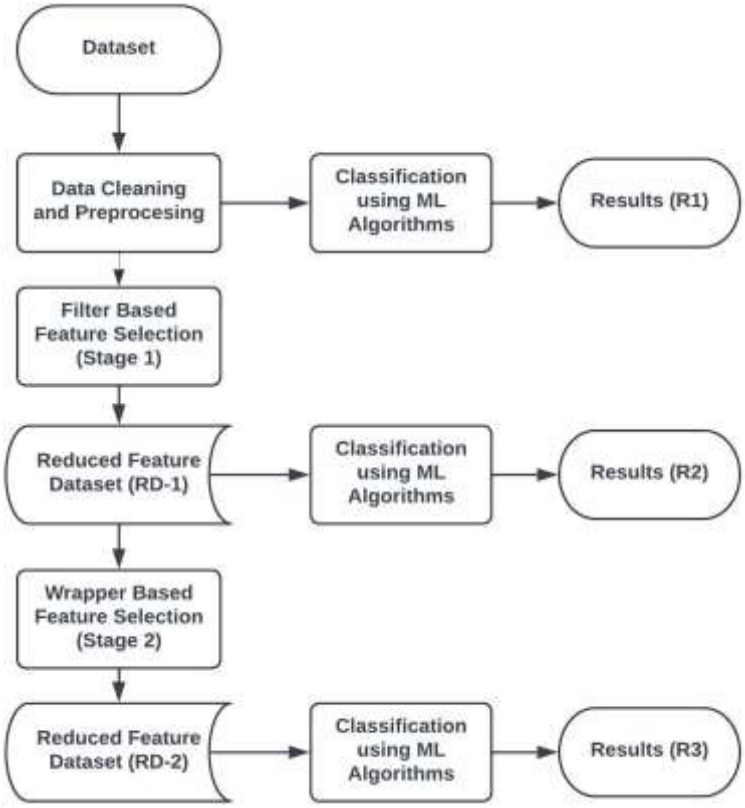

Figure 6.1 Flowchart of Two Staged Feature Selection Method

The datasets have been cleaned and preprocessed to be used for training using ML algorithms. Different traditional ML algorithms are employed to classify the preprocessed data, and the results are stored in R1 for comparison. A filter-based method for FS is applied to the preprocessed dataset. Features are selected based on the feature importance assigned to each feature by the model. This process generates a new subset of the dataset viz. RD-1. The same traditional ML algorithms are used to classify RD-1, and the results are stored in R2. A wrapper-based FS method is then performed on RD-1 to further reduce the number of features, while also aiming to improve the performance of the models. The selected features from RD-1 form a new feature subset (RD-2). The traditional ML algorithms are again applied to RD-2, and the results are stored in R3. By analysing R1, R2, and R3, the significance and importance of the proposed methodology can be determined.

6.3 Dataset Preprocessing

Dataset is a crucial component in ML. The performance of ML models heavily relies on the quality and size of the dataset. There are numerous ID datasets available publicly; in this study, we use CIC-IDS2017 dataset, CSE-CIC-IDS2018 dataset, and CIC-DDoS2019 dataset. These datasets were chosen because they contain several newer attacks in comparison to other publicly available datasets that were used as benchmark datasets in the past for research purposes. The details of the 3 intrusion datasets are tabulated in Table 6.1

Table 6.1 Numerical Details of Intrusion Detection Dataset

Datasets	**CIC-IDS2017**	**CSE-CIC-IDS2018**	**CIC-DDoS2019 (Reflection)**	**CIC-DDoS2019 (Exploitation)**
No. of Records	2,830,723	16,233,022	50,063,112	20,364,525
No. of features	79	78	82	88
No. of Benign	2,359,269	13,473,658	56,863	56,965
No. of Attacks	471,454	2,759,364	50,006,249	20,307,560
No. of Class	14	14	12	7

The three datasets are all intrusion detection datasets created by CIC (Canadian Institute of Cybersecurity). CIC-IDS2017 contains network traffic data that were captured on a real network, along with associated labels. The label shows whether the traffic data is benign or malicious. CSE-CIC-IDS2018 contains a larger and more diverse set of network traffic data than CICIDS2017, including both benign and malicious traffic. The CICDDoS2019 dataset is focused specifically on Distributed DoS (DDoS) attack detection. It contains traffic data from a variety of DDoS attacks, as well as normal traffic data.

The CIC-IDS2017 dataset contains 8 CSV files which are combined into 1 CSV file and the CSE-CIC-IDS2018 dataset contains 10 CSV files which is also combined into 1 CSV file. In the CIC-DDoS2019 dataset, the Reflection (Table 6.1) dataset is combined into 1 CSV file for training and the Exploitation (Table 6.1)

dataset is combined into 1 csv file for testing. In these datasets, the Exploitation dataset contains attacks which are not present in Reflection dataset, this can be considered as zero-day attack. From these datasets, all the features which contain only 1 type of value i.e. feature with same value for all records, are removed. In addition to that, serial features like “Timestamp” and “External IP” were also removed.

The records containing null values and infinity values are removed. There are some instances with non-numeric data type which are converted into “int” data type. All the “Benign” Label are converted into “0” and all the attacks are converted into “1” irrespective of the attack type.

All the three datasets are of extreme imbalances as shown in Table 7.1. The instances labeled "0" in the CIC-IDS2017 and CSE-CIC-IDS2018 datasets were downsampled, while the instances labeled "1" were upsampled using the Synthetic Minority Oversampling Technique (SMOTE). Similarly, in the case of the CIC-DDoS2019 dataset, the training set (Reflection) and the testing set (Exploitation) were resampled using a combination of random undersampling and SMOTE. For instance, on CIC-IDS-2017 dataset, before converting the class label into 0 and 1, classes with fewer records such as Heartbleed, Infiltration, etc were upsampled using SMOTE to increase their presentation, while class with overwhelming records such as Benign and DoS, were downsampled. The outcome of this resampling process toward achieving a balanced dataset is tabulated in Table 3 and the class distribution of CIC-IDS2017 dataset before and after (regrouped) the pre-processing and resampling is depicted using bar graph in Fig 6.2. The attack instances are carefully downsampled so that each attack will still be present in the binary class resampled. Balancing the dataset is important to ensure that the ML model is not biased towards the majority class and to improve precision and recall, especially for the minority class.

Table 6.2 Datasets after resampling

Datasets	**Benign**	**Attack**
CIC-IDS2017	235,182	235,182
CSE-CIC-IDS2018	274,693	274,693
CIC-DDoS2019 (Reflection)	93,167	93,167
CIC-DDoS2019 (Exploitation)	86,495	86,495

The resampling results of the datasets are shown in Table 6.2. CIC-DDoS2019 datasets (Reflection and Exploitation) are shown as separate dataset as they are used for training and testing respectively.

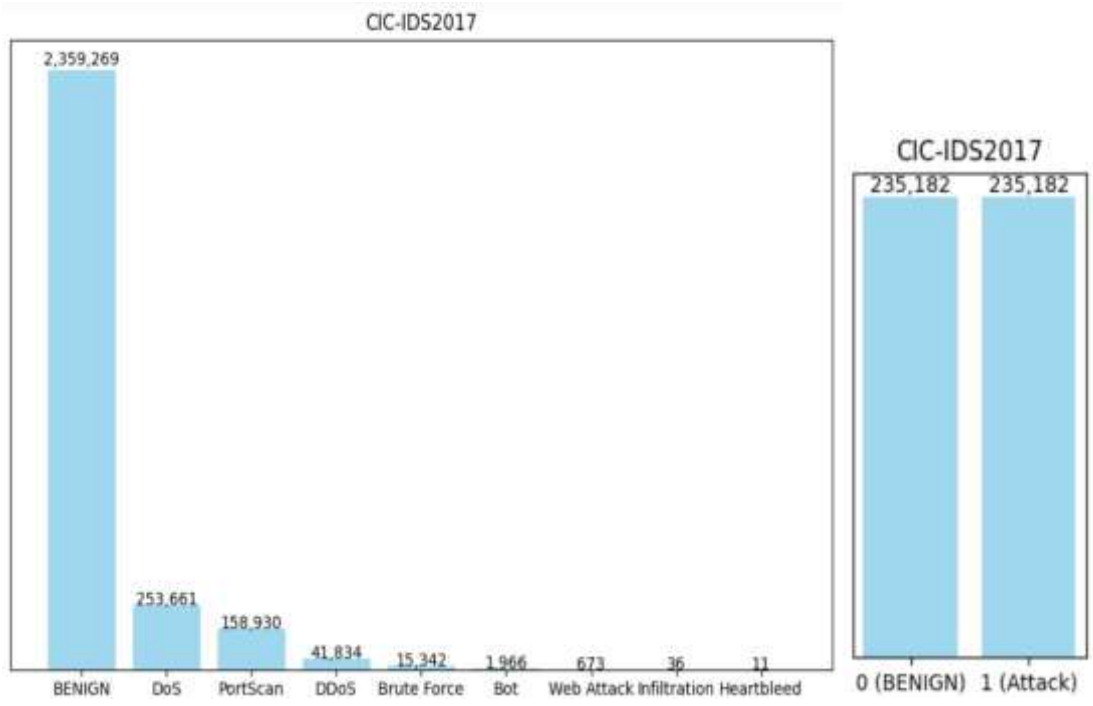

Figure 6.2 Class Distribution before (left) and after (right) pre-processing and resampling

The class distribution of CIC-IDS2017 dataset before and after pre-processing and resampling is shown in Figure 6.2. From the attack class, minority class such as Heartbleed, Infiltration, etc are oversampled while DoS and PortScan are downsampled. The Benign class is also heavily downsampled and the resulting class distribution is shown to the right of Figure 6.2.

6.4 Feature Subset Selection

Dimensionality reduction or feature subset selection can be accomplished in five ways: FM, WM, embedded method, hybrid method, and ensemble method [47]. The purpose of FS is to identify the most important features, reducing the overall number of features while enhancing classification performance. This reduction of features will also lower the complexity and computation cost. In this study, a novel two-staged FS method is proposed that combines Filter-based method and Wrapper-based method

6.4.1 Filter Based Approach

In contrast to other FS methods, the filter-based approach is the most straightforward and economical. Filter algorithms evaluate the relationships among a set of features using independent measures, such as information measures, distance measures, or consistency measures [176]. The Gini index, one of the filter-based methods is a measure of the impurity of a specific class after splitting a dataset along a particular attribute [177]. It gives value to features ranging from 0-1, where features with higher value signify that it is effective at splitting the data into subsets. This Gini coefficient is used as the first stage (Stage 1) of the FS method. Mathematically, if ”D” is the dataset and ”c” is different class labels, then, the Gini coefficient can be represented as:

$$Gini(D) = 1 - \sum_{i=1}^{C} pi^2 \quad (10)$$

where *pi* is the proportion of samples in class i in the dataset D

6.4.2 Wrapper Based Approach

Wrapper-Based Methods of FS are a class of algorithms that select subsets of features by evaluating the performance of a ML model using different combinations of features. WM can produce good results but at the cost of high computational complexity [178]. Common WM include recursive feature elimination and sequential FS [179]. In this paper, GA, a WM is used as the second stage of FS. The GA has different operation as shown in Fig 7.3:

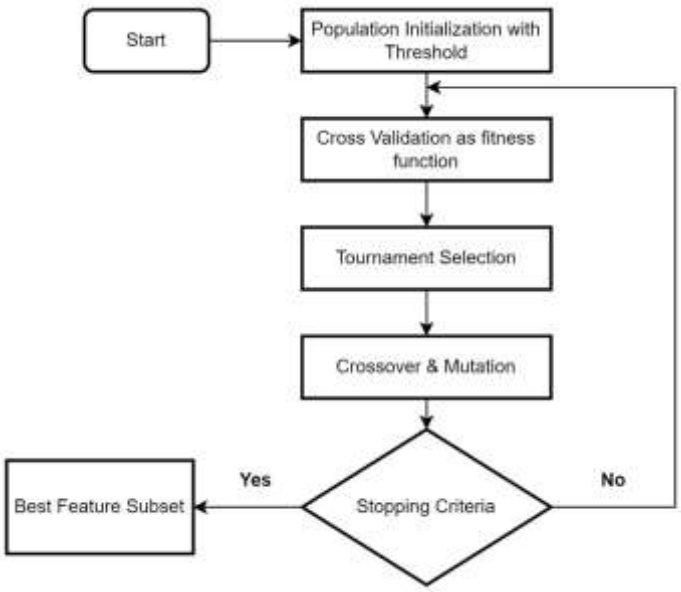

Figure 6.3 Flowchart of Genetic Algorithm

The steps involved in the process of Genetic Algorithms are:

- Population Initialization: The population is the total number of possible solutions which are generated randomly. Each solution is typically represented as a chromosome, which is a string of binary digits (0's and 1's) and each individual digit is called a gene.
- Fitness Function: Each individual in the population is evaluated based on how well it solves the problem. The fitness function assigns a fitness score to each individual, which reflects how well it performs relative to other individuals in the population.
- Selection: This operation selects individuals to reproduce new individuals. Individuals with higher fitness scores are more likely to be selected for reproduction.
- Crossover: This operator combines genetic information from two parent chromosomes to create new offspring chromosomes. A portion of one parent's chromosome is swapped with a portion of the other parent's chromosome.

- Mutation: The mutation operator works by randomly selecting one or more genes within an individual chromosome and changing their values. For example, 0 may be changed to 1, or vice versa.
- Termination: The GA is terminated when a stopping criterion is met. This can be achieved by using convergence or by specifying a fixed number of generations.

GA's are typically computationally expensive, and their randomized operations can result in different outputs each time they run. However, when properly configured with appropriate parameters, they can consistently produce optimal solutions. Now, combining the Gini and Genetic Algorithm (GIGA) for FS, it results in a two-staged FS.

Pseudocode for GIGA:

Input: Dataset D, Feature Set F
Output: Optimal Feature Subset $F*$

Stage 1: Filter-Based Feature Selection using Gini Impurity
  a. For each feature $f \in F$:
    Compute Gini score (f) using the Gini Impurity formula.
  b. Retain features $F_{\text{stage1}} \subseteq F$ where (f) > (threshold).

