Applying Statistical Techniques on Traffic Features for Intrusion Detection

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

OF

MASTER OF SCIENCE

IN

MATHEMATICS

Submitted by:

SAKSHI(2K22/MSCMAT/35)

Under the supervision of

DR. ANSHUL ARORA

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

MAY, 2024

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Sakshi, 2K22/MSCMAT/35 student 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 me 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 **Sakshi**

Date: 25TH May 2023 2K22/MSCMAT/35

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 "**Applying Statistical Techniques on Traffic Features for Intrusion Detection"** which is submitted by Sakshi, 2K22/MSCMAT/35[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

I express my sincere gratitude to Dr. Anshul Arora (Assistant Professor, Department of Applied Mathematics), for his valuable guidance and timely suggestions during the entire duration of our dissertation work, without which this work would not have been possible. I would also like to convey our deep regards to all other faculty members of the Department of Applied Mathematics, who have bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our part to finish this work. Finally, I would also like to thank our friends for their advice and pointing out our mistakes.

CONTENTS

LIST OF TABLES

 LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 COMMENCEMENT

Introduction to Intrusion Detection Systems

In today's digitally interconnected world, the security of information systems is paramount. As organizations and individuals increasingly rely on digital platforms for communication, commerce, and data storage, the threat landscape has expanded, making cybersecurity a critical priority. Intrusion Detection Systems (IDS) have emerged as essential tools in this context, designed to detect unauthorized access and malicious activities within a network or system. An IDS works by continuously monitoring network traffic and system behaviors, analyzing data for signs of suspicious activity, and alerting administrators to potential threats. This proactive approach allows for the early identification of security breaches, enabling timely responses to mitigate damage and protect valuable information assets.

The Need for Intrusion Detection Systems

The need for IDS arises from the growing complexity and frequency of cyber threats. Cyber-attacks have become more sophisticated, targeting vulnerabilities in systems and networks to gain unauthorized access, steal sensitive data, or disrupt operations. Traditional security measures like firewalls and antivirus software, while necessary, are often insufficient on their own. They primarily focus on preventing attacks but may not detect or respond to intrusions that have already bypassed these defenses. An IDS complements these measures by providing an additional layer of security through real-time monitoring and analysis, which is crucial in identifying and addressing potential threats that might otherwise go unnoticed.

Moreover, regulatory compliance is another significant driver for implementing IDS. Many industries are governed by stringent regulations that mandate the protection of sensitive data, such as the General Data Protection Regulation (GDPR), the Health Insurance Portability and Accountability Act (HIPAA), and the Payment Card Industry Data Security Standard (PCI DSS). An IDS helps organizations meet these regulatory requirements by ensuring continuous monitoring and reporting of security incidents, thereby avoiding legal penalties and safeguarding their reputation.

Additionally, the economic impact of cyber-attacks cannot be overstated. Businesses can suffer substantial financial losses due to downtime, data breaches, and the cost of remediation. By detecting and addressing threats promptly, an IDS helps minimize these financial risks. Furthermore, as cyber threats continue to evolve, IDS solutions also advance, incorporating machine learning and artificial intelligence to enhance their detection capabilities. This adaptability is crucial in maintaining robust security in the face of constantly changing attack vectors.

In conclusion, the integration of Intrusion Detection Systems into an organization's cybersecurity strategy is indispensable. They not only enhance the ability to detect and respond to threats but also ensure compliance with regulatory standards and protect against significant financial losses. As cyber threats continue to grow in sophistication, the role of IDS in maintaining the integrity and security of digital environments becomes ever more critical.

Contributions of the Thesis

Keeping these limitations in mind, an anomaly-based intrusion detection model has been proposed in this Thesis.

The main contributions of the Thesis are summarized below:

1. The network traffic features were extracted from the traffic files of normal and intrusion traffic.

2. Statistical techniques, Kendall’s Tau Test, were applied to the set of traffic features to rank them. The ranking of features was obtained with these statistical tests.

3. A novel algorithm for intrusion detection was proposed by applying machine learning classifiers to the ranked features.

