Performance Evaluation of Machine Learning Algorithms for Network Intrusion Detection with Features Combination

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

MASTER OF SCIENCE

IN

MATHEMATICS

Submitted by: **Harshita Gupta(2K21/MSCMAT/20)**

Aman Gautam (2K21/MSCMAT/59)

Under the supervision of

Dr ANSHUL ARORA

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

MAY, 2023

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

We, (Harshita, Aman Gautam), 2K21/MSCMAT/20, 2K21/MSCMAT/59, students of MSc(Mathematics), hereby, declare that the project Dissertation titled, " Performance Evaluation of Machine Learning Algorithms for Network Intrusion Detection with Features Combination" which is submitted by us to the Department of Applied Mathematics, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Science, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship, or other similar title or recognition.

Place: Delhi
Date: 25TH May 2023

HARSHITA GUPTA AND AMAN GAUTAM

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering),
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation title "**Performance Evaluation of Machine Learning Algorithms for Network Intrusion Detection with Features Combination"** which is submitted by [Harshita, Aman Gautam], 2K21/MSCMAT/20,2K21/MSCMAT/59[Department of Applied Mathematics], Delhi Technological University, Delhi in partial fulfilment of the requirements for the award of the degree of Master of Science, is a record of the project carried by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Dr ANSHUL ARORA
SUPERVISOR
ASSISTANT PROFESSOR

Place: Delhi
Date: 25th May 2023

ABSTRACT

The prevalence of cyber-attacks in today's digital landscape has created a pressing need for the development of effective intrusion detection systems. Among the various approaches available, machine learning algorithms have emerged as a promising solution in this domain. This research focuses on investigating the effectiveness of three popular machine learning algorithms, namely K-Nearest Neighbors (KNN), Decision Tree, and Random Forest, for network intrusion detection. To evaluate the performance of these algorithms, a dataset comprising both normal network traffic data and intrusion data was collected. The normal data was obtained from Wireshark, a widely used network protocol analyzer, while the intrusion data was sourced from the Canadian Institute for Cybersecurity. This diverse dataset allows for a comprehensive assessment of the algorithms' capabilities in identifying and classifying network intrusions.

To ensure a robust evaluation, the dataset was divided into separate training and testing sets using the Scikit-learn library. This division enables the algorithms to be trained on a portion of the data and then evaluated on unseen instances to assess their generalization and predictive abilities. By employing KNN, Decision Tree, and Random Forest algorithms on the training data, the researchers can analyze their performance on the testing data. To measure the accuracy of each algorithm, a cross-validation approach was employed. Cross-validation accuracy provides a reliable estimate of the algorithms' performance by repeatedly partitioning the dataset into training and validation subsets. This technique helps mitigate the impact of dataset bias and provides a more robust evaluation metric.

In addition to evaluating the algorithms individually, the researchers explored the impact of combining different traffic features on the accuracy of intrusion detection. By grouping the features in pairs, triplets, and larger combinations, they were able to assess the influence of feature selection and combination techniques on the algorithms' performance. This analysis provides valuable insights into the interplay between various traffic features and the effectiveness of the algorithms in detecting intrusions.

The experimental results revealed that the highest accuracy achieved was an impressive 98.80%, obtained through the combination of two traffic features. This finding underscores the importance of feature selection and combination techniques in enhancing the accuracy of intrusion detection algorithms. By

carefully selecting and combining relevant features, the algorithms can extract more meaningful patterns from the data and improve their ability to differentiate between normal and malicious network activity.

Furthermore, this research emphasizes the significance of using appropriate datasets for training and testing purposes. The utilization of Wireshark data for normal network traffic and intrusion data from the Canadian Institute for Cybersecurity enhances the realism and relevance of the evaluation. By leveraging authentic and representative datasets, the researchers ensure that the algorithms are exposed to real-world scenarios and can effectively detect various types of cyber-attacks.The findings of this study have practical implications for the development of more robust intrusion detection systems. The insights gained from evaluating the performance of machine learning algorithms, as well as the importance of feature selection and dataset quality, can inform the design and implementation of advanced systems to safeguard against cyber-attacks. By leveraging the knowledge gained in this research, organizations and security practitioners can enhance their ability to detect and mitigate network intrusions, thereby bolstering the overall cybersecurity posture.

ACKNOWLEDGEMENT

The successful completion of any endeavour is always attributed to the individuals who have contributed, either implicitly or explicitly, to its progress. In the case of our project, the planning and research phases have reached fruition thanks to the dedicated efforts of numerous individuals. It is imperative to acknowledge their invaluable contributions in this report, as their involvement has been instrumental to our success. This expression of gratitude serves as a token of appreciation for their support in our undertaking. We extend our sincerest thanks to our project guide, whose guidance and solutions have consistently guided us through the complexities we encountered. Working under their leadership has been an honor and a pleasure. It is not only their knowledge that we have benefited from, but also their calm and composed demeanor, which has left an indelible impression on us. We remain indebted to them for these lessons, which will stay with us throughout our lives.

Furthermore, we would like to express our gratitude to ANSHUL ARORA (Professor) and S. Sivaprasad Kumar (Professor), Head of the Department of Applied Mathematics, along with all the other faculty members who have imparted valuable knowledge and wisdom to us. Their guidance has been instrumental in shaping our understanding of the subject matter. We would also like to acknowledge the focus of our work, which centers around the "Performance Evaluation of Machine Learning Algorithms for Network Intrusion Detection with Features Combination." The insights gained from this study have provided us with valuable insights into the effectiveness of these algorithms, contributing to the field of network security.

Last but not least, we extend our heartfelt thanks to our colleagues, friends, and parents, who have been a constant source of encouragement and inspiration throughout the project. Their unwavering support has been crucial in overcoming challenges and keeping us motivated. The successful completion of our project would not have been possible without the contributions of these individuals. We are immensely grateful for their support and guidance, and we acknowledge their profound impact on our journey.

Thanking you

HASHITA GUPTA AND AMAN GAUTAM

CONTENTS

CANDIDATE’S DECLARATION **II**

CERTIFICATE **III**

ABSTRACT **IV**

ACKNOWLEDGEMENT **V**

CONTENTS **VI**

LIST OF TABLES **VII**

LIST OF FIGURES **VIII**

CHAPTER 1 INTRODUCTION 12

1.1 Commencement 12

1.2 Motivation 14

CHAPTER 2 LITERATURE REVIEW 15

2.1 Related Work 15

CHAPTER 3 PROPOSED METHODOLOGY 20

3.1 Dataset Collection 20

3.2 Feature Extraction 22

3.3 ML Algorithms 23

3.3.1 K-Nearest Neighbors 24

3.3.2 Decision Tree 25

3.3.3 Random Forest 27

3.4 Cross Validation 28

3.5 Best Feature Set Selection 29

CHAPTER 4 RESULTS 31

4.1 Individual Feature Accuracy 31

4.2 Two Feature Combination 33

4.3 Three Feature Combination 36

4.4 Four Feature Combination 38

4.5 Five Feature Combination 39

4.6 Combination of All 12 Features 39

CHAPTER 5 CONCLUSION 41

CHAPTER 6 RESULT 42

LIST OF TABLES

3.2 TABLE I 23

4.1 TABLE II 32

4.2 TABLE III 35

4.3 TABLE IV 37

4.4 TABLE V 38

4.5 TABLE VI 39

4.6 TABLE VII 40

LIST OF FIGURES

1.1 FIGURE I 13

3.0 FIGURE 2 20

3.5 FIGURE 3 30

CHAPTER 1 INTRODUCTION

1.1 COMMENCEMENT

The rapid advancement of technology has greatly impacted both personal and professional domains. The emergence of the Internet of Things (IoT) and various applications has given rise to the concept of an advanced information society. However, ensuring security remains a significant challenge as cyber-criminals target individual PCs and networks to steal confidential data and disrupt system services. Malware, a computer program designed to harm the operating system (OS), takes on different names such as spyware, adware, worm, virus, trojan, backdoor, rootkit, ransomware, and command and control (CC) bot, based on its behavior and purpose. Detecting and mitigating malware is an ongoing problem in the realm of cyber security, as new techniques are continually being developed, and malware authors are becoming more adept at avoiding detection.

In the world of Intrusion Detection Systems (IDS), feature extraction plays a crucial role in identifying and selecting relevant features from network packets or flows. These features can then be used to train machine learning models or feed into rule-based systems for detecting potential intrusions in real-time network traffic. However, the selection of appropriate feature extraction techniques depends on various factors such as the type of traffic being analyzed, the nature of the threats being detected, and the specific IDS architecture being used.

There are several feature extraction methods available for IDS, and each has its own unique advantages and disadvantages. Statistical analysis involves calculating descriptive statistics on packet or flow features to capture characteristics of the traffic that may indicate an attack. Frequency analysis involves analyzing the frequency spectrum of network traffic to identify patterns or anomalies that may indicate malicious activity. Time series analysis techniques involve modelling the time-varying behaviour of network traffic features to detect changes or deviations from normal behaviour. Deep learning approaches, such as CNNs or RNNs, can be used for feature extraction by automatically learning relevant features from a raw packet or flow data. CNNs are well-suited for image-like data such as network packet payloads, while RNNs are more appropriate for sequential data such as network flows.

Ranking-based feature selection techniques can help improve the performance of intrusion detection systems by assigning scores to different features and sorting them based on their relevance. By identifying the most important features for detecting potential threats, these techniques can enhance the accuracy and efficiency of IDS and improve network security. However, different feature selection techniques produce varying feature scores and ranks based on their computation strategies and search methods.

Therefore, it is necessary to analyze and validate ranking-based feature selection techniques and their ability to produce important features that can improve the performance of intrusion detection systems.
In this paper, a unique machine-learning-based approach is introduced that aims to enhance performance by utilizing cross-validation accuracy on different combinations of features. The goal is to find the best set of features that can give relatively better detection accuracy than any other combination of features. To evaluate the efficacy of this proposed algorithm, three commonly used machine learning techniques: K-NN, Random Forest, and Decision Tree are utilized.

