Zero-Day Attack Detection Framework for Network Intrusion Detection Systems

by

Akinwale Mayomi Aisida

M.Sc in Information Technology, National Open University, Nigeria, 2021

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

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisor: Saqib Hakak, Ph.D., Faculty of Computer Science

Examining Board: Rongxing Lu, Ph.D., Faculty of Computer Science, Chair
Przemyslaw RafalPochec, Ph.D., Faculty of Computer Science
Mohsen Mohammadi, Ph.D. P.Eng, Department of Mechanical Engineering

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

December, 2024

Abstract

This study addresses the critical challenge of detecting zero-day attacks in Network Intrusion Detection Systems (NIDS) using machine learning (ML). With the NIDS market projected to reach US$5.93 billion by the year 2028 and cyber threats costing US$4.35 million per breach, improved detection is vital. A robust ML framework was developed, utilizing extensive feature engineering to reduce feature sets by 50-70% without performance loss. Zero-day scenarios were simulated using systematic attack-type exclusion, with training, validation, and testing split 60-20-20. Random Forest and XGBoost achieved high F1-scores (> 0.98) and Zero-Day Detection Rates (Z-DR). On UNSW-NB15, Random Forest achieved 100% Z-DR for seven of nine attack types; XGBoost excelled on NF-UNSW-NB15-v2. CNN and Voting Classifiers underperformed on Z-DR despite high accuracy. Kolmogorov-Smirnov tests confirmed key features' importance. This research advances NIDS by enhancing zero-day detection and improving network security.

Dedication

This thesis is dedicated to GOD, my refuge and stronghold; as well to my wife and kids.

Acknowledgements

As I conclude this significant chapter in my academic journey, it is a privilege to express my profound gratitude to those who have been instrumental in the completion of my Master's thesis. I am immensely grateful to my supervisor, Dr. Saqib Hakak for his invaluable guidance, support, and expertise throughout my research at the University of New Brunswick's Faculty of Computer Science. His mentorship has been crucial in shaping my academic journey. My affiliation with the Canadian Institute for Cybersecurity has been a pivotal part of my learning experience, offering an enriching environment for growth and innovation in cybersecurity research. I want to give special thanks to my family, Dr. Mahdi Rabbani, Dr. Hamed Jelodar, Sandi MacKinnon, Chitom Obi, Zubair, and Civic Tech Volunteers for their unwavering support and inspiration. Their achievements and encouragement have been a guiding light in my academic pursuits. I would also like to acknowledge OpenAI's ChatGPT for providing insights and suggestions during the development of this thesis. Lastly, I am profoundly indebted to my family for their spiritual and emotional support, which has been my cornerstone throughout this academic endeavour. In closing, I reflect on this journey with a sense of accomplishment and gratitude. The knowledge and experiences gained during my Master's studies are treasures that I will carry forward in my future endeavors. I am deeply thankful to all who have contributed to this journey.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

One of the most important technologies in use today is a Network Intrusion Detection System (NIDS), which operates to monitor network traffic with the ability to identify malicious activity, such as denial of service (DoS), bruteforce, spoofing and other attack categories. The global NIDS market, valued at US$2.7 billion in the year 2023, is expected to develop at an annual rate of 8.3% until the year 2028 [1]. Alarmingly, such cyber threats cost businesses an average of US$4.35 million per data breach in the year 2022 [2]. As per recent studies [3],[4],[5],[6], the major challenge for NIDS is detection of zero-day attacks. A zero-day attack [7],[8] is defined as any new attack that seeks to exploit unknown or known vulnerabilities in a network system and it is always difficult to defend. They pose a critical risk to network security, exploiting vulnerabilities that system defenders are unaware of. These attacks can remain undetected for an average of 287 days [9] and presents a serious risk to network security. These have a higher chance of not being noticed by traditional NIDS, as NIDS rely on attack signatures that are known [10].

Some of the most viable solutions to help NIDS improve their capabilities have shown to be machine learning (ML) based approaches [11],[12]. Machine learning models provide a promising solution due to their capacity to recognize intricate patterns and generalize from known attack data to identify new and unknown threats. Some studies have shown that ML-based NIDS will improve the detection rate up to 30% compared to traditional signature-based systems [13],[14],[15].

1.2 Problem Statement

Despite the critical need for robust network security, current NIDS solutions face several significant challenges:

1. Inability to detect zero-day attacks: Signature-based detection, widely used in most current NIDS, relies on known threat patterns, making it ineffective against novel attacks.

2. High false positive rates: Many existing NIDS produce an unacceptable number of false alarms. Some systems generate false positives as high as 90% [16], leading to alert fatigue and potentially masking real threats.

3. Scalability issues: As network traffic volumes increase, many NIDS struggle to maintain real-time performance. Most systems degrade in performance by up to 40% under high loads [17], creating potential blind spots in network security during peak traffic periods.

4. Lack of adaptability: The rapidly changing nature of cyber threats requires security solutions that can adapt and learn from new attack patterns. Traditional systems often lack this flexibility, leaving them increasingly obsolete as new attack vectors emerge.

These challenges collectively highlight the urgent need for innovative approaches to NIDS. The integration of machine learning techniques into NIDS offers a potential solution to these problems, promising improved detection rates, reduced false positives, and better adaptability to emerging threats. As cyber threats continue to grow in complexity and scale, the development of more sophisticated, adaptive, and efficient NIDS solutions becomes essential for maintaining robust network security. This research aims to address these challenges by developing a machine learning-based framework for detecting zero-day attacks, potentially revolutionizing the field of network intrusion detection.

1.3 Objective of the Study

This study aims to develop a robust framework for detecting zero-day attacks using machine learning techniques. The primary objectives are:

- To develop a machine learning-based framework for zero-day attack detection.
- To evaluate the model in detecting unseen attack patterns through a systematic rotation method simulating zero-day scenarios.

By achieving these objectives, this study aims to contribute to the advancement of NIDS technologies, potentially improving the detection of zero-day attacks and enhancing overall network security postures.

1.4 Scope of the Study

This study adopts a comprehensive approach to zero-day attack detection using machine learning techniques. The research focuses on five diverse machine learning models: Random Forest (RF), Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), XGBoost and Voting classifier. These models have been selected based on their proven effectiveness in network intrusion detection scenarios. For instance, Random Forest has demonstrated up to 99.8% detection rates in previous studies [7], while CNNs have achieved 99.2% accuracy in attack classification [18]. The study utilizes three prominent datasets to ensure a robust evaluation: UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018. These datasets offer a diverse representation of network traffic patterns and attack scenarios. The UNSW-NB15 dataset, containing 2,540,044 records with 49 features, provides a mix of real normal activities and synthetic contemporary attacks [17]. The NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS 2018 datasets offer NetFlow-based records, crucial for evaluating NIDS in flow-based monitoring scenarios commonly found in large-scale enterprise environments. Extensive feature engineering techniques are applied to enhance detection accuracy. This includes the extraction of additional time-based and connection-based features, following methods proposed by [17]. For the NF-UNSW-NB15-v2 dataset, 20 new statistical features are computed over 5-minute windows, as suggested by [19]. A multi-step feature selection process involving correlation analysis, ANOVA F-tests, Recursive Feature Elimination (RFE) and Random Forest importance scores, is implemented to optimize the feature set. This process typically reduces the feature set by 50-70%, enhancing model performance [20]. To simulate zero-day attack scenarios, the study employs a systematic rotation method. Each known attack type is excluded from the training data in turn, treating it as a potential zero-day attack. The models are then trained on the modified datasets and evaluated on their ability to detect the excluded attack type. This process is repeated for all attack types in each dataset, providing a comprehensive assessment of zero-day detection capabilities. Previous studies using similar methodologies have reported improvements in zero-day detection rates of up to 15-20% compared to traditional NIDS [21]. This approach allows for a rigorous evaluation of the models' ability to generalize to unseen attack patterns, closely mimicking real-world zero-day scenarios.

1.5 Summary of Contributions

This study makes several key contributions to the field of NIDS.

- We developed a comprehensive machine learning framework for zero-day attack detection, addressing a critical gap in current NIDS capabilities. This framework incorporates five diverse models (Random Forest, MLP, CNN, XGBoost and Voting classifiers), each selected for their proven effectiveness in related tasks.

- We conducted a thorough evaluation of these models across three distinct datasets (UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018), providing insights into their performance under various network conditions and attack scenarios. This multi-dataset approach enhances the generalizability of our findings.

- We implemented an extensive feature engineering process tailored to NIDS datasets. This includes the extraction of additional time-based and connection-based features, and the computation of new statistical features over specific time windows. Our feature selection process, combining correlation analysis, ANOVA F-tests, Recursive Feature Elimination (RFE) and Random Forest importance scores, typically reduces the feature set by 50-70%, optimizing model performance [20].

- We presented a comparative analysis of model performance across multiple datasets, offering important insights into the strengths and limitations of each approach in detecting zero-day attacks. This analysis can guide future research and practical implementations of ML-based NIDS.

1.6 Thesis Organization

This thesis is organized into five chapters:

Chapter 1: Presents the introduction of the study, problem statement, objectives, scope of the study and the summary of the contributions.

Chapter 2: Presents background and literature review focusing on NIDS, Zero-Day Attacks, and machine learning techniques.

Chapter 3: Details the methodology employed in this research, including dataset selection, preprocessing steps, and our approach to feature engineering. It also describes the implementation of our zero-day attack simulation method.

Chapter 4: Presents the experiments conducted and analyzes the results. It includes a comparative evaluation of the four machine learning models across the selected datasets, with a focus on their performance in detecting simulated zero-day attacks.

Chapter 5: Concludes the thesis, summarizing key findings and their implications for the field of network security. It also outlines directions for future research, including potential improvements to the proposed framework and its application in real-world scenarios.

Chapter 2

Literature review

2.1 Overview

This chapter provides a comprehensive background and critical analysis of the existing literature with regards to network intrusion detection systems (NIDS), zero-day attacks, and the application of machine learning in NIDS. It explores the fundamental concepts, challenges, and recent developments in these areas, setting the stage for our research on machine learning-based zero-day attack detection. Also the surveyed literature focuses on traditional approaches to NIDS.

2.2 Background

In today's digital environment, network security is vital to prevent network attacks. So far, NIDS has continued to play a pivotal role in safeguarding networks from various malicious activities that often attack network systems. For instance, traditional NIDS techniques, such as signature-based and anomaly-based methods, have been widely used to detect intrusions by comparing network activities against known attack signatures or identifying deviations from normal behavior [22]. However, these traditional approaches face significant limitations, including their inability to adapt to new, previously unseen threats, and their reliance on extensive labelled data for training. In recent years, machine learning (ML) has emerged as a powerful tool to enhance the capabilities of NIDS. By leveraging vast amounts of data, ML models can learn complex patterns and improve the accuracy of intrusion detection. Supervised learning, unsupervised learning, and semi-supervised learning have all been applied to NIDS, with algorithms like Decision Trees, Support Vector Machines and Neural Networks showing promising results [23]. Despite these advances, traditional ML models still require large datasets of labelled instances to perform effectively, which poses a challenge in dynamic and evolving threat landscapes. Zero Shot Learning (ZSL) [5],[24] is a novel approach to address the limitations of traditional ML models in NIDS. ZSL enables models to recognize and categorize new, unseen threats based on their understanding of previously learned classes without the need for labelled examples of every possible intrusion type. Thus, this research is carried out to further discover the benefits of ZSL for network security.

2.3 Network Intrusion Detection Systems (NIDS) and Its Types

NIDS are vital tools in the realm of cybersecurity, designed to detect unauthorized access and malicious activities within a network [25],[10],[22]. Ahmad et al. [26] noted that these systems play a crucial role in maintaining the integrity, confidentiality, and availability of information by identifying potential threats and alerting administrators to take corrective actions. The concept of NIDS can be broadly categorized into two main types: signature-based and anomaly-based detection [27],[28]. Few studies have also proposed the category of hybrid [29],[30],[31].

1. <u>Signature-based NIDS</u> operate by comparing incoming network traffic against a database of known attack signatures. This method is highly effective in detect-

ing known threats due to its specificity [31],[32],[33]. However, its effectiveness diminishes when encountering novel or modified attacks that do not match existing signatures, leading to significant challenges in adapting to new threats [34].

2. Anomaly-based NIDS in contrast, establish a baseline of normal network behavior and monitor for deviations from this norm. This approach enables the detection of previously unknown threats by identifying unusual patterns in network traffic [35]. While this method offers greater adaptability, it is prone to generating false positives, such as benign variations in network activity can be incorrectly flagged as potential threats [36],[37].

3. Hybrid NIDS combines the strengths of both signature-based and anomaly-based techniques to enhance detection capabilities and reduce false positives [30],[29]. By integrating these approaches, hybrid systems aim to provide a more comprehensive and resilient solution for network security [38]. This integration allows for the precise detection of known threats while maintaining the flexibility to identify new and emerging attacks. Despite their importance, NIDS face ongoing challenges, including the need for constant updates to signature databases and the computational demands of analyzing large volumes of network traffic in real-time [26],[39]. Moreover, the increasing sophistication of cyber-attacks necessitates continuous advancements in NIDS technologies to keep pace with evolving threats [40],[41].

2.4 Machine Learning Techniques in NIDS

Machine Learning (ML) has become a transformative approach in enhancing NIDS, offering advanced techniques to improve detection accuracy and adaptability. ML methods in NIDS can be broadly categorized into supervised, unsupervised, and semi-supervised learning, each with its distinct mechanisms and applications [42],[43].

2.4.1 Supervised Learning

Supervised learning involves using a labelled dataset to train a model, where each training example is paired with a corresponding output label [44]. This method is prevalent in NIDS for its ability to build predictive models based on historical attack data. Common algorithms used include Decision Trees, Support Vector Machines (SVM), and Neural Networks. Decision Trees, for instance, classify network traffic based on a series of binary decisions derived from training data [45]. SVMs are effective in creating hyperplanes that separate different classes of data, which is useful for differentiating between normal and malicious activities [27],[46]. Neural Networks, including deep learning models, are employed for their capability to model complex relationships in large datasets [47]. While supervised learning can achieve high accuracy, it requires extensive labelled data, which can be a significant limitation in dynamic environments where new attack types frequently emerge [48],[21].

2.4.2 Unsupervised Learning

Unsupervised learning, without requiring labelled data, identifies patterns and structures within the data through clustering and association techniques. Common approaches include K-Means clustering and Principal Component Analysis (PCA). K-Means clustering groups similar data points together, which helps in identifying patterns of anomalous behavior that deviate from normal clusters [49]. PCA is used to reduce data dimensionality while retaining essential features, which helps in detecting anomalies that might otherwise be obscured by high-dimensional noise [50],[51]. Unsupervised learning is advantageous in discovering novel attack patterns and adapting to new threats without requiring prior knowledge. However, it often results in higher false positive rates due to the lack of prior classification, which can make it challenging to differentiate between benign, anomalies and actual attacks [52],[53].

2.4.3 Semi-Supervised Learning

Semi-supervised learning combines component of both supervised and unsupervised learning, leveraging a limited amount of labelled data together with a larger set of unlabelled data. This approach aims to improve model accuracy and generalization by utilizing the strengths of both methods. Techniques such as Self-Training and Co-Training are used in semi-supervised learning for NIDS [54]. Self-Training involves iteratively using the model's predictions on unlabelled data to refine the learning process [55]. Co-Training, on the other hand, involves training multiple models on different feature sets and iteratively exchanging labels to enhance learning [56]. Semi-supervised learning is particularly useful in scenarios where obtaining labelled data is expensive or impractical, making it a valuable tool for NIDS in evolving threat environments. However, it still faces challenges related to the initial labeling accuracy and the integration of labelled and unlabelled data [57].

2.5 Zero-Day Attacks

Zero-day attacks are a form of cybersecurity threat used to manipulate previously unknown vulnerabilities in software or systems. They are the most hazardous because they occur before a developer can build and release patches, leaving the systems open for a very long time. They were named "zero-day" because developers have had "zero days" to tackle and patch the bug. The sophistication and frequency of zero-day attacks have grown manyfold in the last couple of years. Project Zero at Google reported that in the year 2022 alone, 68 zero-day vulnerabilities were exploited in the wild, a 12 percent rise compared to the year 2021 [58]. Since no signature exists for an unknown vulnerability, such attacks easily manage to bypass traditional signature-based detection techniques. The consequences of zero-day attacks might be widespread and devastating. In the year 2017, ransomware WannaCry targeted Windows systems through a zero-day vulnerability, spread to more than 200,000 computers in at least 150 countries, and accumulated the estimated US$4 billion in damage [59]. In an even more recent attack, the SolarWinds supply chain attack from 2020 exploited several zero-day vulnerabilities that compromised some government agencies and major corporations and exposed full potential for this kind of attack [60]. IBM reports that the average time to identify and contain a data breach, which often involves zero-day exploits, in the year 2022 was 287 days [2]. This period of vulnerability underlines the urgent need for more effective means of detection, especially means capable of identifying novel patterns of attack on a real-time basis.