1.2 MOTIVATION

In the digital age, cybersecurity has become a critical concern for individuals, businesses, and governments alike. The increasing dependence on computer networks and information systems has made them attractive targets for cybercriminals, motivating the development of Intrusion Detection Systems (IDS). The frequency and sophistication of cyber-attacks are on the rise, necessitating a robust defense mechanism like IDS to identify and mitigate these threats before they can cause significant damage. Organizations across various sectors handle sensitive data, including personal information, financial records, and intellectual property, where unauthorized access or breaches can lead to severe consequences, including financial loss, legal penalties, and reputational damage. An IDS provides an additional layer of security to protect this critical information and helps in compliance with stringent regulatory requirements such as GDPR, HIPAA, and PCI DSS, ensuring continuous monitoring and detection of security incidents.

Moreover, cyber-attacks can disrupt business operations, leading to significant downtime and loss of productivity. By detecting intrusions early, an IDS enables swift response and remediation, minimizing operational disruptions and maintaining business continuity. It not only detects potential intrusions but also provides detailed information about the nature and scope of the attack, crucial for incident response teams to understand the threat, contain it, and develop strategies to prevent future occurrences. While preventive measures such as firewalls and antivirus software are essential, they may not be sufficient to detect and respond to all types of cyber threats. An IDS complements these measures by providing a more comprehensive security posture, often at a lower cost compared to the potential losses from a successful cyber-attack. Cyber threats are continuously evolving, with attackers developing new techniques to bypass traditional security measures. An IDS is designed to adapt to these changes, using advanced algorithms and machine learning to identify novel attack patterns and ensure ongoing protection. For businesses and their stakeholders, knowing that there is an effective system in place to detect and respond to security incidents provides significant peace of mind, which is invaluable in maintaining trust and confidence among customers, partners, and investors. Thus, the development of an Intrusion Detection System is driven by the need to protect digital assets in an increasingly hostile cyber environment, making it a crucial component of a comprehensive cybersecurity strategy.

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. Seneviratne et al. [1] introduced SHERLOCK, a malware detection framework achieving 91% accuracy for binary classification. Ibrahim et al. [2] 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. The authors in [3] 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. [4] 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. UFILA was developed by authors in [5] for detecting and classifying Android malware by introducing new features. In [6], 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. A model based on the FCG function to detect Android malware was proposed by authors in [7]. while the authors in [8] presented an XGBoost model to detect Android malware and investigated the effect of feature selection on classification. . Bar et. Al [9] 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. Demirci et al. [10] proposed a method for identifying

malicious code using stacked bidirectional long short-term memory and generative pre-trained transformer-based deep learning language models. Yang et al. Meanwhile, in [11], 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: USTCTFC2016, ISCX2012, and CICIDS2017. Iqbal et al. [12] developed SpyDroid, a real-time malware detection framework with a detection module that identifies malicious apps. In [13], 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. In [14], 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. . Ullah et al. [15] proposed IDS-INT, a system that employs transfer learning with transformerbased 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 [16], 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. In [17], 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. Chen et.al. [18] 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. Ban et al. [19] evaluated the contribution of different features in familial analysis using a convolutional neural network on a real-world malware dataset. Elnaggar et.al. [20] 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. . Zhong et al [21] 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. The authors in [22] developed an AnDroid Packer framework to detect packed samples. [23] 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. DFAIDDK 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. Soni et.al. [24] proposed a framework for malware classification using opcode and API calls features. Libri et.al. [25] introduced pAElla, a system that detects real-time malware in an IoT-based monitoring system using power measurements and autoencoders.

Nie et al. [26] 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.

[27] presented Joint NIDS, a joint traffic classification architecture that utilizes two submodels. Lastly, the authors in [28] 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.

The authors in [29] proposed the use of statistical methods such as ANOVA and Chi-Square tests to organize network traffic features. In [30], 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.

Chen et.al. [31] 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.

Peng et.al. [32] 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. [33] 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.

Al [34] 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 2coded color mapping and was evaluated using the Wi-Fi Intrusion Data Set (AWID2. Zhang et al. [35] 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. A model based on the FCG function to detect Android malware was proposed by authors in [36]. Yang et. Al [37] 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.

The authors in [38] proposed a novel intrusion detection model that combines BiSRU and CNN to process network traffic logs effectively.The authors in [39] introduced a deep reinforcement learning (DRL) technique for anomaly network intrusion detection, allowing the system to adapt to different network traffic behaviours.