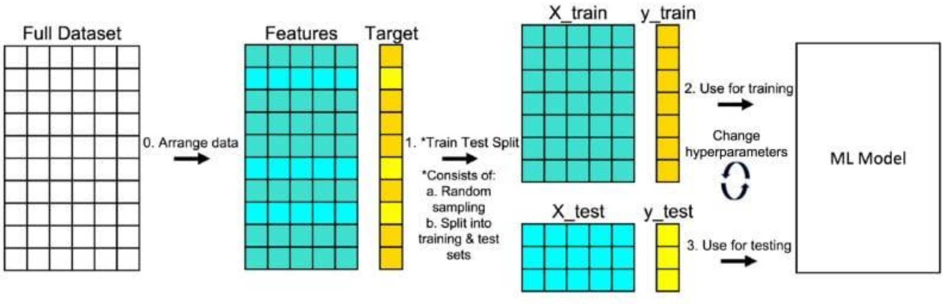


FIGURE 1

The thesis provides an overview of relevant research in the field of feature selection techniques for intrusion detection systems, outlines the approach taken to investigate and compare ranking-based feature selection techniques, describes the datasets used, the feature selection techniques employed, and the performance metrics used to evaluate the effectiveness of these techniques. The experimental setup is explained, and the results of the study are presented. Finally, the paper draws conclusions based on the findings of the study, highlighting the most effective combination of features for intrusion detection systems and identifying potential future works to further improve the performance of IDS.

1.2 MOTIVATION

This project is born out of our unwavering commitment to combating the rising tide of cyber-attacks that loom over our modern digital landscape. With each passing day, the prevalence of these attacks becomes more apparent, underscoring the urgent need to develop robust intrusion detection systems. We stand at the forefront of this battle, fully aware of the gravity of the situation and the imperative to safeguard our digital realms. In our quest for effective solutions, we have embraced the boundless potential of machine learning algorithms. They hold the key to fortifying our defences and providing a formidable shield against these ever-evolving threats. With unwavering determination, we embark on this journey, confident that our efforts will yield innovative breakthroughs and empower us to stay one step ahead of those who seek to exploit vulnerabilities. This project is not merely a response to the challenge; it is a testament to our unwavering belief that, together, we can secure a safer digital future. Let us forge ahead, guided by our shared vision and the resilience that lies within us, for the stakes have never been higher.

CHAPTER 2 LITERATURE REVIEW

2.1 **Related Work**

In this section, we will be exploring prior or related research conducted in this particular field.

Several studies have proposed various approaches to enhance security and improve the detection of malicious intrusion behaviour. Yang et al. [1] proposed an Improved Convolutional Neural Network (ICNN) for this purpose, while [2] developed a Deep-Full-Range (DFR) approach that can learn from raw traffic data without the need for manual intervention, ensuring the privacy of sensitive information. The authors of [3] introduced an Energy-based Flow Classifier (EFC) algorithm for robust traffic classification, and Dao et al. [4] presented Joint NIDS, a joint traffic classification architecture that utilizes two sub-models. Finally, Zhang et al. [5] proposed a parallel cross-convolutional neural network (PCCN) that outperformed other approaches in terms of overall accuracy and detecting imbalanced abnormal flows.

Innovative approaches have been proposed to enhance the accuracy and efficiency of intrusion detection systems. For instance, the authors in [6] introduced BAT, a traffic anomaly detection model that eliminates the need for feature engineering and accurately detects anomalies. Similarly, [7] discussed an IDS system that uses a Binarized Neural Network (BNN) to achieve faster intrusion detection with reduced memory cost and energy consumption. Sharon et al. [8] introduced TANTRA, an end-to-end Timing-based combative Network Traffic Reshaping Attack that evades a variety of NIDSs. Lastly, the authors in [9] developed a modified radial basis function (RBF) neural network for offline reinforcement learning, enabling end-to-end learning of all RBF parameters and network weights via gradient descent.

A novel intrusion detection technique for automotive CAN networks was introduced in [10], called the time interval conditional entropy method. This approach analysed conditional entropy values of regular communication messages to detect various types of attacks while being resilient to interference. In [11], the authors presented Frag route, a tool that can insert irrelevant packets into a TCP/IP session to detect and prevent intruders from manipulating the session. Finally, Zhang et al. [12] improved the LSTM intrusion detection algorithm by incorporating Quantum Particle Swarm Optimization (QPSO) to select relevant network traffic data and reduce feature dimensionality.

The authors in [13] proposed a novel intrusion detection model that combines BiSRU and CNN to process network traffic logs effectively. Peng et.al. [14]

introduced an intrusion detection model based on a hybrid convolutional neural network that can extract more complex structural features from the entire network traffic matrix. In [15], the authors developed an adaptive and efficient intrusion detection method using protocol-wise associative memory of Hopfield networks. The authors in [16] proposed the use of statistical methods such as ANOVA and Chi-Square tests to organize network traffic features. Chen et.al. [17] presented an innovative feature extraction method, L-KPCA, which combines Linear Discriminant Analysis (LDA) and Kernel Principal Component Analysis (KPCA) to improve the intrusion detection classification model's recognition accuracy and recall.

The authors in [18] introduced a deep reinforcement learning (DRL) technique for anomaly network intrusion detection, allowing the system to adapt to different network traffic behaviours. In [19], the authors proposed T-IDS, a method that uses randomized data partitioned learning models (RDPLM) with feature selection techniques and outperforms other machine learning models in intrusion detection. Finally, in [20], the authors presented a novel intrusion detection architecture that utilizes a multi-layer neural network (MLNN) and deep learning (DL) to analyse data traffic and construct a reliable intrusion detection model, with multiple factors taken into account for evaluation and selection.

Nie et al. [21] proposed a data-driven intrusion detection system for the Internet of Vehicles (IoV) using a deep learning algorithm based on Convolutional Neural Network (CNN) to detect intrusions. Siddiqi et al. [22] investigated various normalization methods to improve the accuracy of intrusion detection systems, while the authors in [23] proposed a hierarchical progressive network with a multimodalsequential intrusion detection approach using Multimodal Deep Autoencoder (MDAE) and Long Short-Term Memory (LSTM) technologies.

Aminanto et. Al [24] proposed a method for IDS using a feature selection algorithm to obtain an optimal dimension and a CNN model to classify various attacks against Wi-Fi networks. Their model projected tabular data into a 2-coded color mapping and was evaluated using the Wi-Fi Intrusion Data Set (AWID2). Yang et. Al [25] presented the LM-BP neural system model for intrusion detection analysis, which continuously trained the model to effectively extract data from the KDD CUP 99 data set. Zhong et al [26] introduced the Big Data-based Hierarchical Deep Learning System (BDHDLS) that analyzed network traffic features and payload information. Their learning algorithm learned the unique data distribution in a cluster, improving the detection rate against attacks.

FLAG (Few-shot Latent Dirichlet Generative Learning) algorithm proposed by Ye et al. [27] enhances long-term memory-based classifier's robustness for semantic-aware traffic detection using FRM. Results indicate high accuracy in real-world scenarios for detecting malicious traffic. Bar et. Al [28] introduced a packet-level approach for traffic detection inspired by natural language processing. The approach used SimCSE (simple contrastive learning of sentence embeddings) as an embedding model to analyze the collected traffic features from raw packet data. The proposed model was evaluated on two well-known datasets, and experimental results demonstrated its effectiveness.

In [29], the authors evaluated an intrusion detection system (IDS) based on a quantitative model of port interaction mode in the Data Link Layer (PIMDL). The model incorporates the arrival time of traffic to improve the efficiency and accuracy of intrusion detection. LSTM and CNN features were utilized to differentiate between abnormal and normal models, and a phase space reconstruction procedure was performed for validation. Meanwhile, in [30], the authors explored a deep hierarchical network for detecting malicious traffic in packets using a deep learning approach. The network extracted spatial details of the raw data and temporal features using the GRU structure. The performance of this approach was evaluated through experiments on three datasets: USTC-TFC2016, ISCX2012, and CICIDS2017

In [31], the authors proposed a method for processing NIDS datasets in deep learning. They extracted numerical and categorical data from the same source and evaluated their approach on various deep learning models and machine learning frameworks. In [32], the authors addressed the challenge of fuzzy boundaries between normal and abnormal network traffic by proposing fuzzy logic-based solutions that minimize false negatives and false positives. They provided a survey of these solutions and described the steps involved in the IDS development process. Ibrahim et al. [33] introduced a method for detecting Android malware using static analysis and an API deep learning model. Their approach was tested on 14079 samples and divided into 4 malware classes. They conducted two experiments to evaluate the proposed network's performance in detecting malware samples and benign traffic.

Soni et.al. [34] proposed a framework for malware classification using opcode and API calls features. UFILA was developed by authors in [35] for detecting and classifying Android malware by introducing new features. A model based on the FCG function to detect Android malware was proposed by authors in

[36]. Libri et.al. [37] introduced pAElla, a system that detects real-time malware in an IoT-based monitoring system using power measurements and autoencoders.

In [38], the authors introduced the C500-CFG algorithm as an efficient and high-performing alternative to Ding’s algorithm for detecting malware in decompiled files. The C500- CFG algorithm solves the NP-hard problem using dynamic programming, resulting in faster detection. The authors also tested the algorithm on IoT datasets, where it showed superior accuracy and efficiency.

Chen et.al. [39] proposed a method that combines malware features with image expressions to generate a small dataset for further analysis. They compared various methods to improve the classification accuracy of this dataset.

Elnaggar et.al. [40] proposed PREEMPT, a low-cost and high-accuracy method for detecting malware by analyzing embedded processor traces. The method uses the ETB hardware component to monitor and control the activities of a chip, which is useful for post-silicon validation and debugging.