Stage 2: Wrapper-Based Feature Selection using Genetic Algorithm
  a. Initialize population P with random subsets of F_{stage1}.
  b. Repeat for G generations:
    i. Evaluate the fitness of each subset $S \in P$ using cross validation accuracy with DT algorithm.
    ii. Select top-performing subsets using Tournament selection.
    iii. Apply crossover to generate offspring subsets.
    iv. Mutate subsets to introduce variability.
  c. Identify the subset $F*$ with the highest fitness score.
Return: $F*$

The proposed GIGA method stands out from similar methods by integrating a Gini Impurity-based filter stage to reduce the search space, thereby minimizing computational cost. It employs a DT based fitness function for targeted evaluation, ensuring high-quality feature subsets. Parameters like mutation rate and crossover

strategy are fine-tuned for efficient search of solution. This hybrid method enhances performance and adaptability, making it superior to the conventional GA approaches in terms of accuracy, efficiency, and robustness.

6.5 Implementation and Experimental Results

The experiments were conducted using Python Jupyter Notebook on a PC equipped with Intel i7th 12th Gen processor, 32 Gb Ram, and RTX 3060Ti GPU. After data pre-processing (Sec 3.2) is done, the CIC-IDS2017 dataset, as well as the CSE-CIC-IDS2018 dataset, is split into a training set and a test set, and is evaluated with different ML algorithms using 5-fold cross validation to reduce the chance of overfitting. The CIC-DDoS2019 dataset has separate training set and test set, with the test set containing attacks which are not present in the training set which can be considered zero-day attack. The classification results with all features of CIC-IDS2017 dataset, CSE-CIC-IDS2018 dataset and CIC-DDoS2019 dataset are tabulated in Table 6.3

Table 6.3 Classification results with all features

CIC-IDS2017 (71 features)							
Classifiers	**Accuracy**	**FPR**	**FNR**	**Precision**	**Recall**	**F1-Score**	**Time (sec)**
LR	78.93	14.89	27.293	83.012	72.761	77.55	194.2
NB	58.47	81.76	1.29	54.69	98.70	70.38	3.8
K-NN	97.56	3.43	1.42	96.63	98.57	97.59	755.7
DT	99.318	0.714	0.66	99.287	99.350	99.32	88.9
RF	99.38	1.06	0.16	98.45	99.83	99.13	583.6
CSE-CIC-IDS2018 (70 features)							
LR	78.79	9.31	33.09	87.77	66.90	75.93	132.6
NB	69.47	54.01	7.03	63.25	92.96	75.28	3.02
K-NN	96.70	1.91	4.68	98.03	95.31	96.65	114.36
DT	96.01	3.88	4.08	96.10	95.91	96.01	60.2
RF	97.04	1.49	4.41	98.46	95.58	96.99	590.1
CIC-DDoS2019 (69 features)							
LR	72.99	1.37	52.64	97.18	47.35	63.67	12.2
NB	57.36	80.22	5.04	54.20	94.95	69.01	0.3
K-NN	78.82	2.02	40.33	96.72	59.66	73.79	16.1
DT	97.95	0.03	0.03	99.97	95.92	97.90	1.28
RF	99.77	0.37	0.07	99.62	99.92	99.76	19.6

In the next stage, the datasets were divided into train_set and test_set, the feature importance is calculated for each feature on the train_set for each dataset using Gini Index. In CIC-IDS2017 dataset, 31 features out of 71 features are selected by removing features with feature importance values lower than 0.001. For CSE-IDS2018, 30 features out of 70 features are selected by removing features with feature importance values lower than 0.001. Similarly for CIC-DDoS2019, 16 features out of 69 features are selected by removing features with feature importance value = zero (0). In CIC-DDoS2019 dataset, the feature importance is calculated on the Reflection (training) dataset. The top 10 high importance value features evaluated by GINI are shown in Table 6.4.

Table 6.4 Top 10 Feature importance value by Gini Index on each dataset

CIC-IDS2017	**CSE-CIC-IDS2018**	**CIC-DDoS2019**
Min Packet Length = 0.331	Subflow Fwd Byts = 0.314	Min Packet Length = 0.9322
Init_Win_bytes_backward = 0.223	Dst Port = 0.204	Source Port = 0.0352
Source Port = 0.134	Init Fwd Win Byts = 0.119	Inbound = 0.0220
Destination Port = 0.076	Fwd Seg Size Min = 0.077	Destination Port = 0.0098
Fwd Packet Length Max = 0.059	Bwd Pkt Len Mean = 0.051	Bwd Header Length = 0.002
Average Packet Size = 0.036	Fwd IAT Std = 0.048	act_data_pkt_fwd = 0.001
Init_Win_bytes_forward = 0.027	Flow Duration = 0.043	Min_seg_size forward = 0.0006
Fwd IAT Min = 0.023	Bwd Seg Size Avg= 0.034	Subflow Bwd Packets = 0.0004
Flow IAT Min = 0.014	Fwd IAT Tot = 0.015	Total Backward Packets = 0.0003
Flow Bytes/s = 0.009	Flow Byts/s = 0.008	Fwd Packet Length Std = 0.0003

Abbreviations: Min = Minimum; Init = Initial; Win = Windows; Fwd = Forward; Max = Maximum; IAT = Inter-Arrival Time; Bytes/s = Bytes per Second; Byts = Bytes; Dst = Destination; Bwd = Backward; Seg = Segment; Avg = Average; Std = Standard Deviation; Tot = Total; act = Active; pkt = Packets

The train_set and the test_set for all the datasets were now reduced by removing the features which do not pass the threshold values. The performance of different ML algorithms after the first stage of FS using Gini is tabulated in Table 6.5.

Table 6.5 Classification Results after I-Stage of feature selection

CIC-IDS2017 (29 features)							
Classifiers	**Accuracy**	**FPR**	**FNR**	**Precision**	**Recall**	**F1-Score**	**Time (sec)**
LR	78.28	17.26	26.16	81.03	73.83	77.26	38.50
NB	58.83	80.65	1.64	54.92	98.35	70.48	0.20
K-NN	97.58	3.48	1.33	96.58	98.66	97.61	22.07
DT	99.38	0.65	0.57	99.34	99.42	99.38	5.29
RF	99.40	1.09	0.10	98.91	99.89	99.40	54.14
CSE-CIC-IDS2018 (31 features)							
LR	80.49	14.24	24.75	84.09	75.24	79.42	46.46
NB	68.91	53.49	8.68	63.07	91.31	74.61	0.29
K-NN	96.71	1.94	4.61	98.00	95.38	96.67	31.12
DT	95.95	4.04	4.04	95.96	95.95	95.95	8.1.0
RF	97.18	1.17	4.46	98.78	95.53	97.13	118.3
CIC-DDoS2019 (16 features)							
LR	66.17	18.22	49.41	73.51	50.58	59.92	1.4
NB	54.21	86.67	4.90	52.31	95.09	67.49	0.1
K-NN	82.16	8.83	26.84	89.22	73.16	80.39	9.4
DT	97.94	0.04	4.06	99.95	95.93	97.89	0.3
RF	99.98	0.00	0.03	100	99.96	99.98	11.3

With the FS using Gini, all the features of the dataset are reduced to less than half significant reduction in the performance of ML algorithms. With the remaining features, GA, a wrapper-based FS method is used to select even more relevant features. 8 features are selected from 29 features of CIC-IDS2017 dataset, 4 features out of 31 features are selected in CSE-CIC-IDS2018 dataset and 8 features out of 16 features are selected in CIC-DDoS2019 dataset. The Proposed GA uses the following parameters, which are configured based on empirical experiments:

- Size of Population: 200 (number of features threshold set to “less than 14”)
- Fitness Function: 5-fold cross validation accuracy of DT Classifier is used as a fitness value. Each individual in the population is evaluated for their fitness score.

- Selection: Tournament selection is used as selection method where tournsize=3.
- Crossover: One-point crossover method is used to generate new individuals.
- Mutation: MutFlipbit operation is used where one or more bits will flip with a probability of 0.05 (or 5%).
- Stopping Criteria: The number of generations is fixed to 10

Gini, the filter-based approach may give the same result (feature value) when running the process again, but the wrapper-based GA for FS may or may not give the same subset of features due to its nature of randomness. The final selected feature subsets are shown in Table 6.6. The two-staged FS greatly improves the computation cost and time. Since the FS is solely using the DT algorithm in both the FS stage, the increase in performance of other ML algorithms cannot be highly expected. The performances of the ML algorithms on the dataset after two-stage FS (GIGA) are shown in Table 6.7.

Table 6.6 Final selected feature subsets using GA (Stage-II)

CIC-IDS2017	**CSE-CIC-IDS2018**	**CIC-DDoS2019**
Source Port	Dst Port	Min Packet Length
Destination Port	Init Fwd Win Byts	Source Port
Flow Duration	Init Bwd Win Byts	Inbound
Total Length of Bwd Packets	Fwd Seg Size Min	Bwd Header Length
Fwd IAT Std		min_seg_size_forward
Min Packet Length		Fwd IAT Mean
Init_Win_bytes_forward		Fwd IAT Min
Init_Win_bytes_backward		Flow Bytes/s

There are 8 features, 4 features, and 8 features selected from CIC-IDS2017 dataset, CSE-CIC-IDS2018 dataset, and CIC-DDoS2019 dataset respectively as shown in Table 6.6. The selected feature subset especially for CSE-CIC-IDS2018 dataset is very less in comparison to the number of original 78 features. This will reduce the computational cost down to almost 20 folds. CIC-IDS2017 and CIC-DDoS2019 datasets also benefitted from the FS achieving a 10-fold reduction in the number of features.

Table 6.7 Classification Results after II-Stage of feature selection

CIC-IDS2017 (8 features)							
Classifiers	**Accuracy**	**FPR**	**FNR**	**Precision**	**Recall**	**F1-Score**	**Time (sec)**
LR	71.61	14.28	42.51	80.09	57.49	66.93	1.3
NB	62.66	70.41	4.25	57.61	95.75	71.94	0.04
K-NN	97.26	3.93	1.55	96.16	98.45	97.29	3.03
DT	99.52	0.53	0.44	99.47	99.56	99.52	0.8
RF	99.52	0.82	0.14	99.18	99.86	99.52	24.12
CSE-CIC-IDS2018 (4 features)							
LR	66.88	63.91	2.35	60.46	97.65	74.68	0.5
NB	66.30	65.68	1.58	59.93	98.42	74.50	0.06
K-NN	97.24	0.96	4.56	99.01	95.44	97.19	19.5
DT	97.19	1.04	4.59	98.92	95.41	97.14	0.26
RF	97.36	0.07	4.52	99.20	95.48	97.31	13.71
CIC-DDoS2019 (8 features)							
LR	69.17	15.43	46.22	77.70	53.78	63.56	0.4
NB	51.08	92.97	4.86	50.57	95.13	66.03	0.1
K-NN	79.53	12.06	28.86	85.49	71.13	77.65	7.5
DT	99.98	0.01	0.01	99.99	99.98	99.98	0.1
RF	99.87	0.11	0.13	99.88	99.86	99.87	7.2

The training time of KNN is the longest among the ML algorithms used, followed by RF. But with the reduced number of features, their training time also reduced to a very low period of time in comparison to the dataset with full features. KNN and RF gives good classification results even though the FS is based on DT. For KNN, it means that the algorithm is able to identify patterns or similarities among the data points based on their proximity to each other, regardless of the relationship between the features. On the other hand, RF is basically a collection of DTs and has the advantage of reducing the risk of overfitting. It can be seen from the experimental results that DT and RF are on par with each other in terms of different performance metrics. In order to better represent the change in model performance with respect to the number of features, the line graphs for Accuracy and FPR particularly for DT and RF are depicted in Fig 6.3, Fig 6.4 and Fig 6.5, respectively (all values are in percentage):

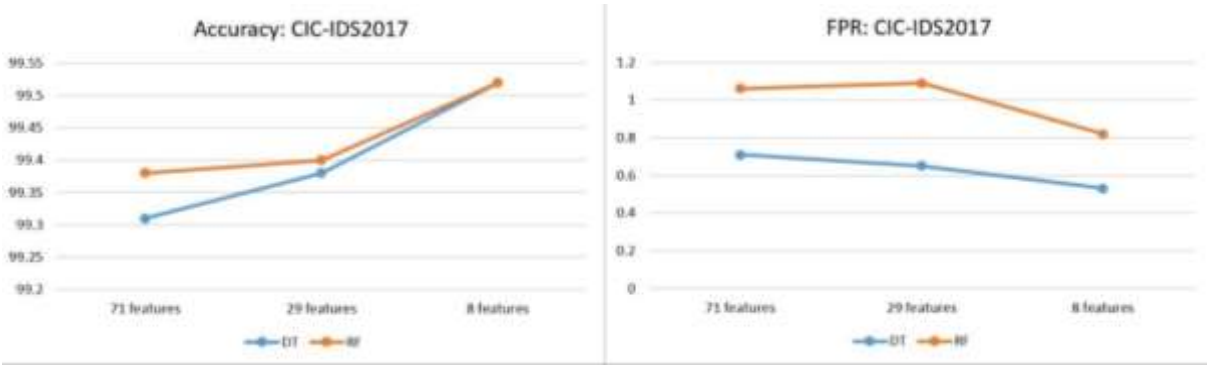

Figure 6.4 Line Graphs of Accuracy and FPR of CIC-IDS2017 dataset

The line Graphs on Figure 6.4 shows the change in accuracy and FPR of CIC-IDS2017 with change in number of features. It can be seen that with the selected features the accuracy has increased significantly. Likewise, the FPR also decreases with decrease in number of features which is a sign of improvements. The line graphs showed the graphs for DT and RF only.