Finally, in [40], 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.

A novel intrusion detection technique for automotive CAN networks was introduced in [41], 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. Yang et al. [42] proposed an Improved Convolutional Neural Network (ICNN) for this purpose. Ban et al. [44] evaluated the contribution of different features in familial analysis using a convolutional neural network on a real-world malware dataset. Sharon et al. [45] introduced TANTRA, an end-to-end Timing-based combative Network Traffic Reshaping Attack that evades a variety of NIDSs.

In [47], the authors developed an adaptive and efficient intrusion detection method using protocol-wise associative memory of Hopfield networks

For instance, the authors in [48] introduced BAT, a traffic anomaly detection model that eliminates the need for feature engineering and accurately detects anomalies.

The authors in [49] proposed a hierarchical progressive network with a multimodalsequential intrusion detection approach using Multimodal Deep Autoencoder (MDAE) and Long Short-Term Memory (LSTM) technologies.

The authors of [50] introduced an Energy-based Flow Classifier (EFC) algorithm for robust traffic classification.

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).

3.1 Dataset Collection :-

In our research project, we leveraged Wireshark, a powerful and versatile packet analyzer, to conduct an in-depth analysis of normal network traffic. Wireshark's extensive capabilities in capturing, analyzing, and interpreting network traffic made it an invaluable tool for our study. By dissecting the structure of network protocols and examining packet headers, payloads, and metadata, we were able to unravel the complexities of data transmission within our network. This thorough examination helped us understand the subtle nuances and patterns that characterize our network's normal operations.

Wireshark's flexibility in handling both live network data and saved packet captures allowed us to revisit specific scenarios and analyze traffic in detail. The tool's wide range of features, including filtering, sorting, and categorizing packets by various criteria, facilitated a granular analysis of our network traffic. This enabled us to identify anomalies, detect potential vulnerabilities, and establish a comprehensive understanding of our network's baseline behavior. Through this meticulous analysis, we gained actionable insights into network security, optimization, and troubleshooting.

Additionally, we obtained intrusion data from the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick. The CIC, renowned for its expertise in cybersecurity, provided us with a diverse dataset of network traffic that included both normal traffic and various cyber attack instances, such as DoS, DDoS, SQL injection, and XSS. This dataset was crucial for evaluating the performance of intrusion detection systems and contributed significantly to our research. By integrating these datasets, we were able to enhance the depth and quality of our study, leading to valuable findings in network behavior and security.

3.2 Feature Extraction :-

Feature extraction was pivotal in our research, allowing us to identify crucial variables that offered valuable insights into underlying data patterns. We meticulously selected features relevant to our research question, employing statistical and machine-learning techniques to ensure a rigorous process. This careful selection resulted in significant improvements in model accuracy and interpretability, highlighting the importance of focusing on the most pertinent variables.

Incorporating the 12 selected features into our analysis notably enhanced model performance compared to using the full set of available features. These features provided meaningful insights into network traffic behavior, illuminating various dimensions and uncovering hidden relationships among variables. This comprehensive approach deepened our understanding of the research domain.

Moreover, focusing on a concise set of informative features streamlined data analysis and interpretation, making the process more intuitive and manageable. The 12 features effectively distilled the complexity of network traffic data, leading to clearer insights and actionable findings. For detailed information on the identified features, please refer to Table I in the document, which summarizes their descriptions, relevance, and significance.

Overall, the feature extraction process was instrumental to our project's success, improving accuracy, interpretability, and manageability of results. The insights gained from these features will guide future research and advancements in our domain.

Table I summarizes the 12-network traffic features we extracted

.

LIST OF TRAFFIC FEATURES

Feature Notation	Feature Extracted
F1	Average Packet Size
F2	Bytes
F3	Average Packet Size Sent
F4	Average Packet Size Received
F5	Ratio of Incoming to Outgoing Packets

F6	Bits sent/sec
F7	Bits received/sec
F8	Flow Duration
F9	Time interval between packet sent
F10	Time interval between packet received

TABLE I

3.3 About Feature Ranking Methods

Here, in this work, we have used statistical test to rank the features obtained.