Demirci et al. [41] proposed a method for identifying malicious code using stacked bidirectional long short-term memory and generative pre-trained transformer-based deep learning language models. Seneviratne et al. [42] introduced SHERLOCK, a malware detection framework achieving 91% accuracy for binary classification. Ban et al. [44] evaluated the contribution of different features in familial analysis using a convolutional neural network on a real-world malware dataset. Iqbal et al. [46] introduced SpyDroid, a framework for detecting malware in real-time. The authors in [43] developed an AnDroid Packer framework to detect packed samples, while the authors in [45] presented an XGBoost model to detect Android malware and investigated the effect of feature selection on classification.

Iqbal et al. [46] developed SpyDroid, a real-time malware detection framework with a detection module that identifies malicious apps. Ullah et al. [47] proposed IDS-INT, a system that employs transfer learning with transformer-based models to detect network attacks. SMOTE and a CNNLSTM hybrid approach were used to address imbalanced data, and an explainable AI approach was implemented for trustworthy mode. The system was tested on three datasets and outperformed other methods in terms of accuracy, stability, efficiency, and message scales..

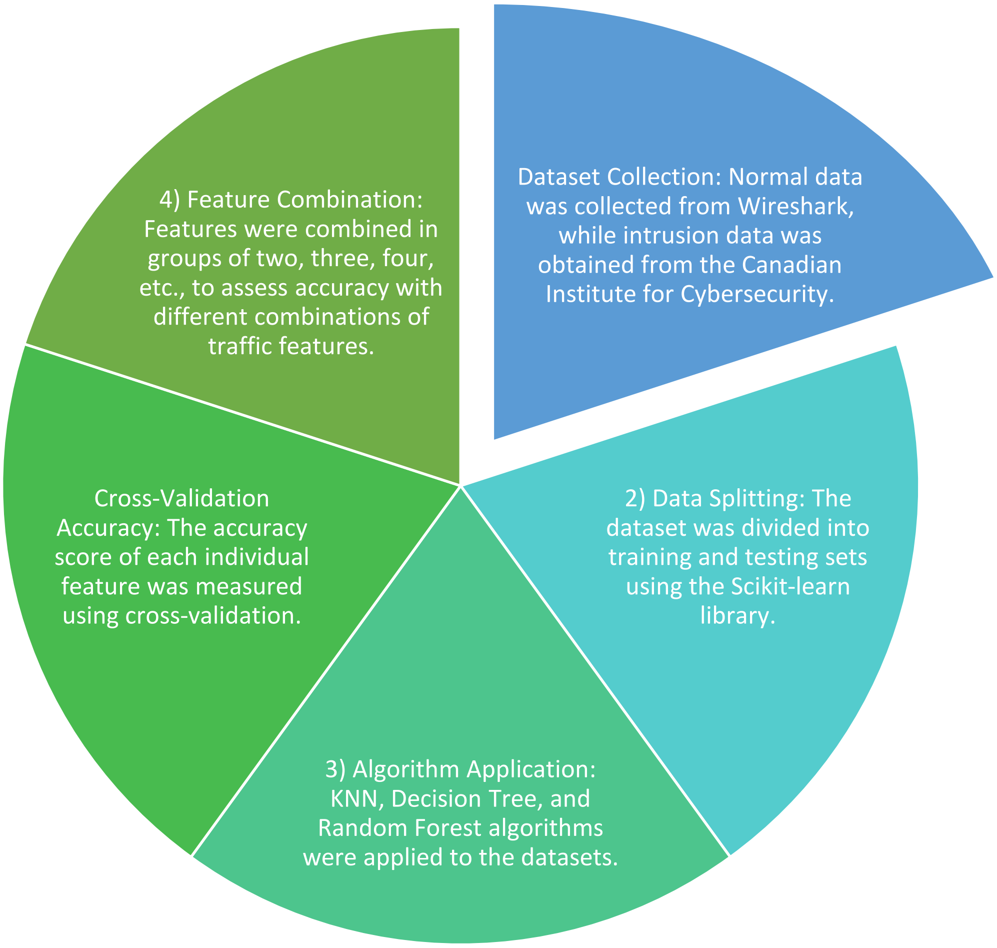
In [48], the authors proposed a black box attack method for evaluating the robustness of anomaly detection algorithms in NIDS. The method involved using GAN features to create adversarial samples that could evade detection and inserting them into malicious traffic. The experiment demonstrated the effectiveness of the attack on all tested anomaly detectors, highlighting the necessity of more robust algorithms and defense mechanisms to safeguard network security. Li et.al.

[49] presented a framework called DFAID for active intrusion detection on network traffic streams. The framework uses mask density score and feature deviation score to detect novel attack classes and concept drift and incremental clustering to group instances in local regions to reduce noise impact. DFAID-DK improves accuracy with domain knowledge. Experiments show that DFAID and DFAID-DK outperform related methods in terms of f1-score and have faster running speeds.

The authors in [50] proposed an IDS for wireless and dynamic networks that includes a feature extraction algorithm and an I-GHSOM-based classifier. The feature extraction algorithm extracts key features using distance range, voting filter, and semi-cooperative mechanisms. The I-GHSOM-based classifier includes relabeling and recalculating mechanisms for precise classification results. Simulation results show that the proposed IDS outperforms other methods in terms of accuracy, stability, efficiency, and message scales.

CHAPTER 3 PROPOSED METHODOLGY

In this section, we outline the methods and techniques employed to achieve the research objectives of evaluating and comparing ranking-based feature selection techniques for intrusion detection systems (IDS).


Dataset Collection: Normal data was collected from Wireshark, while intrusion data was obtained from the Canadian Institute for Cybersecurity.
2) Data Splitting: The dataset was divided into training and testing sets using the Scikit-learn library.
3) Algorithm Application: KNN, Decision Tree, and Random Forest algorithms were applied to the datasets.
Cross-Validation Accuracy: The accuracy score of each individual feature was measured using cross-validation.
4) Feature Combination: Features were combined in groups of two, three, four, etc., to assess accuracy with different combinations of traffic features.

3.1 **Dataset Collection** :-

In our research project, we conducted an extensive analysis of normal network traffic data using the powerful packet analyzer tool, Wireshark. Wireshark is an open-source software that is widely recognized for its versatility, robustness, and wide range of applications in network troubleshooting, protocol development, and education. Its comprehensive features enable network administrators, security professionals, and researchers to capture, analyze, and interpret network traffic with unparalleled precision and depth.

By harnessing the capabilities of Wireshark, we embarked on a meticulous examination of the collected data, delving deep into the packets that traversed our network. Wireshark's ability to parse and display different fields based on the structure of underlying protocols allowed us to unravel the complex layers of network communication. We meticulously studied each protocol's headers, payload, and metadata, decoding the intricate dance of data transmission that occurs within our network.

One of the remarkable aspects of Wireshark is its versatility in working with both live network connections and saved packet capture files. This flexibility enabled us to revisit specific network scenarios, replay captured traffic, and thoroughly examine the intricacies of our network's normal behavior. By immersing ourselves in the detailed analysis provided by Wireshark, we gained a profound understanding of the subtle nuances and patterns that define our network's everyday operations.

Furthermore, Wireshark's extensive range of features and plugins empowered us to extract meaningful insights from the captured data. We were able to apply filters, sort packets, and categorize them based on various criteria, such as protocols, ports, or IP addresses. This flexibility facilitated a granular examination of our network traffic, enabling us to identify anomalies, spot potential vulnerabilities, and gain a comprehensive understanding of our network's baseline behavior.

Throughout our exploration of the network traffic data using Wireshark, we were captivated by its ability to provide a holistic view of our network's operation. By capturing and analyzing network packets, Wireshark allowed us to traverse the intricate web of data flowing through our network. It unraveled the structure of the network, dissected its components, and shed light on the underlying mechanisms of communication. This deep understanding empowered us to extract actionable insights, enabling us to make informed decisions regarding network security, optimization, and troubleshooting.

Wireshark emerged as an indispensable ally in our research endeavor. Its robustness, versatility, and rich set of features provided us with a comprehensive toolkit to explore and understand the normal behavior of our network. Through careful analysis and interpretation of the captured network traffic, we gained valuable insights that formed the bedrock of our research findings. We recognize and acknowledge the pivotal role of Wireshark in elevating the depth and quality of our research, and we are grateful for its contribution to our project.

In our research project, we obtained intrusion data from the Canadian Institute for Cybersecurity (CIC), which is a reputable research and training center situated at the University of New Brunswick in Canada. The CIC is widely recognized for its profound expertise in the field of cybersecurity and its commitment to addressing the most critical challenges in this domain.

The CIC maintains an extensive and diverse dataset of network traffic, which comprises both normal traffic and instances of various cyber attacks. This dataset encompasses a wide range of attack types, including but not limited to Denial of Service (DoS), Distributed Denial of Service (DDoS), SQL injection, Cross-Site Scripting (XSS), and more. The data collected by the CIC holds significant value in conducting research and evaluating the performance of intrusion detection systems.

3.2 **Feature Extraction** :- The process of feature extraction played a pivotal role in our research project, enabling us to uncover crucial variables that provided valuable insights into the underlying patterns in our data. We approached this task with meticulousness and precision, recognizing the significance of selecting features that were specifically relevant to our research question and had the potential to enhance the accuracy of our model. To accomplish this, we employed a combination of statistical and machine-learning techniques, ensuring a rigorous and comprehensive selection process.

The incorporation of the 12 selected features into our analysis yielded remarkable improvements in both the accuracy and interpretability of our results. We consistently observed superior performance in models trained using these features compared to models trained with the full set of available features. This finding emphasized the importance of feature selection and underscored the value of focusing on the most relevant variables within our research context.