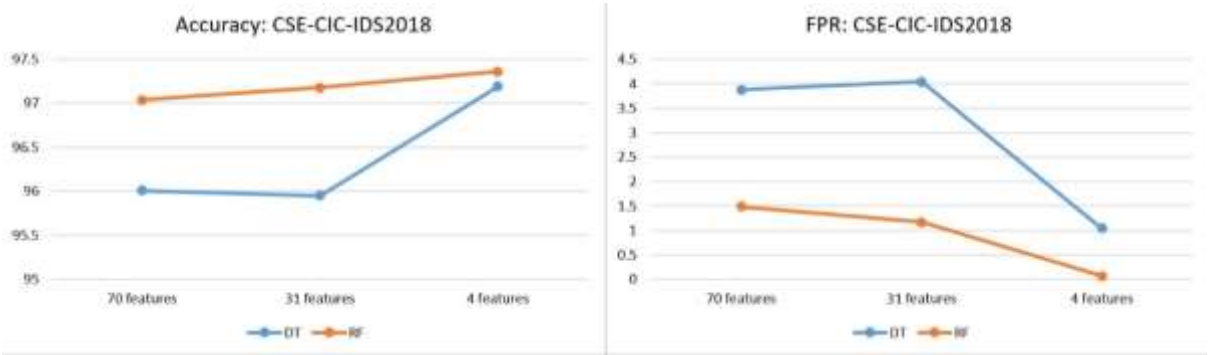

Figure 6.5 Line Graphs of Accuracy and FPR of CSE-CIC-IDS2018 dataset

The line Graphs on Figure 6.5 shows the change in accuracy and FPR of CSE-CIC-IDS2018 with change in number of features. It can be seen that with the selected features the accuracy has increased significantly. Likewise, the FPR also decreases with decrease in number of features which is a sign of improvements. The line graphs showed the graphs for DT and RF only.

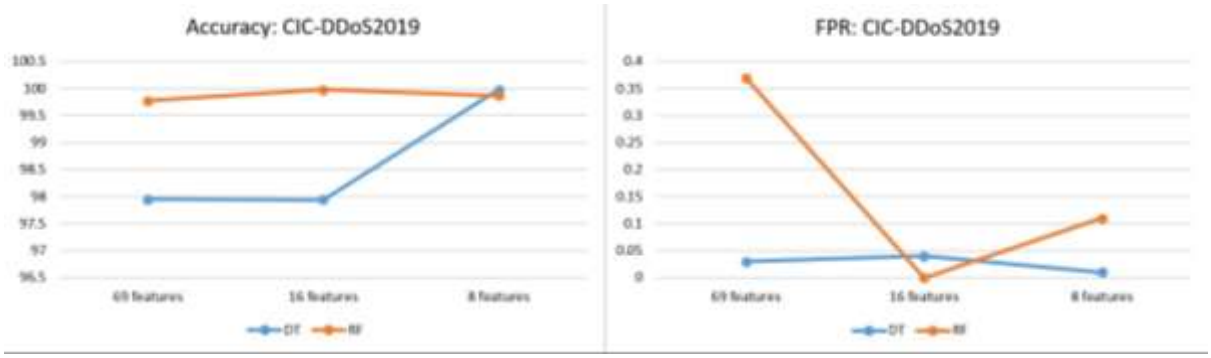

Figure 6.6 Line Graphs of Accuracy and FPR of CIC-DDoS2019 dataset

The line graphs (Fig 6.4, Fig 6.5, and Fig 6.6) shows that the performances of the model irrespective of the dataset are promising. All the left side Graphs represent classification accuracy and the right-side graphs represent FPR. Higher points in left line graph represent better accuracy while lower points in right line graphs represent better FPR. The accuracy and FPR may or may not improve with a reduced number of features, but it also doesn't degrade. Though the performance of some models may remain the same before and after the FS, the model's computational cost is highly reduced.

Interpretability of the proposed method lies in the transparency of FS stages. The Gini Impurity metric clearly explains the importance of the feature and its contribution to impurity reduction; it hence ranks features. For instance, 'Min Packet Length' and "Subflow Fwd Byts" were highly important than others to discriminate benign from malicious traffic. The GA refines these selections by evaluating the impact on detection accuracy. This ensures that the selected features will improve not only computational efficiency but also the classification accuracy resulting in a robust ID system model. Comparison of the proposed method with state of art methods is shown in Table 6.8

Table 6.8 Comparison of proposed method with other state of art methods

Dataset	**Feature Selection Techniques**	**Number of features**	**Model**	**Accuracy**
CIC-IDS2017 [180]	RF Regressor	7	Ensemble	98.30
CIC-IDS2017 [69]	PCA	10	RF	99.60
CIC-IDS2017 [181]	Lasso & RFE	15	RF	96.65
proposed method	GIGA	8	DT	99.52
CSE-CIC-IDS2018 [180]	RF Regressor	7	Ensemble	95.10
CSE-CIC-IDS2018 [182]	-	78	CNN	91.50 (avg)
CSE-CIC-IDS2018 [86]	PCA	2	RF	97.45
proposed method	GIGA	4	RF	97.36
CIC-DDoS2019 [82]	ANOVA	15	XGBoost	99.73
CIC-DDoS2019 [87]	BWWPA	NA	OptFBFN	99.18
CIC-DDoS2019 [183]	Information Gain	25	DT and RF	98.98
proposed method	GIGA	8	DT	99.98

Though the datasets and the objective may be same, the class distribution and the resampling amount may be different which can make big difference in the resulting output.

6.6 Security Analysis

The escalating sophistication and diversity of cyber threats in SDN and IoT driven environments necessitate IDS that are not only precise but also efficient and adaptable. The proposed GIGA method which is an integrated two stage FS approach combining Gini impurity and GA delivers measurable improvements to the overall security posture of IDS by optimizing the detection process across multiple fronts. A primary achievement lies in the model's ability to enhance detection performance while mitigating false classifications. The reduction in FPR and FNR is critical in securing networks, as false positives may lead to resource exhaustion and alert fatigue, whereas false negatives allow undetected intrusions to propagate. Notably, after applying the GIGA method, classifiers such as DT and RF achieved detection

accuracies as high as 99.98% on CIC-DDoS2019, with FNR approaching zero. This highlights the model's robustness in correctly identifying malicious traffic while minimizing security breaches caused by misclassification.

The model also exhibits strong adaptability to unseen or evolving attacks, as demonstrated through the use of the CIC-DDoS2019 Exploitation dataset, which contains zero-day attacks not present in the training set. Despite this challenge, the system retained high detection capability, suggesting that the selected features are not overfitted to known signatures but instead captures more generalizable behavioural patterns. This property is essential for ensuring IDS resilience in dynamic threat environments.

The layered structure of the GIGA method strengthens privacy preservation by minimizing data exposure. By discarding unnecessary features early in the pipeline, the system reduces reliance on potentially sensitive attributes, aligning with privacy mandates in security critical sectors. This characteristic is especially valuable in SDN environments, where centralized data aggregation can elevate privacy risks. From an operational perspective, GIGA also enhances model auditability and transparency. Security analysts and system administrators benefit from a reduced and well ranked feature set that facilitates easier diagnosis and refinement of IDS behaviour.

6.7 Summary

The proposed GIGA can be useful to reduce overfitting and reducing computational costs. In our case, the first stage of FS, GI (Filter-based) quickly select the best features for splitting the data at each node of the DT by removing features with very low feature importance values. This greatly reduces the number of features in the dataset. In the second stage of FS, GA (Wrapper-based) is used to select the best feature subset among plenty of possible solutions. The proposed GIGA FS method significantly reduces the dimensionality of the datasets, improving computational efficiency without compromising accuracy. The numbers of features are reduced to nearly ten-fold and the computation time for CIC-IDS2017, CSE-CIC-IDS2018, and CIC-DDoS2019 dataset are greatly reduced.

The ML algorithms exhibit high-performance results with all features (before FS), which leaves very little room for improvement in terms of accuracy and other metrics. However, significant improvements have been achieved in computational cost and time, while reducing the complexity of the dataset. The CIC-IDS datasets were captured on real network and closely resembles real world scenario, the proposed methods can be useful for building ID system as it exhibits high performance with less features. The experimental results show that LR and NB do not perform well for these datasets, however KNN, DT and RF perform very well in both stage of GIGA.

Though the CIC-IDS datasets closely resemble real world scenarios and have become standard benchmarks for evaluating ID Systems, Layeghy et al. [184] highlight their limitations in replicating real-world network traffic characteristics. These datasets may not fully capture the statistical complexity of real-world traffic, potentially leading to inflated performance metrics. To mitigate this, our study carefully applies feature selection to retain critical statistical properties, ensuring that key traffic patterns remain representative of intrusion behaviours. Furthermore, to validate the robustness of our model, we employ rigorous evaluation techniques, including cross-validation and multiple performance metrics, to prevent overestimation of results. While synthetic datasets may differ from real-world traffic, the consistency of our findings across different feature subsets and validation strategies supports the reliability and practical applicability of our approach in real-world ID Systems' deployment.

How does a two-staged feature selection method combining filter and wrapper techniques impact the detection accuracy and computational efficiency of intrusion detection systems across diverse network datasets? The proposed GIGA, which integrates a filter based Gini Impurity stage with a wrapper based Genetic Algorithm (GA), significantly enhances both detection accuracy and computational efficiency of IDS across multiple network datasets. In the first stage, the Gini filter rapidly eliminates redundant and low importance attributes, reducing the features by over 50% while retaining core features. This initial reduction minimizes search complexity and shortens subsequent optimization time. In the second stage, the GA

wrapper refines the subset by evaluating feature combinations using model based fitness, ensuring the final set maximizes classification accuracy. Across the CIC-IDS2017, CSE-CIC-IDS2018, and CIC-DDoS2019 datasets, this approach achieved up to 10 fold reduction in features and nearly 20 times lower computation time, while maintaining or slightly improving detection accuracy. The method also reduced false positives and improved generalization on unseen attacks. Thus, combining filter and wrapper techniques yields a compact, high quality feature subset that accelerates model training, lowers resource usage, and sustains strong detection capability across diverse network environments.

CHAPTER 7

ZERO DAY ATTACK DETECTION ON SDN ENVIRONMENT

7.1 Problem Statement and Background

The dynamic and programmable nature of SDN introduces both flexibility and increased vulnerability to a broad range of network attacks. Traditional IDS designed for static and hardware-based network architectures often fail to adapt to the evolving threat landscape in SDN, especially in detecting sophisticated and previously unseen (zero-day) attacks. Furthermore, most ML-based IDS models are built and evaluated using datasets that lack realistic SDN-specific traffic or comprehensive attack diversity, limiting their generalizability to real-world SDN environments.

The InSDN dataset, specifically curated for SDN-based environments, provides a unique opportunity to evaluate the effectiveness of ML algorithms under realistic SDN traffic and attack scenarios. However, existing research using the InSDN dataset primarily focuses on conventional binary or multiclass classification tasks. There is a significant research gap in leveraging this dataset for zero-day attack detection, particularly using unsupervised or anomaly-based approaches, which can identify deviations from normal behaviour without relying on prior attack signatures.

IDS in SDNs have garnered increasing attention due to SDN's programmable nature, which, while beneficial for traffic management, exposes networks to new threats. Several researchers have proposed ML-based IDS tailored for SDN environments to detect various types of network attacks, including zero-day attacks. Braga et al. [185] pioneered a lightweight DDoS detection mechanism leveraging OpenFlow's centralized view of traffic, laying the groundwork for ML-based IDS in SDNs. More recent approaches have incorporated supervised ML algorithms, such as RF, SVM, and XGBoost, for both binary and multiclass classification. Tang et al. [186] demonstrated the effectiveness of deep learning and ensemble techniques in detecting diverse attack patterns within SDN environments.

Traditional IDS often fall short in detecting zero-day attacks due to their reliance on known attack signatures. Anomaly detection methods such as One-Class SVM, Autoencoders, and Isolation Forest have gained traction for detecting previously unseen attack behaviours. For instance, Javaid et al. [132] utilized a DL-based anomaly detector that generalized well to unknown threats. Ring et al. [187] emphasized that models trained exclusively on benign data are more effective at identifying zero-day attacks by learning the boundary of normal behaviour.

The InSDN dataset, proposed by Elsayed et al. [110], is specifically designed to evaluate intrusion detection mechanisms in SDN environments. It includes realistic traffic and a wide variety of attack types distributed across three subsets: Metasploitable, OVS, and Normal. This dataset has been adopted in several studies for training and evaluating both supervised and unsupervised learning models. Chen et al. [188] applied feature selection techniques with ML classifiers on InSDN to enhance detection performance. Additionally, a recent study from the University of Regina [189] implemented early intrusion detection methods using a subset of InSDN to assess detection delays and accuracy.

However, most existing research has not sufficiently explored zero-day detection scenarios using anomaly-based methods on InSDN. Moreover, while the dataset contains diverse SDN traffic types, only a few studies have integrated the Metasploitable and OVS subsets to fully represent an SDN environment. The current work addresses these gaps by combining both supervised and unsupervised approaches, including Isolation Forest for detecting zero-day attacks, in addition to performing binary and multiclass classification on the InSDN dataset.

7.2 Methodology

In this chapter, we are going to have comprehensive experimentations on InSDN dataset using RF and XGBoost algorithms viz. Multiclass classification, Binary classification, and Zero-day attack detection using Leave One Attack Out (LOAO) method. All these experiments will be done with and without SMOTE (oversampling). The flowchart for Zero-day Attack detection is shown in Figure 7.1.