3.3.1 Kendall's Tau Test :

Kendall's tau is a statistical measure used to evaluate the strength and direction of association between two variables. Named after the British statistician Maurice Kendall, this coefficient assesses the ordinal relationship between pairs of data. Unlike Pearson's correlation, which measures linear relationships, Kendall's tau focuses on the rank correlation, making it especially useful for ordinal data or non-parametric statistics.

Kendall's tau operates by comparing the concordance and discordance of pairs in the dataset. A pair of observations (x1, y1) and (x2, y2) is considered concordant if the order of x-values and y-values is the same (i.e., if $x1 < x2$ and $y1 < y2$ or if $x1 > x2$ and $y1 > y2$). Conversely, the pair is discordant if the order is reversed (i.e., if $x1 < x2$ and $y1 > y2$ or if $x1 > x2$ and $y1 < y2$). The tau coefficient is calculated based on the difference between the number of concordant and discordant pairs, normalized by the total number of pairs.

The value of Kendall's tau ranges from -1 to 1. A tau value of 1 indicates a perfect positive association, meaning all pairs are concordant. A value of -1 indicates a perfect negative association, with all pairs being discordant. A tau value of 0 suggests no association between the variables.

One of the advantages of Kendall's tau is its robustness to outliers, as it depends solely on the ranks of the data rather than their specific values. This makes it a preferred choice in many fields, such as economics, social sciences, and bioinformatics, where data often deviate from normality.

In summary, Kendall's tau is a valuable tool for measuring the strength and direction of relationships between variables, particularly when dealing with ordinal data or when a non-parametric method is required. Its reliance on rank correlation provides a reliable measure that is less sensitive to outliers and non-linearities in the data.

The formula for Kendall's tau (τ) is: τ=(C-D)/(C+D)

3.4 ML Algorithms :-

A machine learning algorithm is designed to learn from data by identifying patterns within it. In practice, datasets are divided into two subsets: training data and testing data. The training data, usually a larger portion, is used to teach the model, while the testing data evaluates its performance on new, unseen data. In our research project, we allocated 70% of the total dataset for training our machine learning model, using the scikit-learn (sklearn) library.

Scikit-learn is a popular Python library that provides a comprehensive suite of tools for machine learning tasks, including classification, regression, clustering, and dimensionality reduction. It offers a consistent and user-friendly interface for common tasks such as splitting data, scaling, and model selection. Additionally, sklearn includes numerous machine learning algorithms and evaluation metrics, which can be easily customized and extended to fit specific needs. This versatility made it an ideal choice for building and evaluating our model.

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

KNN, short for k-Nearest Neighbors, is a straightforward and intuitive algorithm used for both classification and regression tasks in machine learning. It operates on the principle of similarity, where the class or value of a data point is determined by its proximity to other data points in the feature space. The "k" in KNN represents the number of nearest neighbors considered when making predictions.

In classification tasks, KNN assigns the majority class among the k nearest neighbors to the query point, while in regression tasks, it calculates the average or weighted average of the values of those neighbors. One of the notable advantages of KNN is its simplicity and ease of implementation. Additionally, KNN can adapt to different types of data and does not require the training of a model, making it suitable for scenarios with small to moderate-sized datasets.

However, KNN's performance can be sensitive to the choice of distance metric and the value of k. Additionally, it may suffer from computational inefficiency, especially with large datasets, as it requires calculating distances between the query point and all other data points. Despite these limitations, KNN remains a popular and versatile algorithm, particularly in situations where interpretability and ease of use are prioritized.

2) Decision Tree:

Decision trees build classification models using a tree structure composed of if-then rules that are mutually exclusive. These rules are sequentially learned from training data, with each rule removing the tuples it covers until a termination condition is met. The final structure resembles a tree with nodes and leaves, where attributes at the top have the most impact on classification, determined by the information gain concept. All attributes should be categorical or discretized in advance. However, decision trees can produce many branches, sometimes reflecting noise or outliers.

Belonging to the family of supervised learning algorithms, decision trees can handle both regression and classification tasks. The goal is to construct a model that predicts the class or value of a target variable by learning simple decision rules from the training data. To predict a class label for a record, we start at the tree's root, comparing the root attribute value with the record's attribute value, and follow the corresponding branch to the next node. This sorting continues until a terminal node or leaf is reached, which provides the classification.