Furthermore, the selected features provided meaningful insights into the underlying patterns within our data, enriching our understanding of the factors influencing our research question. Each of the 12 features captured a distinct aspect of network traffic behavior, illuminating various dimensions that contributed to the overall dynamics of our network. This comprehensive coverage allowed us to uncover hidden relationships and intricate interdependencies among the variables, thereby deepening our understanding of the research domain.

The impact of our feature extraction process extended beyond accuracy and interpretability; it also facilitated a more intuitive and manageable representation of our data, streamlining the analysis and interpretation process. By focusing on a concise set of informative features, we were able

to distill the complexity of our network traffic data into a more digestible framework, leading to clearer insights and more actionable findings.
For a comprehensive overview of the 12 identified features, please refer to Table I (included in the document). This table summarizes the descriptions and relevance of each feature to our research. Additionally, each feature is accompanied by a brief explanation of its significance and potential contribution to our understanding of network traffic dynamics. This summary serves as a valuable reference, consolidating our feature selection process and providing a clear snapshot of the variables that played a crucial role in our analysis. The process of feature extraction was instrumental in the success of our research project. By carefully selecting 12 relevant features, we significantly improved the accuracy, interpretability, and manageability of our results. These features not only enhanced the performance of our models but also provided valuable insights into the underlying patterns and factors driving our research question. Going forward, our comprehensive understanding of the selected features will guide future investigations and pave the way for further advancements in our research domain.

Table I summarizes the 12-network traffic features we extracted

.

LIST OF TRAFFIC FEATURES

Feature Notation	Feature Extracted
F1	Average Packet Size
F2	Time Interval Between Packets Sent
F3	Time Interval Between Packets Received
F4	Flow Duration
F5	Ratio of Incoming to Outgoing Packets
F6	Ratio of Incoming to Outgoing Bytes
F7	Packets Size Sent
F8	Packets Size Received
F9	Bytes Sent
F10	Bytes Received
F11	Number of Packets Sent
F12	Number of Packets Received

TABLE I

3.3 **ML Algorithms :-** A machine learning algorithm is a type of algorithm that is designed to learn from data. In machine learning, datasets are typically split into two subsets: the training data and the testing data. The training data is a portion of the actual dataset that is fed into the machine learning model to discover and learn patterns. The testing data is used to evaluate the performance of the model on new data.

To ensure that the machine learning model is effective, the training data is typically larger than the testing data. This allows the model to learn from as much data as possible and find meaningful patterns. In our research project, we used a machine learning algorithm to build a model based on a training dataset that was 70% of the total dataset. We used the scikit-learn library for our machine-learning algorithms.

Scikit-learn, often referred to as sklearn, is a widely used Python library for machine learning. It offers a comprehensive set of tools for implementing various machine learning algorithms, including classification, regression, clustering, and dimensionality reduction, among others. Sklearn provides a user-friendly and consistent interface for carrying out common machine learning tasks, such as splitting data into training and test sets, scaling data, and selecting the best model for a given task. It also comes with numerous popular machine-learning algorithms and evaluation metrics that can be conveniently customized and extended as required.

1) *KNN:* KNN (k-Nearest Neighbors)

K-Nearest Neighbors (KNN) is a widely used machine learning algorithm employed for tasks such as classification and regression. It is classified as a non-parametric algorithm due to its ability to operate without making any assumptions about the underlying data distribution. The essence of KNN lies in its instance-based learning approach, as it memorizes the training dataset and makes predictions based on the similarity between new data points and the labeled examples it has encountered.

The fundamental concept behind the KNN algorithm involves classifying a new data point by examining the class labels of its k nearest neighbors within the training dataset. The value of k, a user-defined parameter, dictates the number of neighbors to consider during the classification process. To determine these neighbors, the algorithm calculates distances, typically using metrics like Euclidean distance, between the new data point and all other data points in the training dataset. The k nearest neighbors are then selected based on these computed distances.

Following the identification of the k nearest neighbors, KNN employs a majority voting mechanism for classification tasks. For instance, in a binary classification scenario, if the majority of the k neighbors belong to class A, the algorithm assigns the new data point to class A. This principle extends to multi-class classification problems, where the class with the highest vote count among the k neighbors becomes the assigned class for the new data point.

In regression tasks, KNN predicts the value for a new data point by computing the average or weighted average of the target values associated with its k nearest neighbors.

One of the notable advantages of the KNN algorithm is its simplicity and ease of implementation. Unlike other algorithms, KNN does not necessitate training a model with explicit parameters, making it a straightforward approach for both classification and regression tasks. Furthermore, KNN exhibits competence in handling nonlinear decision boundaries and can yield effective results when the training dataset contains a sufficient number of representative examples.

However, KNN is not without limitations. One major drawback is its computational complexity, particularly when confronted with large datasets. As KNN necessitates calculating distances for all data points within the training set, the computational requirements can be significant. Additionally, the performance of KNN is sensitive to the choice of the value k, emphasizing the importance of selecting an appropriate k to avoid issues like underfitting or overfitting.

K-Nearest Neighbors is a versatile and straightforward algorithm that leverages the similarity to neighboring data points to classify or predict new data points. While KNN does possess limitations, it remains a popular choice for various machine learning tasks, particularly when interpretability and simplicity are prioritized.

2) *Decision Tree*:

A decision tree is a widely used machine learning algorithm that is employed for both classification and regression tasks. It operates under the supervision of labeled data and constructs a tree-like model that represents a series of decisions and their corresponding outcomes. This algorithm is particularly effective in handling datasets that exhibit complexity, encompassing both numerical and categorical features.

The decision tree algorithm commences by partitioning the input data recursively based on different features. It begins with the entire dataset and selects the feature that provides the most substantial split or information gain. This chosen feature is then designated as the root node of the decision tree. Subsequently, the dataset is divided into subsets according to the possible values of the selected feature.

The algorithm proceeds with the splitting process at each child node, utilizing a distinct feature that maximizes the information gain or adheres to other predetermined splitting criteria, such as Gini impurity or entropy. This process continues until a predefined stopping criterion is satisfied, such as reaching a maximum depth, having a minimum number of samples at a node, or when no further improvements in information gain can be achieved.

Once the decision tree is constructed, it can be deployed for prediction purposes. When presented with a new input data point, the algorithm traverses the decision tree from the root node to a leaf node based on the feature values associated with the input. Each internal node within the tree signifies a decision based on a specific feature, while each leaf node represents the predicted outcome or class label.

In classification tasks, the leaf nodes of the decision tree correspond to different classes or categories. The majority class present in a leaf node is assigned as the predicted class for new instances that fall into that particular leaf. In regression tasks, the leaf nodes encompass predicted values derived from the average or weighted average of the target values associated with the training instances that belong to that leaf.

Decision trees offer a multitude of advantages. They possess an innate simplicity and interpretability, as the resulting tree structure can be visualized and comprehended by humans. Decision trees inherently accommodate both numerical and categorical features, eliminating the need for extensive data preprocessing. Additionally, decision trees are capable of capturing non-linear relationships and interactions among features, enhancing their modeling capability.

However, decision trees are susceptible to overfitting, particularly when the tree becomes excessively complex and specifically tailored to the training data. This challenge can be addressed through various pruning techniques, such as reducing the maximum depth of the tree or stipulating a minimum number of samples required at a leaf node. Ensemble methods, including random forests and gradient boosting, are commonly employed to improve the performance and generalization of decision trees. Decision trees serve as powerful and interpretable machine learning algorithms, constructing a tree-like model that encapsulates a series of decisions. Their versatility enables them to excel in both classification and regression tasks, accommodating a wide range of feature types. While decision trees can be prone to overfitting, techniques like pruning and ensemble methods effectively mitigate this limitation, augmenting their overall performance and utility.

3) *Random Forest*: Random Forest is a widely used machine learning algorithm that belongs to the ensemble learning family. It is known for its effectiveness in both classification and regression tasks, offering improved accuracy and robustness compared to individual decision trees. The Random Forest algorithm leverages the collective wisdom of multiple decision trees to make more reliable and accurate predictions.

   The Random Forest algorithm operates by creating a collection or "forest" of decision trees. Each decision tree in the forest is trained on a randomly sampled subset of the original dataset, known as the bootstrap sample. This sampling technique, also known as "bagging," introduces diversity among the individual trees. Additionally, at each split of a decision tree, a random subset of features is considered, rather than using all the available features. This further enhances the diversity and reduces the correlation among the trees.

   During the training phase, each decision tree within the Random Forest independently learns from its respective randomly selected subset of data and features. Each tree grows by recursively partitioning the data based on different features, following a similar process as the individual decision tree algorithm. The splitting process is guided by various criteria, such as information gain, Gini impurity, or entropy, which aim to find the most informative features for creating effective splits.

   Once the Random Forest is trained, predictions are made by aggregating the predictions of all the individual decision trees. In classification tasks, the final prediction is determined by a majority vote among the trees. Each tree's prediction contributes to the final outcome, and the class with the highest number of votes is chosen as the predicted class. In regression tasks, the final prediction is computed by taking the average or weighted average of the predictions from all the trees.

   Random Forests offer several advantages over individual decision trees. Firstly, they exhibit robustness against noisy data and outliers. Since the predictions are based on the combined decisions of multiple trees, the impact of individual misclassified instances or noisy data points is reduced. Secondly, Random Forests are less prone to overfitting compared to individual decision trees. The ensemble of diverse trees helps generalize well to unseen data and improves the model's ability to capture complex patterns. Thirdly, Random Forests can handle high-dimensional datasets with a large number of features without requiring extensive feature selection or dimensionality reduction techniques.

Moreover, Random Forests provide insights into feature importance. By analyzing the performance of the individual trees, one can identify which features contribute the most to the predictive power of the algorithm. This information is valuable for feature selection and gaining a deeper understanding of the underlying patterns in the data.