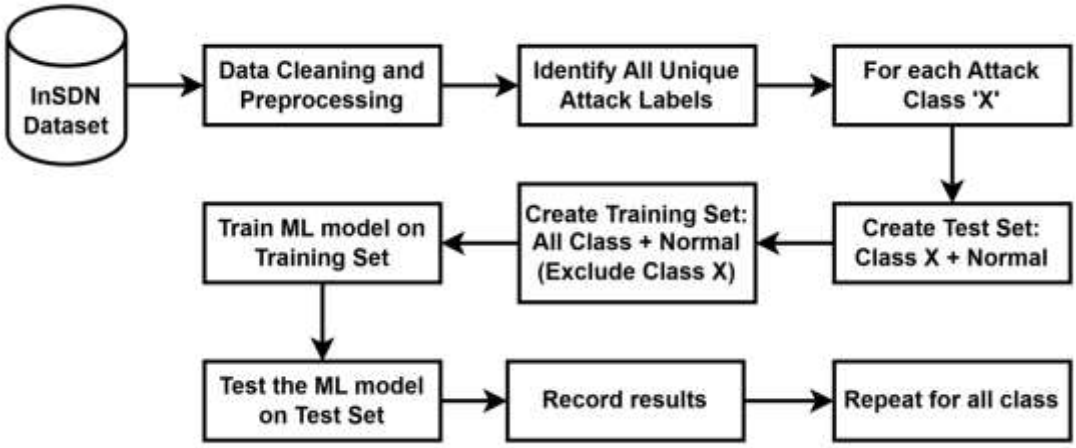

Figure 7.1 Flowchart for Zero-Day Attack Detection

The InSDN dataset is preprocessed and cleaned so that it became more suitable for evaluation using the ML models. As shown in Figure 7.1, all the attack labels are identified, and a test set is created for each attack label in association with the Normal label. Training set is created by excluding one of the attack class, and an ML model is trained on that training set. A test set is also created with an attack that is not included in the training set, the trained model is tested on that particular test set, this is done to stimulate a zero-day attack detection. This method is repeated for each attack type.

7.3 Dataset Preprocessing

The InSDN dataset a publicly available dataset that is specifically designed for research in IDS within the SDN environments. There are 3 CSV files viz. Metasploitable-2, OVS, and Normal. These 3 CSV files were combined in a single CSV file. Table 7.1 shows the data distribution of the InSDN dataset.

Table 7.1 Data distribution of InSDN dataset

Label	**No. of records**
DDoS	121,942
Probe	98,129
Normal	68,424
DoS	53,616
BFA	1,405
Web-Attack	192

BOTNET	164
U2R	17
Total	**343,889**

There are 84 features (including the class label) in this dataset. Out of these 84 features, there are 12 features which have constant value throughout the dataset; these features are removed from the dataset. There are also 6 features which are ID of the dataset and timestamp, which do not contribute to the learning model. Table 7.2 shows the features which are removed.

Table 7.2 Removed Features

Features	**Reason**
Flow ID	Unique ID, causes overfitting
Src IP	Not generalizable
Src Port	Can be random/spoofed
Dst IP	Not generalizable
Dst Port	Can be random/spoofed
Timestamp	Not informative raw
Fwd PSH Flags	Constant value
Fwd URG Flags	Constant value
CWE Flag Count	Constant value
ECE Flag Cnt	Constant value
Fwd Byts/b Avg	Constant value
Fwd Pkts/b Avg	Constant value
Fwd Blk Rate Avg	Constant value
Bwd Byts/b Avg	Constant value
Bwd Pkts/b Avg	Constant value
Bwd Blk Rate Avg	Constant value
Init Fwd Win Byts	Constant value
Fwd Seg Size Min	Constant value

There are 12 features where all the values are “0” constantly throughout the dataset. There are also 6 features which act as identity and cannot contribute in ML classifications. These 18 features are removed from the dataset as shown in Table 7.2. There is no record/instance that has missing values in this dataset, and all the remaining features are numerical which greatly reduce the steps of data preprocessing. The dataset is scaled using StandardScaler where all the features will have values with mean=0 and Standard deviation=1. The algorithm for the proposed method is shown in Algorithm 7.1.

Algorithm 7.1. Proposed method

Input:

- InSDN dataset $D=\{X,Y\}$, where X = features, Y= attack class labels
- Classifier set C ={Random Forest, XGBoost}
- Zero-day target class $Y_z \in Y$

Output:

- Predicted Labels $Y`$ for test set
- Performance metrics (Accuracy, Precision, Recall, F1-Score, FPR, FNR)

1: Data Preprocessing:

- Remove invalid, null, and duplicate records from D
- Encode categorical attributes numerically
- Normalize feature values

2: Zero-Day Attack Simulation (using LOAO):

- Select a target class $Y_z \in Y$ to simulate a zero-day attack
- Construct training set: $D_{train} = D \backslash Y_z$
- Construct test set: $D_{test} = Y_z$ U Random sample of benign class

3: Data Resampling:

- Apply SMOTE to D_{train} to balance class distribution (optional)

4: Model Training:

For each classifier $c \in C$:

- Train c on D_{train} using selected features
- Predict labels $Y`_c$ on D_{test}

5: Evaluation:

- Compare $Y`_c$ with ground truth labels from D_{test}
- Compute performance metrices

6: Result Comparison and Analysis:

- Repeat Steps 2–5 for each attack type $Y_z \in Y$
- Identify best-performing model for each zero-day attack scenario

The proposed algorithm detects zero-day attacks using a LOAO strategy combined with RF and XGBoost classifiers. One attack type is excluded from the training data to simulate an unseen scenario, while the model is trained on the remaining classes. SMOTE is applied to balance class distributions, enhancing the model's generalization. The excluded class, along with benign traffic, forms the test set. After training, both classifiers predict labels on this set, and performance is evaluated using metrics like accuracy, recall, precision, F1-score, FPR, and FNR. This process is

repeated for each attack class, providing a comprehensive evaluation of the model's resilience against various zero-day threats. The approach effectively demonstrates the model's ability to detect novel attacks with high accuracy and low FNR, making it suitable for real world SDN environments.

7.4 Implementation and Experimental Results

In this chapter, we are doing a comprehensive experimentation on InSDN dataset using RF and XGBoost algorithm. The experiments were conducted using Python Jupyter Notebook on a PC equipped with Intel i7th 12th Gen processor, 32 Gb Ram, and RTX 3060Ti GPU. Parameter tuning is done throughout each experiment. SMOTE is used in the training set to find out the difference with and without the oversampling.

7.4.1 Multiclass Classification

Each class is given numerical numbers using Label encoder, the training set comprises 80% of the data and the testing set comprises 20% of the data. Table 7.3 and Table 7.4 shows the result of the experiment using RF and XGBoost.

Table 7.3 Multiclass classification without resampling

Models	Class	Precision	Recall	F1	Support
RF	BFA	0.91	0.85	0.88	281
	BOTNET	0.97	0.94	0.95	33
	DDoS	1.00	1.00	1.00	24389
	DoS	1.00	1.00	1.00	10723
	Normal	1.00	1.00	1.00	13685
	Probe	1.00	1.00	1.00	19626
	U2R	0.60	1.00	0.75	3
	Web-Attack	0.88	0.97	0.93	38
XGBoost	BFA	0.98	0.87	0.92	281
	BOTNET	1.00	1.00	1.00	33
	DDoS	1.00	1.00	1.00	24389
	DoS	1.00	1.00	1.00	10723
	Normal	1.00	1.00	1.00	13685
	Probe	1.00	1.00	1.00	19626
	U2R	1.00	1.00	1.00	3
	Web-Attack	1.00	1.00	1.00	38

Table 7.4 Multiclass classification with resampling using SMOTE

Models	Class	Precision	Recall	F1	Support
RF	BFA	0.98	0.99	0.99	24388
	BOTNET	1.00	1.00	1.00	24388
	DDoS	1.00	1.00	1.00	24389
	DoS	1.00	1.00	1.00	24389
	Normal	1.00	1.00	1.00	24389
	Probe	0.99	0.98	0.99	24388
	U2R	1.00	1.00	1.00	24389
	Web-Attack	1.00	1.00	1.00	24388
XGBoost	BFA	0.98	0.99	0.99	24388
	BOTNET	1.00	1.00	1.00	24388
	DDoS	1.00	1.00	1.00	24389
	DoS	1.00	1.00	1.00	24389
	Normal	1.00	1.00	1.00	24389
	Probe	0.99	0.98	0.99	24388
	U2R	1.00	1.00	1.00	24389
	Web-Attack	1.00	1.00	1.00	24388

The experimental results of multiclass classification of InSDN dataset using RF and XGBoost, before and after resampling is shown in Table 7.1 and Table 7.2 respectively. Both the algorithms achieve exceptionally high detection rate. There is small difficulty in detecting less samples like U2R, BFA, etc. which is well addressed by utilizing SMOTE oversampling. This indicates that using SMOTE oversampling can enhance the detection rate especially for small classes whose class are underrepresented or there is no enough instances for the ML model to learn the patterns well.

7.4.2 Binary Classification

In Binary classification, the Normal data is labelled 0 and all the attacks were labelled 1. In this experiment, the confusion matrix is shown instead of table as the misclassification is very less in numbers in case of binary classification. Figure 7.2 and Figure 7.3 comprises the confusion matrices of binary classification using RF and RF+SMOTE, and XGBoost and XGBoost+SMOTE respectively.

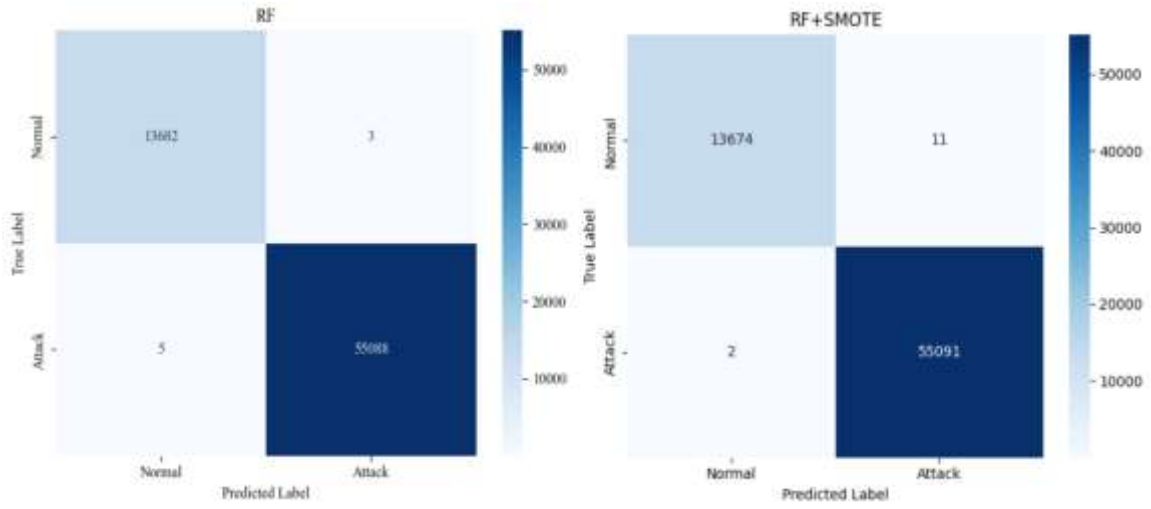

Figure 7.2 Confusion Matrices of Binary classification using RF and RF+SMOTE

Since the differences in the computation results are extremely small, confusion matrix is used to better represent the changes. Figure 7.2 shows 2 confusion matrix, which are confusion matrix of RF (left) and confusion matrix of RF+SMOTE (right). It can be seen that the false negative improves while the false positive increases when comparing before and after SMOTE. The same pattern can be seen in Figure 7.3 which is confusion matrix of XGBoost, before and after SMOTE.

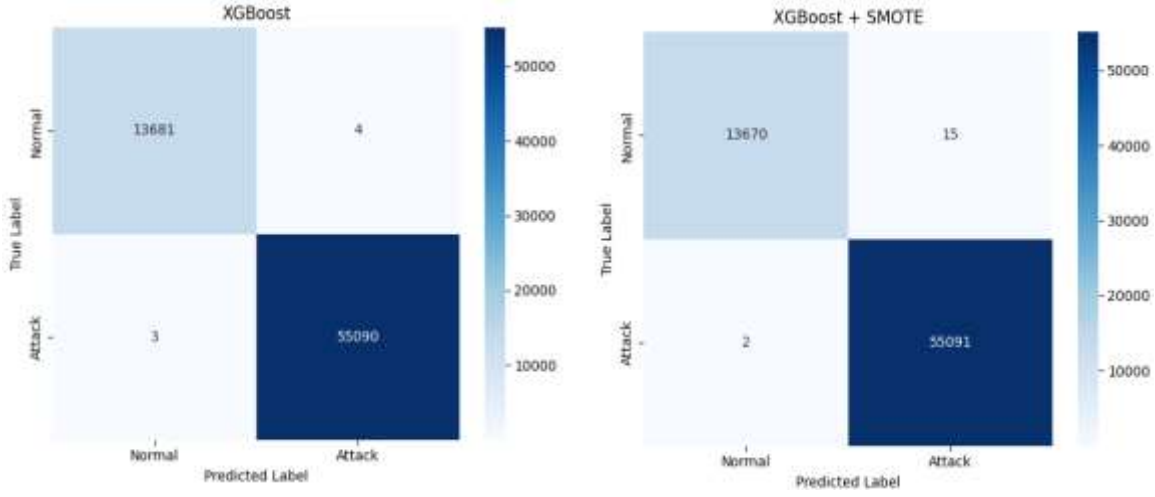

Figure 7.3 Confusion Matrices of Binary classification using XGBoost and XGBoost+SMOTE

In all the cases of binary classification, the detection accuracy is very high. However the inclusion of SMOTE oversampling in the training set has negative impact on the

detection of the Normal (Benign) Class, while it has positive impact on the detection of Attack class. The reason behind this can be that there are some samples create by SMOTE oversampling, which highly resembles the normal instances but is actually an attack samples.

7.4.3 Zero-Day Attack Detection

For zero-day attack detection using the InSDN dataset, LOAO technique is used. In this experiment, RF and XGB classifiers are used with and without SMOTE. In LOAO technique, the training set contains all the classes except one attack class (say DoS). The test set will contain only the Benign (Normal) Class and the DoS class. In this scenario, DoS acts as a zero-day attack as its behaviour is not trained in the model. The comprehensive experiments using InSDN dataset is shown in following tables.