2.6 Feature Engineering for NIDS

Feature engineering plays a crucial role in the effectiveness of ML models for NIDS. Well-designed features can significantly improve model accuracy and reduce false positive rates. A study by [19] found that appropriate feature engineering could improve zero-day attack detection rates by up to 20%. In NIDS, features are typically categorized into three types:

1. Packet-Based Features: These are derived from individual network packets and include attributes like packet size, protocol type, and inter-arrival time. Packet-based features are crucial for detecting attacks that manifest at the packet level, such as certain types of distributed denial of service (DDoS) attacks.

2. Flow-Based Features: Derived from network flows, these features include connection duration, total bytes transferred, and flow direction. Flow-based features are particularly useful for detecting attacks that occur over multiple pack-

ets or connections, such as data exfiltration attempts.

3. Custom Features: These are additional features derived from domain knowledge or specific security requirements. For example, [17] proposed a set of time-based features for the UNSW-NB15 dataset, which improved detection rates for certain attack types by up to 15%.

2.6.1 Feature selection

It is a critical step in the feature engineering process which common techniques include:

- Correlation Analysis: This technique identifies and removes highly correlated features to reduce redundancy. Zhou et al. [20] reported that removing features with correlation coefficients > 0.95 could reduce the feature set by 30-40% without significant loss in model performance.

- ANOVA F-Test: This statistical test helps select features that contribute most to distinguishing between normal and attack traffic. Sarhan et al. [19] used ANOVA F-tests to select the top 30 features from the NF-UNSW-NB15-v2 dataset, improving model efficiency while maintaining high detection rates.

- Recursive Feature Elimination (RFE): An iterative method that progressively removes the least important features. A study by [61] found that RFE could reduce the feature set by up to 70% while maintaining or even improving model accuracy.

These feature engineering techniques, when applied effectively, can significantly enhance the performance of ML models in NIDS, particularly in the challenging task of detecting zero-day attacks.

2.7 Related works

In [62], the authors explored attribute learning methods to detect unknown attack types. The authors focused on a Zero-Shot Learning (ZSL) architecture to design a NIDS to overcome the limitations in anomaly detection faced by current methods. This architecture features a pipeline that utilizes Random Forest (RF) for feature selection and a spatial clustering attribute conversion method. The results indicate that the proposed method surpasses state-of-the-art approaches in anomaly detection. The attribute learning framework transforms network data samples into unsupervised cluster attributes. The attribute learning framework converts network data samples into unsupervised cluster attributes. The NSL-KDD dataset was used to evaluate this framework, successfully detecting DoS (apache2) and Probe (saint) attacks with an overall accuracy of 34.71%.

Alok et al. [63] proposed a hybrid evolutionary approach for detecting zero-day attacks on wired/wireless network systems. Long Short-Term Memory (LSTM) techniques must be able to identify harmful actions that depart from well established patterns and behaviours, displaying a degree of adaptability and foresight unequalled by traditional security systems.

The work done by [64] aligns with efforts to enhance the effectiveness of NIDS against unknown threats, particularly zero-day attacks, while also focusing on mitigating false positives. In the meantime, [65] presents Deep Transductive Transfer Learning (DTTL), which creates soft labels for target instances by using cluster correspondence. This technique transfers knowledge from labelled source domains to unlabelled target domains by using a DNN classifier and taking advantage of domain correlation. By aligning domains and taking advantage of correlations, this strategy transfers knowledge to detect zero-day attacks, whereas LSTM-based techniques are superior at identifying temporal patterns in network traffic data.

Luu et al. [24] proposed a zero-shot deep learning approach to detect botnet samples from both known and unknown families on previously unseen devices. They introduced a Deep Sparse Contrastive Auto-Encoder (DSCAE) model for botnet detection, enhancing performance by extracting meaningful latent feature representations. Additionally, they evaluated their approach against traditional machine learning methods and a simple Auto-Encoder. The results show that the proposed model improves performance in most cases and often achieves the highest metric scores.

Xian et al. [66] proposed a framework leveraging semantic embeddings for ZSL in network security. Their approach utilized attributes and textual descriptions of known attack types to create semantic spaces. They applied these embeddings to classify novel attack types by comparing them to known classes in the semantic space. The study highlighted the effectiveness of semantic embeddings in bridging the gap between seen and unseen attacks and demonstrated improved performance in detecting novel attacks compared to traditional methods.

In this [67], the authors evaluated the performance of ML-based Network Intrusion Detection Systems (NIDS) in detecting zero-day attacks. They utilized Zero-Shot Learning (ZSL) to simulate zero-day attack scenarios, using a sparse autoencoder model that maps attributes of known attacks into a semantic space to enable the detection of unknown attacks. The ML models learn distinctive information between attack and benign classes by mapping feature and attribute spaces. The study utilised attacks from the NSL-KDD dataset released in 1998, which contains four types of attack scenarios, to simulate a zero-day environment. Results showed an average accuracy of 88.3% across all attacks in the dataset.

In this paper [68], an investigated hybrid approaches that combine ZSL with traditional anomaly detection techniques was proposed. Their research integrated semantic embeddings with anomaly detection frameworks to enhance the detection of previously unknown attacks. By combining these approaches, the study aimed to leverage the strengths of both methods, achieving better detection rates for novel attacks while maintaining robustness against known attack types. The results showed that hybrid models could offer a balanced approach, addressing some of the limitations of pure ZSL methods.

In [69], the authors proposed a robust model for detecting zero-day attacks by utilizing high-traffic volumes to identify heavy-hitter, high-volume attacks and using low-volume attacks to generate patterns through a graph-based approach. The framework used two main phases: a pattern generation phase and an evaluation phase. Accuracy detection is assessed using patterns developed during training. Testing on a real-time attack dataset achieved accuracies of 91.33% for binary classification and 90.35% for multi-class classification. When evaluated on the CICIDS18 benchmark dataset, the model achieved 91.62% accuracy for binary classification and 88.98% for multi-class classification.

In [5], the authors proposed a novel zero-shot learning approach to evaluate the performance of ML-based Network Intrusion Detection Systems (NIDS) in detecting zero-day attack scenarios. During the attribute learning phase, the models map network data features to semantic attributes that differentiate between known attacks and benign behavior. In the inference stage, the models established relationships between known and zero-day attacks to classify them as malicious. A new evaluation metric, Zero-day Detection Rate (Z-DR), is introduced to assess the effectiveness of the learning model in identifying unknown attacks. The proposed framework is evaluated using two key machine learning (ML) models and two modern NIDS datasets. The results indicate that, for certain zero-day attack groups identified in this study, ML-based NIDS are ineffective in detecting them as malicious. Further analysis reveals that attacks with a low Z-DR exhibit significantly different feature distributions and a wider range of Wasserstein Distance compared to other attack classes.

2.8 Limitations of Traditional NIDS

Traditional Network Intrusion Detection Systems (NIDS), despite their crucial role in network security, face several significant limitations that impact their effectiveness in the ever-evolving landscape of cyber threats. These limitations are particularly evident in signature-based and anomaly-based detection methods. Signature-based NIDS relies on a database of known attack patterns or signatures to detect malicious activities. This approach is highly effective for identifying well-known threats due to its precision in matching signatures [70],[22]. However, its reliance on pre-existing signatures makes it inherently reactive, as it cannot detect new or unknown threats that have not yet been cataloged [71],[15]. This inability to recognize zero-day exploits and novel attack vectors is a critical vulnerability, leaving networks exposed to emerging threats until signatures are updated. Anomaly-based NIDS, which monitors network traffic for deviations from established normal behavior, aims to address some of the shortcomings of signature-based systems. While this method can potentially identify previously unknown threats, it is plagued by high false positive rates [72]. The baseline of normal behavior is challenging to define accurately, and legitimate network activities can often be misclassified as anomalies, leading to frequent false alarms. This not only burdens security teams with excessive alerts but also risks desensitizing them to real threats due to alert fatigue. Another limitation common to both signature-based and anomaly-based NIDS is their computational demands. Analyzing vast amounts of network traffic in real-time requires significant processing power and resources [73]. This can be particularly challenging in large, high-speed networks where the volume of data to be processed is substantial. The resource-intensive nature of NIDS can lead to performance bottlenecks and delays in threat detection, undermining their effectiveness in protecting network infrastructure. Furthermore, both types of NIDS struggle with encrypted traffic. As encryption becomes more prevalent to protect data privacy, NIDS that rely on inspecting packet contents for signatures or anomalies face challenges in analyzing encrypted data streams [74]. This limitation reduces the visibility of NIDS into network activities, allowing sophisticated attackers to hide their actions within encrypted traffic. Lastly, traditional NIDS often lack adaptability. Cyber threats are continually evolving, with attackers developing new techniques to bypass detection mechanisms. Traditional NIDS, especially those dependent on static signatures, require constant updates and tuning to remain effective [75]. This ongoing maintenance is resource-intensive and can lead to periods of vulnerability if updates are not applied promptly.

2.9 Limitations of Existing Research and Identified Gaps

Current NIDS research faces several significant limitations. A primary challenge is the lack of comprehensive models that efficiently detect zero-day attacks across diverse datasets. While studies like [20] achieved 85% detection rates for simulated zero-day attacks on UNSW-NB15, performance often degrades when models are applied to different network environments. This highlights the need for more generalizable approaches. Over-reliance on labelled datasets for supervised learning models remains a persistent issue. Sarhan et al. [76] noted that up to 70% of real-world network traffic may be unlabelled, limiting the effectiveness of supervised techniques. This gap underscores the importance of developing semi-supervised and unsupervised methods for NIDS. High computational requirements pose challenges for real-time NIDS implementation, particularly for deep learning models. Vinayakumar et al. [77] reported that their CNN-based model, while achieving 99.2% accuracy, required 15 seconds to process 1 minute of network traffic on high-end GPUs, making it impractical for many real-world scenarios.

Opportunities for improvement lie in integrating advanced feature engineering with more scalable models. XGBoost has shown promise in this regard, with [78] demonstrating a 93% detection rate for novel attacks while maintaining lower computational overhead compared to deep learning approaches. More research is needed on decentralized machine learning approaches like Federated Learning (FL) to address data privacy concerns. While Elenguer et al. [79] showed FL's potential in IoT environments, achieving 92% detection rates for novel attacks, scalability and communication efficiency remain significant challenges.

Sarhan et al. [5] still faces some challenges like low detection rate for certain attacks, limited generalized datasets, no feature selection technique, however, the proposed methodology aims to address these research gaps. In this study, we have considered the recent work of [5] as a baseline study. This work has a few challenges such as a low detection rate for certain attacks, limited generalized datasets, and no future selection technique. The proposed methodology aims to address these challenges to develop an efficient NIDS for zero day detection

Chapter 3

Zero Day Detection Framework

3.1 Overview

This chapter presents a rotation-based machine learning-based framework to detect zero-day attacks in NIDS. As illustrated in Figures in 3.1 and 3.2, a systematic approach was employed to ensure clarity and precision in the execution of the experiment. Five machine learning methods RF, MLP, CNN, XGBoost and Voting Classifiers were used. Therefore, five models were trained and tested on three different datasets: UNSWNB-15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018.

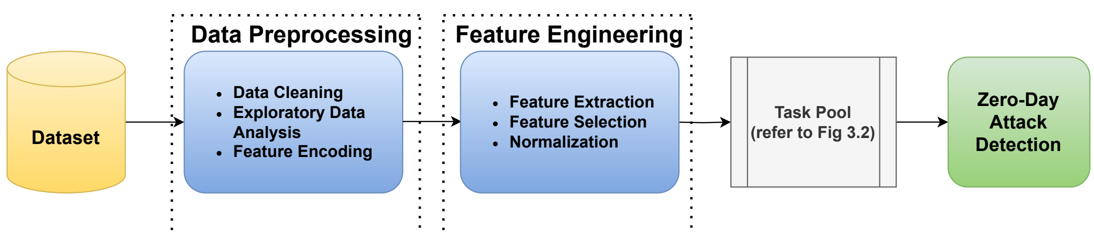

Figure 3.1: Zero-Day Detection Framework

In addition, our approach basically hinges on systematically excluding one attack type after another at training time to simulate zero-day scenarios. For this reason, we rotate the sets to asynchronously evaluate the generalization of the models to unseen attack patterns. We incorporated extensive feature engineering, including feature extraction and selection techniques in optimizing model performance.

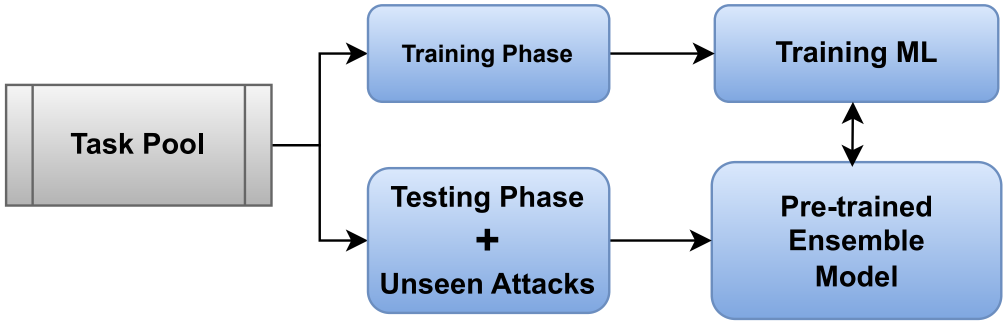

Figure 3.2: Rotational Task Pool

3.2 Zero-Day-Based NIDS Framework

The proposed framework is presented in Figure 3.1. Our framework has the following components:

3.2.1 Datasets

Our methodology employed the utilization of three distinct datasets, each serving crucial role in training and validation of our models.

This study utilized three prominent datasets in the field of Network Intrusion Detection Systems (NIDS): UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018. These datasets were selected for their contemporary representation of modern network traffic patterns and diverse attack scenarios.

- Dataset 1 (UNSW-NB15): The UNSW-NB15 dataset, created by [80], is a hybrid dataset containing both real modern normal activities and synthetic contemporary attack behaviors. It consists of 2,540,044 records, which are divided into 2,218,761 (87.35%) normal traffic instances and 321,283 (12.65%) attack instances. The dataset features 49 attributes, including packet-based features, flow-based features, and additional generated features. It encompasses nine types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The UNSW-NB15 dataset is particularly relevant to NIDS research due to its comprehensive coverage of modern attack types and its inclusion of both CSV and packet capture (PCAP) files, allowing for in-depth analysis [17].

- Dataset 2 (NF-UNSW-NB15-v2): The NF-UNSW-NB15-v2 dataset is a NetFlow version of the original UNSW-NB15 dataset, generated by [81]. It contains 2,390,275 flow records, with 2,295,222 (96.02%) benign flows and 95,053 (3.98%) attack flows. The dataset features 43 NetFlow-based attributes, making it particularly suitable for evaluating NIDS in environments where only flow-level data is available. The attack types remain consistent with the original UNSW-NB15 dataset. This dataset is crucial for our study as it allows us to assess the effectiveness of our zero-day detection approach in flow-based network monitoring scenarios, which are more common in large-scale enterprise environments [76].

- Dataset 3 (NF-CSE-CIC-IDS 2018): This dataset is a NetFlow version of the CSE-CIC-IDS 2018 dataset, created by [76]. It contains 8,392,401 total flows out of which 1,019,203 (12.14%) are attack samples and 7,373,198 (87.86%) are benign ones. The dataset features 83 NetFlow-based attributes, making it particularly suitable for evaluating NIDS in large-scale network environments. It includes a wide range of attack scenarios such as DoS, DDoS, Brute Force, Botnet, and Web Attacks, providing a comprehensive representation of modern cyber threats.

The use of these three datasets provides a comprehensive foundation for evaluating our zero-day attack detection methodology. By leveraging datasets with different characteristics and attack types, we aim to develop and validate a robust approach that can generalize across various network environments and threat landscapes

3.2.2 Preprocessing

Preprocessing is one of the high-priority steps for maintaining the quality and consistency of datasets, critical for ensuring robust model performance. Our preprocessing pipeline was fully automated and included comprehensive logging to guarantee reproducibility. The pipeline involved handling missing values, outlier detection, encoding, scaling, and addressing class imbalances across the three datasets: UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018.