Each node in a decision tree acts as a test case for an attribute, with descending edges representing possible answers. This recursive process continues for each subtree rooted at a new node. Some key assumptions in using decision trees include treating the entire training set as the root initially, preferring categorical feature values (with continuous values being discretized), and using statistical approaches to place attributes at the root or internal nodes. The accuracy of the tree depends significantly on strategic splits, which are made using various algorithms to increase the similarity of resultant sub-nodes.

Decision trees decide on node splits by evaluating all available variables and selecting the split that results in the most similar sub-nodes, thereby enhancing the node's relevance to the target variable. This process ensures the formation of effective and accurate decision rules.

3) Random Forest:

Random Forest is an ensemble learning method used for classification, regression, and other tasks by constructing multiple decision trees during training. It addresses the issue of overfitting associated with single decision trees by averaging the results of many trees. Although Random Forest is generally less reactive than gradient-boosted trees, its performance can vary based on data characteristics.

As the name suggests, Random Forest comprises many individual decision trees that work together as a collective. Each tree provides a class prediction, and the class with the most votes becomes the model's final prediction. This ensemble approach outperforms any single decision tree because it relies on the collective decision of many models, reducing the risk of errors from individual trees.

For Random Forest to perform effectively, the features used must contain actual signals rather than random noise, and the predictions made by individual trees should have low correlation with each other. This ensures that while some trees may be incorrect, the majority will be correct, allowing the ensemble to make accurate predictions overall.

4) Support Vector Machine:

Support Vector Machine (SVM) is a powerful and widely-used supervised learning algorithm employed for both classification and regression tasks in machine learning. Its primary objective is to find the optimal hyperplane that best separates data points into different classes or predicts a continuous target variable. SVM achieves this by identifying the hyperplane that maximizes the margin, which is the distance between the hyperplane and the closest data points (support vectors).

One of the key strengths of SVM is its ability to handle high-dimensional data efficiently, making it suitable for tasks with complex feature spaces. Moreover, SVM can effectively handle non-linear relationships between features through the use of kernel functions, which map the input space into a higher-dimensional space where data points are more easily separable.

SVM is particularly useful in scenarios where the number of features exceeds the number of samples, as it avoids overfitting by focusing on the support vectors, which are the most informative data points for determining the optimal hyperplane. Additionally, SVM's decision boundary is determined by a subset of training data points, making it robust to outliers.

Despite its effectiveness, SVM may be computationally intensive, especially with large datasets, and requires careful selection of parameters such as the choice of kernel function and regularization parameter. Nevertheless, SVM remains a versatile and widely-applied algorithm, valued for its ability to handle complex data and produce accurate predictions across various domains..

5) Logistic Regression:
Logistic Regression is a fundamental statistical technique utilized for binary classification tasks in machine learning and statistics. Despite its name, logistic regression is a linear model that predicts the probability of a binary outcome based on one or more predictor variables. It works by modeling the relationship between the independent variables and the logarithm of the odds of the dependent variable being in a particular category.

One of the key advantages of logistic regression is its simplicity and interpretability, making it an accessible choice for many practitioners. Additionally, logistic regression can handle both categorical and continuous predictor variables, making it versatile for various types of data.

Logistic regression estimates parameters using the maximum likelihood estimation method, which optimizes the likelihood function to find the parameters that best fit the observed data. The output of logistic regression is a probability score between 0 and 1, representing the likelihood of the binary outcome.

While logistic regression is powerful for binary classification tasks, it can be extended to handle multi-class classification through techniques like one-vs-rest or multinomial logistic regression. Moreover, logistic regression is robust to overfitting, particularly when using regularization techniques like L1 or L2 regularization.

Despite its simplicity, logistic regression may struggle with non-linear relationships between predictors and the outcome, which may require feature engineering or the use of more complex models. Nonetheless, logistic regression remains a valuable tool in the machine learning toolkit, appreciated for its simplicity, interpretability, and effectiveness in many classification tasks.