Table 7.5 DDoS as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	27	121942	381
RF+SMOTE	68424	2	121942	49
XGBoost	68424	2	121942	22
XGBoost + SMOTE	68424	2	121942	11

Table 7.6 Probe as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	0	98129	12622
RF+SMOTE	68424	0	98129	9387
XGBoost	68424	44	98129	48
XGBoost + SMOTE	68424	33	98129	16

Table 7.7 DoS as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	2	53616	7663
RF+SMOTE	68424	3	53616	11570
XGBoost	68424	19	53616	3610
XGBoost + SMOTE	68424	2	53616	9548

Table 7.8 BFA as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	2	1405	0
RF+SMOTE	68424	2	1405	132
XGBoost	68424	36	1405	0
XGBoost + SMOTE	68424	2	1405	0

Table 7.9 Web-Attack as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	2	192	4
RF+SMOTE	68424	2	192	4
XGBoost	68424	2	192	1
XGBoost + SMOTE	68424	2	192	2

Table 7.10 BOTNET as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	2	164	0
RF+SMOTE	68424	2	164	0
XGBoost	68424	2	164	0
XGBoost + SMOTE	68424	2	164	0

Table 7.11 U2R as Zero-Day attack

Models	**Normal Count**	**False Positive Count**	**Attack Count**	**False Negative Count**
RF	68424	2	17	0
RF+SMOTE	68424	2	17	0
XGBoost	68424	2	17	1
XGBoost + SMOTE	68424	2	17	1

In Table 7.5 and Table 7.6, where DDoS and Probe acts as zero-day attack respectively, SMOTE oversampling on the training set has high positive impact and increases the detection rate significantly. However, in Table 7.7, Table 7.8, and Table 7.9, the SMOTE oversampling has mixed impact i.e. it has positive impact and negative impact, especially Table 7.7 where DoS acts as zero-day attack, it has negative impact on both RF and XGBoost. And in Table 7.10 and Table 7.11 where BOTNET and U2R acts as zero-day attack respectively, the SMOTE oversampling does not have any impact on the classification results. Overall, the zero-day attack detection in this dataset using RF and XGBoost is a success as the detection accuracy is very high in all the cases except when DoS acts as zero-day attack.

7.5 Security Analysis

Detecting zero-day intrusions in SDN environments presents unique challenges due to the dynamic nature of network behaviour and the unpredictability of novel attack patterns. The security effectiveness of the proposed model lies in its capacity to address these challenges by incorporating both traditional classification techniques and strategic simulation of unknown threats through Leave-One-Attack-Out (LOAO) evaluation. One of the most compelling security contributions of this work is its success in identifying previously unseen attack types. The LOAO strategy enables the model to simulate realistic zero-day scenarios by withholding specific attack classes during training. This mirrors real world cases where novel threats deviate from established patterns. The results demonstrate that both RF and XGBoost models exhibit strong detection capabilities across diverse zero-day attacks, including high impact classes such as DDoS and stealthier types like BFA and U2R.

Particularly, XGBoost, with its gradient based optimization, consistently achieved low FNR, indicating its capacity to recognize deviations from normal behaviour even when no prior exposure exists.

The methodology does not rely on handcrafted signatures or rule based detection, which are inherently brittle against evolving threats. Instead, the model uses generalized learning from high dimensional, SDN specific traffic, allowing it to respond to novel patterns without prior knowledge. This characteristic is critical in safeguarding programmable networks, where attack surfaces shift dynamically. The scalability and adaptability of the system further strengthen its security profile. The experimental setup accommodates large scale data processing, as shown by the model's ability to handle the comprehensive InSDN dataset with over 300,000 instances. The efficient feature handling and model training pipelines indicate feasibility for deployment in distributed SDN architectures.

The proposed zero-day detection framework demonstrates strong security achievements: it accurately distinguishes novel threats, minimizes false decisions, and adapts effectively to imbalanced and complex traffic scenarios. By integrating LOAO validation and selective oversampling, it offers a robust foundation for IDS targeting modern SDN infrastructures, where flexibility must be matched with intelligent and anticipatory defence mechanisms.

7.6 Summary

In this chapter, we presented an in-depth evaluation of two ML algorithms for IDS, specifically focusing on zero-day attack detection in SDN environments. Our work encompassed multiclass classification, binary classification, and zero-day detection experiments using RF and XGBoost algorithms, with and without SMOTE oversampling applied during training. The InSDN dataset, tailored for SDN-based networks, provides a realistic benchmark by including heterogeneous traffic types and a diverse range of attacks. Our methodology took full advantage of this dataset by combining supervised learning (for binary and multiclass classification) with a LOAO technique to simulate zero-day scenarios.

In multiclass classification, both RF and XGBoost achieved near-perfect accuracy across almost all attack classes, especially when the dataset was balanced using SMOTE. The minority classes such as U2R and BOTNET are well classified after resampling, showing a significant improvements in recall and F1-scores. This indicates that oversampling plays a crucial role in handling class imbalance and improving generalization in multiclass settings. However, in binary classification, where labels were simplified to benign vs. attack, SMOTE brought mixed results. Although it improved the model's ability to detect attacks, it slightly reduced precision for benign instances. Despite this tradeoff, both RF and XGBoost maintained high performance overall.

By excluding a specific attack class from the training set and evaluating performance on test data containing only that attack type and normal traffic, we perform a zero-day attack scenario. Across different attacks like DDoS, Probe, DoS, and BFA, XGBoost consistently outperformed RF in terms of both FPR and FNR. This highlights XGBoost's superior capacity to generalize and distinguish unseen patterns, possibly due to its gradient boosting framework. SMOTE improved performance in zero-day scenarios as well, especially for attacks like DDoS and BFA, where it significantly reduced false negatives. This also shows a better representation of various attack during training can have positive impact.

How can machine learning models be generalized to detect zero-day attacks in dynamically evolving SDN environments without prior knowledge of attack signatures? Detecting zero-day attacks in SDN environments can be challenging due to the dynamic and constantly changing nature of network configurations, traffic patterns, and control mechanisms. Conventional IDSs that rely on predefined attack signatures are ineffective in such conditions, as they can only identify known threats. To overcome this limitation, machine learning models can be generalized by focusing on learning behavioural patterns of normal network activity rather than learning specific attack characteristics. By establishing a robust understanding of normal SDN behaviour, the model can identify significant deviations that may indicate previously unseen or modified attacks. Training on diverse and realistic datasets like InSDN further enhances generalization, as the model is exposed to a

wide range of traffic conditions and attack behaviours. This enables the model to detect new threats by recognizing anomalies rather than relying on known patterns. Ensemble and tree based models such as Random Forest and XGBoost inherently support generalization through their ability to combine multiple decision trees trained on different data subsets. This reduces overfitting and improves robustness to unseen inputs

CHAPTER 8

CONCLUSION AND FUTURE WORKS

IDS are designed to monitor and analyse network traffic with the objective of identifying abnormal behaviour and potential cyber threats. Over the past decade, a wide range of ML techniques have been applied to develop IDS, largely due to the increasing complexity and rapid evolution of cyber-attacks. This thesis explores the application of ML methodologies to create specialized IDS solutions tailored for specific security challenges. In this thesis, we have explored the development of an efficient IDS tailored for modern network environments using ML techniques. The proliferation of connected devices and the exponential rise in cyber threats necessitate the deployment of advanced, intelligent detection systems capable of identifying both known and unknown attacks. This work focused on leveraging the InSDN dataset; a comprehensive benchmark dataset that reflects real-world SDN-based traffic, to train and evaluate multiple classification models.

Chapter 2 reviewed existing literature on ML in IDS, Ensemble methods, Resampling method, feature selection method, IoT and SDN and the use of ML/DL techniques in cybersecurity. It identified current challenges such as lack of generalization, high FPR, and the limited capacity of existing models to handle imbalanced datasets or detect zero-day attacks. This review helped in formulating the problem statement and informed the design of the proposed approach. It highlights the research gaps from the existing literature. Chapter 3 presented the Dataset used in this thesis, these datasets are all intrusion detection dataset. It describes the dataset and the features present in the dataset. It also presents the description of the networking attack that appear in those datasets.

Chapter 4, Chapter 5, Chapter 6, and Chapter 7 presented, analysed, and implemented the proposed method in this thesis like Ensemble ML model, Hybrid Sampling method, and Two-staged FS. The experimental results from all experiments of the proposed turns out to be positive. The findings showed that ensemble methods consistently outperformed other models in classification. Although detection accuracy of ensemble method was slightly lower than individual

ML algorithms in some case, the models demonstrated reasonable generalization to unseen attack types. Chapter 5 investigated the effects of hybrid and two-stage feature selection strategies to enhance model performance. It evaluated methods like Gini and correlation-based filters combined with wrapper techniques to further optimize the feature set. The analysis confirmed that carefully designed feature selection pipelines contribute meaningfully to detection accuracy and model robustness. Chapter 6 explored additional aspects of system deployment, including runtime efficiency and computational cost. It discussed the practical challenges of integrating ML based IDS models into real-time SDN environments. The chapter also examined how model complexity, training time, and latency trade-offs can affect the feasibility of real-world implementation. And lastly Chapter 7 presents a comprehensive experimentation on InSDN dataset which is a dataset specifically created from SDN environment. It also presented a zero-day attack detection using the LOAO method.

Despite the promising results, certain limitations persist. While traditional ML models offer reasonable accuracy, they rely heavily on hand-crafted features and often struggle with dynamic traffic patterns or concept drift. Furthermore, although the models achieved high precision in controlled test conditions, their performance in real-time, high-throughput network environments remains to be validated. There are several promising directions in which this research can be extended:

- Integration with DL Techniques: Future studies can explore DL models such as CNNs, RNNs, or Transformer-based architectures, which can automatically extract hierarchical features from raw network traffic and adapt better to complex patterns.
- Online and Real-Time Detection: Implementing the IDS in a live SDN environment with real-time packet processing capabilities is a crucial step. Stream-based models or online learning algorithms could be employed to adapt continuously to changing traffic behaviour.
- Federated Learning and Privacy-Preserving Techniques: As privacy becomes a growing concern, federated learning offers a mechanism for distributed

model training without exposing raw traffic data, thus preserving data confidentiality while enabling collaborative learning across multiple domains.

- Multi-Stage and Hybrid Detection Frameworks: Combining signature-based and anomaly-based detection in a hybrid or multi-stage system can improve the robustness of IDS by benefiting from the strengths of both paradigms.
- Adversarial Robustness and Model Hardening: Future IDS models must be evaluated and hardened against adversarial attacks that aim to bypass detection. Techniques like adversarial training and robust optimization can help mitigate these risks.

In summary, this research lays a strong foundation for building intelligent, adaptive, and efficient IDS using ML. As networks evolve and threats become more sophisticated, the continuous enhancement of IDS through advanced AI and secure learning paradigms will be pivotal to safeguarding modern cyber infrastructures.

REFERENCES

1. Anderson, J. P. (1980). Computer security threat monitoring and surveillance. Technical Report, James P. Anderson Company.
2. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., & Rajarajan, M. (2013). A survey of intrusion detection techniques in cloud. Journal of Network and Computer Applications, 36(1), 42–57.
3. Buczak, A. L., & Guven, E. (2015). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Communications Surveys & Tutorials, 18(2), 1153–1176.
4. García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. Big data analytics, 1, 1-22.
5. Jerez, J. M., Molina, I., García-Laencina, P. J., Alba, E., Ribelles, N., Martín, M., & Franco, L. (2010). Missing data imputation using statistical and machine learning methods in a real breast cancer problem. Artificial Intelligence in Medicine, 50(2), 105–115.
6. Zheng, A., & Casari, A. (2018). Feature engineering for machine learning: principles and techniques for data scientists. " O'Reilly Media, Inc.".
7. Patro, S. G. O. P. A. L., & Sahu, K. K. (2015). Normalization: A preprocessing stage. arXiv preprint arXiv:1503.06462.
8. Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. Computers & Electrical Engineering, 40(1), 16–28.
9. Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. bioinformatics, 23(19), 2507-2517.
10. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321–357.
11. Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets (Vol. 10, No. 2018). Cham: Springer.
12. He, H., & Garcia, E. A. (2009). Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284.
13. Kubat, M., & Matwin, S. (1997, July). Addressing the curse of imbalanced training sets: one-sided selection. In Icml (Vol. 97, No. 1, p. 179).
14. Mani, I., & Zhang, I. (2003, August). kNN approach to unbalanced data distributions: a case study involving information extraction. In Proceedings of workshop on learning from imbalanced datasets (Vol. 126, No. 1, pp. 1-7). United States: ICML.
15. Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81–106.
16. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining (pp. 413–422). IEEE.
17. Regression, N. (1992). An Introduction to Kernel and Nearest-Neighbor. The American Statistician, 46(3), 175-185.
18. Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression. John Wiley & Sons.

19. Rish, I. (2001). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41–46).
20. Dudoit, S., Fridlyand, J., & Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association, 97(457), 77–87.
21. Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32.
22. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys (CSUR), 41(3), 1–58.
23. Stallings, W. (2020). Network Security Essentials: Applications and Standards (6th ed.). Pearson Education
24. Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (idps). NIST special publication, 800(2007), 94.
25. Abdallah, E. E., & Otoom, A. F. (2022). Intrusion detection systems using supervised machine learning techniques: a survey. Procedia Computer Science, 201, 205-212.
26. Ahmad, I., Namal, S., Ylianttila, M., & Gurtov, A. (2015). Security in software defined networks: A survey. IEEE Communications Surveys & Tutorials, 17(4), 2317-2346.
27. Wang, T. (2016). Benefits and the security risk of software-defined networking. Isaca Journal, 4, 59.
28. Alamiedy, T. A., Anbar, M., Alqattan, Z. N., & Alzubi, Q. M. (2020). Anomaly-based intrusion detection system using multi-objective grey wolf optimisation algorithm. Journal of Ambient Intelligence and Humanized Computing, 11(9), 3735-3756.
29. Satilmiş, H., Akleylek, S., & Tok, Z. Y. (2024). A systematic literature review on host-based intrusion detection systems. Ieee Access, 12, 27237-27266.
30. Kumar, S., Gupta, S., & Arora, S. (2021). Research trends in network-based intrusion detection systems: A review. Ieee Access, 9, 157761-157779.
31. Almutairi, A. H., & Abdelmajeed, N. T. (2017, October). Innovative signature based intrusion detection system: Parallel processing and minimized database. In 2017 International Conference on the Frontiers and Advances in Data Science (FADS) (pp. 114-119). IEEE.
32. Jyothsna, V. V. R. P. V., Prasad, R., & Prasad, K. M. (2011). A review of anomaly based intrusion detection systems. International Journal of Computer Applications, 28(7), 26-35.
33. Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255–260.
34. Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Tibshirani, R., & Friedman, J. (2009). Unsupervised learning. The elements of statistical learning: Data mining, inference, and prediction, 485-585.
35. Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction (Vol. 1, No. 1, pp. 9-11). Cambridge: MIT press.
36. Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160(1), 3-24.