3.2.2.1 Missing Data Handling

Missing data are rows with empty cells or incomplete data where data or complete data should be present in a dataset [82].

For UNSW-NB15, significant missing values were detected. In particular, the attack_cat feature had 2,218,764 missing values, which were replaced with 'normal' to preserve data completeness. Other features with high proportions of missing data included service (49.07%), ct_flw_http_mthd (53.08%), and is_ftp_login (56.29%). Table 3.1 summarizes the missing data distribution for key features. After removing rows with missing data (excluding attack_cat), the dataset size was reduced by 82.8%, from 2,540,047 to 436,822 rows.

Table 3.1: Missing values for different features

Feature	**Missing Values**	**Percentage**
service	1,246,397	49.07%
ct_flw_http_mthd	1,348,145	53.08%
is_ftp_login	1,429,879	56.29%

For the NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS 2018 datasets, no missing values were identified, maintaining the original row counts of 2,390,275 and 8,392,401, respectively. Figure 3.3 highlights the preprocessing data.

Figure 3.3: Preprocessing Data

3.2.2.2 Outlier Detection and Treatment

Outliers were detected and handled using two primary methods: the Interquartile Range (IQR) method, which detected the presence of extreme values by identifying points below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR and Z-score method which flagged value with absolute z-scores greater than 3 for futures which assume normal distribution [83], [84]. In the UNSW-NB15 dataset, significant outliers were identified, particularly in the dur feature, where 451,297 outliers were detected using the IQR method, and 6,458 outliers were identified using the Z-score method. Similarly, in the NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS 2018 datasets, large numbers of outliers were observed, especially in critical features such as TCP_FLAGS, L4_SRC_PORT, and L7_PROTO. For instance, the NF-UNSW-NB15-v2 dataset had 561,892 outliers in the TCP_FLAGS feature detected by the IQR method, and NF-CSE-CIC-IDS2018 recorded 1,637,413 outliers for the L4_SRC_ PORT feature.

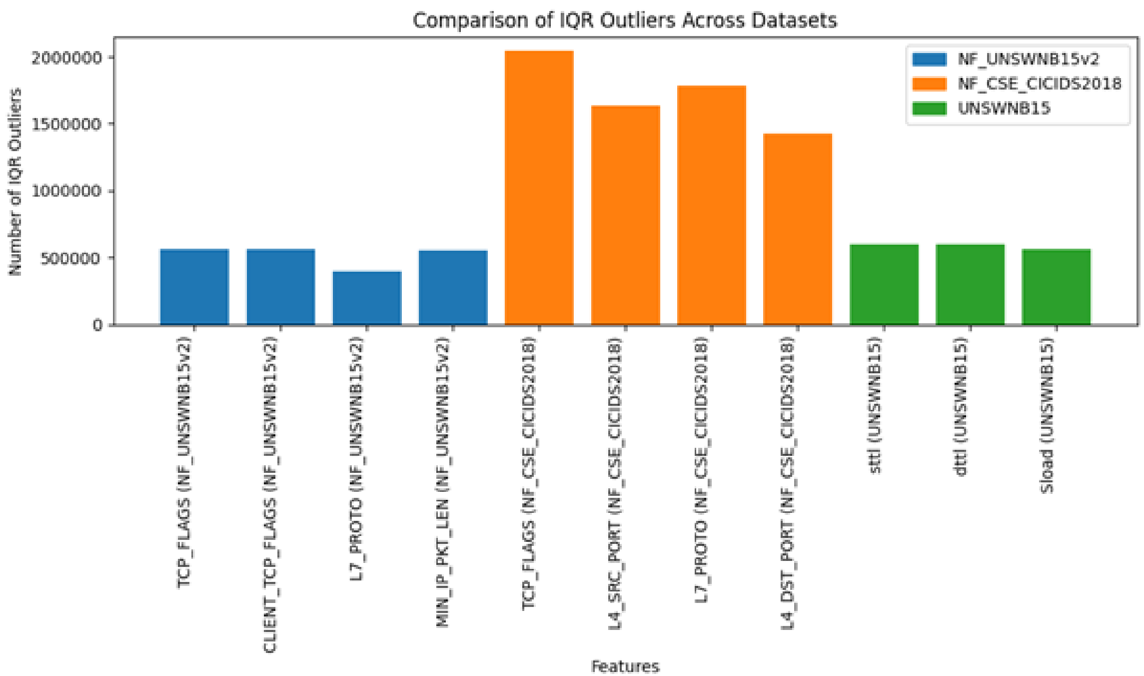

Figure 3.4: Outlier Detection

Figure 3.4 provides a visual representation of these outliers, aiding in decisions on whether to remove or cap them based on predefined thresholds. Outlier treatment contributed to improving the data quality of the datasets, without significantly reducing their size. Notably, both the NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS 2018 datasets retained all of their original rows following outlier treatment, ensuring the completeness of the datasets for further analysis. This provides a clear summary of the outlier detection process across the datasets and highlights the importance of managing outliers to improve data quality while maintaining dataset size

3.2.2.3 Exploratory Data Analysis (EDA) and Feature Encoding

We conducted exploratory data analysis (EDA) to gain insight into the distributions and statistical properties of features in each dataset. For numerical features, boxplots and histograms revealed skewness, which was addressed through log transformation. Categorical features, such as protocol_type from UNSW-NB15, were encoded using Label Encoding for ordinal variables and One-Hot Encoding for nominal variables. One-hot encoding was limited to categories present in at least 1% of the samples to minimize dimensionality.

The encoded datasets were saved as separate CSV files, with all transformations logged for reproducibility. Additionally, descriptive statistics for key continuous variables, such as IN_BYTES and OUT_BYTES, are provided in Tables 3.2, 3.3 and 3.4 to illustrate the variability and central tendencies of features across the datasets.

Table 3.2: Descriptive Statistics for Continuous Variables (UNSW-NB15)

Feature	**Count**	**Mean**	**Std Dev**	**Min**	**25%**	**50%**	**75%**	**Max**
sport	436,822	32,593.14	19,075.83	53	15,948.25	32,849	48,661	65,535
dsport	436,822	4,669.10	13,272.10	21	53	53	80	65,504
dur	436,822	0.59	2.12	0	0.001	0.007	0.85	59.99
sbytes	436,822	5,515.32	37,167.47	56	146	424	2,934	9,779,191
dbytes	436,822	29,288.70	165,688.20	0	178	824	8,824	14,655,420
sttl	436,822	35.87	30.92	31	31	31	31	255
dttl	436,822	29.93	20.68	0	29	29	29	254

Table 3.3: Descriptive Statistics for Continuous Variables (NF-UNSW-NB15-v2)

Feature	**Count**	**Mean**	**Std Dev**	**Min**	**25%**	**50%**	**75%**	**Max**
IN_BYTES	2,390,275	4,621.66	70,214.09	1	481	1,684	3,286	30,241,410
OUT_BYTES	2,390,275	35,265.65	156,515.50	0	312	2,456	14,968	14,658,520
FLOW_DURA TION_MS	2,390,275	259,111.70	1,022,610	0	0	0	0	4,294,965

Table 3.4: Descriptive Statistics for Continuous Variables (NF-CSE-CIC-IDS 2018)

Feature	**Count**	**Mean**	**Std Dev**	**Min**	**25%**	**50%**	**75%**	**Max**
IN_BYTES	8,392,401	3,761.34	130,629.80	0	65	153	1,360	24,371,270
OUT_BYTES	8,392,401	4,346.90	257,957.80	0	0	120	700	214,122,900
FLOW_DURA TION_MS	8,392,401	2,832,315	2,030,250	0	0	4,291,835	4,294,906	4,294,967

3.2.2.4 Class Imbalance Handling

All three datasets exhibited significant class imbalances, particularly the UNSW-NB15 dataset, where only 2.40% of instances represented attacks. We addressed this imbalance using Random Undersampling (RUS), which reduced the majority class (benign traffic) to achieve more balanced class distributions. For instance, in NF-CSE-CIC-IDS 2018, attack traffic represented 12.14% of the dataset post-undersampling.

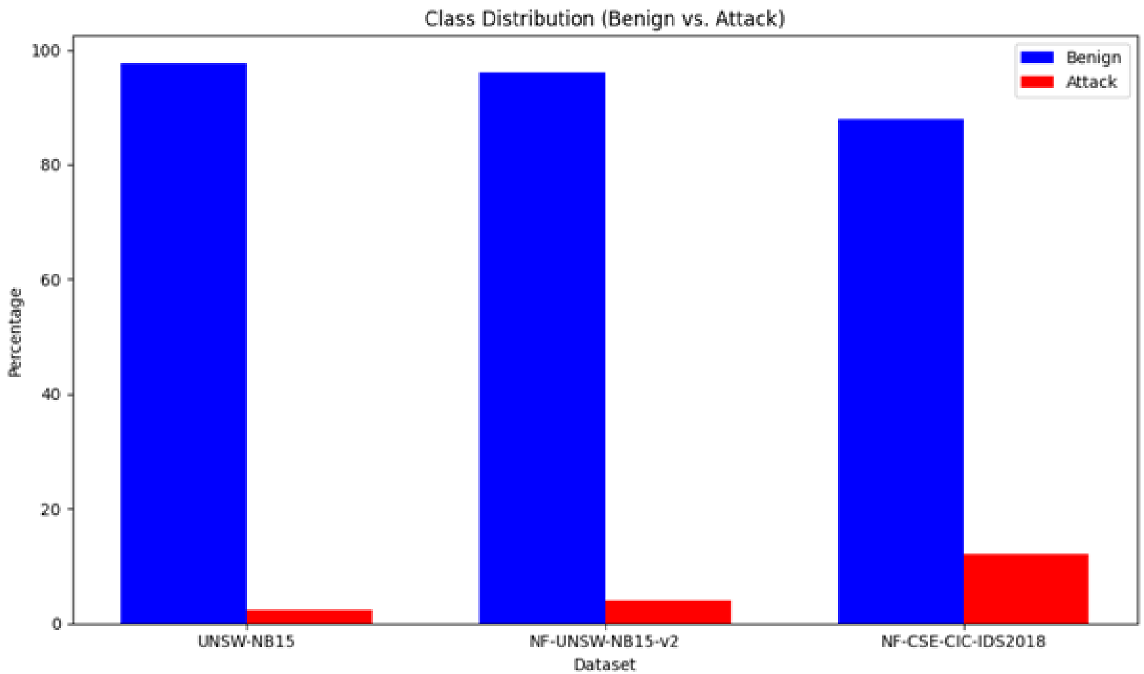

Figure 3.5: Class Distribution of Dataset

To further mitigate class imbalance effects during model training, we computed class weights utilizing the compute_class_weight function from scikit-learn to assign higher weights to underrepresented classes.

These weights improved the model's ability to detect minority attack classes. Figure 3.5 illustrates the distribution of attack categories in UNSW-NB15 and NF-UNSW-NB15-v2. Generic attacks are the most common in UNSW-NB15 (1.64%), while Exploits dominate in NF-UNSW-NB15-v2 (1.32%). Both datasets contain a diverse range of attack types, including Fuzzers, DoS, and Reconnaissance, albeit in varying proportions.

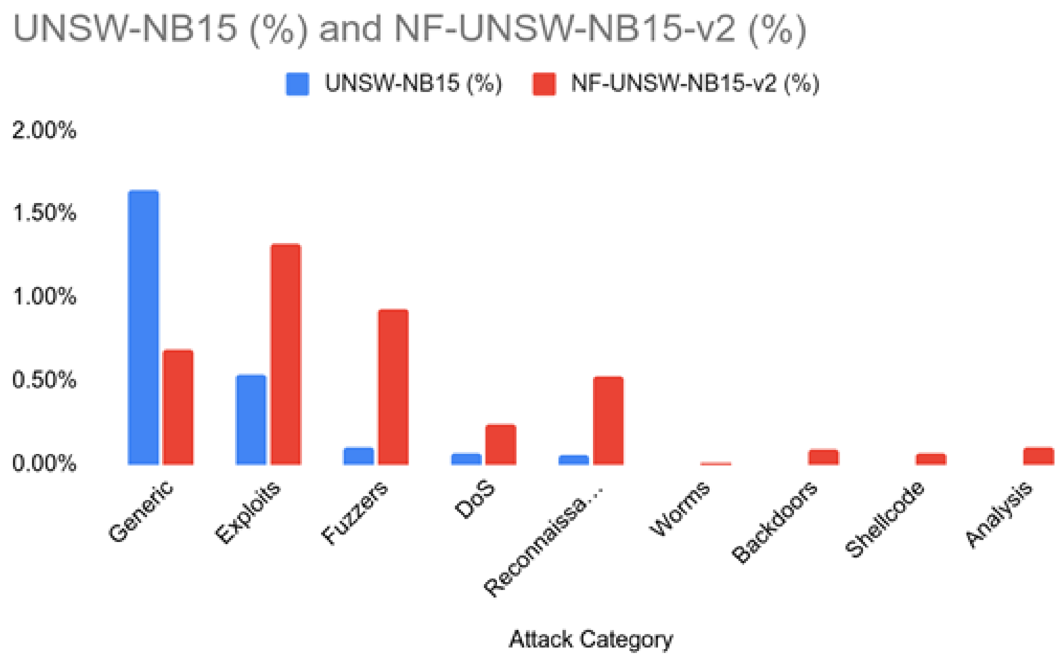

Figure 3.6: Class Distribution of Dataset

3.2.2.5 Summary of Preprocessing Results

The preprocessing results revealed stark differences in data quality and attack distributions across the three datasets as shown in Figure 3.6. UNSW-NB15 required extensive cleaning, reducing the dataset size significantly, while NF-UNSW-NB15-v2 and NF-CSE-CIC-IDS 2018 retained their original sizes. Despite retaining more rows, both datasets exhibited substantial outliers, which were appropriately addressed.

3.2.3 Feature Engineering Results

Feature engineering played a crucial role in enhancing the predictive power of our machine learning models across the three datasets: UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018. This process involved creating additional features that capture important characteristics of network traffic, potentially improving the models' ability to detect zero-day attacks.

3.2.3.1 Scaling and Normalization

To ensure consistency across feature scales, Min-Max scaling was applied to the UNSW-NB15 and NF-UNSW-NB15-v2 datasets, transforming values into the range [0, 1]. For the CIC-IDS 2018 dataset, Z-score normalization was used, transforming features to have a mean of 0 and a standard deviation of 1. These transformations aimed to facilitate more effective machine learning model performance.

1. For the UNSW-NB15 dataset, we engineered 9 new features as shown in Table 3.5. These features focused on flow-level metrics, such as flow_bytes_per_second and flow_pkts_per_second, which provide insights into data throughput and activity. We also included ratio-based features like src_dst_pkt_ratio and src_dst_byte_ratio to capture traffic symmetry, which can be indicative of certain attack types. Jitter and inter-arrival time features (total_jitter and total_inter arrival_time) were added to detect variations in network traffic often associated with malicious activities.

2. The NF-UNSW-NB15-v2 dataset, being a NetFlow version, required a different approach. We created 10 new features as shown in Table 3.6, including flow_duration_seconds and throughput metrics (bytes_per_second and packets_per_second). TCP flag features (tcp_syn_flag and tcp_ack_flag) were introduced to identify patterns in connection setup, crucial for detecting attacks like SYN flooding. We also added packet size ratio features and retransmission metrics, which can help in identifying abnormal behaviors such as scanning or evasion techniques.

3. For the NF-CSE-CIC-IDS 2018 dataset, we engineered 12 new features as

Table 3.5: Feature Engineering Summary for UNSW-NB15

New Feature Name	Description	Calculation Method
flow_bytes_per_second	Flow bytes per second (sum of source and destination bytes)	(sbytes + dbytes) / dur
flow_pkts_per_second	Flow packets per second (sum of source and destination packets)	(Spkts + Dpkts) / dur
src_dst_pkt_ratio	Ratio of source to destination packets	Spkts / Dpkts
src_dst_byte_ratio	Ratio of source to destination bytes	sbytes / dbytes
flow_bytes_per_packet	Bytes per packet for the entire flow	(sbytes + dbytes) / (Spkts + Dpkts)
avg_src_pkt_size	Average packet size for source	sbytes / Spkts
total_jitter	Total jitter (source and destination)	Sjit + Djit
total_interarrival_time	Total inter-packet arrival time	Sintpkt + Dintpkt
tcp_rtt	TCP round-trip time (RTT)	tcprtt (pre-existing)

shown on Table 3.7. This included categorical features for ports and protocols (src_port_ category, dst_port_category, protocol_group) to simplify the analysis of traffic based on service types and major protocols. We expanded on TCP flag decoding (tcp_syn_flag, tcp_ack_flag, tcp_fin_flag) to capture vital connection setup and teardown activities. Flow-level metrics similar to those in the other datasets were also included.