6) Naïve Bayes:

Naive Bayes is a popular and efficient machine learning algorithm used for classification tasks, particularly in text categorization and spam filtering. Despite its simplicity, Naive Bayes can achieve competitive performance and is widely adopted due to its ease of implementation and scalability.

The algorithm is based on Bayes' theorem, which calculates the probability of a hypothesis given the evidence. In the context of classification, Naive Bayes assumes that features are conditionally independent given the class label, hence the term "naive". This simplifying assumption allows Naive Bayes to make predictions quickly and with relatively low computational cost.

One of the key advantages of Naive Bayes is its ability to handle high-dimensional data efficiently, making it suitable for tasks with large numbers of features. Moreover, Naive Bayes performs well even with limited training data, making it robust in scenarios where data is sparse.

Naive Bayes comes in several variants, including Gaussian Naive Bayes for continuous features, Multinomial Naive Bayes for discrete features, and Bernoulli Naive Bayes for binary features.

While Naive Bayes may not always capture complex relationships between features, it serves as a strong baseline model and is often used in combination with more sophisticated techniques in ensemble methods. Overall, Naive Bayes remains a valuable and widely-used algorithm in the machine learning community for its simplicity, efficiency, and competitive performance in many classification tasks

3.5 Proposed Approach

In reference to section 3.2, where statistical tests were conducted, we structured the data such that features were ranked based on their performance in these tests. The top-ranked feature was placed at the forefront of the column, followed by progressively lower-ranked features. Subsequently, we constructed another table displaying the accuracy of each feature when individually assessed by machine learning algorithms, namely Decision Tree, SVM, and Random Forest.

Following the determination of single-feature accuracies, we initiated a process involving the combination of the highest-ranked feature with the subsequent highest-ranked feature from each statistical test column. For each pairing, we recalculated the accuracy of the combined features using Decision Tree, SVM, and Random Forest. Two possible scenarios emerged from this process:

1) If the accuracy of the combined features was equal to or lower than that of the highest-ranked feature alone, the combination was disregarded.
2) If the accuracy of the combined features surpassed that of the highest-ranked feature alone, the combination was retained.

This process was iterated until the lowest-ranked feature was reached, generating three distinct sets for each column. These sets were then combined, following the aforementioned method, and duplicate features were removed. The resulting set constituted the final selection of features deemed most effective for detecting network intrusions.

CHAPTER 4 RESULTS

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

The Tau-value is inversely proportional to rank of feature. Therefore, the features are placed in a descending order from to bottom, where top most feature is the best ranked according to the ANOVA test and the last feature is the least in ranking.

Ranking of Features When applied Kendall's Tau tests gives you the Tau Value:

Feature Notation	Feature Extracted	Tau Values	Rank
F1	Average Packet Size	-0.141	5
F2	Bytes	-0.105	7
F3	Average Packet Size Sent	-0.034	2
F4	Average Packet Size Received	-0.030	3
F5	Ratio of Incoming to Outgoing Packets	-0.011	6
F6	Bits sent/sec	0.029	9
F7	Bits received/sec	0.068	10
F8	Flow Duration	0.004	8
F9	Time interval between packet sent	-0.025	4
F10	Time interval between packet received	-0.049	1

4.2. Detection Results with Individual Features :-

Here we have calculated accuracy for different features individually using three different algorithms (machine learning classifiers) Decision Tree, SVM and Random Forest. All the of these classifiers give different results for the highest accuracy feature. For Decision Tree, KNN and Random Forest, the result for F10 is **98.124%, 97.163% and 84.32%** respectively while for DT F3 remains highest accuracy feature

with 96.21%. Now if we compare between all the Algorithms, we find that F10 is the most accurate individual feature of all.

Features	DT	SVM	KNN	LR	NB	RF
F1	90.082	90.609	90.082	73.06	64.4	74.22
F2	91.786	89.080	90.786	78.09	63.61	74.14
F3	96.214	93.174	94.147	81.95	69.91	81.6
F4	89.080	90.876	73.453	79.97	61.1	79.97
F5	88.789	73.477	88.789	75.98	54.28	77.91
F6	88.740	89.686	97.282	75.89	60.01	73.09
F7	88.740	80.805	73.477	76.30	66.43	76.21
F8	84.22	90.075	91.859	77.91	62.10	75.64
F9	91.370	92.753	90.418	87.94	69.18	82.18
F10	**98.124**	82.732	**97.163**	89.50	79.20	84.32

4.2 Two Features Combination:

In our classification analysis, we thoroughly examined a dataset comprising both normal and intrusion data. The main goal of our study was to explore how various combinations of features affect the accuracy of our classification model. We specifically concentrated on combining two features at a time to identify the most effective feature pairs for our classification task.