37. Ahmed, U., Nazir, M., Sarwar, A., Ali, T., Aggoune, E. H. M., Shahzad, T., & Khan, M. A. (2025). Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering. Scientific Reports, 15(1), 1726.
38. Kaushik, S., Bhardwaj, A., Almogren, A., Bharany, S., Altameem, A., Rehman, A. U., ... & Hamam, H. (2025). Robust machine learning based Intrusion detection system using simple statistical techniques in feature selection. Scientific Reports, 15(1), 3970.
39. Rysbekov, S., Aitbanov, A., Abdiakhmetova, Z., & Kartbayev, A. (2025). Advancing network security: a comparative research of machine learning techniques for intrusion detection. International Journal of Electrical & Computer Engineering (2088-8708), 15(2).
40. Taylor, W., Hussain, A., Gogate, M., Dashtipour, K., & Ahmad, J. (2023). Intrusion detection systems using machine learning. In Decision Making and Security Risk Management for IoT Environments (pp. 75-98). Cham: Springer International Publishing.
41. Dawkhar, S. L., Borate, N., Bangad, N., Patil, P. W., & Joshi, S. (2024). Designing IDS to Analyze Malicious Attacks using Machine Learning. International Journal For Science Technology And Engineering, 12(5), 88–93. https://doi.org/10.22214/ijraset.2024.61461
42. Singh, A., Prakash, J., Kumar, G., Jain, P. K., & Ambati, L. S. (2024). Intrusion detection system: a comparative study of machine learning-based IDS. Journal of Database Management (JDM), 35(1), 1-25.
43. Golande, S. V., Vaidya, S., Pardeshi, A., Katkade, V., & Pawar, V. (2024). An Efficient Network Intrusion Detection and Classification System using Machine Learning. learning, 4(1).
44. Iqbal, Z., Sajid, A., Zakki, M. N., Zafar, A., & Mehmood, A. (2024). Role of Machine and Deep Learning Algorithms in Secure Intrusion Detection Systems (IDS) for IOT & Smart Cities. International Journal of Information Technology, Research and Applications, 3(4), 1-16.
45. Hemraj, S., Gaurav Gupta, A., Rana, A., & Gupta, A. (2024). Demystifying Intrusion Detection Process using Machine Learning Techniques. In 2024 11th International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 629-633). IEEE.
46. Breiman, L. (1996). Bagging predictors. Machine learning, 24, 123-140.
47. Rosy, J. V., & Kumar, S. (2023). Intrusion Detection System Using Ensemble Learning Approaches. International Journal of Advanced Research in Computer and Communication Engineering, 12(7). https://doi.org/10.17148/ijarcce.2023.12719
48. Pathak, P., & Shrivas, A. K. (2024, June). Intrusion Detection System Based Ensemble Model for Classification of Attacks. In 2024 OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 4.0 (pp. 1-6). IEEE.
49. Ntayagabiri, J. P., Bentaleb, Y., Ndikumagenge, J., & El Makhtoum, H. (2025). OMIC: A Bagging-Based Ensemble Learning Framework for Large-Scale IoT Intrusion Detection. Journal of Future Artificial Intelligence and Technologies, 1(4), 401-416.

50. Zewdu, A. C., & Kumssa, H. K. (2024). An Ensemble Method for Supervised Learning for Intrusion Detection and Network Forensics.
51. Reddy, D. K. K., Behera, H. S., Pratyusha, G. S., & Karri, R. (2021). Ensemble bagging approach for IoT sensor based anomaly detection. In Intelligent Computing in Control and Communication: Proceeding of the First International Conference on Intelligent Computing in Control and Communication (ICCC 2020) (pp. 647-665). Springer Singapore
52. Al-Syouf, R., Aljarrah, O. Y., Bani-Hani, R., & Alma'aitah, A. (2025). Ensemble Machine Learning Models Utilizing a Hybrid Recursive Feature Elimination (RFE) Technique for Detecting GPS Spoofing Attacks Against Unmanned Aerial Vehicles. Sensors, 25(8), 2388.
53. Kanade, A., Vibhute, A. D., & Kanade, S. (2025). Novel ensemble bagging-logistic regression algorithm for NoSQL database security. Applied Intelligence, 55(6), 492.
54. Zhang, Z., Kong, S., Xiao, T., & Yang, A. (2024). A Network Intrusion Detection Method Based on Bagging Ensemble. Symmetry, 16(7), 850.
55. Gaikwad, D. P., & Thool, R. C. (2015). Intrusion detection system using bagging with partial decision treebase classifier. Procedia Computer Science, 49, 92-98.
56. Schapire, R. E. (1990). The strength of weak learnability. Machine learning, 5, 197-227.
57. Akinrotimi, A. O., Oluwaseun, O. R., & Bamidele, A. J. (2024). A Comparative Analysis of Adaboost and XGBoost Meta-Algorthms for Improving Network Security. Journal of Interdisciplinary Research (ISSN: 2408-1906), 9(2), 1-10.
58. Ismanto, E., Al Amien, J., & Vitriani, V. (2024). A Comparison of Enhanced Ensemble Learning Techniques for Internet of Things Network Attack Detection. MATRIK: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, 23(3), 543-556.
59. Thanh, H., & Lang, T. (2019). Use the ensemble methods when detecting DoS attacks in Network Intrusion Detection Systems. EAI Endorsed Transactions on Context-aware Systems and Applications, 6(19).
60. Al-Sharif, M., & Bushnag, A. (2024). Enhancing cloud security: A study on ensemble learning-based intrusion detection systems. IET Communications, 18(16), 950-965.
61. Hossain, M. A., & Islam, M. S. (2023). A novel hybrid feature selection and ensemble-based machine learning approach for botnet detection. Scientific Reports, 13(1), 21207
62. Okey, O. D., Maidin, S. S., Adasme, P., Lopes Rosa, R., Saadi, M., Carrillo Melgarejo, D., & Zegarra Rodríguez, D. (2022). BoostedEnML: Efficient technique for detecting cyberattacks in IoT systems using boosted ensemble machine learning. Sensors, 22(19), 7409.
63. Bhati, B. S., Chugh, G., Al-Turjman, F., & Bhati, N. S. (2021). An improved ensemble based intrusion detection technique using XGBoost. Transactions on emerging telecommunications technologies, 32(6), e4076.
64. Wolpert, D. H. (1992). Stacked generalization. Neural networks, 5(2), 241-259.

65. Kotangale, N., Sonekar, S. V., Sawwashere, S., & Baig, M. M. (2024). Intrusion Detection using Ensemble Machine Learning. Indian Scientific Journal Of Research In Engineering And Management, 08(07), 1–13. DOI: 10.55041/IJSREM36806
66. Pansare, S., Malik, A., & Batra, I. (2024, May). Hybrid Machine Learning Algorithm for Intrusion Detection Systems. In 2024 International Conference on Communication, Computer Sciences and Engineering (IC3SE) (pp. 359-364). IEEE.
67. Laha, B., Basu, D., Biswas, S., Gupta, P., & Sadhukhan, B. (2023, August). Intrusion Detection in IoT Systems Using Ensemble Machine Learning Techniques. In 2023 IEEE 4th Annual Flagship India Council International Subsections Conference (INDISCON) (pp. 1-7). IEEE.
68. Thockchom, N., Singh, M. M., & Nandi, U. (2023). A novel ensemble learning-based model for network intrusion detection. Complex & Intelligent Systems, 9(5), 5693-5714.
69. Almomani, A., Akour, I., Almomani, O., & Alauthman, M. (2023). Ensemble-Based Approach for Efficient Intrusion Detection in Network Traffic. Intelligent Automation and Soft Computing, 37(2), 2499–2517.
70. Urmi, W. F., Uddin, M. N., Uddin, M. A., Talukder, M. A., Hasan, M. R., Paul, S., ... & Imran, F. (2024). A stacked ensemble approach to detect cyber attacks based on feature selection techniques. International Journal of Cognitive Computing in Engineering, 5, 316-331.
71. Ghazi, M. R., & Raghava, N. S. (2023). A Scalable and Stacked Ensemble Approach to Improve Intrusion Detection in Clouds. Information Technology and Control, 52(4), 898-914.
72. Oriola, O. (2020). A stacked generalization ensemble approach for improved intrusion detection. International Journal of Computer Science and Information Security (IJCSIS), 18(5).
73. Rajadurai, H., & Gandhi, U. D. (2022). A stacked ensemble learning model for intrusion detection in wireless network. Neural computing and applications, 34(18), 15387-15395.
74. Kuhn, M., & Johnson, K. (2013). Applied predictive modeling (Vol. 26, p. 13). New York: Springer.
75. Zuech, R., Hancock, J., & Khoshgoftaar, T. M. (2021). Detecting web attacks using random undersampling and ensemble learners. Journal of Big Data, 8(1), 75.
76. Rafrastara, F. A., Supriyanto, C., Paramita, C., Astuti, Y. P., & Ahmed, F. (2023). Performance Improvement of Random Forest Algorithm for Malware Detection on Imbalanced Dataset using Random Under-Sampling Method. Jurnal Informatika: Jurnal Pengembangan IT, 8(2), 113-118.
77. Hartono, H., Zuhanda, M. K., & Ongko, E. (2025). Optimization Class Imbalance Using Undersampling Tomek Links on Backpropagation. Indonesian Journal of Industrial Intelligence, 1(1).
78. More, A. (2016). Survey of resampling techniques for improving classification performance in unbalanced datasets. arXiv preprint arXiv:1608.06048.

79. Vuttipittayamongkol, P., & Elyan, E. (2020). Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. Information Sciences, 509, 47-70.
80. Aziz, M. N., & Ahmad, T. (2021). Clustering under-sampling data for improving the performance of intrusion detection system. Journal of Engineering Science and Technology, 16(2), 1342-1355.
81. Silva, B. R., Silveira, R. J., da Silva Neto, M. G., Cortez, P. C., & Gomes, D. G. (2021). A comparative analysis of undersampling techniques for network intrusion detection systems design. Journal of Communication and Information Systems, 36(1), 31-43.
82. Bagui, S., & Li, K. (2021). Resampling imbalanced data for network intrusion detection datasets. Journal of Big Data, 8(1), 6.
83. Khedkar, S.A., Chandane, M., & Gawande, R. (2024) Integrated Spatial and Temporal Features Based Network Intrusion Detection System Using SMOTE Sampling. International Journal of Computer Network and Information Security (IJCNIS), 16(2), 14-27.
84. Liu, J., Gao, Y., & Hu, F. (2021). A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM. Computers & Security, 106, 102289.
85. Yang, S. J., & Cha, K. J. (2021). Gmote: Gaussian based minority oversampling technique for imbalanced classification adapting tail probability of outliers. arXiv preprint arXiv:2105.03855.
86. Abdelkhalek, A., & Mashaly, M. (2023). Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning. The journal of Supercomputing, 79(10), 10611-10644.
87. Chen, H., You, G. R., & Shiue, Y. R. (2024). Hybrid Intrusion Detection System Based on Data Resampling and Deep Learning. International Journal of Advanced Computer Science & Applications, 15(2).
88. Cao, B., Li, C., Song, Y., & Fan, X. (2022). Network intrusion detection technology based on convolutional neural network and BiGRU. Computational Intelligence and Neuroscience, 2022(1), 1942847.
89. Aldallal, A. (2022). Toward Efficient Intrusion Detection System Using Hybrid Deep Learning Approach. Symmetry, 14(9), 1916.
90. Abdulmajeed, I., & Husien, I. (2022). MLIDS22- IDS Design by Applying Hybrid CNN-LSTM model on Mixed-Datasets. Informatica, 46(8).
91. Damtew, Y. G., Chen, H., & Yuan, Z. (2023). Heterogeneous ensemble feature selection for network intrusion detection system. International Journal of Computational Intelligence Systems, 16(1), 9.
92. Siddiqi, M. A., & Pak, W. (2020). Optimizing Filter-Based Feature Selection Method Flow for Intrusion Detection System. Electronics, 9(12), 2114.
93. Lyu, Y., Feng, Y., & Sakurai, K. (2023). A Survey on Feature Selection Techniques Based on Filtering Methods for Cyber Attack Detection. Information, 14(3), 191.
94. Almaghthawi, Y., Ahmad, I., & Alsaadi, F. E. (2022). Performance Analysis of Feature Subset Selection Techniques for Intrusion Detection. Mathematics, 10(24), 4745.