3.2.3.2 Feature Selection Outcomes

The feature selection process was implemented as a multi-step pipeline to ensure only the most relevant and informative features were retained. We began with correlation analysis, removing features with high multicollinearity (correlation coefficient > 0.95) to eliminate redundant information. Features that showed strong correlation with the target variable (correlation > 0.5) were prioritized. Following this, we applied

Table 3.6: Feature Engineering Summary for NF-UNSW-NB15-v2

New Feature Name	Description	Calculation Method
flow_duration_seconds	Flow duration in seconds	FLOW_DURATION_MILLI SECONDS/1000
bytes_per_second	Throughput (bytes per second)	(IN_BYTES+OUT_BYTES)/ flow_duration_seconds
packets_per_second	Throughput (packets per second)	(IN_PKTS+OUT_PKTS)/ flow_duration_seconds
tcp_syn_flag	TCP SYN flag (whether SYN flag is set)	TCP_FLAGS & 0x02
tcp_ack_flag	TCP ACK flag (whether ACK flag is set)	TCP_FLAGS & 0x10
small_packet_ratio	Ratio of small packets to total packet count	NUM_PKTS_UP_TO_128_ BYTES/total_packet_count
large_packet_ratio	Ratio of large packets to total packet count	NUM_PKTS_1024_TO_1514_ BYTES/total_packet_count
ttl_diff	Difference between max and min TTL	MAX_TTL - MIN_TTL
total_retransmitted_bytes	Total retransmitted bytes (sum of retransmitted in/out bytes)	RETRANSMITTED_IN_ BYTES+RETRANSMITTED_ OUT_BYTES
total_retransmitted_packets	Total retransmitted packets (sum of retransmitted in/out packets)	RETRANSMITTED_IN_PKTS +RETRANSMITTED_ OUT_PKTS

the ANOVA F-value test to rank features based on their discriminative power and selected the top 30 features. The third step utilized Random Forest importance scores, retaining features above the 25th percentile to ensure we kept only the most influential predictors. Finally, we employed Recursive Feature Elimination (RFE) with a Random Forest classifier to select the top 10 features in a more rigorous, iterative manner. To make our final feature selection, we took an ensemble approach. Features that were selected by at least three of the four methods (correlation analysis, ANOVA, Random Forest importance, and RFE) were retained for the final feature set. This conservative approach ensured that our selected features were robust and relevant, having been validated through multiple statistical and machine learning techniques. By combining these different selection methods, we were able to create a feature set that balances predictive power with model simplicity, enhancing both the

Table 3.7: Feature Engineering Summary for NF-CSE-CIC-IDS 2018

New Feature Name	Description	Calculation Method
src_port_category	Categorical grouping of source ports	Port categorization function
dst_port_category	Categorical grouping of destination ports	Port categorization function
protocol_group	Grouping of protocols (e.g., TCP, UDP, ICMP)	Protocol categorization function
tcp_syn_flag	TCP SYN flag (whether SYN flag is set)	TCP_FLAGS & 0x02
tcp_ack_flag	TCP ACK flag (whether ACK flag is set)	TCP_FLAGS & 0x10
tcp_fin_flag	TCP FIN flag (whether FIN flag is set)	TCP_FLAGS & 0x01
flow_duration_seconds	Flow duration in seconds	FLOW_DURATION_MILLI SECONDS / 1000
bytes_per_second	Bytes transferred per second	(IN_BYTES+OUT_BYTES)/ flow_duration_seconds
packets_per_second	Packets transferred per second	(IN_PKTS+OUT_PKTS)/ flow_duration_seconds
total_bytes	Total bytes transferred (inbound + outbound)	IN_BYTES + OUT_BYTES
in_out_bytes_ratio	Inbound to outbound byte ratio	IN_BYTES/ (OUT + 1)
avg_packet_size	Average packet size (inbound and outbound)	(IN_BYTES + OUT_BYTES)/ (IN_PKTS + OUT_PKTS)

performance and interpretability of our models.

Table 3.8 presents a comparison of features that passed at least three selection methods for each dataset.

The results reveal significant differences in feature relevance across the datasets. UNSW-NB15 and NF-CSE-CIC-IDS 2018 showed considerable overlap in their selected features, with 10 common features including sttl, ct_state_ttl and src_dst_byte_ratio. This similarity suggests that these features are crucial for detecting network intrusions across different environmenter. In contrast, NF-UNSW-NB15-v2 exhibited a distinct set of important features, primarily focused on Time to Live (TTL) values and packet characteristics. Features such as MIN_TTL, MAX_TTL, and MIN_IP_PKT_LEN were uniquely important in this dataset. This divergence can be attributed to the flow-based nature of NF-UNSW-NB15-v2, as opposed to the

Table 3.8: Feature Selection Comparison Across Datasets

Feature	UNSW-NB15	NF-UNSW-NB15-v2	NF-CSE-CIC-IDS2018
sttl	✓ (4 methods)		✓ (4 methods)
ackdat	✓ (3 methods)		✓ (3 methods)
ct_state_ttl	✓ (4 methods)		✓ (4 methods)
state_INT	✓ (3 methods)		✓ (3 methods)
tcp_rtt	✓ (3 methods)		✓ (3 methods)
tcprtt	✓ (3 methods)		✓ (3 methods)
dttl	✓ (3 methods)		✓ (3 methods)
src_dst_byte_ratio	✓ (3 methods)		✓ (3 methods)
dmeansz	✓ (3 methods)		✓ (3 methods)
flow_bytes_per_packet	✓ (3 methods)		✓ (3 methods)
MIN_TTL		✓ (4 methods)	
MAX_TTL		✓ (4 methods)	
MIN_IP_PKT_LEN		✓ (3 methods)	
SHORTEST_FLOW_PKT		✓ (3 methods)	
NUM_PKTS_UP_TO_128_BYTES		✓ (3 methods)	
LONGEST_FLOW_PKT		✓ (3 methods)	
DST_TO_SRC_AVG_THROUGHPUT		✓ (3 methods)	
OUT_PKTS		✓ (3 methods)	

packet-based approach of the other datasets. Notably, some features demonstrated consistent importance across multiple datasets. For instance, sttl and ct_state_ttl were selected by all four methods in both UNSW-NB15 and NF-CSE-CIC-IDS 2018. This consistency underscores their potential as universal indicators for network intrusion detection.

3.2.4 Rotational Task Pool

This is one of the key stages as shown Figure 3.2 in the proposed framework, where rotation of attack categories take place. It serves as a middleware that receives pre-processed data (after data cleaning, feature extraction, feature selection, and scaling), divides this data into specific tasks for training and testing machine learning, including unseen or zero-day attacks and be evaluated for zero day detection [85]. Rotational Task Pool primary function is to systematically prepare data for both training and testing phases while maintaining the integrity of our zero-day attack simulation approach. In the training phase, the Relational Task Pool organizes the preprocessed data by deliberately excluding one attack type at a time. This systematic exclusion creates scenarios where specific attack patterns are treated as "unseen", effectively simulating zero-day attack conditions. The excluded attack type is reserved for testing, while the remaining attack patterns are used for model training. This rotation ensures that each attack type serves as a simulated zero-day attack during different experimental iterations. For the testing phase, the Relational Task Pool prepares specialized datasets that combine known attack patterns with the previously excluded attack type. This approach enables us to evaluate our models' ability to detect novel attack patterns while maintaining accurate detection of known threats. The Rotational Task Pool maintains strict separation between training and testing data to prevent data leakage and ensure the validity of our zero-day detection evaluation. To maintain experimental consistency, the Rotational Task Pool implements stratified sampling techniques, ensuring representative distribution of attack types in both training and testing sets. This organization method allows for systematic evaluation of our models' generalization capabilities while simulating real-world scenarios where networks must detect both known and unknown attack patterns.

3.2.4.1 Model Selection and Training

We selected and implemented five diverse machine learning models for our study:

1. Random Forest (RF): An ensemble learning method implemented using sci-kit-learn, with 50 trees and a maximum depth of 8. This configuration was chosen to balance model complexity and computational efficiency.

2. XGBoost: An optimized gradient boosting framework, configured with 50 estimators and a maximum depth of 8 for each tree. We utilized the scale_pos_weight parameter to handle class imbalance, setting it based on the ratio of negative to

positive instances.

3. Multilayer Perceptron (MLP): A feedforward neural network implemented in Keras, with two hidden layers (64 and 32 neurons) and ReLU activation. The model was trained for 5 epochs using binary cross-entropy as the loss function and the Adam optimizer.

4. Convolutional Neural Network (CNN): Implemented in Keras with two 1D convolutional layers (32 and 64 filters) followed by max pooling and dense layers (64 and 32 neurons). The model was trained using the Adam optimizer with binary cross-entropy loss.

5. Voting Classifier: An ensemble method combining the predictions of the RF, XGBoost, MLP, and CNN models using soft voting. This approach leverages the strengths of each individual model to produce more robust predictions[86].

For all models, we incorporated class weights computed based on the class distribution to address the imbalance in our datasets. These weights were applied during model training to ensure equal importance was given to minority classes.

Hyperparameter tuning was performed using Bayesian Optimization, implemented with the Hyperopt library [87]. We defined a search space for each model and used the Tree of Parzen Estimators (TPE) algorithm to optimize the F1-score over 100 iterations. This method was chosen for its efficiency in exploring high-dimensional hyperparameter spaces.

3.3 Performance Metrics

To thoroughly assess the effectiveness of our models in detecting zero-day attacks, we utilized several performance metrics. These included accuracy, precision, recall, F1-score, Area Under the Receiver Operating Characteristic Curve (AUC-ROC), and a novel metric we introduced called the Zero-Day Detection Rate (Z-DR). Below, we describe each metric, its significance, and its calculation formula. Accuracy measures the overall correctness of the model's predictions. It is calculated as the ratio of correctly classified instances (both true positives and true negatives) to the total number of predictions. The formula for accuracy is:

$$\text{Accuracy} = \frac{\text{Number of Correctly Classified Instances}}{\text{Total Number of Instances}} \tag{3.1}$$

While accuracy is a useful general indicator of performance, it can be misleading for imbalanced datasets a common scenario in network intrusion detection. In these cases, precision, recall, and F1-score are more informative. Precision Precision quantifies the accuracy of the positive predictions made by the model. It is defined as the proportion of correctly predicted attacks out of all instances classified as attacks. The formula for precision is:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{3.2}$$

where TP = True Positives (correctly predicted attacks) TN = True Negatives (correctly predicted benign instances) FP = False Positives (benign instances incorrectly predicted as attacks) FN = False Negatives (attacks incorrectly predicted as benign). A high precision indicates that most of the instances classified as attacks are indeed true attacks, reflecting the model's ability to minimize false alarms. Recall, also known as sensitivity or true positive rate, measures the ability of the model to correctly identify actual attacks. It is defined as the ratio of true positives to the total number of actual attack instances. The formula for recall is:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3.3}$$

A high recall value indicates that the model is effective at detecting most of the attack instances, minimizing false negatives. F1-Score The F1-score is the harmonic mean of precision and recall. It provides a balanced measure, particularly useful when dealing with imbalanced datasets where precision and recall alone might be insufficient to capture the model's overall performance. The formula for F1-score is:

$$\text{F1-score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \tag{3.4}$$

The F1-score helps to balance both false positives and false negatives, which is crucial for assessing a model that must handle both attack detection and false alarm reduction. Other metrics to be considered are

True Positive Rate (Sensitivity):

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.5}$$

False Positive Rate:

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{3.6}$$

Matthews Correlation Coefficient (MCC):

$$\text{MCC} = \frac{(\text{TP} \times \text{TN}) - (\text{FP} \times \text{FN})}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{3.7}$$

Balanced Accuracy:

$$\text{BA} = \frac{\text{TPR} + \text{TNR}}{2} \tag{3.8}$$

where

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{3.9}$$

Area Under the Receiver Operating Characteristic Curve (AUC-ROC) AUC-ROC is a performance metric that evaluates the model's ability to distinguish between different classes across various threshold settings. The Receiver Operating Characteristic (ROC) curve is a plot of the True Positive Rate (TPR) against the False Positive Rate (FPR) at different classification thresholds, while AUC (Area Under Curve) quantifies the overall ability of the model to separate classes.

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}} \tag{3.10}$$

An AUC-ROC value of 1.0 indicates a perfect classifier, while 0.5 represents a model no better than random guessing. This metric is valuable as it is independent of any specific classification threshold, making it ideal for comparing models under different operational settings. Zero-Day Detection Rate (Z-DR) To evaluate the models' ability to detect previously unseen attack types, we introduced a novel metric called the Zero-Day Detection Rate (Z-DR). The Z-DR measures how well the model generalizes to new, previously unseen attacks, which is the main focus of our research. The formula for Z-DR is:

$$\text{Z-DR} = \frac{\text{True Positives of Zero-Day Attacks}}{\text{Total Zero-Day Attack Instances}} \tag{3.11}$$

Overall Zero-Day Detection Rate:

$$\text{Z-DR}_{\text{avg}} = \frac{1}{n} \sum_{k=1}^{n} \text{Z-DR}_k \tag{3.12}$$

where n is the total number of attack types.

This metric provides critical insights into the robustness of the model and its capability to identify novel threats that were not part of the training data. All models (Random Forest, XGBoost, MLP, CNN, and Voting Classifier) were evaluated using a consistent set of performance metrics: Accuracy, Precision, Recall, F1-score, AUC-ROC, and the novel Zero-Day Detection Rate (Z-DR). Due to the class imbalance in our datasets, we used weighted averages for Precision, Recall, and F1-score, providing a more balanced assessment of model performance across all classes [88]. The Z-DR metric was particularly crucial in assessing each model's capability to detect previously unseen attack types, directly addressing our research focus on zero-day attack detection.

3.3.1 Attack Types and Descriptions

Table 3.9 shows the attack types and its descriptions in the datasets used for this research.

Table 3.9: Attack Types and Descriptions

Attack Type	**Description**
Exploits	Attacks that take advantage of vulnerabilities in software or systems to gain unauthorized access.
Generic	Broadly defined attacks that may not fit into specific categories, often involving various tactics.
Fuzzers	Techniques used to discover vulnerabilities by inputting random data to provoke unexpected behavior.
Backdoor	Hidden methods of bypassing normal authentication or securing remote access to a computer.
DoS	Denial of Service attacks that overwhelm a target system, making it unavailable to legitimate users.
Reconnaissance	Information-gathering attacks that gather data about the target to plan further attacks.
Shellcode	Small pieces of code used as the payload in an exploit, often designed to execute a command on the target system.
Worms	Self-replicating malware that spreads across networks without user intervention.
Analysis	Attacks focused on analyzing and exploiting system weaknesses, often as part of a broader strategy.
Infiltration	Gaining unauthorized access to a system or network to steal data or disrupt services.
Bot	A form of malware that allows a remote attacker to control a compromised computer (bot) to perform various tasks.
Brute Force - Web	Attempts to gain unauthorized access by systematically guessing login credentials on web applications.
Brute Force - XSS	Exploits vulnerabilities in web applications by injecting malicious scripts, often through user input fields.
DDOS attack-HOIC	High-Output Internet Control Message Protocol (ICMP) attacks that flood a target with traffic to overwhelm it.
SQL Injection	An attack that exploits vulnerabilities in a web application's database layer by injecting malicious SQL queries.
DoS attacks-Hulk	A specific type of DoS attack that targets web servers by sending numerous requests to exhaust resources.
DoS attacks-SlowHTTPTest	A variant of DoS that sends slow requests to the server, aiming to exhaust its resources over time.
DDoS attacks-LOIC-HTTP	Distributed Denial of Service (DDoS) attacks using Low Orbit Ion Cannon (LOIC) to flood a target with HTTP requests.
DDOS attack-LOIC-UDP	Similar to the above but utilizes UDP packets to overwhelm the target, making it unavailable.
FTP-BruteForce	Attacks that attempt to guess FTP credentials using automated tools to gain unauthorized access.
SSH-Bruteforce	Attacks that try to gain unauthorized access to a server via Secure Shell (SSH) by guessing passwords.

Chapter 4

Results

4.1 Overview

This chapter presents the outcomes of our extensive experiments on zero-day attack detection using machine learning techniques. Our study encompassed three distinct datasets: UNSW-NB15, NF-UNSW-NB15-v2, and NF-CICIDS 2018. For each dataset, we implemented and evaluated five different models: Random Forest, Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), XGBoost and an ensemble combining these models.