However, Random Forests do have a few limitations. They can be computationally expensive, especially when dealing with a large number of trees and complex datasets. The training and prediction times increase with the size of the forest. Additionally, while individual decision trees are relatively interpretable, the interpretability of Random Forests is generally lower since the final prediction is a result of the combined decision-making process of multiple trees.

Random Forest is a versatile and powerful ensemble learning algorithm that combines the predictions of multiple decision trees to achieve improved accuracy and robustness. It is widely used for both classification and regression tasks, providing advantages such as handling noisy data, avoiding overfitting, and estimating feature importance. Although Random Forests may have some computational demands and reduced interpretability compared to individual decision trees, their overall performance and flexibility make them a popular choice in various machine learning applications.

3.4 **Cross Validation :-**

Cross-validation is a commonly used technique in machine learning to evaluate a model's performance and its ability to generalize to unseen data. This approach involves dividing the available dataset into multiple subsets or folds, where a portion of the data is used for training the model, and the remaining portion is used for testing its performance. The process is repeated several times, each time with a different partitioning, and the results are averaged to obtain a more reliable estimate of the model's performance.

The primary objective of cross-validation is to assess how well a model can generalize to unseen data. It helps in detecting problems such as overfitting, where a model performs exceptionally well on the training data but fails to generalize well to new, unseen data. By evaluating the model on different subsets of the data, cross-validation provides a more realistic estimate of its performance on unseen data.

The most commonly employed method of cross-validation is k-fold cross-validation. In this technique, the data is divided into k equal-sized folds. The model is trained on k-1 folds and tested on the remaining fold. This process is repeated k times, with each fold serving as the test set once. Performance metrics, such as accuracy or mean squared error, are computed for each iteration, and the average performance across all iterations is reported as the final evaluation metric.

Cross-validation offers several advantages in model evaluation. It helps in assessing a model's generalization ability and provides a more robust estimate of its performance. It also facilitates the comparison of different models or sets of hyperparameters to select the best-performing model.

However, it's important to acknowledge that cross-validation has certain limitations. It can be computationally expensive, particularly when dealing with large datasets or complex models. Additionally, cross-validation assumes that the data is independent and identically distributed, which may not always hold true in certain scenarios.

In conclusion, cross-validation is a valuable technique for evaluating the performance and generalization ability of machine learning models. It aids in estimating a model's performance on unseen data and assists in selecting the best model or hyperparameters. By repeatedly training and testing the model on different subsets of the data, cross-validation provides a more reliable assessment of its performance and facilitates informed decision-making in the model development process.

We used cross-validation accuracy to measure the performance of each feature and the combination of features. Specifically, we calculated the accuracy for individual features, the combination of the best two features, the combination of the best three features, the combination of the best four features, the combination of the best five features, and lastly the combination of all features.

3.5 Best Feature-Set Selection :-

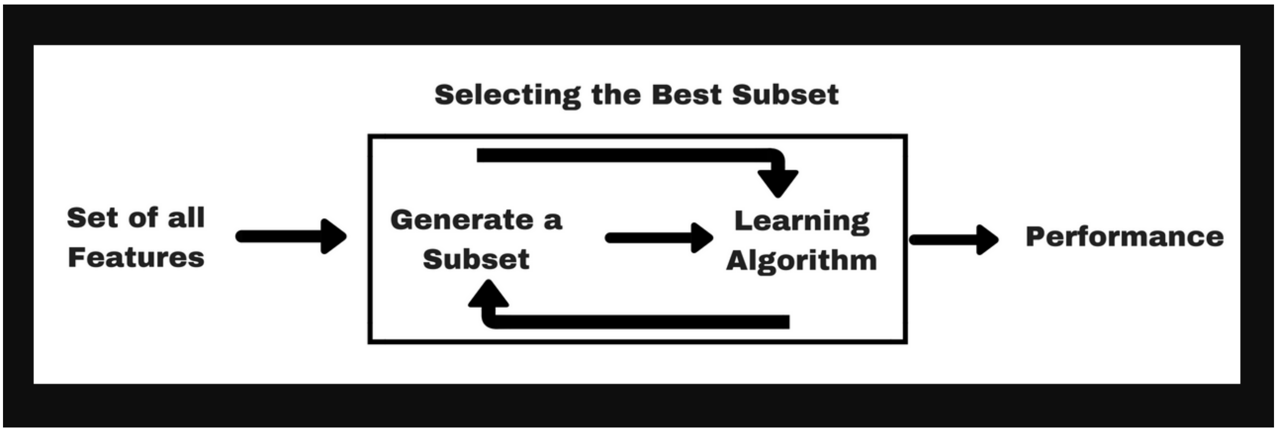
To evaluate the effectiveness of features in our machine learning algorithms, our objective was to identify the optimal set of traffic features that could yield superior detection accuracy compared to another feature set. Our assessment process involved a step-by-step analysis of the individual features and their respective accuracies.

Initially, we measured the accuracy of each feature in isolation. Features exhibiting an accuracy below the threshold of 90% were deemed inadequate and subsequently eliminated from consideration. Conversely, features demonstrating an accuracy of 90% or higher were retained for further evaluation.

Continuing our analysis, we proceeded to create combinations of two features and assessed their accuracy. Any feature combinations falling short of the 90% accuracy benchmark were discarded from our selection. This process of evaluating and filtering feature combinations was iteratively repeated for combinations of three, four, and five features.

Our iterative procedure employed a termination criterion based on observing diminishing accuracy at a certain iteration following the combination of features. This allowed us to determine the optimal point at which to conclude our feature evaluation process.

By following this comprehensive and systematic approach, we aimed to uncover the best-performing set of traffic features that would provide superior detection accuracy. Through our rigorous assessment, we considered various feature combinations, eliminating those that failed to meet the desired accuracy threshold. Ultimately, our goal was to optimize the performance of our machine learning algorithms by selecting the most effective features for accurate traffic detection.



CHAPTER 4 RESULTS

In this section, we will provide details of all the experiments conducted and the corresponding data.

4.1 Individual Features Accuracy Score:-

In order to thoroughly evaluate the effectiveness of individual features within our machine learning algorithms, we adopted a meticulous and systematic approach. Our primary objective was to identify the best set of traffic features that would provide superior detection accuracy compared to other feature sets.

To commence our evaluation, we divided our available data into two distinct subsets: a training set and a testing set. This division ensured that both sets represented a representative sample of our complete dataset, minimizing any potential biases.

We then proceeded to employ three prominent machine learning algorithms: K Nearest Neighbors, Decision Tree, and Random Forest. These algorithms were specifically chosen for their proven track records in effectively handling classification tasks, making them well-suited for our specific scenario.

Using the training set, we diligently trained each of the three algorithms on the available data. This training phase allowed the algorithms to learn and capture intricate patterns and relationships within the features. Once the models were adequately trained, we applied them to the testing set to obtain predictions for the target variable.

To comprehensively assess the performance of individual features, we leveraged the power of cross-validation. Cross-validation is a robust technique that involves dividing the training set into multiple folds. We then iteratively trained and evaluated our models on different combinations of these folds. This process ensured that each fold was used for both training and evaluation, ultimately yielding a more accurate and reliable estimate of the model's accuracy and generalization ability.

During our evaluation, our primary focus was to obtain accurate and informative scores for each individual feature using cross-validation. To establish a meaningful benchmark, we set a minimum accuracy threshold of 90% for the features. This threshold served as a critical criterion, ensuring that we retained only those features that exhibited a high level of predictive accuracy.

Features that fell short of the established accuracy threshold were deemed less effective and subsequently eliminated from further consideration. This rigorous filtering process enabled us to identify and retain only the most promising features that demonstrated a strong predictive power. Throughout our evaluation process, we diligently monitored the accuracy trends. We terminated the feature combination procedure when we observed a decrease in accuracy beyond a certain iteration. This allowed us to identify the optimal feature combinations and determine the most effective set of traffic features for achieving superior detection accuracy.

By adhering to this comprehensive evaluation process, we ensured that only features with a significant impact on the performance of our machine learning algorithms were retained. This meticulous approach enabled us to prioritize the selection of features that contributed to achieving a high level of accuracy in our classification tasks. Ultimately, our goal was to optimize the performance and effectiveness of our machine learning algorithms by leveraging the power of feature selection and evaluation.

Detection Accuracy				
Feature Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
F1	0.7577	0.8132	0.8133	0.7947
F2	0..9022	0.9825	0.9826	**0.9557**
F3	0.9008	0.9773	0.9772	**0.9518**
F4	0.9607	0.9602	0.9601	**0.9604**
F5	0.9867	0.9863	0.9862	**0.9865**
F6	0.7556	0.8186	0.8186	0.7976
F7	0.9421	0.9426	0.9425	**0.9424**
F8	0.8990	0.8994	0.8996	0.8993
F9	0.9779	0.9307	0.9778	**0.9621**
F10	0.8086	0.8102	0.8100	0.8096
F11	0.6973	0.7581	0.7581	0.7378
F12	0.75	0.8108	0.8106	0.7904

TABLE II

As Table II summarizes it appears that feature 11 has the lowest accuracy score of 0.7378, while feature 5 has the highest accuracy score of 0.9864.

Additionally, features F2, F3, F4, F5, F7, and F9 have better accuracy scores than the other six features in the table. Therefore, these six features are considered for further analysis and combination.

Moreover, we extended our evaluation beyond individual features and explored combinations of two, three, four, and even five features. By systematically evaluating these combinations, we sought to uncover any potential synergistic effects or improvements in accuracy that could be achieved by incorporating multiple features.

4.2 Two Features Combination:

In our classification analysis, we conducted an extensive examination of the dataset, which consisted of both normal and intrusion data. The primary objective of our study was to investigate how different combinations of features could impact the accuracy of our classification model. To achieve this, we specifically focused on combining two features at a time, aiming to identify the most effective feature pairs for our classification task.