Combination Notation	DT	SVM	KNN	LR	NB	RF	Mean
F10	**0.9812**	0.8273	**0.9716**	**0.8950**	**0.7920**	**0.8432**	0.88505
F10 F3	0.6582	0.6580	0.6617	0.7986	0.7754	0.6752	0.7045
F10 F4	0.6697	**0.8688**	0.6706	0.7899	0.7644	0.6283	0.73195
F10 F9	0.7521	0.8166	0.8144	0.8133	0.7803	0.7972	0.79565
F10 F1	0.8258	0.8214	0.8254	0.6454	0.6112	0.8362	0.7609
F10 F5	0.8249	0.8246	0.8238	0.6749	0.7706	0.83	0.7914
F10 F2	0.6160	0.6340	0.6305	0.7841	0.7738	0.8121	0.7084
F10 F8	0.7904	0.7893	0.7912	0.6102	0.6117	0.8142	0.7345
F10 F6	0.7990	0.7991	0.8003	0.8008	0.7456	0.8046	0.7915
F10 F7	0.7975	0.8008	0.8006	0.8102	0.7649	0.6891	0.7711

The accuracy of F10F4 has improved to 86.88% using SVM. We will now combine three features to see if the accuracy improves further.

Combination Notation	SVM
F10 F4 F3	0.7145
F10 F4 F9	0.7041
F10 F4 F1	0.6749
F10 F4 F5	0.6102
F10 F4 F2	0.6454
F10 F4 F8	0.6752
F10 F4 F6	0.5912
F10 F4 F7	0.6419

CHAPTER 5 CONCLUSION

In this thesis, we highlight the critical importance of network security amidst the increasing prevalence of network attacks. Such attacks can result in significant financial and practical losses for organizations, associations, and individuals. Traditional security measures like antiviruses and firewalls, once sufficient, are now

inadequate to protect against these evolving threats. This necessitates the implementation of intelligent countermeasures to safeguard networks and crucial systems.

Chapter 1 delves into the concept and role of Intrusion Detection Systems (IDS). An IDS is any device or software application designed to perform intrusion detection. We explored various types of IDSs, including signature-based, anomaly-based, and hybrid detection systems. Essentially, the primary goal of an IDS is to identify intrusions before they can inflict serious damage on the network.

Chapter 2 reviews existing literature on network traffic-based intrusion detection, encompassing over 50 scholarly papers. In Chapter 3, we introduced 12 features, explaining their names and meanings. We then employed statistical tests to rank these features based on their effectiveness in detecting network intrusions, the central aim of our research. Specifically, we utilized the ANOVA test and the Chi-Square test. After explaining the principles, formulas, and functions of these tests, we introduced the machine learning classifiers used in our research: Decision Tree, SVM, and Random Forest.

The core of Chapter 3 is our methodology for ranking features. Features are organized such that the least efficient feature is at the bottom, while the most efficient is at the top. We created three columns for this ranking: one for the ANOVA test, another for the Chi-Square normal test, and a third for the Chi-Square malware test. Additionally, we prepared a table evaluating single features individually using the three machine learning classifiers.

Considering that feature combinations might yield higher effectiveness, we tested various feature combinations using the classifiers again. This process resulted in three different sets of columns. When a new combination exhibited higher accuracy than the original feature, it was ranked higher. If the accuracy remained the same or decreased, the combination was discarded.

This method was repeated for the three columns, ultimately identifying a set of features most effective for network intrusion detection. In Chapter 4, we presented all the tables, calculations, and evidence leading to our conclusion. Our research found

that the feature "Time interval between packets received" achieved the highest accuracy of 98.12% when using the Decision Tree classifier, outperforming all other features and combinations.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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 MultiClass 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 inVehicle 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 RealTime 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