95. Umar, M. A., Zhanfang, C., & Liu, Y. (2020, January). Network intrusion detection using wrapper-based decision tree for feature selection. In Proceedings of the 2020 International Conference on Internet Computing for Science and Engineering (pp. 5-13).
96. Fatima, Z., & Ali, A. (2022). Effective Metaheuristic Based Classifiers for Multiclass Intrusion Detection. arXiv preprint arXiv:2210.02678.
97. Mokbal, M., Wang, D., Osman, M., Yang, P., & Alsamhi, S. H. (2022). An efficient intrusion detection framework based on embedding feature selection and ensemble learning technique. The International Arab Journal of Information Technology, 19(2), 237-248.
98. Zhang, Y., Zhang, H., & Zhang, B. (2022). An effective ensemble automatic feature selection method for network intrusion detection. Information, 13(7), 314.
99. Liu, Z., Thapa, N., Shaver, A., Roy, K., Siddula, M., Yuan, X., & Yu, A. (2021). Using Embedded Feature Selection and CNN for Classification on CCD-INID-V1-A New IoT Dataset. Sensors, 21(14), 4834.
100. Zhao, X., Su, H., & Sun, Z. (2022). An Intrusion Detection System Based on Genetic Algorithm for Software-Defined Networks. Mathematics, 10(21), 3941.
101. Chouikik, M., Ouaissa, M., Ouaissa, M., Boulouard, Z., & Kissi, M. (2024). Detection and mitigation of DDoS attacks in SDN based intrusion detection system. Bulletin of Electrical Engineering and Informatics, 13(4), 2750–2757.
102. Alshammari, T. M., & Alserhani, F. M. (2022). Scalable and robust intrusion detection system to secure the iot environments using software defined networks (SDN) enabled architecture. Int. J. Comput. Networks Appl, 9(6), 678-688.
103. Radhi, M., & Mohammed, A. (2022). An efficient deep learning approach for network intrusion detection system on software defined network. Int J Netw Secur Appl, 14, 1-14.
104. Abdallah, M., An Le Khac, N., Jahromi, H., & Delia Jurcut, A. (2021, August). A hybrid CNN-LSTM based approach for anomaly detection systems in SDNs. In Proceedings of the 16th International Conference on Availability, Reliability and Security (pp. 1-7).
105. Latah, M., & Toker, L. (2020). An efficient flow-based multi-level hybrid intrusion detection system for software-defined networks. CCF Transactions on Networking, 3(3), 261-271.
106. Abbas, O. G., Khorzom, K., & Assora, M. (2020). Machine Learning based Intrusion Detection System for Software Defined Networks. International Journal of Engineering Research and Technology, 9(9), 1033-1036.
107. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp, 1(2018), 108-116.
108. Sharafaldin, I., Lashkari, A. H., Hakak, S., & Ghorbani, A. A. (2019, October). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In 2019 international carnahan conference on security technology (ICCST) (pp. 1-8). IEEE

109. Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. Future Generation Computer Systems, 100, 779-796.
110. Elsayed, M. S., Le-Khac, N. A., & Jurcut, A. D. (2020). InSDN: A novel SDN intrusion dataset. IEEE access, 8, 165263-165284.
111. Hovav, A., & D'Arcy, J. (2003). The impact of denial-of-service attack announcements on the market value of firms. Risk Management and Insurance Review, 6(2), 97-121.
112. Maleki, M., & Shahidi, S. M. (2022). Analysis of IDS alerts by generalising features and discovering emerging patterns. International Journal of Reasoning-based Intelligent Systems, 14(1), 56–65.
113. Schuba, C. L., Krsul, I. V., Kuhn, M. G., Spafford, E. H., Sundaram, A., & Zamboni, D. (1997, May). Analysis of a denial of service attack on TCP. In Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No. 97CB36097) (pp. 208-223). IEEE.
114. Kang, H. S., Kim, K., & Kim, S. R. (2024). A New Mitigation Method against DRDoS Attacks Using a Snort UDP Module in Low-Specification Fog Computing Environments. Electronics, 13(15), 2919.
115. Saurabh, K., Kumar, T., Singh, U., Vyas, O. P., & Khondoker, R. (2022, June). Nfdlm: A lightweight network flow based deep learning model for ddos attack detection in iot domains. In 2022 IEEE World AI IoT Congress (AIIoT) (pp. 736-742). IEEE.
116. Hamza, A. A., & surayh Al-Janabi, R. J. (2024). Detecting brute force attacks using machine learning. In BIO Web of Conferences (Vol. 97, p. 00045). EDP Sciences.
117. Hamza, A. A., & s urayh Al-Janabi, J. (2024). Detecting Brute Force Attacks on SSH and FTP Protocol Using Machine Learning: A Survey. Journal of Al-Qadisiyah for Computer Science and Mathematics, 16(1), 21-31.
118. Kumar, J. H., & Ponsam, J. G. (2023, January). Cross site scripting (XSS) vulnerability detection using machine learning and statistical analysis. In 2023 International Conference on Computer Communication and Informatics (ICCCI) (pp. 1-9). IEEE.
119. Bertino, E., & Sandhu, R. (2005). Database security-concepts, approaches, and challenges. IEEE Transactions on Dependable and secure computing, 2(1), 2-19.
120. Zargar, S. T., Joshi, J., & Tipper, D. (2013). A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE communications surveys & tutorials, 15(4), 2046-2069.
121. Presekal, A., Ştefanov, A., Rajkumar, V. S., Semertzis, I., & Palensky, P. (2024). Advanced Persistent Threat Kill Chain for Cyber-Physical Power Systems. IEEE Access. 12, 177746 – 177771.
122. Durumeric, Z., Li, F., Kasten, J., Amann, J., Beekman, J., Payer, M., ... & Halderman, J. A. (2014, November). The matter of heartbleed. In Proceedings of the 2014 conference on internet measurement conference (pp. 475-488).

123. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2011). Surveying port scans and their detection methodologies. The Computer Journal, 54(10), 1565-1581.
124. Pittman, J. M. (2023). Machine learning and port scans: A systematic review. arXiv preprint arXiv:2301.13581.
125. Roy, S., Sharmin, N., Acosta, J. C., Kiekintveld, C., & Laszka, A. (2022). Survey and taxonomy of adversarial reconnaissance techniques. ACM Computing Surveys, 55(6), 1-38.
126. Laštovička, M., Husák, M., Velan, P., Jirsík, T., & Čeleda, P. (2023). Passive operating system fingerprinting revisited: Evaluation and current challenges. Computer Networks, 229, 109782.
127. Damopoulos, D., Kambourakis, G., & Gritzalis, S. (2013). From keyloggers to touchloggers: Take the rough with the smooth. Computers & security, 32, 102-114.
128. Al-Harrasi, A., Shaikh, A. K., & Al-Badi, A. (2021). Towards protecting organisations' data by preventing data theft by malicious insiders. International Journal of Organizational Analysis, 31(3), 875-888.
129. Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., & Das, K. (2000). The 1999 DARPA off-line intrusion detection evaluation. Computer networks, 34(4), 579-595.
130. Mahoney, M. V., & Chan, P. K. (2003, November). Learning rules for anomaly detection of hostile network traffic. In Third IEEE International Conference on Data Mining (pp. 601-604). IEEE.
131. Kreibich, C., & Crowcroft, J. (2004). Honeycomb: creating intrusion detection signatures using honeypots. ACM SIGCOMM computer communication review, 34(1), 51-56.
132. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016, May). A deep learning approach for network intrusion detection system. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS) (pp. 21-26).
133. Mukherjee, S., & Sharma, N. (2012). Intrusion detection using naive Bayes classifier with feature reduction. Procedia Technology, 4, 119-128.
134. Khan, L., Awad, M., & Thuraisingham, B. (2007). A new intrusion detection system using support vector machines and hierarchical clustering. The VLDB journal, 16, 507-521.
135. Muda, Z., Yassin, W., Sulaiman, M. N., & Udzir, N. I. (2011, July). Intrusion detection based on K-Means clustering and Naïve Bayes classification. In 2011 7th international conference on information technology in Asia (pp. 1-6). IEEE.
136. Kostas, K. (2018). Anomaly detection in networks using machine learning. Research Proposal, 23, 343.
137. Gautam, R. K. S., & Doegar, E. A. (2018, January). An ensemble approach for intrusion detection system using machine learning algorithms. In 2018 8th International conference on cloud computing, data science & engineering (confluence) (pp. 14-15). IEEE.

138. Tama, B. A., Comuzzi, M., & Rhee, K. H. (2019). TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. IEEE access, 7, 94497-94507.
139. Yang, J., Sheng, Y., & Wang, J. (2020). A GBDT-paralleled quadratic ensemble learning for intrusion detection system. IEEE Access, 8, 175467-175482.
140. Thakkar, A., & Lohiya, R. (2020). A review of the advancement in intrusion detection datasets. Procedia Computer Science, 167, 636-645.
141. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.
142. Haq, N. F., Onik, A. R., Hridoy, M. A. K., Rafni, M., Shah, F. M., & Farid, D. M. (2015). Application of machine learning approaches in intrusion detection system: a survey. IJARAI-International Journal of Advanced Research in Artificial Intelligence, 4(3), 9-18.
143. Beulah, J. R., & Punithavathani, D. S. (2020). An efficient mixed attribute outlier detection method for identifying network intrusions. International Journal of Information Security and Privacy (IJISP), 14(3), 115-133.
144. Baich, M., Hamim, T., Sael, N., & Chemlal, Y. (2022). Machine Learning for IoT based networks intrusion detection: a comparative study. Procedia Computer Science, 215, 742-751.
145. Mahmoud, M., Kasem, M., Abdallah, A., & Kang, H. S. (2022, July). Ae-lstm: Autoencoder with lstm-based intrusion detection in iot. In 2022 International Telecommunications Conference (ITC-Egypt) (pp. 1-6). IEEE.
146. Liu, X., & Du, Y. (2023). Towards effective feature selection for IoT botnet attack detection using a genetic algorithm. Electronics, 12(5), 1260.
147. Hnamte, V., Najar, A. A., Nhung-Nguyen, H., Hussain, J., & Sugali, M. N. (2024). DDoS attack detection and mitigation using deep neural network in SDN environment. Computers & Security, 138, 103661.
148. Fernando, G. P., Florina, A. M., & Liliana, C. B. (2024). Evaluation of the performance of unsupervised learning algorithms for intrusion detection in unbalanced data environments. IEEE Access.
149. Tsogbaatar, E., Bhuyan, M. H., Taenaka, Y., Fall, D., Gonchigsumlaa, K., Elmroth, E., & Kadobayashi, Y. (2021). DeL-IoT: A deep ensemble learning approach to uncover anomalies in IoT. Internet of Things, 14, 100391.
150. Anitha, A. A., & Arockiam, L. (2022). A review on intrusion detection systems to secure IoT networks. International Journal of Computer Networks and Applications, 9(1), 38-50.
151. Alosaimi, S., & Almutairi, S. M. (2023). An intrusion detection system using BoT-IoT. Applied Sciences, 13(9), 5427.
152. Leevy, J. L., Khoshgoftaar, T. M., & Hancock, J. (2022). Iot attack prediction using big bot-iot data. International Journal of Internet of Things and Cyber-Assurance, 2(1), 45-61.
153. Ibrahimi, K., & Benaddi, H. (2022, December). Improving the ids for bot-iot dataset-based machine learning classifiers. In 2022 5th International Conference on Advanced Communication Technologies and Networking (CommNet) (pp. 1-6). IEEE.

154. Karthik, N. (2024). The Effect of Anomaly Detection and Data Balancing in Prediction of Diabetes. International Journal of Computer Information Systems and Industrial Management Applications, 16(2), 13-13.
155. Yu, L., Zhou, R., Chen, R., & Lai, K. K. (2022). Missing data preprocessing in credit classification: One-hot encoding or imputation?. Emerging Markets Finance and Trade, 58(2), 472-482.
156. Ferreira, P., Le, D. C., & Zincir-Heywood, N. (2019, October). Exploring feature normalization and temporal information for machine learning based insider threat detection. In 2019 15th International Conference on Network and Service Management (CNSM) (pp. 1-7). IEEE.
157. Pokhrel, S., Abbas, R., & Aryal, B. (2021). IoT security: botnet detection in IoT using machine learning. arXiv preprint arXiv:2104.02231.
158. Waskom, M. L. (2021). Seaborn: statistical data visualization. Journal of Open Source Software, 6(60), 3021.
159. Kulkarni, D. D., & Jaiswal, R. K. (2023). An intrusion detection system using extended Kalman filter and neural networks for IoT networks. Journal of Network and Systems Management, 31(3), 56.
160. Atuhurra, J., Hara, T., Zhang, Y., Sasabe, M., & Kasahara, S. (2024). Dealing with imbalanced classes in bot-IoT dataset. arXiv preprint arXiv:2403.18989.
161. Jafarian, T., Masdari, M., Ghaffari, A., & Majidzadeh, K. (2020). Security anomaly detection in software-defined networking based on a prediction technique. International Journal of Communication Systems, 33(14), e4524.
162. Panwar, S. S., Negi, P. S., Panwar, L. S., & Raiwani, Y. P. (2019). Implementation of machine learning algorithms on CICIDS-2017 dataset for intrusion detection using WEKA. International Journal of Recent Technology and Engineering Regular Issue, 8(3), 2195-2207.
163. Stiawan, D., Idris, M. Y. B., Bamhdi, A. M., & Budiarto, R. (2020). CICIDS-2017 dataset feature analysis with information gain for anomaly detection. IEEE Access, 8, 132911-132921.
164. Maseer, Z. K., Yusof, R., Bahaman, N., Mostafa, S. A., & Foozy, C. F. M. (2021). Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. IEEE access, 9, 22351-22370.
165. Kim, J., Shin, Y., & Choi, E. (2019). An intrusion detection model based on a convolutional neural network. Journal of Multimedia Information System, 6(4), 165-172.
166. Krishnan, D. V. G., Hemamalini, S., Cheraku, P., Priya, K. H., Ganesan, S., & Balamanigandan, R. (2023). Attack detection using DL based feature selection with improved convolutional neural network. International Journal of Electrical and Electronics Research, 11(2), 308-314.
167. Kumari, P., Jain, A. K., Pal, Y., Singh, K., & Singh, A. (2024). Towards Detection of DDoS Attacks in IoT with Optimal Features Selection. Wireless Personal Communications, 137(2), 951-976.
168. Khoei, T. T., Aissou, G., Hu, W. C., & Kaabouch, N. (2021, May). Ensemble learning methods for anomaly intrusion detection system in smart grid. In 2021 IEEE international conference on electro information technology (EIT) (pp. 129-135). IEEE.