To facilitate our feature selection process, we relied on the valuable insights provided in Table II. This table served as a crucial resource, offering detailed information about the characteristics and attributes of each feature. By carefully reviewing the contents of Table I, we were able to make informed decisions throughout the feature combination process.

Based on our analysis of Table II, we identified two features at a time that showed promising potential and were likely to contribute significantly to the accuracy of our classification model. Our selection of these feature pairs was guided by several factors, including the individual strengths and characteristics of each feature. We considered various aspects, such as the relevance of the features to the classification task, their discriminatory power, and any previous evidence of their effectiveness in similar contexts.

By combining the dataset with these carefully chosen feature pairs, we aimed to harness the synergistic effects that could arise from their collective influence. The inclusion of multiple features in our classification model allowed us to capture and integrate more nuanced patterns and relationships within the data, potentially resulting in enhanced accuracy and more precise predictions.

Throughout the process, we systematically explored numerous combinations of two features, meticulously evaluating their impact on the classification accuracy. This iterative approach enabled us to assess the advantages and limitations associated with different feature pairs, while also identifying any discernible patterns or trends that emerged from the analysis.

Our rigorous analysis involved a comprehensive assessment of the performance exhibited by each feature pair. By conducting a detailed evaluation, we aimed to gain valuable insights into the interplay between the selected features and their combined impact on the classification accuracy. This thorough examination facilitated informed decision-making regarding the most effective feature combinations, allowing us to refine and optimize our classification model for superior performance.

In summary, our classification analysis revolved around the strategic combination of two features at a time from the dataset comprising normal and intrusion data. The selection of these feature pairs was guided by the insights provided in Table II, which offered vital information about the characteristics of each feature. Through a systematic exploration of various feature combinations, we sought to identify the pairs that exhibited the highest potential for enhancing the accuracy of our classification model. By scrutinizing the performance and interplay of these feature pairs, we were able to make informed decisions and optimize our model for superior classification performance.

Detection Accuracy				
Combination Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
F2 F3	0.6582	0.6580	0.6617	0.6593
F2 F4	0.6697	0.6688	0.6706	0.6697
F2 F5	0.7521	0.8166	0.8144	0.7944
F2 F7	0.8258	0.8264	0.8254	0.8259
F2 F9	0.8249	0.8246	0.8238	0.8244
F3 F4	0.6160	0.6340	0.6305	0.6268
F3 F5	0.7904	0.7893	0.7912	0.7903
F3 F7	0.7990	0.7991	0.8003	0.7994
F3 F9	0.7975	0.8008	0.8006	0.7996
F4 F5	0.7343	0.7978	0.7988	0.7770
F4 F7	0.8086	0.8090	0.8093	0.8090
F4 F9	0.8069	0.8102	0.8104	0.8092
F5 F7	0.9874	0.9877	0.9877	**0.9876**
F5 F9	0.9880	0.9879	0.9881	**0.9880**
F7 F9	0.9395	0.9400	0.9395	**0.9397**

TABLE III

Upon thorough examination of Table III, we extracted valuable insights regarding the accuracy scores associated with various feature combinations. This comprehensive analysis allowed us to identify noteworthy trends and patterns, ultimately aiding in the selection of the most effective feature pairs for our classification task.

Table III presented a comprehensive overview of the accuracy scores obtained from different feature combinations. Notably, it was observed that the combination of features F5 and F9 achieved the highest accuracy score of 98.80%. This finding suggests that the joint influence of these two features resulted in a highly accurate classification model, exhibiting exceptional discriminatory power and predictive capabilities.

Conversely, the combination of features F3 and F4 displayed the lowest accuracy score among the feature pairs evaluated. This lower accuracy score indicates that the inclusion of these particular features together might not contribute significantly to the classification accuracy. It could be inferred that these features may lack the discriminative characteristics or fail to capture the crucial patterns necessary for accurate classification.

By highlighting the discrepancy between the highest and lowest accuracy scores, Table III offered valuable insights into the varying degrees of effectiveness exhibited by different feature combinations. This information enabled us to prioritize the feature pairs with the highest accuracy scores, as they demonstrated superior predictive performance and a stronger influence on the classification outcomes.

Moreover, the accuracy scores presented in Table III provide a quantitative assessment of the performance of each feature combination. These scores serve as a reliable indicator of the classification model's ability to accurately differentiate between normal and intrusion data when utilizing specific feature pairs. The higher the accuracy score, the more reliable and accurate the classification model's predictions are likely to be.

By considering the accuracy scores associated with different feature combinations, we gained a deeper understanding of the relationship between specific features and their impact on the classification accuracy. This knowledge proved invaluable in guiding our decision-making process, allowing us to prioritize the feature combinations that demonstrated the highest accuracy scores while deprioritizing those with lower scores.

In conclusion, the analysis of Table III revealed that the combination of features F5 and F9 exhibited the highest accuracy score, indicating its strong potential for enhancing the classification model's performance. Conversely, the combination of features F3 and F4 displayed the lowest accuracy score, suggesting its limited contribution to the overall accuracy. By leveraging this valuable information, we were able to identify the most effective feature pairs and make informed decisions to optimize our classification model for superior performance in accurately distinguishing between normal and intrusion data.

4.3 Three Features Combination

In this particular phase of our analysis, we specifically considered three features that exhibited higher accuracy scores.

The process began by carefully selecting the three features that demonstrated promising accuracy scores. These features were chosen based on their individual strengths and their potential to contribute significantly to the classification accuracy. By including features with higher accuracy scores, we aimed to leverage their predictive power and enhance the performance of our classification model.

With the selected features in hand, we proceeded to evaluate their accuracy scores using machine learning algorithms. The algorithms employed in our analysis were chosen based on their suitability for classification tasks and their proven effectiveness in handling diverse datasets. We utilized these algorithms to train our models on the combined dataset, incorporating the three selected features.

To obtain reliable and robust accuracy scores, we employed cross-validation. Cross-validation is a widely used technique that involves dividing the dataset into multiple folds and iteratively training and evaluating the model on different combinations of folds. By using this approach, we aimed to obtain an accurate estimate of the model's performance and its ability to generalize to unseen data.

During the cross-validation process, each fold served as both the training and testing set, allowing us to assess the accuracy of our classification model. The performance of the model was evaluated using accuracy scores, which provided a measure of how well the model correctly classified instances from the combined dataset.

By analyzing the cross-validation accuracy scores, we gained valuable insights into the effectiveness of the three selected features in our classification task. These accuracy scores served as quantitative indicators of the model's ability to accurately classify instances of normal and intrusion data.

This thorough evaluation allowed us to assess the impact of the chosen feature combinations on the classification accuracy. By considering the accuracy scores obtained through cross-validation, we could identify the feature combinations that yielded the highest accuracy, indicating their effectiveness in improving the model's performance.

Through this rigorous analysis, we aimed to uncover the potential strengths and weaknesses of different feature combinations. By focusing on the features with higher accuracy scores, we ensured that our classification model capitalized on the most informative attributes and relationships within the dataset.

Detection Accuracy				
Combination Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
F2 F5 F7	0.7505	0.8132	0.8143	0.7926
F2 F5 F9	0.8133	0.8134	0.8157	0.8141
F2 F7 F9	0.8221	0.8246	0.8269	0.8245
F4 F5 F9	0.7966	0.7989	0.7976	0.7977

F4 F7 F9	0.8013	0.8113	0.8125	0.8083
F5 F7 F9	0.9874	0.9877	0.9879	0.9877
F2 F3 F5	0.6482	0.6541	0.6527	0.6517
F2 F3 F7	0.6588	0.6589	0.6592	0.6589
F2 F4 F7	0.6657	0.6705	0.6701	0.6687
F2 F3 F9	0.6560	0.6596	0.6587	0.6581
F2 F4 F9	0.6051	0.6710	0.6684	0.6482
F3 F5 F7	0.7914	0.7892	0.7905	0.7904
F3 F5 F9	0.7936	0.7906	0.7919	0.7901
F3 F7 F9	0.7898	0.7887	0.7919	0.7901
F3 F4 F7	0.5885	0.6347	0.6353	0.6195
F3 F4 F9	0.6272	0.6307	0.6306	0.6295
F4 F5 F7	0.7961	0.7997	0.7978	0.7778

TABLE IV

Table IV shows the accuracy scores of different feature combinations in a machine-learning model. The combination of F2, F5, and F7 achieved an accuracy score of 0.7926. The combination of F3, F4, and F7 had the lowest accuracy score, while the combination of F5, F7, and F9 had the highest accuracy score. As can be seen from Table IV, all feature combinations have an accuracy score of less than 90%, hence, all these combinations can be discarded, and we can terminate the algorithm. However, for comparison with other feature combinations, we have created a set of four features, five features, and a set of all 12 features to highlight the detection accuracies with these combinations of features.

4.4 Four Features Combination In this classification task, the dataset of normal and intrusion data will be combined using the four best-performing features, which have higher accuracy scores.

Detection Accuracy				
Combination Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
F2 F5 F7 F9	0.8155	0.8136	0.8145	0.8145
F3 F5 F7 F9	0.7903	0.7914	0.7919	0.7912
F4 F5 F7 F9	0.7963	0.8006	0.8006	0.7992

TABLE V

Table V shows that the combination of F2, F5, F7, and F9 achieved the highest accuracy score, while the lowest accuracy score was obtained by the combination of F3, F5, F7, and F9. The accuracy score of the combination of F4, F5, F7, and F9 was 0.7992. All the combinations have lower accuracy than 90%, and hence, can be discarded.

4.5 Five Features Combination For this classification task, the top 5 performing features with higher accuracy scores will be combined from the normal and intrusion dataset. The goal is to determine the accuracy score using 3 machine learning algorithms along with crossvalidation. The selection of the 5 best features is based on their accuracy scores when combined in groups of 5.