4.2 Experimental Setup

Each dataset, such as UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018, was exactly divided into 60% for training, 20% for validation, and 20% for testing. We performed the split by stratification according to attack types using the Scikit-learn library's function, (train_test_split), to ensure proper class distribution across the splits. A stratified approach helps to include a proportional representation of different attack classes in each subset, making this particularly useful for model training and balanced evaluation. This experiment was run on preprocessed datasets with some samples deleted.

4.3 Zero-Day Simulation Using Rotation Approach

To simulate zero-day attacks, we employed a systematic rotation method across all three datasets. This approach treats each known attack type as a potential zero-day attack in turn. The methodology was designed to evaluate the model's ability to detect unseen attack patterns, mimicking real-world scenarios where new attack types continually emerge.

For a given NIDS dataset, the Rotational Task Pool framework for attack classes can be defined as follows:

Training Set:

$$D_{\text{train}} = \{(x, y) \mid x \in X_{\text{train}}, y \in Y_{\text{train}} = \{b, a_1, a_2, \ldots, a_n\} \setminus \{a_u\}\} \tag{1}$$

Testing Set:

$$D_{\text{test}} = \{(x, y) \mid x \in X_{\text{test}}, y \in Y_{\text{test}} = \{b, a_1, a_2, \ldots, a_n \ + \{a_u\}\} \tag{2}$$

where $X_{\text{train}} \subseteq X$ and $X_{\text{test}} \subseteq X$.

Let x represent a data instance (flow) selected from the training set X_{train} and the testing set X_{test}. X represents all data instances, y corresponds to the associated labels, Ytrain represents the set of class labels observed in the training phase, and Ytest represents the set of class labels used in the testing phase. The training set Y_{train} comprises benign traffic b and n attack classes $a_1, \ldots, a_n$, excluding the unseen attack class a_u, X represents all data instances, y corresponds to the associated labels. In contrast, the test dataset D_{test} includes instances from all classes without excluding any attack class. By omitting an attack class a_u from the training phase, we simulate a zero-day attack scenario, where the ML model has not been trained on the specific attack a_u. The training set D_{train} and the testing set D_{test} remain mutually exclusive during the entire experiment.

Algorithm: Training and Evaluation of Zero-Day Attack Detection Models

---

1: $Input$: $Dataset\ D$

2: Split D into $Training\ set$ (60%), $Validation\ set$ (20%), $Testing\ set$ (20%) using stratified sampling

3: $Initialize$: $Training\ set = D_{train}$, $Validation\ set = D_{val}$, $Testing\ set = D_{test}$,

4: $Return\ D_{train}$, D_{val}, D_{test}

5: $End\ Procedure$

6: $Start\ Procedure$: $Exclude\ Zero\ Day\ (D, attack\_type)$

7: $Remove\ all\ samples\ of\ attack\_type\ from\ D_{train}\ and\ D_{val}$.

8: $Return\ D_{train}$ and D_{val}

9: $End\ Procedure$

10: $Initialize\ models$: $RF, XGBoost, MLP, CNN and\ Voting\ classifiers$

11: $Train\ models\ on\ D_{train}$

12: $Validate\ models\ using\ D_{val}$

13: $Test\ models\ on\ initialized\ D_{test}$, $which\ includes\ all\ attack\ types$

14: $Evaluate\ model$

15: $Rotation$: $Repeat\ steps$ 6-13 $for\ each\ attack\ type\ in\ the\ dataset$

16: $End\ Procedure$

This process was applied uniformly across all datasets and models, ensuring a comprehensive evaluation of each model's ability to detect zero-day threats. By rotating the exclusion of attack types, we ensured that all attacks were treated as zero-day scenarios in separate iterations, providing valuable insights into each model's generalization capabilities.

4.3.1 UNSW-NB15 Dataset Results

The UNSW-NB15 dataset results offer comprehensive insights into the performance of five machine learning models in detecting zero-day attacks: Random Forest (RF), XGBoost, Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), and a Voting classifier. We evaluated these models using two primary metrics: F1-score and Zero-Day Detection Rate (Z-DR), across nine different attack types.

The Random Forest model as shown in Table 4.1 demonstrated exceptional performance, with F1-scores consistently above 0.984 and near-perfect Z-DR scores. It achieved perfect Z-DR scores for seven out of nine attack types, indicating excellent generalization to unseen attacks.

Table 4.1: Performance evaluation of RF on UNSW-NB15

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9852	99.99
Reconnaissance	0.9851	100.00
DoS	0.9852	99.97
Generic	0.9847	100.00
Shellcode	0.9851	100.00
Fuzzers	0.9851	100.00
Worms	0.9852	100.00
Backdoors	0.9851	100.00
Analysis	0.9851	100.00

XGBoost performance in Table 4.2 showed similarly impressive results, with consistent F1-scores of 0.9851 and above and very high Z-DR scores. It achieved perfect Z-DR scores for five attack types, demonstrating strong zero-day attack detection capabilities.

Table 4.2: Performance evaluation of XGBoost on UNSW-NB15

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9852	99.94
Reconnaissance	0.9852	100.00
DoS	0.9852	99.97
Generic	0.9851	99.97
Shellcode	0.9852	100.00
Fuzzers	0.9852	99.50
Worms	0.9852	100.00
Backdoors	0.9852	100.00
Analysis	0.9852	100.00

The MLP performance in Table 4.3 achieved slightly lower but still impressive F1-scores ranging from 0.9828 to 0.9850. Notably, it attained perfect Z-DR scores for all attack types except DoS, where it still achieved a high score of 0.9994.

Table 4.3: Performance evaluation of MLP on UNSW-NB15

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9850	99.43
Reconnaissance	0.9836	100.00
DoS	0.9844	99.97
Generic	0.9828	100.00
Shellcode	0.9838	100.00
Fuzzers	0.9842	100.00
Worms	0.9842	100.00
Backdoors	0.9830	100.00
Analysis	0.9835	100.00

The CNN performance in Table 4.4 showed high F1-scores comparable to other models, ranging from 0.9827 to 0.9855. However, its Z-DR scores were significantly lower, ranging from 0.1935 to 0.2181, indicating poor performance in detecting zero-day attacks despite high overall accuracy.

Table 4.4: Performance evaluation of CNN on UNSW-NB15

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9854	19.74
Reconnaissance	0.9954	19.57
DoS	0.9953	19.60
Generic	0.9827	20.09
Shellcode	0.9856	20.52
Fuzzers	0.9855	19.69
Worms	0.9854	20.69
Backdoors	0.9852	21.81
Analysis	0.9854	19.35

The Voting classifier performance in Table 4.5 achieved the highest F1-scores for some attack types (e.g., 0.9867 for Exploits and DoS). However, like CNN, it showed very low Z-DR scores, ranging from 0.1935 to 0.2181, indicating poor zero-day attack detection. Comparing across models, Random Forest, XGBoost, and MLP consistently outperforms CNN and Voting classifier in Z-DR scores, despite all models showing similar F1-scores. This highlights the importance of using specialized metrics like Z-DR in evaluating NIDS performance for novel threats.

Table 4.5: Performance evaluation of Voting classifier on UNSW-NB15

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9867	19.76
Reconnaissance	0.9867	19.58
DoS	0.9867	19.67
Generic	0.9850	20.09
Shellcode	0.9867	20.52
Fuzzers	0.9867	19.69
Worms	0.9867	20.69
Backdoors	0.9867	21.81
Analysis	0.9867	19.35

4.3.2 NF-UNSW-NB15-v2 Dataset Results

The NF-UNSW-NB15-v2 dataset, a NetFlow version of the original UNSW-NB15, provided valuable insights into the performance of our models in detecting zero-day attacks. We evaluated five models: Random Forest (RF), XGBoost, Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), and Voting classifier. The results are presented in Tables 4.6 to 4.10, showing the performance metrics for each model across different attack types simulated as zero-day attacks.

The Random Forest model as shown in Table 4.6 demonstrated exceptional performance across all attack types, maintaining high F1-scores (> 0.9946), and Zero-Day Detection Rates (Z-DR) > 99.68. Notably, it achieved perfect Z-DR scores for six out of nine attack types, indicating its robust generalization capability for unseen attacks.

Table 4.6: Performance evaluation of RF on NF-UNSW-NB15-v2

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9947	99.97
Generic	0.9947	100.00
Fuzzers	0.9948	99.69
Backdoor	0.9947	100.00
DoS	0.9948	99.91
Reconnaissance	0.9948	100.00
Shellcode	0.9947	100.00
Worms	0.9948	100.00
Analysis	0.9948	100.00

XGBoost showed similar performance to Random Forest, with slightly lower but still impressive metrics in Table 4.7. It achieved perfect Z-DR scores for five attack types and maintained high F1-scores (> 0.994) across all categories.

Table 4.7: Performance evaluation of XGBoost on NF-UNSW-NB15-v2

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9940	99.92
Generic	0.9940	100.00
Fuzzers	0.9941	98.43
Backdoor	0.9941	100.00
DoS	0.9950	99.91
Reconnaissance	0.9940	100.00
Shellcode	0.9948	100.00
Worms	0.9948	100.00
Analysis	0.9940	100.00

In contrast, the MLP model showed inconsistent results in Table 4.8. While it achieved high Z-DR scores across most attack types, F1-scores was considerably lower, especially for attacks like Exploits, Fuzzers, Shellcode, Worms and Analysis. This indicated that although the MLP model was effective in detecting new attacks, it struggled to maintain good F1-score in overall classification.

Table 4.8: Performance evaluation of MLP on NF-UNSW-NB15-v2

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.0030	100.00
Generic	0.0030	83.52
Fuzzers	0.0135	99.69
Backdoor	0.0030	100.00
DoS	0.0031	100.00
Reconnaissance	0.0034	100.00
Shellcode	0.0030	100.00
Worms	0.0030	100.00
Analysis	0.0030	100.00

The CNN model demonstrated high F1-scores (> 0.977) across all attack types, as shown in Table 4.9. However, its Z-DR scores were notably lower than those of other models, ranging from 0.0868 to 0.2125. This suggests that, despite CNN's strong performance in the overall classification, it struggled to generalize effectively to unseen attack patterns.

Table 4.9: Performance evaluation of CNN on NF-UNSW-NB15-v2

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9846	04.04
Generic	0.9829	20.28
Fuzzers	0.9885	07.88
Backdoor	0.9871	18.26
DoS	0.9833	19.64
Reconnaissance	0.9772	09.74
Shellcode	0.9797	18.50
Worms	0.9876	03.05
Analysis	0.9837	17.09

The Voting classifier, combining the strengths of multiple models, demonstrated high F1-scores (> 0.993) across all attack types in Table 4.10. However, its Z-DR scores (ranging from 0.1646 to 0.2148) were lower than individual models like Random Forest and XGBoost, suggesting that the ensemble approach did not necessarily improve zero-day attack detection.

Table 4.10: Performance evaluation of Voting classifier on NF-UNSW-NB15-v2

Attack Type	**F1-Score**	**Z-DR**
Exploits	0.9938	19.76
Generic	0.9937	20.30
Fuzzers	0.9949	18.50
Backdoor	0.9937	21.48
DoS	0.9948	19.80
Reconnaissance	0.9937	20.22
Shellcode	0.9945	18.50
Worms	0.9945	16.46
Analysis	0.9937	19.70

4.3.3 NF-CSE-CIC-IDS 2018 Dataset Results

As shown in Table 4.11, Random Forest demonstrated consistent performance across various attack types. Its F1-scores were generally high, ranging from 0.7166 to 0.9917. Notably, RF showed excellent Z-DR for several attack types, achieving 100% detection for attacks such as DoS attacks-Hulk, SSH-Bruteforce and others. However, it struggled with some attacks, such as Bot and Infiltration, where the Z-DR was 0%.

Table 4.11: Performance evaluation of RF on NF-CSE-IDS 2018

Attack Type	**F1-Score**	**Z-DR**
Infiltration	0.9917	0.02
Bot	0.9678	0.00
Brute Force - Web	0.9679	98.65
Brute Force - XSS	0.9682	99.70
DDOS attack-HOIC	0.9718	65.85
SQL Injection	0.9740	12.50
DoS attacks-Hulk	0.9628	100.00
DoS attacks-SlowHTTPTest	0.9642	100.00
DDoS attacks-LOIC-HTTP	0.7166	0.17
DDOS attack-LOIC-UDP	0.9728	100.00
DoS attacks-GoldenEye	0.9683	100.00
DoS attacks-Slowloris	0.9692	100.00
FTP-BruteForce	0.9205	100.00
SSH-Bruteforce	0.9665	100.00

XGBoost, as shown in Table 4.12, demonstrated strong performance with F1-scores ranging from 0.7342 to 0.9968. Its Z-DR was notably high, achieving 100% for 6 out of 14 attack types and over 87% for three additional types.

Table 4.12: Performance evaluation of XGBoost on NF-CSE-IDS 2018

Attack Type	**F1-Score**	**Z-DR**
Infiltration	0.9924	0.03
Bot	0.9446	86.39
Brute Force - Web	0.9412	99.81
Brute Force - XSS	0.9429	100.00
DDOS attack-HOIC	0.9468	87.80
SQL Injection	0.9448	87.50
DoS attacks-Hulk	0.9187	99.95
DoS attacks-SlowHTTPTest	0.9219	100.00
DDoS attacks-LOIC-HTTP	0.7342	0.17
DDOS attack-LOIC-UDP	0.9451	100.00
DoS attacks-GoldenEye	0.9390	100.00
DoS attacks-Slowloris	0.9410	100.00
FTP-BruteForce	0.8833	100.00
SSH-Bruteforce	0.9159	0.05

In Table 4.13 MLP's performance exhibited high F1-scores (0.9677 to 0.9801) for most attack types, with a significant drop in performance for DDoS attacks-LOIC-HTTP with F1-score (0.3175). MLP's Z-DR was 100% for DoS attacks-Hulk, DoS attacks-SlowHTTPTest, DDoS attacks-LOIC-HTTP, DoS attacks-Slowloris FTP-BruteForce, DoS attacks-GoldenEye and SSH-Bruteforce, but it failed to detect several other attack types as zero-day threats, with 0% Z-DR for 6 out of 14 attack types.

Table 4.13: Performance evaluation of MLP on NF-CSE-IDS 2018

Attack Type	**F1-Score**	**Z-DR**
Infiltration	0.9813	0.00
Bot	0.9799	0.00
Brute Force - Web	0.9777	0.00
Brute Force - XSS	0.9765	0.00
DDOS attack-HOIC	0.9731	0.00
SQL Injection	0.9720	0.00
DoS attacks-Hulk	0.9775	100.00
DoS attacks-SlowHTTPTest	0.9762	100.00
DDoS attacks-LOIC-HTTP	0.3175	100.00
DDOS attack-LOIC-UDP	0.9725	32.79
DoS attacks-GoldenEye	0.9793	100.00
DoS attacks-Slowloris	0.9800	100.00
FTP-BruteForce	0.9776	100.00
SSH-Bruteforce	0.9791	100.00

In Table 4.14, CNN's performance was inconsistent across attack types, while it achieved high F1-scores (0.9609 to 0.9656) for some attacks, it performed poorly on others, with F1-scores as low as 0.0339 for DDoS attacks-LOIC-HTTP. Its Z-DR values were generally low, ranging from 0% to 20.04%, indicating limited ability to detect zero-day attacks.

Table 4.14: Performance evaluation of CNN on NF-CSE-IDS 2018

Attack Type	**F1-Score**	**Z-DR**
Infiltration	0.9091	0.00
Bot	0.82182	0.00
Brute Force - Web	0.0346	19.86
Brute Force - XSS	0.8218	0.00
DDOS attack-HOIC	0.9860	0.00
SQL Injection	0.0343	22.22
DoS attacks-Hulk	0.9852	20.03
DoS attacks-SlowHTTPTest	0.8218	0.00
DDoS attacks-LOIC-HTTP	0.0442	19.99
DDOS attack-LOIC-UDP	0.0343	18.30
DoS attacks-GoldenEye	0.9048	0.00
DoS attacks-Slowloris	0.0343	20.02
FTP-BruteForce	0.8218	0.00
SSH-Bruteforce	0.0343	19.96

In Table 4.15, Voting classifier's performance demonstrated high F1-scores (0.9919 to 0.9930) across all attack types. Its Z-DR performance was moderate, ranging from 0.00 to 20.04, suggesting a balanced approach to zero-day attack detection.