169. Khammassi, C., & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. Computers & security, 70, 255-277.
170. Kasongo, S. M., & Sun, Y. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. Journal of Big Data, 7(1), 105.
171. Samadi Bonab, M., Ghaffari, A., Soleimanian Gharehchopogh, F., & Alemi, P. (2020). A wrapper-based feature selection for improving performance of intrusion detection systems. International Journal of Communication Systems, 33(12), e4434.
172. Rabie, O. B. J., Selvarajan, S., Hasanin, T., Alshareef, A. M., Yogesh, C. K., & Uddin, M. (2024). A novel IoT intrusion detection framework using Decisive Red Fox optimization and descriptive back propagated radial basis function models. Scientific Reports, 14(1), 386.
173. Selvarajan, S. (2024). A comprehensive study on modern optimization techniques for engineering applications. Artificial Intelligence Review, 57(8), 194.
174. Shitharth, S., Kshirsagar, P. R., Balachandran, P. K., Alyoubi, K. H., & Khadidos, A. O. (2022). An innovative perceptual pigeon galvanized optimization (PPGO) based likelihood Naïve Bayes (LNB) classification approach for network intrusion detection system. IEEE Access, 10, 46424-46441.
175. Selvarajan, S., Shaik, M., Ameerjohn, S., & Kannan, S. (2020). Mining of intrusion attack in SCADA network using clustering and genetically seeded flora-based optimal classification algorithm. IET Information Security, 14(1), 1-11.
176. Tangirala, S. (2020). Evaluating the impact of GINI index and information gain on classification using decision tree classifier algorithm. International Journal of Advanced Computer Science and Applications, 11(2), 612-619.
177. Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. Cybern. Inf. Technol, 19(1), 3-26.
178. Jain, D., & Singh, V. (2018). Feature selection and classification systems for chronic disease prediction: A review. Egyptian Informatics Journal, 19(3), 179-189.
179. Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. Journal of biomedical informatics, 35(5-6), 352-359.
180. Indra Gandhi, K., Balaji, S., Srikanth, S., & Varshini, V. S. (2023, April). Ensemble Machine Learning-Based Network Intrusion Detection System. In International Conference on Frontiers of Intelligent Computing: Theory and Applications (pp. 135-144). Singapore: Springer Nature Singapore
181. Neelaveni, R., Abhinav., & Sahas (2023). Analysis of Efficient Intrusion Detection System using Ensemble Learning. International Journal For Science Technology And Engineering, 11(5), 1521–1530. https://doi.org/10.22214/ijraset.2023.51858

182. Alotaibi, Y., & Ilyas, M. (2023). Ensemble-learning framework for intrusion detection to enhance internet of things' devices security. Sensors, 23(12), 5568.
183. Liu, X., Li, T., Zhang, R., Wu, D., Liu, Y., & Yang, Z. (2021). A GAN and feature selection-based oversampling technique for intrusion detection. Security and Communication Networks, 2021(1), 9947059.
184. Layeghy, S., Gallagher, M., & Portmann, M. (2024). Benchmarking the benchmark—Comparing synthetic and real-world Network IDS datasets. Journal of Information Security and Applications, 80, 103689.
185. Braga, R., Mota, E., & Passito, A. (2010, October). Lightweight DDoS flooding attack detection using NOX/OpenFlow. In IEEE local computer network conference (pp. 408-415). IEEE.
186. Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., & Ghogho, M. (2016, October). Deep learning approach for network intrusion detection in software defined networking. In 2016 international conference on wireless networks and mobile communications (WINCOM) (pp. 258-263). IEEE.
187. Ring, M., Wunderlich, S., Scheuring, D., Landes, D., & Hotho, A. (2019). A survey of network-based intrusion detection data sets. Computers & security, 86, 147-167.
188. Harahsheh, K., Al-Naimat, R., & Chen, C. H. (2024). Using feature selection enhancement to evaluate attack detection in the internet of things environment. Electronics, 13(9), 1678.
189. Towhid, M. S., & Shahriar, N. (2023, May). Early detection of intrusion in sdn. In NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium (pp. 1-6). IEEE.

BIO DATA OF THE CANDIDATE

Name	: **R. LALDUHSAKA**
Date of Birth	: 01/07/1993
Phone	: 9862195255
Email	: duskeralte777@gmail.com
Permanent Address	: H. No T-42, Bethlehem Vengthlang, Aizawl Mizoram
Marital Status	: Married
Educational Details	:

1. B.Tech : North Eastern Regional Institute of Science and Technology (NERIST)
2. M.Tech : National Institute of Technology, Silchar
3. Ph.D Course Work : Mizoram University

List of Publications Based on Thesis

Journal Publications

1. **Lalduhsaka, R**., Bora, N., & Khan, A. K. (2022). Anomaly-based intrusion detection using machine learning: An ensemble approach. International *Journal of Information Security and Privacy (IJISP),* 16(1), 1-15. (ESCI)
2. **Lalduhsaka, R**., Khan, A. K., & Chawngsangpuii, R. (2025). Hybrid Sampling Approach to Enhance Intrusion Detection System in IoT Networks. *International Journal of Computer Information Systems and Industrial Management Applications,* 17, 17-17. (SCOPUS)
3. **Lalduhsaka, R**., & Khan, A. K. (2025). Enhanced Intrusion Detection System Using a Two-Staged Feature Selection Method. Security and Privacy, 8(3), e70025. (ESCI)
4. **R**. **Lalduhsaka**, Ajoy Kumar Khan, Candy Lalrempuii, R Chawngsangpuii. "Ensemble based Zero-Day attack detection in Software Defined Networks using the InSDN dataset", Turkish Journal of Electrical Engineering & Computer Sciences. (Communicated)

Conference Paper Presented

1. Lalduhsaka, R., Khan, A. K., & Roy, A. K. (2021, October). Issues and challenges in building a model for intrusion detection system. In 2021 5th International Conference on Information Systems and Computer Networks (ISCON) (pp. 1-5). IEEE.
2. R. Lalduhsaka, Lalchhandama Lailung, & Ajoy K. Khan (2023). Study on Machine Learning based Intrusion Detection System on Software Defined Networks. *International Conference on Advanced Computing (ICAC 2023).* Bharathiar

PARTICULARS OF THE CANDIDATE

NAME OF THE CANDIDATE **: R. LALDUHSAKA**

DEGREE : Ph.D.

DEPARTMENT : COMPUTER ENGINEERING

TITLE OF THE THESIS : DEVELOPMENT OF EFFICIENT PROTOCOL FOR INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

DATE OF ADMISSION : 06/11/2020

APPROVAL OF RESEARCH PROPOSAL

1. DRC : 12th APRIL, 2021
2. BoS : 27th APRIL, 2021
3. SCHOOL BOARD : 20th MAY, 2021

MZU REGISTRATION NUMBER : 2001033

Ph.D. REFGISTRATION NUMBER : MZU/Ph.D./1654 of 06.11.2020)

EXTENSION : NIL

(Dr. VD AMBETH KUMAR)

HEAD

DEPARTMENT OF COMPUTER ENGINEERING

ABSTRACT

DEVELOPMENT OF EFFICIENT PROTOCOLS FOR INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

AN ABSTRACT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

R. LALDUHSAKA

MZU REGISTRATION NO.: 2001033

Ph.D. REGISTRATION NO.: MZU/Ph.D./1654 of 06/11/2020

DEPARTMENT OF COMPUTER ENGINEERING

SCHOOL OF ENGINEERING AND TECHNOLOGY

JUNE, 2025

DEVELOPMENT OF EFFICIENT PROTOCOLS FOR INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

BY

R. LALDUHSAKA

Department of Computer Engineering

Supervisor

Prof. AJOY KUMAR KHAN

Joint Supervisor

Dr. R. CHAWNGSANGPUII

Submitted

In partial fulfillment of the requirement of the Degree of Doctor of Philosophy in Computer Engineering of Mizoram University, Aizawl.

The rapid growth of internet connected devices and IoT technologies have led to an increased risk of sophisticated cyber threats. Traditional security tools like firewalls and antivirus software are insufficient against modern attacks such as zero-day exploits and ransomware. This thesis focuses on enhancing Intrusion Detection Systems (IDS) using Machine Learning (ML) to improve detection accuracy while reducing false positive rates (FPR), addressing key challenges like data imbalance, high dimensionality, and zero-day detection. A comprehensive ML based IDS framework is proposed, involving data preprocessing, feature selection (FS), resampling, and model evaluation. Multiple ML algorithms including Decision Tree (DT), Random Forest (RF), K-Nearest Neighbour (KNN), Logistic Regression (LR), Naïve Bayes (NB), QDA, and XGBoost are analyzed using standard metrics. Key contributions include: (1) an ensemble IDS combining NB, QDA, and DT for improved performance; (2) a two-stage feature selection method (GIGA) integrating Gini Impurity and Genetic Algorithm; (3) a hybrid resampling technique combining SMOTE and NearMiss to handle class imbalance in the Bot-IoT dataset; and (4) a zero-day detection approach using Leave-One-Attack-Out (LOAO) strategy on the InSDN dataset for SDN environments. Experimental results across multiple benchmark datasets (CIC-IDS2017, CSE-CIC-IDS2018, CIC-DDoS2019, Bot-IoT, InSDN) demonstrate significant improvements in accuracy, FPR, and computational efficiency. This work highlights the need for attack-specific models and adaptive ML integration in SDN to build practical, scalable, and robust IDS solutions. The introduction chapter is followed by the literature review chapter which study and discuss the recent researches on relevant areas.

Chapter 3 investigates the datasets used in this thesis, viz. CIC-IDS2017, CSE-CIC-IDS2018, CIC-DDoS2019, Bot-IoT, and InSDN dataset. They contain a wide variety of real world network attacks, including DoS, DDoS, brute force, infiltration, botnet, and more. Each dataset is described in terms of its structure, feature set, and class distribution, with a particular focus on their imbalance and diversity. Detailed analysis is conducted on the types of attacks present in these datasets to highlight their significance in IDS research.

Chapter 4 proposes lightweight ensemble-based IDS using MLg techniques to improve detection accuracy and reduce FPR. The model addresses key limitations of existing IDS such as high false positives, overfitting, and inefficiency in handling diverse attack types. The ensemble method integrates NB, QDAs, and ID3 as base classifiers, with RF serving as the meta-classifier. FS is performed using a Random Forest Regressor to reduce dimensionality and retain the most important attributes, significantly optimizing training and detection time. The proposed model is evaluated using two benchmark datasets: CIC-IDS2017 and CSE-CIC-IDS2018. These datasets include real-world traffic with a wide range of attack scenarios. Preprocessing steps such as data cleaning, normalization, and binary label conversion are applied to ensure consistency. FS reduces the feature space from over 80 features to the top 7 or 8, preserving more than 95% of cumulative importance. Experimental results show that the ensemble model outperforms individual classifiers in both accuracy and efficiency. It achieves an accuracy of 98.3% with a FPR of 2% on CIC-IDS2017, and 95.1% accuracy with a 6.9% FPR on CSE-CIC-IDS2018. Comparisons with existing approaches confirm its effectiveness in minimizing cost and time while maintaining high detection rates.

Chapter 5 addresses a critical challenge in ML based IDS for IoT; class imbalance in datasets such as Bot-IoT, where benign instances are vastly outnumbered by attack records. To overcome this, a novel hybrid sampling method combining Near Miss-1 undersampling and SMOTE oversampling is proposed, effectively rebalancing the dataset to an 80:20 attack to benign ratio. This strategy maintains data integrity while enhancing the minority class representation. The methodology includes preprocessing the Bot-IoT dataset, cleaning and encoding relevant features, and applying the hybrid sampling technique across the full dataset and specific attack subsets (DoS, DDoS, Scan, and Theft). Four classifiers viz. RF, LR, NB, and KNN are evaluated using standard performance metrics. Experimental results show that RF consistently outperforms other models in accuracy, precision, recall, and false positive/negative rates, achieving 99.98% accuracy and only 0.02% FPR on the full dataset. Interestingly, NB and LR perform better on the Theft subset due to its smaller size and reduced complexity. The proposed hybrid approach

demonstrates superior performance compared to state-of-the-art methods and proves effective in reducing both false positives and false negatives. This work not only enhances detection capability in highly imbalanced IoT scenarios but also suggests the value of multi-model IDS architectures tailored to specific attack types and dataset characteristics.

Chapter 6 presents a novel two-staged FS approach, termed GIGA, for enhancing the efficiency and accuracy of IDS in complex and high-dimensional network environments. With the rapid growth of SDN, IoT, and cloud infrastructures, the detection of cyber threats is challenged by massive volumes of heterogeneous and noisy data. The proposed method addresses the limitations of traditional IDS by combining a filter-based Gini impurity technique and a wrapper-based Genetic Algorithm (GA) to identify the most relevant features while minimizing computational costs and maintaining high detection performance. Experiments were conducted on three benchmark intrusion detection datasets: CIC-IDS2017, CSE-CIC-IDS2018, and CIC-DDoS2019. Initially, the Gini index reduced the number of features by more than half, followed by a GA-driven selection for further refinement. This two-stage FS process led to a nearly tenfold reduction in feature dimensions and significantly improved training times without compromising accuracy. Classifiers such as DT and RF achieved up to 99.98% accuracy with near to zero FNR, even in zero-day attack scenarios. The method's effectiveness was validated using 5-fold cross-validation, and comparisons with existing FS techniques demonstrated superior results.

Chapter 7 investigates the challenge of detecting zero-day attacks in SDN environments using ML approaches. Traditional IDS are often ineffective against emerging, unknown threats due to their reliance on predefined attack signatures. Leveraging the InSDN dataset, specifically curated to represent realistic SDN traffic and diverse attack types. This study evaluates the performance of two ML classifiers, RF and XGBoost, across multiclass, binary, and zero-day detection scenarios. A comprehensive methodology is employed, including preprocessing steps to clean and normalize data, and oversampling using SMOTE to address class imbalance. Zero-day detection is simulated using a LOAO approach, where one attack type is

excluded from training and used solely for testing alongside benign traffic. Both RF and XGBoost show high classification performance in multiclass and binary settings, with XGBoost demonstrating superior recall and F1-scores, particularly for minority classes. In zero-day scenarios, XGBoost consistently outperforms RF in detecting previously unseen attacks like DDoS, Probe, and U2R, achieving lower FNR. While SMOTE improves detection for underrepresented attack classes, it slightly impacts precision for benign instances. Security analysis confirms the robustness of the proposed models in identifying novel threats without reliance on prior attack signatures. Overall, this work demonstrates that combining supervised ML techniques with anomaly-based strategies and balanced datasets can effectively detect zero-day attacks in dynamic SDN environments. The LOAO strategy, supported by RF and XGBoost models, offers a reliable foundation for intelligent, anticipatory intrusion detection systems in programmable network infrastructures. This chapter is followed by Conclusion and Future Works.