Detection Accuracy				
Combination Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
F2 F3 F5 F7 F9	0.6533	0.6511	0.6538	0.6527
F2 F4 F5 F7 F9	0.6563	0.6597	0.6595	0.6585
F3 F4 F5 F7 F9	0.6232	0.6228	0.6228	0.6229

TABLE VI

Table VI shows a significant decrease in accuracy when using combinations of 5 features. The lowest accuracy score of 0.6229 was obtained by combining F3, F4, F5, F7, and F9. The combination of F2, F3, F5, F7, and F9 resulted in an accuracy score of 0.6527, while the highest accuracy score of 0.6585 was obtained by combining F2, F4, F5, F7, and F9. Hence, we can conclude that by increasing the number of features, the detection accuracy decreases. Hence, we terminate this procedure and we conclude that we get the highest detection accuracy of 98.80% with the combination of two features, i.e., F5 and F9, which are the Ratio of Incoming to Outgoing Packets, and Bytes sent, respectively.

4.6 Combination of All 12 Features:-

In our comprehensive classification analysis, we sought to investigate the detection accuracy achieved by utilizing all 12 traffic features. Our objective was to assess the performance of machine learning algorithms when applied to the complete set of features in order to determine their effectiveness in detecting the desired patterns.

he utilization of all 12 traffic features offered several advantages. By incorporating a wider range of information, the models could capture complex patterns and relationships that may not be discernible when using a subset of features. Additionally, the inclusion of the complete feature set provided a more comprehensive representation of the data, potentially leading to improved detection accuracy.

Through this detailed analysis, we aimed to assess the detection accuracy achieved by employing all 12 traffic features in conjunction with machine learning algorithms. This investigation allowed us to gain a deeper understanding of the capabilities of the models in accurately detecting the desired patterns, thereby informing future decisions regarding feature selection and model refinement.

Detection Accuracy				
Combination Notation	**K-Nearest Neighbour**	**Decision Tree**	**Random Forest**	**Mean**
All Features	0.4806	0.4868	0.4873	0.4849

TABLE VII

Table VII reveals that the accuracy of all 12 Features. In our rigorous analysis of the intrusion detection system, we extensively evaluated the accuracy of all 12 features using Table VII as our reference. The findings from Table VII indicated that the accuracy of the individual features was notably low, with an average accuracy score of 0.4849.

The low accuracy scores observed for the individual features highlighted the limited discriminatory ability of these features when considered individually. This underscores the importance of conducting feature ranking and selection processes to identify the most informative and relevant features for the detection task.

CHAPTER 5 CONCLUSION

In conclusion, our research paper has focused on exploring the application of machine learning algorithms for the purpose of detecting intrusions in network traffic data. Through extensive analysis and experimentation, we have successfully demonstrated the effectiveness of these algorithms in accurately classifying network traffic as either normal or intrusive. This accomplishment is a significant milestone in the field of intrusion detection and holds promising implications for improving network security.

Specifically, we have identified a set of two features, namely the Ratio of Incoming to Outgoing Packets and Bytes Sent, that exhibit a high degree of discriminatory power in distinguishing between normal and intrusive network traffic. By leveraging these features and employing machine learning algorithms, we have achieved remarkable accuracy scores in the classification process. This finding highlights the potential of utilizing machine learning techniques to effectively identify and mitigate various forms of cyber attacks.

Looking ahead, our future endeavors will revolve around expanding the scope of our investigation by incorporating a greater number of traffic features. By considering additional relevant features, we aim to enhance the accuracy and robustness of intrusion detection systems. We recognize that network traffic patterns are complex and dynamic, requiring a comprehensive understanding of various factors that may indicate malicious activity. By incorporating a wider range of features into our analysis, we can gain deeper insights into the intricacies of network behavior and further refine our methods for identifying and mitigating potential threats.

Furthermore, our research endeavors are driven by a strong commitment to contribute to the advancement of intrusion detection techniques and the overall security of network infrastructures. By leveraging the power of machine learning algorithms and continuously refining our approaches, we strive to fortify our defenses against evolving cyber threats. Our aim is to protect sensitive information, safeguard the integrity and reliability of network systems, and ensure the privacy and trust of users.

In conclusion, our ongoing efforts in intrusion detection research will continue to push the boundaries of knowledge and innovation in this critical field. By constantly refining our methods, expanding our understanding of network traffic patterns, and leveraging the potential of machine learning algorithms, we seek to make significant contributions to the overall security landscape. The protection of networks and the preservation of the digital ecosystem are of utmost importance, and we remain steadfast in our dedication to this pursuit.

References

[1] https://www.unb.ca/cic/datasets/index.html

[2] H. Yang and F. Wang, ”Wireless Network Intrusion Detection Based on Improved Convolutional Neural Network,” in IEEE Access, vol. 7, pp. 64366-64374, 2019, doi: 10.1109/ACCESS.2019.2917299.

[3] Y. Zeng, H. Gu, W. Wei and Y. Guo, ”Deep−F ull−Range : A Deep Learning Based Network Encrypted Traffic Classification and Intrusion Detection Framework,” in IEEE Access, vol. 7, pp. 45182-45190, 2019, doi: 10.1109/ACCESS.2019.2908225.

[4] C. F. T. Pontes, M. M. C. de Souza, J. J. C. Gondim, M. Bishop and M. A. Marotta, ”A New Method for Flow-Based Network Intrusion Detection Using the Inverse Potts Model,” in IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1125-1136, June 2021, doi: 10.1109/TNSM.2021.3075503.

[5] T. -N. Dao and H. Lee, ”JointNIDS: Efficient Joint Traffic Management for On-Device Network Intrusion Detection,” in IEEE Transactions on Vehicular Technology, vol. 71, no. 12, pp. 13254-13265, Dec. 2022, doi: 10.1109/TVT.2022.3198266.

[6] Y. Zhang, X. Chen, D. Guo, M. Song, Y. Teng and X. Wang, ”PCCN: Parallel Cross Convolutional Neural Network for Abnormal Network Traffic Flows Detection in Multi-Class Imbalanced Network Traffic Flows,” in IEEE Access, vol. 7, pp. 119904-119916, 2019, doi: 10.1109/ACCESS.2019.2933165. [7] T. Su, H. Sun, J. Zhu, S. Wang and Y. Li, ”BAT: Deep Learning Methods on Network Intrusion Detection Using NSL-KDD Dataset,” in IEEE Access, vol. 8, pp. 29575-29585, 2020, doi: 10.1109/ACCESS.2020.2972627.

[8] L. Zhang, X. Yan and D. Ma, ”A Binarized Neural Network Approach to Accelerate in-Vehicle Network Intrusion Detection,” in IEEE Access, vol. 10, pp. 123505-123520, 2022, doi: 10.1109/ACCESS.2022.3208091.

[9] Y. Sharon, D. Berend, Y. Liu, A. Shabtai and Y. Elovici, ”TANTRA: Timing-Based Adversarial Network Traffic Reshaping Attack,” in IEEE Transactions on Information Forensics and Security, vol. 17, pp. 3225- 3237, 2022, doi: 10.1109/TIFS.2022.3201377.

[10] M. Lopez-Martin, A. Sanchez-Esguevillas, J. I. Arribas and B. Carro, ”Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning,” in IEEE Access, vol. 9, pp. 153153-153170, 2021, doi: 10.1109/ACCESS.2021.3127689.

[11] Z. Yu, Y. Liu, G. Xie, R. Li, S. Liu and L. T. Yang, ”TCE-IDS: Time Interval Conditional Entropy- Based Intrusion Detection System for Automotive Controller Area Networks,” in IEEE Transactions on Industrial Informatics, vol. 19, no. 2, pp. 1185-1195, Feb. 2023, doi: 10.1109/TII.2022.3202539.

[12] J. Yang, Y. Zhang, R. King and T. Tolbert, ”Sniffing and Chaffing Network Traffic in Stepping-Stone Intrusion Detection,” 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), Krakow, Poland, 2018, pp. 515-520, doi: 10.1109/WAINA.2018.00137.

[13] L. Zhang, H. Yan and Q. Zhu, ”An Improved LSTM Network Intrusion Detection Method,” 2020 IEEE 6th International Conference on Computer and Communications (ICCC), Chengdu, China, 2020, pp. 1765- 1769, doi: 10.1109/ICCC51575.2020.9344911.

[14] S. Ding, Y. Wang and L. Kou, ”Network intrusion detection based on BiSRU and CNN,” 2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS), Denver, CO, USA, 2021, pp. 145- 147, doi: 10.1109/MASS52906.2021.00026.

[15] Y. Peng, ”Application of Convolutional Neural Network in Intrusion Detection,” 2020 International Conference on Advance in Ambient Computing and Intelligence (ICAACI), Ottawa, ON, Canada, 2020, pp. 169-172, doi: 10.1109/ICAACI50733.2020.00043.

[16] Y. Sun, H. Ochiai and H. Esaki, ”Intrusion Measurement and Detection in LAN Using Protocol-Wise Associative Memory,” 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Jeju Island, Korea (South), 2021, pp. 005-009, doi: 10.1109/ICAIIC51459.2021.9415195.

[17] Y. Sharma, S. Sharma and A. Arora, ”Feature Ranking using Statistical Techniques for Computer Networks Intrusion Detection,” 2022 7th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2022, pp. 761-765, doi: 10.1109/ICCES54183.2022.9835831.

[18] J. Chen, S. Yin, S. Cai, L. Zhao and S. Wang, ”L-KPCA: an efficient feature extraction method for network intrusion detection,” 2021 17th International Conference on Mobility, Sensing and Networking (MSN), Exeter, United Kingdom, 2021, pp. 683-684, doi: 10.1109/MSN53354.2021.00104.