Table 4.15: Performance evaluation of Voting classsifier on NF-CSE-IDS 2018

Attack Type	**F1-Score**	**Z-DR**
Infiltration	0.9919	0.00
Bot	0.9907	0.00
Brute Force - Web	0.9926	18.98
Brute Force - XSS	0.9926	18.17
DDOS attack-HOIC	0.9926	10.43
SQL Injection	0.9930	0.00
DoS attacks-Hulk	0.9926	20.03
DoS attacks-SlowHTTPTest	0.9929	20.04
DDoS attacks-LOIC-HTTP	0.9430	0.44
DDOS attack-LOIC-UDP	0.9926	18.30
DoS attacks-GoldenEye	0.9926	20.04
DoS attacks-Slowloris	0.9926	20.02
FTP-BruteForce	0.9926	19.95
SSH-Bruteforce	0.9926	19.96

4.4 Analysis

4.4.1 Introduction

This section uses Kolmogorov-Smirnov (K-S) test to investigate how selected network features are distributed across different types of attacks, especially for spotting zero-day threats. The goal of the Kolmogorov-Smirnov (K-S) test is to find any major differences in how features behave between known attacks and possible zero-day attacks. However, understanding these differences can help improve NIDS by highlighting which network characteristics change most across attack types.

4.4.2 Overview of K-S Test Results

The Kolmogorov-Smirnov test is a non-parametric method that compares the cumulative distributions of two groups [89]. K-S was used to look at how certain features differ between known attacks (our training set) and each specific attack type treated as a potential zero-day (test set). With this approach, we can pick out features that behave differently across specific attack types, giving clues on which features might help us detect new, unseen attacks. The K-S test gives us two important metrics: the K-S statistic and the p-value. The K-S statistic measures the largest distance between the cumulative distributions of the two groups. The p-value shows the likelihood of observing these results if there were no actual difference between the two distributions. In our case, a low p-value (typically < 0.05) points to a significant difference in feature distributions between known and possible zero-day attacks. The analysis focuses on features chosen through a feature engineering and selection process, tested across multiple attack types in the dataset. By looking closely at the K-S results for each feature within each attack type, we aim to see which features stand out most for different kinds of attacks. This approach could help boost the system's ability to catch zero- day threats.

4.4.3 Detailed Analysis by Attack Category

This section examines the Kolmogorov-Smirnov (K-S) test results for each attack category across the UNSW-NB15, NF-UNSW-NB15v2, and NF-CSE-CIC-IDS 2018 datasets. We analyze how feature distributions differ between benign and attack traffic, and discuss the implications for creating machine learning models capable of detecting zero-day attacks.

1. UNSW-NB15 Dataset

   The Kolmogorov-Smirnov (K-S) test analysis on the UNSW-NB15 dataset highlighted big distributional differences across attack categories for six important network traffic features Figure 4.1. K-S statistics for features like sttl and ct_state_ttl were high, between 0.84 and 0.96, showing strong distinctions between normal and malicious traffic patterns [17]. These two features consistently showed the most significant discrimination, especially for Generic attacks, which reached a K-S statistic of 0.96. Timing-related features like ackdat and tcprtt had moderate K-S statistics, ranging from 0.35 to 0.84, suggesting their effectiveness differs by attack type. This variability lines up with earlier studies showing that timing characteristics vary across cyber attack types. Features like dttl and dmeansz also had decent K-S values, from 0.37 to 0.86, with Generic and Worm attacks displaying the strongest differences. In terms of machine learning, these distributional differences back the use of tree-based models, like Random Forests, which work well with distinct feature distributions [90]. However, the consistent distribution patterns across attack types, shown by clustered K-S values, could make zero-day attack detection more challenging. While high K-S statistics (> 0.8) for key features help in identifying malicious activity, the overlap in distribution patterns could limit the model's ability to identify new attack vectors.

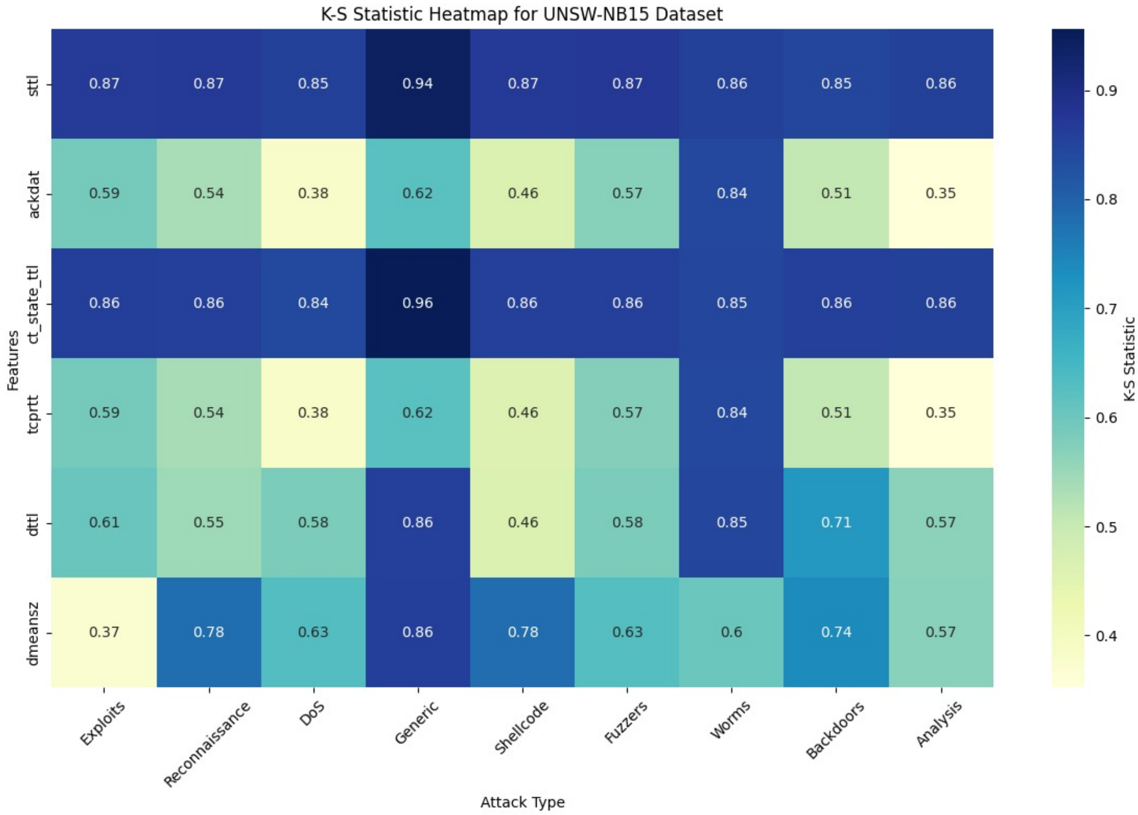

Figure 4.1: K-S heatmap for UNSW-NB15

All six features showed statistically significant differences ($p < 0.05$) between normal and attack traffic, though the K-S statistic itself offers more practical value than just p-values for this analysis. By quantifying distributional differences, the K-S statistic provides clearer insights for feature selection and model building than binary significance alone.

2. NF-UNSW-NB15v2 Dataset

The NF-UNSW-NB15v2 dataset as shown in Figure 4.2, revealed significant differences in eight features, including MIN_TTL, MAX_TTL, MIN_IP_PKT_LEN, and flow-related metrics. Most attack categories showed p-values of 0 for all eight features, indicating strong distributional differences. Analysis of the NF-UNSW-NB15v2 dataset using Kolmogorov-Smirnov (K-S) statistics revealed significant discriminatory power in key network features. TTL-related features

(MIN_TTL, MAX_TTL) demonstrated consistently high K-S values (0.94-0.97) across all attack types, while MIN_IP_PKT_LEN showed strong discrimination ability (0.94-0.96). Flow-based metrics exhibited varying effectiveness, with DST_TO_SRC_AVG_THROUGHPUT showing lower discrimination for Exploits (0.23) but higher values for Analysis attacks (0.83). These findings suggest specific feature combinations could enhance detection accuracy for different attack types.

Interestingly, Shellcode and Worms attacks displayed unique patterns. Shellcode attacks had varying p-values across different feature sets, while Worms showed significant differences in all features but with non-zero p-values. These nuanced differences in feature distributions could be valuable for creating models that not only detect attacks but also classify them more accurately.

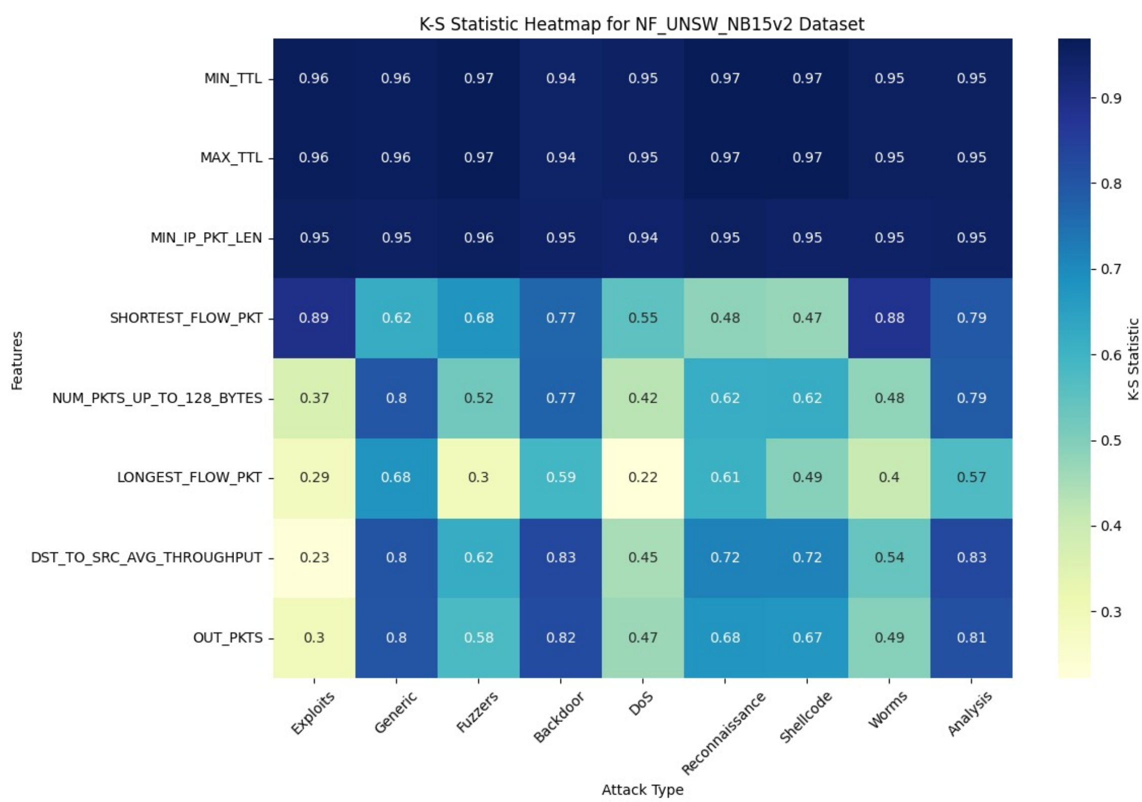

Figure 4.2: K-S heatmap for UNSW-NB15-V2

For zero-day attack detection, these varied distributions suggest that a model trained on this dataset might be more adept at identifying novel threats. The diverse feature distributions across attack types provide a richer set of patterns for the model to learn, potentially improving its ability to recognize unusual traffic that does not fit known benign or attack profiles.

3. NF-CSE-CIC-IDS 2018 Dataset

   The NF-CSE-CIC-IDS 2018 dataset analysis as shown in Figure 4.3, using Kolmogorov-Smirnov (K-S) statistics, identified seven key features with significant distributional differences across multiple attack categories. Among these, the feature L4_DST_PORT exhibited outstanding discriminatory power, with K-S values of 0.97 to 0.98 for DoS attacks such as Hulk and GoldenEye, indicating its strong potential to distinguish these attacks from benign traffic. In contrast, this feature showed a notably lower discriminatory power for Infiltration attacks (K-S value of 0.24), which implies that additional features may be necessary to effectively identify this particular attack type. FLOW_DURATION_MILLISEC ONDS provided consistent discrimination across most attack categories, with moderate-to-high K-S values ranging from 0.56 to 0.77. The highest K-S value observed for this feature was 0.77, specifically for Bot attacks, underscoring its utility in identifying certain types of attacks. In contrast, IN_BYTES displayed more variability in its effectiveness, showing a K-S value of 1.0 for DDoS attacks-LOIC-HTTP (indicating complete separation from benign traffic) but a lower value of 0.25 for DoS attacks- GoldenEye. This variation in IN_BYTES highlights the necessity of incorporating multiple features to achieve comprehensive attack detection across different categories. Additionally, TCP_FLAGS showed high discriminatory effectiveness for some attacks, such as Bot (0.80) and DDoS attack-HOIC (0.77), while performing less effectively for DDoS attack-LOIC-UDP (0.38). Traffic volume metrics like OUT_PKTS and OUT_BYTES offered stable, moderate discriminatory power, with K-S values ranging from 0.60 to

0.72 across most attack categories, serving as reliable indicators for detecting traffic anomalies.

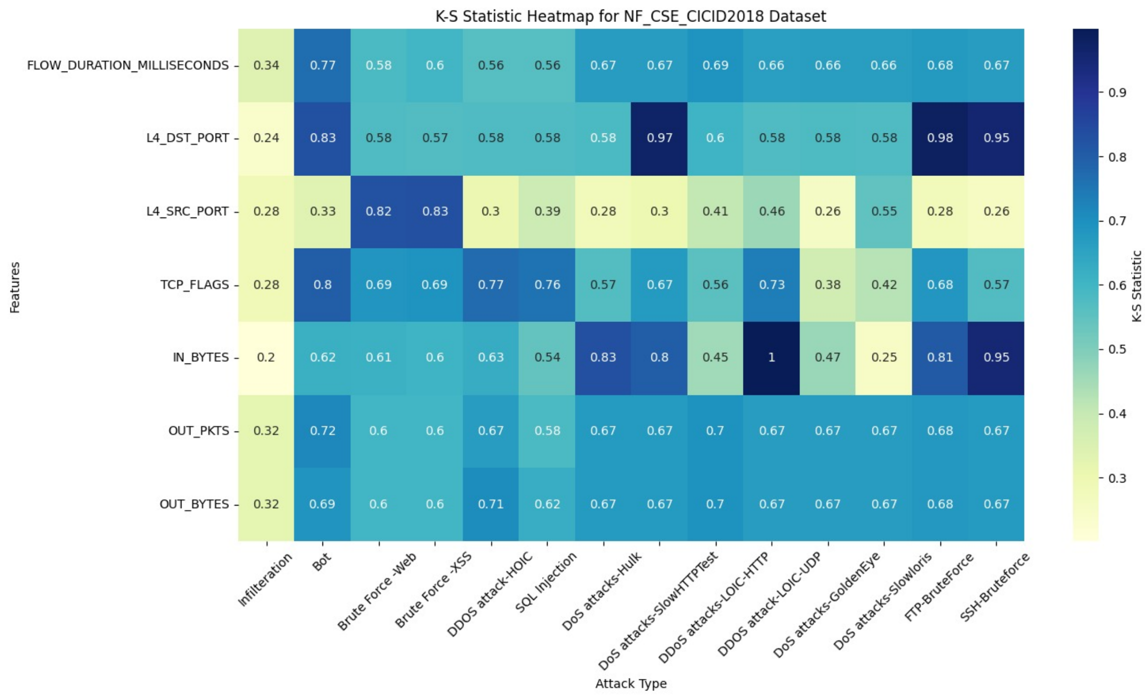

Figure 4.3: K-S heatmap for NF-CSE-CIC-IDS 2018

4.4.4 Implications for Model Creation

The clear differences in feature distributions across all datasets suggest that machine learning models especially Random Forests can effectively distinguish between benign and malicious traffic. However, these consistent patterns also pose challenges for spotting zero-day attacks. On the one hand, these strong distributional differences provide a solid base for building reliable intrusion detection models. On the other, they highlight a key issue: the need for more sophisticated approaches to handle unknown, novel threats. Balancing a model's capacity to identify known attack patterns while staying flexible enough to catch zero-day threats is essential for creating robust, adaptable cybersecurity systems.

4.5 Comparative Analysis

This section shows the comparative analysis between the baseline studies [5] and our proposed methodology (P). We focused on six metrics: Zero-Day Detection Rate (Z-DR), Accuracy (Acc), F1-Score (F1-Sc), False Alarm Rate (FAR), Detection Rate (DR), and Area Under Curve (AUC).