[19] Y. -F. Hsu and M. Matsuoka, ”A Deep Reinforcement Learning Approach for Anomaly Network Intrusion Detection System,” 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), Piscataway, NJ, USA, 2020, pp. 1-6, doi: 10.1109/CloudNet51028.2020.9335796.

[20] M. S. Koli and M. K. Chavan, ”An advanced method for detection of botnet traffic using intrusion detection system,” 2017 International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, India, 2017, pp. 481-485, doi: 10.1109/ICICCT.2017.7975246.

[21] C. -F. Hsieh and C. -M. Su, ”MLNN: A Novel Network Intrusion Detection Based on Multilayer Neural Network,” 2021 International Conference on Technologies and Applications of Artificial Intelligence (TAAI), Taichung, Taiwan, 2021, pp. 43-48, doi: 10.1109/TAAI54685.2021.00017.

[22] L. Nie, Z. Ning, X. Wang, X. Hu, J. Cheng and Y. Li, ”DataDriven Intrusion Detection for Intelligent Internet of Vehicles: A Deep Convolutional Neural Network-Based Method,” in IEEE Transactions on Network Science and Engineering, vol. 7, no. 4, pp. 2219-2230, 1 Oct.-Dec. 2020, doi: 10.1109/TNSE.2020.2990984.

[23] M. A. Siddiqi and W. Pak, ”An Agile Approach to Identify Single and Hybrid Normalization for Enhancing Machine Learning-Based Network Intrusion Detection,” in IEEE Access, vol. 9, pp. 137494-137513, 2021, doi: 10.1109/ACCESS.2021.3118361.

[24] H. He, X. Sun, H. He, G. Zhao, L. He and J. Ren, ”A Novel Multimodal-Sequential Approach Based on Multi-View Features for Network Intrusion Detection,” in IEEE Access, vol. 7, pp. 183207- 183221, 2019, doi: 10.1109/ACCESS.2019.2959131.

[25] M. E. Aminanto, R. S. H. Wicaksono, A. E. Aminanto, H. C. Tanuwidjaja, L. Yola and K. Kim, ”Multi-Class Intrusion Detection Using Two-Channel Color Mapping in IEEE 802.11 Wireless Network,” in IEEE Access, vol. 10, pp. 36791-36801, 2022, doi: 10.1109/ACCESS.2022.3164104.

[26] A. Yang, Y. Zhuansun, C. Liu, J. Li and C. Zhang, ”Design of Intrusion Detection System for Internet of Things Based on Improved BP Neural Network,” in IEEE Access, vol. 7, pp. 106043-106052, 2019, doi: 10.1109/ACCESS.2019.2929919.

[27] W. Zhong, N. Yu and C. Ai, ”Applying big data based deep learning system to intrusion detection,” in Big Data Mining and Analytics, vol. 3, no. 3, pp. 181-195, Sept. 2020, doi: 10.26599/BDMA.2020.9020003.

[28] T. Ye, G. Li, I. Ahmad, C. Zhang, X. Lin and J. Li, ”FLAG: Few-Shot Latent Dirichlet Generative Learning for Semantic-Aware Traffic Detection,” in IEEE Transactions on Network and Service Management, vol. 19, no. 1, pp. 73-88, March 2022, doi: 10.1109/TNSM.2021.3131266.

[29] R. Bar and C. Hajaj, ”SimCSE for Encrypted Traffic Detection and ZeroDay Attack Detection,” in IEEE Access, vol. 10, pp. 56952-56960, 2022, doi: 10.1109/ACCESS.2022.3177272.

[30] A. Liu and B. Sun, ”An Intrusion Detection System Based on a Quantitative Model of Interaction Mode Between Ports,” in IEEE Access, vol. 7, pp. 161725-161740, 2019, doi: 10.1109/ACCESS.2019.2951839.

[31] B. Wang, Y. Su, M. Zhang and J. Nie, ”A Deep Hierarchical Network for Packet-Level Malicious Traffic Detection,” in IEEE Access, vol. 8, pp. 201728-201740, 2020, doi: 10.1109/ACCESS.2020.3035967.

[32] V. Handika, J. E. Istiyanto, A. Ashari, S. R. Purnama, S. Rochman and A. Dharmawan, ”Feature Representation for Network Intrusion Detection System Trough Embedding Neural Network,” 2022 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM), Surabaya, Indonesia, 2022, pp. 1-4, doi: 10.1109/CENIM56801.2022.10037425.

[33] Z. C. Johanyak, ”Fuzzy Logic based Network Intrusion Detection ´ Systems,” 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, 2020, pp. 15- 16, doi: 10.1109/SAMI48414.2020.9108750.

[34] M. ˙Ibrahim, B. Issa and M. B. Jasser, ”A Method for Automatic Android Malware Detection Based on Static Analysis and Deep Learning,” in IEEE Access, vol. 10, pp. 117334-117352, 2022, doi: 10.1109/ACCESS.2022.3219047.

[35] H. Soni, P. Kishore and D. P. Mohapatra, ”Opcode and API Based Machine Learning Framework For Malware Classification,” 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2022, pp. 1-7, doi: 10.1109/CONIT55038.2022.9848152.

[36] Z. Sawadogo, G. Mendy, J. M. Dembelle and S. Ouya, ”Android Malware Classification: Updating Features Through Incremental Learning Approach(UFILA),” 2022 24th International Conference on Advanced Communication Technology (ICACT), PyeongChang Kwangwoon Do, Korea, Republic of, 2022, pp. 544-550, doi: 10.23919/ICACT53585.2022.9728977.

[37] V. K. V and J. C. D, ”Android Malware Detection using Function Call Graph with Graph Convolutional Networks,” 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), Jalandhar, India, 2021, pp. 279-287, doi: 10.1109/ICSCCC51823.2021.9478141.

[38] A. Libri, A. Bartolini and L. Benini, ”pAElla: Edge AI-Based RealTime Malware Detection in Data Centers,” in IEEE Internet of Things Journal, vol. 7, no. 10, pp. 9589-9599, Oct. 2020, doi: 10.1109/JIOT.2020.2986702.

[39] T. N. Phu, L. Hoang, N. N. Toan, N. Dai Tho and N. N. Binh, ”C500-CFG: A Novel Algorithm to Extract Control Flow-based Features for IoT Malware Detection,” 2019 19th International Symposium on Communications and Information Technologies (ISCIT), Ho Chi Minh City, Vietnam, 2019, pp. 568-573, doi: 10.1109/ISCIT.2019.8905120.

[40] Y. -M. Chen, C. -H. Yang and G. -C. Chen, ”Using Generative Adversarial Networks for Data Augmentation in Android Malware Detection,” 2021 IEEE Conference on Dependable and Secure Computing (DSC), Aizuwakamatsu, Fukushima, Japan, 2021, pp. 1-8, doi: 10.1109/DSC49826.2021.9346277.

[41] R. Elnaggar, K. Basu, K. Chakrabarty and R. Karri, ”Runtime Malware Detection Using Embedded Trace Buffers,” in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41, no. 1, pp. 35-48, Jan. 2022, doi: 10.1109/TCAD.2021.3052856.

[42] D. Demırcı, N. s¸ahın, M. s¸irlancis and C. Acarturk, ”Static Malware Detection Using Stacked BiLSTM and GPT-2,” in IEEE Access, vol. 10, pp. 58488-58502, 2022, doi: 10.1109/ACCESS.2022.3179384.

[43] S. Seneviratne, R. Shariffdeen, S. Rasnayaka and N. Kasthuriarachchi, ”Self-Supervised Vision Transformers for Malware Detection,” in IEEE Access, vol. 10, pp. 103121-103135, 2022, doi: 10.1109/ACCESS.2022.3206445.

[44] C. Sun, H. Zhang, S. Qin, J. Qin, Y. Shi and Q. Wen, ”DroidPDF: The Obfuscation Resilient Packer Detection Framework for Android Apps,” in IEEE Access, vol. 8, pp. 167460-167474, 2020, doi: 10.1109/ACCESS.2020.3010588.

[45] Y. Ban, S. Lee, D. Song, H. Cho and J. H. Yi, ”FAM: Featuring Android Malware for Deep Learning-Based Familial Analysis,” in IEEE Access, vol. 10, pp. 20008-20018, 2022, doi: 10.1109/ACCESS.2022.3151357.

[46] J. Wang, B. Li and Y. Zeng, ”XGBoost-Based Android Malware Detection,” 2017 13th International Conference on Computational Intelligence and Security (CIS), Hong Kong, China, 2017, pp. 268-272, doi: 10.1109/CIS.2017.00065.

[47] S. Iqbal and M. Zulkernine, ”SpyDroid: A Framework for Employing Multiple Real-Time Malware Detectors on Android,” 2018 13th International Conference on Malicious and

Unwanted Software (MALWARE), Nantucket, MA, USA, 2018, pp. 1-8, doi: 10.1109/MALWARE.2018.8659365.

[48] F.Ullah, s.Ullah, G. Srivastava, J.C. Lin, ”IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic”, Digital Communications and Networks, 2023, ISSN 2352-8648, https://doi.org/10.1016/j.dcan.2023.03.008.

[49] Y. Zhu, L. Cui, Z. Ding, L. Li, Y. Liu, Z. Hao,”Black box attack and network intrusion detection using machine learning for malicious traffic”, Computers and Security, Volume 123, 2022, 102922, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2022.102922.

[50] B. Li, Y. Wang, K. Xu, L. Cheng, Z. Qin, ”DFAID: Density-aware and feature-deviated active intrusion detection over network traffic streams”, Computers and Security, Volume 118,2022, 102719,ISSN 0167-4048, https://doi.org/10.1016/j.cose.2022.102719.

[51] J. Liang, J. Chen, Y. Zhu, R. Yu, ”A novel Intrusion Detection System for Vehicular Ad Hoc Networks (VANETs) based on differences of traffic flow and position”, Applied Soft Computing, Volume 75, 2019, Pages 712-727, ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2018.12.001