UNSW-NB 15 Dataset Results

Focusing on the results shown in Table 4.16, we observed some significant differences between our approach and baseline.

- Zero-Day Detection Rate (Z-DR): The Proposed methodology (P) shows a notable improvement in Z-DR across all attack types. For most attacks (Exploits, Reconnaissance, Generic, Shellcode, Worms, Backdoor, Analysis), it achieves 100%, meaning it effectively identifies zero-day threats. The baseline studies [5], however, performed inconsistently, with Z-DR ranging from as low as 20.10% for "Fuzzers" to a high of 99.04% for "Backdoor." Notably, "Fuzzers" sees the most significant increase under P, suggesting that P substantially improves zero-day detection capability.

- Accuracy (Acc): The baseline studies [5] achieved high accuracy, typically around 98-99% across attack types, but our approach (P) slightly underperformed with accuracies ranging from 98.43% to 98.49%. Although this difference is small, it consistently trends lower, indicating that while P improves zero-day detection, it might slightly compromise accuracy across known attacks. P's accuracy remains stable across attack types, suggesting it is well-calibrated for various attack scenarios.

- F1-Score (F1-Sc): F1-Score, reflecting a balance of precision and recall, improves significantly with P, especially for attack types with lower scores in [5] like "Fuzzers" (74% in [5] to 98.52% in P). This improvement is consistent across attack types, showing that P maintains high precision and recall, which is critical for balanced attack detection, even with previously challenging attacks.

- False Alarm Rate (FAR): FAR values reveal an interesting trade-off: the [5] model maintains lower FARs (between 0.15% and 0.60%), while P consistently has a slightly higher FAR of about 1.74% to 1.80% across all attacks. This slight increase in false alarms with P may be an effect of its higher sensitivity, especially in zero-day attack detection, which often requires more aggressive detection thresholds.

- Detection Rate (DR): DR measures how effectively the models detect the attacks in general. Similar to Z-DR, DR scores are substantially higher in P, often close to 100% (99.99%). The [5] model shows more variance in DR, with lower values for attacks like "DoS" (87.70%) and "Fuzzers" (59.16%), whereas P consistently improves detection to near-perfect levels. This implies that P provides a more dependable defense across various attack types.

- Area Under the Curve (AUC): AUC scores are indicative of the model's discrimination capability between classes. P significantly improved over [5], maintaining high values around 99.10% to 99.13% across attacks. [5]'s AUC scores, though good, are generally lower, suggesting that P is more robust in distinguishing between attack and non-attack classes across the dataset.

- Summary: Z-DR and DR in P performed better than [5], especially for attacks that [5] struggled with, such as "Fuzzers" and "DoS." This improvement demonstrated P's enhanced zero-day detection capabilities, an essential quality for handling previously unseen threats. Accuracy vs. FAR in P sacrifices a small amount of accuracy and increases FAR slightly to achieve better Z-DR and DR. This trade-off could be justified depending on the use case; for instance, in high-security environments, higher false alarms might be acceptable if it leads to better zero-day detection. F1-Score and AUC: P demonstrates significant gains in F1-Score and AUC, showing it has a strong balance of precision and recall and excellent classification capability. These improvements indicate that P is a more reliable model for handling a variety of cyber attacks with consistent precision.

Table 4.16: Comparison of Model Performance Metrics

Attack Type	**Z-DR (%)**		**Acc (%)**		**F1-Sc (%)**		**FAR (%)**		**DR (%)**		**AUC (%)**	
	[5]	**P**	**[5]**	**P**	**[5]**	**P**	**[5]**	**P**	**[5]**	**P**	**[5]**	**P**
Exploits	90.31	**99.99**	98.73	**98.49**	0.92	**98.52**	0.47	**1.73**	89.09	**99.99**	94.00	**99.13**
Fuzzers	20.10	**100.00**	96.94	**98.48**	0.74	**98.52**	0.15	**1.74**	59.16	**99.99**	80.00	**99.13**
Generic	95.90	**100.00**	98.93	**98.43**	0.93	**98.47**	0.36	**1.80**	90.09	**99.99**	95.00	**99.10**
Reconnai	91.82	**100.00**	98.93	**98.48**	0.91	**98.51**	0.48	**1.74**	90.17	**99.99**	95.00	**99.13**
DoS	92.80	**99.97**	99.00	**98.48**	0.91	**98.52**	0.35	**1.74**	87.70	**99.99**	94.00	**99.13**
Analysis	84.35	**100.00**	99.06	**98.48**	0.91	**98.51**	0.53	**1.74**	91.36	**99.99**	95.00	**99.13**
Backdoor	99.04	**100.00**	99.03	**98.48**	0.91	**98.52**	0.60	**1.74**	92.10	**99.99**	96.00	**99.13**
Shellcode	97.15	**100.00**	99.08	**98.48**	0.91	**98.51**	0.48	**1.74**	90.67	**99.99**	95.00	**99.13**
Worms	98.25	**100.00**	99.06	**98.48**	0.90	**98.51**	0.51	**1.74**	90.60	**99.99**	95.00	**99.13**

Overall, P outperforms [5] in critical metrics for zero-day detection, attack detection consistency, and classification reliability across various attack types, with a minor compromise in FAR and accuracy. This makes P a compelling choice for environments where zero-day detection and F1-Score are prioritized.

UNSW-NB 15 Dataset Results

Table 4.17 detailed the comparative analysis of the baseline studies [5] and our proposed methodology :

- Zero-Day Detection Rate (Z-DR): P consistently outperforms [5] across all attack types, with significant differences in the Z-DR metric. The greatest improvements are observed in Fuzzers and Reconnaissance attack types, where P achieves nearly perfect scores of 99.50% and 100%, respectively, compared to 14.77% and 89.08% in [5]. This result suggests that P has a strong advantage in handling previously unseen attacks, likely due to its ensemble nature and capability to generalize better on new data.

- Accuracy (Acc): Accuracy for both models is relatively high across the board. However, despite the slight differences, P has a marginally higher accuracy in most cases. For attack types like Exploits and Generic, [5] achieves competitive accuracy (99.07% and 97.64%), but P still slightly outperforms with consistency (98.48%).

- F1-Score (F1-Sc): P demonstrates a higher F1-Score across all attack types, indicating a better balance between precision and recall for each classification task. The F1-score improvements are notable in Fuzzers and DoS, with P achieving 98.52% against [5]'s 73% and 93%, respectively, showing that P is more effective in handling complex or imbalanced classes.

- False Alarm Rate (FAR): P's FAR values are slightly higher, ranging around 1.73% to 1.74%, whereas [5] maintains a lower FAR (ranging from 0.06% to 0.38%). While [5]'s lower FAR indicates fewer false positives, the overall high DR and Z-DR of P suggest it prioritizes recall over precision, leading to better zero-day detection at the cost of more false alarms.

- Detection Rate (DR): P has a considerably higher detection rate across all attacks, with nearly perfect detection in most cases (close to 99.99%), while [5] ranges between 57.58% to 92.58%, this makes P particularly effective in identifying attacks like Fuzzers and DoS, where [5] struggles. The near-perfect DR in P indicates its robustness in classifying attacks, making it a more reliable model in high-stakes security environments.

- Area Under Curve (AUC): AUC scores for P are consistently higher, around 99.76% to 99.79% for most attack types, while [5]'s AUC is relatively lower (around 79% to 96%). P's higher AUC values reflect its superior performance across a range of thresholds, providing a well-balanced model with both high sensitivity and specificity.

- Summary: P shows clear superiority in most metrics, especially in Z-DR, DR, F1-Score, and AUC, indicating that it's better suited for detecting both known and zero-day attacks. The main trade-off is a slightly higher FAR compared to [5], meaning P may produce more false alarms. However, the trade-off seems justified by P's comprehensive advantages in detection and classification accuracy, making it highly effective for security applications requiring reliable attack detection across various attack types.

Table 4.17: Comparison of Model Performance Metrics

Attack Type	**Z-DR (%)**		**Acc (%)**		**F1-Sc (%)**		**FAR (%)**		**DR (%)**		**AUC (%)**	
	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**
Exploits	94.43	**99.94**	99.07	**98.48**	0.94	**98.52**	0.33	**1.73**	91.95	**99.99**	0.96	**99.77**
Fuzzers	14.77	**99.50**	96.92	**98.49**	0.73	**98.52**	0.06	**1.72**	57.58	**99.95**	0.79	**99.78**
Generic	59.06	**99.97**	97.64	**98.48**	0.82	**98.51**	0.38	**1.74**	73.19	**99.99**	0.86	**99.76**
Reconnai	89.08	**100.00**	99.05	**98.48**	0.93	**98.52**	0.36	**1.74**	90.26	**99.99**	0.95	**99.78**
DoS	96.89	**99.97**	99.25	**98.48**	0.93	**98.52**	0.36	**1.73**	92.52	**99.99**	0.96	**99.78**
Analysis	81.37	**100.00**	99.22	**98.48**	0.92	**98.52**	0.36	**1.74**	91.37	**99.99**	0.96	**99.79**
Backdoor	99.60	**100.00**	99.28	**98.48**	0.93	**98.52**	0.37	**1.74**	92.58	**99.99**	0.96	**99.79**
Shellcode	90.80	**100.00**	99.25	**98.48**	0.92	**98.52**	0.35	**1.73**	91.59	**99.99**	0.96	**99.79**
Worms	100.00	**100.00**	99.28	**98.48**	0.93	**98.52**	0.37	**1.74**	92.24	**99.99**	0.96	**99.79**

4.5.1 Concluding Remarks for UNSW-NB15 Dataset

The proposed methodology (P) demonstrates a substantial improvement in critical metrics such as Zero-Day Detection Rate (Z-DR), F1-Score, and Detection Rate (DR), emphasizing its effectiveness in addressing zero-day threats. While it trades off a slight reduction in accuracy and an increase in False Alarm Rate (FAR), these compromises are justified by the significant gains in zero-day attack detection. The results suggest that P is particularly well-suited for environments prioritizing high zero-day detection capabilities, even at the cost of marginally higher false positives. Overall, this performance establishes P as a robust and reliable approach for modern Network Intrusion Detection Systems (NIDS).

NF-UNSW-NB 15-V2 Dataset Results

Table 4.18 shows the detailed comparative analysis of the baseline studies [5] and our proposed methodology (P) across each metric:

- Zero-Day Detection Rate (Z-DR): P significantly outperforms [5] in Z-DR across almost all attack types. For example, Fuzzers and Exploits see the most improvement, with P reaching a Z-DR of 99.69% and 99.97%, respectively, compared to only 51.32% and 59.28% in [5]. P achieves a perfect score (100%) in several categories, such as Reconnaissance, Generic, Shellcode, Worms, and Backdoor, indicating a more robust ability to detect unknown attacks.

- Accuracy (Acc): accuracy of P is marginally higher than that of [5] across all attack types, with scores around 99.45% to 99.46% compared to [5]'s range of 98.07% to 99.77%. The improvement is most noticeable for the Exploits and Fuzzers attack types, where P maintains high accuracy close to 99.46%, while [5] is slightly lower at 98.07% and 98.38%, respectively.

- F1-Score (F1-Sc): P shows a consistently higher F1-Score, indicating an enhanced balance between precision and recall. For instance, P achieves 99.48% in Fuzzers and DoS, while [5] falls short with 85% and 97%, respectively. The improved F1-Score in P across all attack types reflects better handling of class imbalances and an overall more reliable classification model.

- False Alarm Rate (FAR): [5] has a marginally lower FAR across most attack types, ranging from 0.10% to 0.16%, while P varies between 0.56% and 0.57%. This suggests that [5] may produce slightly fewer false positives than P, but this lower FAR may come at the cost of reduced zero-day detection rates and accuracy, especially for challenging attack types like Exploits and Fuzzers.

- Detection Rate (DR): P outperforms [5] in detection rate across all attack types, with near-perfect scores for most attacks. For example, P achieves 100% DR for Reconnaissance, Generic, and Shellcode attacks, whereas [5] achieves only 98.15%, 98.05%, and 97.70% for these respective attacks. The consistently high DR in P signifies a more robust detection ability, particularly for known attack types, and is crucial in environments where missed detections pose significant security risks.

- Area Under Curve (AUC): AUC scores are exceptionally high for both models, but P holds a slight edge, consistently achieving around 99.68% to 99.72% across all attack types, compared to [5]'s range of 87% to 99%. The higher AUC in P reinforces its overall advantage, as it performs more effectively across a range of threshold values, providing a balance between true positive and false positive rates.

- Summary: P surpasses [5] in nearly every metric, especially Z-DR, F1-Score, DR, and AUC, indicating superior zero-day and known attack detection capabilities. However, [5] exhibits a slightly lower FAR, meaning it may result in fewer false positives, which could be desirable in scenarios where over-detection is a concern. Nonetheless, P's strong performance across metrics like Z-DR and DR, coupled with higher AUC scores, makes it a more robust and effective model for attack detection across varied types, particularly in critical security settings.

NF-UNSW-NB 15-V2 Dataset Results

Table 4.19 shows a comparative analysis of the performance of the baseline studies [5] and our proposed methodology.

Table 4.18: Comparison of Model Performance Metrics

Attack Type	Z-DR (%)		Acc (%)		F1-Sc (%)		FAR (%)		DR (%)		AUC (%)	
	[5]	P	[5]	P	[5]	P	[5]	P	[5]	P	[5]	P
Exploits	59.28	**99.97**	98.07	**99.45**	0.84	**99.47**	0.11	**0.57**	73.33	**99.98**	0.87	**99.71**
Fuzzers	51.32	**99.69**	98.38	**99.46**	0.85	**99.48**	0.10	**0.56**	74.61	**99.92**	0.87	**99.68**
Generic	99.11	**100.00**	99.75	**99.45**	0.98	**99.47**	0.15	**0.57**	98.05	**100.00**	0.99	**99.71**
Reconnai	99.57	**100.00**	99.77	**99.46**	0.98	**99.48**	0.15	**0.57**	98.15	**100.00**	0.99	**99.72**
DoS	93.68	**99.91**	99.71	**99.46**	0.97	**99.48**	0.15	**0.56**	96.85	**99.99**	0.98	**99.72**
Analysis	87.95	**100.00**	99.75	**99.46**	0.97	**99.48**	0.14	**0.56**	97.15	**100.00**	0.99	**99.72**
Backdoor	99.49	**100.00**	99.76	**99.45**	0.97	**99.47**	0.16	**0.57**	97.84	**100.00**	0.99	**99.71**
Shellcode	95.94	**100.00**	99.75	**99.46**	0.97	**99.48**	0.16	**0.57**	97.70	**100.00**	0.99	**99.72**
Worms	100.00	**100.00**	99.77	**99.46**	0.97	**99.48**	0.14	**0.56**	97.59	**100.00**	0.99	**99.72**

- Z-DR: P has a significantly higher Z-DR than [5] across most attack types. For example, in Exploits, P achieves 99.92%, outperforming [5]'s 81.47%. In Worms and Generic attacks, XGBoost consistently reaches 100%, highlighting its strength in recognizing unknown patterns, making it more effective for Zero-Day Attack Detection.

- Accuracy (Acc): P shows slightly better accuracy than [5] across all attack types, with values around 99.38% to 99.49%, compared to [5]'s range from 98.89% to 99.63%. This minor edge in accuracy for P implies better general performance, especially in DoS attacks, where P scores 99.49% versus [5]'s 99.55%.

- F1-Score (F1-Sc): P outperforms P in F1-Score across all attack types, achieving scores around 99.40% to 99.50% compared to [5]'s range from 91% to 97%. The F1-Score improvement in P is notable in Fuzzers (99.41% vs.91%) and Exploits (99.40% vs.92%), showing a better balance between precision and recall for these challenging attacks.

- False Alarm Rate (FAR): [5] has a slightly lower FAR than P for some attack types, suggesting fewer false positives. For example, [5] has a FAR of 0.29% in Exploits and Backdoor attacks, whereas P has 0.64% for the same. However, P performs better in DoS and Shellcode attacks, with FAR as low as 0.53% and 0.56%, respectively. This indicates that while [5] may excel in minimizing false positives, P maintains a balanced FAR across all attack types.

- Detection Rate (DR): P demonstrates a higher DR across nearly all attack types, indicating a more effective detection capability. For example, P reaches 99.97% in Exploits and 100% in Generic and Shellcode, surpassing [5]'s rates of 87.74%, 98.85%, and 97.94%, respectively. This enhanced DR in P is particularly important for high-risk attacks like Reconnaissance and Backdoor, where missed detections are critical, giving it a clear advantage.

- Area Under Curve (AUC): Both models show high AUC values, but P slightly edges out [5] with near-perfect scores of around 99.93% to 99.95% compared to [5]'s 93% to 99%. The consistent AUC in P reflects better performance across threshold settings, underscoring its robustness in distinguishing between true positives and false positives, especially in DoS and Shellcode attacks.

- Summary: P outperforms [5] in most metrics, particularly Z-DR, F1-Score, DR, and AUC, making it highly effective for Zero-Day Attack Detection and reliable classification across known attacks. [5] shows a slightly lower FAR in some cases, which may be preferable when minimizing false positives is crucial. However, the strong overall performance of P, especially in critical metrics like Z-DR and DR, suggests that it is better suited for environments requiring high accuracy and comprehensive attack detection.

Table 4.19: Comparison of Model Performance Metrics

Attack Type	**Z-DR (%)**		**Acc (%)**		**F1-Sc (%)**		**FAR (%)**		**DR (%)**		**AUC (%)**	
	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**	[5]	**P**
Exploits	81.47	**99.92**	98.89	**99.38**	0.92	**99.40**	0.29	**0.64**	87.74	**99.97**	0.94	**99.93**
Fuzzers	76.19	**98.43**	98.96	**99.39**	0.91	**99.41**	0.19	**0.62**	85.72	**99.62**	0.93	**99.93**
Generic	99.57	**100.00**	99.62	**99.38**	0.97	**99.40**	0.33	**0.64**	98.85	**100.00**	0.99	**99.94**
Reconnai	99.75	**100.00**	99.60	**99.38**	0.96	**99.40**	0.30	**0.64**	97.89	**100.00**	0.99	**99.93**
DoS	90.68	**99.91**	99.55	**99.49**	0.95	**99.50**	0.31	**0.53**	96.53	**99.98**	0.98	**99.95**
Analysis	88.47	**100.00**	99.60	**99.38**	0.95	**99.40**	0.34	**0.64**	98.23	**100.00**	0.99	**99.94**
Backdoor	97.28	**100.00**	99.59	**99.38**	0.95	**99.41**	0.29	**0.64**	96.90	**100.00**	0.98	**99.94**
Shellcode	98.60	**100.00**	99.63	**99.46**	0.96	**99.48**	0.30	**0.56**	97.94	**100.00**	0.99	**99.95**
Worms	100.00	**100.00**	99.63	**99.46**	0.95	**99.48**	0.32	**0.56**	98.46	**100.00**	0.99	**99.95**

4.5.2 Concluding Remarks For UNSW-NB15-v2 Dataset

The evaluation of machine learning (ML) techniques for detecting zero-day attacks in Network Intrusion Detection Systems (NIDS) revealed significant performance differences across the models employed in this research. Random Forest (RF) and XGBoost emerged as the most effective models, consistently achieving high performance across all datasets. Notably, RF achieved a 100% Zero-Day Detection Rate (Z-DR) for multiple attack categories in the UNSW-NB15 dataset, while XGBoost demonstrated similar excellence in the NF-UNSW-NB15-v2 dataset. Their ability to handle diverse feature distributions and maintain stability under zero-day scenarios underscores their robustness.

In contrast, Convolutional Neural Networks (CNN) and Voting Classifiers, despite their high accuracy in known attack detection, showed limited ability to generalize to zero-day scenarios, resulting in lower Z-DR scores. This disparity highlights the challenge of balancing model complexity with generalization capabilities, particularly for deep learning methods.

The performance metrics used were accuracy, precision, recall, F1 score, Area Under the Receiver Operating Characteristic Curve (AUC-ROC), and the novel Z-DR metric, which provided comprehensive insights into model efficacy. Z-DR was particularly crucial in assessing adaptability to unseen threats, emphasizing the importance of specialized metrics for zero-day detection. The findings suggest that ensemble methods like RF and XGBoost are highly suited for NIDS applications, while further refinement is needed to enhance the adaptability of deep learning models.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This study offers a detailed framework for detecting zero-day attacks within Network Intrusion Detection Systems (NIDS) using a range of machine learning models, including Random Forest, MLP, CNN, XGBoost, and a Voting classifier. Using multiple datasets UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018, this research aimed to improve zero-day attack detection through extensive feature engineering, preprocessing, and applying diverse machine learning methods. The results show that Random Forest and XGBoost consistently perform best, particularly in zero-day scenarios, achieving high F1-scores and Zero-Day Detection Rates (Z-DR). While MLP, CNN, and Voting classifier proved effective for detecting known attacks, their performance in identifying novel threats varied. The Kolmogorov-Smirnov (K-S) test confirmed the significant feature differences between benign and attack traffic, highlighting the importance of selecting strong, distinctive features to better separate normal and abnormal behaviors. Each model type has its strengths and limitations: Random Forest and XGBoost are more adaptable to zero-day detection, while CNN and Voting classifier showed lower Z-DR scores, indicating that further tuning may be needed, especially for unknown attack types. In summary, this study makes a valuable contribution to NIDS adaptability and effectiveness, laying the groundwork for more resilient network security systems that can better defend against emerging cyber threats.

5.2 Summary of the Study

The study explores an advanced framework for zero-day attack detection in Network Intrusion Detection Systems (NIDS) using machine learning (ML). In this study, we identified new features to detect zero-day attacks, with a focus on feature engineering, dataset preprocessing, and the use of machine learning models like Random Forest, XGBoost, Multilayer Perceptron, CNN, and Voting Classifier.

Summary of Key Findings

Aspect	Details
Research Goal	Development of a robust machine learning framework for detecting zero-day attacks in NIDS.
Datasets	Utilized UNSW-NB15, NF-UNSW-NB15-v2, and NF-CSE-CIC-IDS 2018 for comprehensive evaluation.
Feature Engineering	Employed advanced techniques including feature scaling, extraction, and selection to optimize performance.
Models Evaluated	Random Forest, XGBoost, MLP, CNN, and Voting Classifier.

Performance Highlights	RF and XGBoost achieved high F1-scores (>0.98) and strong Zero-Day Detection Rates (Z-DR).
Key Insight	Poor Z-DR observed in CNN and Voting Classifiers despite high accuracy.
Impact	Enhanced generalization capabilities of ML models, addressing key gaps in traditional NIDS.

5.3 Future Work

In this section, we listed potential directions for future research aimed at advancing the resilience of network intrusion detection systems (NIDS) in cybersecurity. Three primary areas of focus are identified, including comparing performance using other datasets, Integration of Unsupervised Learning Techniques, and Dynamic Model Adaptation.

- Integration of Unsupervised Learning Techniques: Future research should prioritize enhancing the generalization capabilities of detection models by incorporating unsupervised anomaly detection methods or exploring hybrid learning approaches. Leveraging these techniques may improve zero-day attack detection by enabling models to recognize novel patterns without labelled data. Additionally, exploring advanced architectures, such as deep learning and generative models, could further bolster zero-day detection performance, uncovering subtle deviations in network traffic that might indicate emerging threats.

- Dataset: Future work involves evaluating the proposed approach with datasets that cover special purpose network intrusion detection systems (NIDS) (e.g., IoT and Critical Infrastructure networks), which will comprise insights into adapting the proposed models, as well as proposing and adapting other ML techniques to use for zero-day attack detection.

- Dynamic Model Adaptation: Developing adaptive models that evolve alongside changing network environments and emerging attack vectors is essential. Future research should focus on creating mechanisms for real-time model retraining and updating, enabling faster adaptation to new threats and enhancing overall resilience against zero-day attacks with lower false positive and false negative rates.

Bibliography

[1] https://www.marketsandmarkets.com/. *Artificial Intelligence*, 2023.

[2] https://www.ibm.com/annualreport/2022/past-reports.html. *IBM*, 2022.

[3] Naushad Alam and Muqeem Ahmed. Zero-day network intrusion detection using machine learning approach. 11(8):194–201.

[4] Mike Nkongolo and Mahmut Tokmak. Zero-day threats detection for critical infrastructures. In Aurona Gerber and Marijke Coetzee, editors, *South African Institute of Computer Scientists and Information Technologists*, volume 1878, pages 32–47. Springer Nature Switzerland. Series Title: Communications in Computer and Information Science.

[5] Mohanad Sarhan, Siamak Layeghy, Marcus Gallagher, and Marius Portmann. From zero-shot machine learning to zero-day attack detection. 22(4):947–959.

[6] Mahdi Soltani, Behzad Ousat, Mahdi Jafari Siavoshani, and Amir Hossein Jahangir. An adaptable deep learning-based intrusion detection system to zero-day attacks. *Journal of Information Security and Applications*, 76:103516, 2023.

[7] Shadi Aljawarneh, Monther Aldwairi, and Muneer Bani Yassein. Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science*, 25:152–160, 2018.

[8] Payam Vahdani Amoli, Timo Hamalainen, Gil David, Mikhail Zolotukhin, and Mahsa Mirzamohammad. Unsupervised network intrusion detection systems for zero-day fast-spreading attacks and botnets. *JDCTA (International Journal of Digital Content Technology and its Applications*, 10(2):1–13, 2016.

[9] https://www2.dtexsystems.com/2023ponemonreport. *ponemonreport*, 2023.

[10] Ansam Khraisat, Iqbal Gondal, and Peter Vamplew. An anomaly intrusion detection system using c5 decision tree classifier. In *Trends and Applications in Knowledge Discovery and Data Mining: PAKDD 2018 Workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, June 3, 2018, Revised Selected Papers 22*, pages 149–155. Springer, 2018.

[11] Tuan-Hong Chua and Iftekhar Salam. Evaluation of machine learning algorithms in network-based intrusion detection system.

[12] Sabrine Ennaji, Fabio De Gaspari, Dorjan Hitaj, Luigi V Mancini, et al. Adversarial challenges in network intrusion detection systems: Research insights and future prospects. *arXiv preprint arXiv:2409.18736*, 2024.

[13] Besik Beridze and Mikheil Donadze. Network anomaly detection utilizing machine learning methods. In *2023 IEEE East-West Design & Test Symposium (EWDTS)*, pages 1–7. IEEE.

[14] Besik Beridze and Mikheil Donadze. Network anomaly detection utilizing machine learning methods. In *2023 IEEE East-West Design & Test Symposium (EWDTS)*, pages 1–7. IEEE, 2023.

[15] Muhammad Ali, Mansoor-ul Haque, Muhammad Hanif Durad, Anila Usman, Syed Muhammad Mohsin, Hana Mujlid, and Carsten Maple. Effective network intrusion detection using stacking-based ensemble approach. *International Journal of Information Security*, 22(6):1781–1798, 2023.

[16] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[17] Nour Moustafa and Jill Slay. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 2016.

[18] Ravi Vinayakumar, Mamoun Alazab, K Padannayil Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[19] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. Towards a standard feature set for network intrusion detection system datasets. *Mobile networks and applications*, pages 1–14, 2022.

[20] Hongfang Zhou, Xiqian Wang, and Rourou Zhu. Feature selection based on mutual information with correlation coefficient. *Applied Intelligence*, 52:1–18, 03 2022.

[21] Rasheed Ahmad, Izzat Alsmadi, Wasim Alhamdani, and Lo'ai Tawalbeh. Zero-day attack detection: a systematic literature review. *Artificial Intelligence Review*, 56(10):10733–10811, 2023.

[22] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1):1–22, 2019.

[23] amp; Mohamed A. Berry, M. Supervised and unsupervised learning for data science. https://doi.org/10.1007/978-3-030-22475-2. *Data Science*, 2020.

[24] Chi Duc Luu, Van Quan Nguyen, Truong Son Pham, and Nhien-An Le-Khac. A zero-shot deep learning approach for unknown IoT botnet attack detection. In *2023 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 278–283. IEEE.

[25] Stephen Wanjau, Geoffrey Wambugu, and Aaron Oirere. Network intrusion detection systems: A systematic literature review of hybrid deep learning approaches. *International Journal of Emerging Science and Engineering*, 10:1–16, 06 2022.

[26] Zeeshan Ahmad, Adnan Shahid Khan, Cheah Wai Shiang, Johari Abdullah, and Farhan Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.

[27] Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708–713, 2015.

[28] Ali Abbasi, Jos Wetzels, Wouter Bokslag, Emmanuele Zambon, and Sandro Etalle. On emulation-based network intrusion detection systems. In *Research in Attacks, Intrusions and Defenses: 17th International Symposium, RAID 2014, Gothenburg, Sweden, September 17-19, 2014. Proceedings 17*, pages 384–404. Springer, 2014.

[29] Elijah Maseno, Zenghui Wang, and Hongyan Xing. A systematic review on hybrid intrusion detection system. *Security and Communication Networks*, 2022:1–23, 05 2022.

[30] Prakash Kalavadekar and Dr Sane. Building an effective intrusion detection system using combined signature and anomaly detection techniques. *International Journal of Innovative Technology and Exploring Engineering*, 8:429–435, 08 2019.

[31] Sajad Einy, Cemil Oz, and Yahya Dorostkar Navaei. The anomaly- and signature-based ids for network security using hybrid inference systems. *Mathematical Problems in Engineering*, 2021:1–10, 2021.

[32] Yazan Otoum and Amiya Nayak. As-ids: Anomaly and signature based ids for the internet of things. *Journal of Network and Systems Management*, 29(3):23, 2021.

[33] Philokypros Ioulianou, Vasileios Vasilakis, Ioannis Moscholios, and Michael Logothetis. A signature-based intrusion detection system for the internet of things. *Information and Communication Technology Form*, 2018.

[34] bluegoatcyber. https://bluegoatcyber.com/blog/signature-vs-anomaly-based-detection-which-is-more-effective. *bluegoatcyber*.

[35] Rafath Samrin and D Vasumathi. Review on anomaly based network intrusion detection system. In *2017 international conference on electrical, electronics, communication, computer, and optimization techniques (ICEECCOT)*, pages 141–147. IEEE, 2017.

[36] Mansi Bhavsar, Kaushik Roy, John Kelly, and Odeyomi Olusola. Anomaly-based intrusion detection system for iot application. *Discover Internet of things*, 3(1):5, 2023.

[37] Elhadj Benkhelifa, Thomas Welsh, and Walaa Hamouda. A critical review of practices and challenges in intrusion detection systems for iot: Toward universal and resilient systems. *IEEE communications surveys & tutorials*, 20(4):3496–3509, 2018.

[38] Seiba Alhasan, Gaddafi Abdul-Salaam, Yaw Missah, and Mohammad Anisi. Hybrid network intrusion detection systems: A systematic review. pages 1–35, 01 2024.

[39] Waleed Bul'ajoul, Anne James, and Mandeep Pannu. Improving network intrusion detection system performance through quality of service configuration and parallel technology. *Journal of Computer and System Sciences*, 81(6):981–999, 2015.

[40] Frank Cremer, Barry Sheehan, Michael Fortmann, Arash N Kia, Martin Mullins, Finbarr Murphy, and Stefan Materne. Cyber risk and cybersecurity: a systematic review of data availability. *The Geneva papers on risk and insurance. Issues and practice*, 47(3):698, 2022.

[41] Yatin Kalra, Saket Upadhyay, and Pushpinder Patheja. Advancements in cyber attacks and security. *International Journal of Innovative Technology and Exploring Engineering*, 9:1520–1528, 02 2020.

[42] Nari Sivanandam Arunraj, Robert Hable, Michael Fernandes, Karl Leidl, and Michael Heigl. Comparison of supervised, semi-supervised and unsupervised learning methods in network intrusion detection system (nids) application. *Anwendungen und Konzepte der Wirtschaftsinformatik*, (6):10–19, 2017.

[43] Samir-B. amp; Fares B. Omar, B. Idss for scada systems: Security analysis, challenges and solutions. 1-6. https://doi.org/10.1109/inis46949.2021.10151947. pages 1–35, 01 2021.

[44] Amer FAH Alnuaimi and Tasnim HK Albaldawi. An overview of machine learning classification techniques. In *BIO Web of Conferences*, volume 97, page 00133. EDP Sciences, 2024.

[45] Bahzad Charbuty and Adnan Abdulazeez. Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, 2(01):20–28, 2021.

[46] Majjed Al-Qatf, Yu Lasheng, Mohammed Al-Habib, and Kamal Al-Sabahi. Deep learning approach combining sparse autoencoder with svm for network intrusion detection. *Ieee Access*, 6:52843–52856, 2018.

[47] Hamed Taherdoost. Deep learning and neural networks: Decision-making implications. *Symmetry*, 15(9):1723, 2023.

[48] Vratika Gupta, Vinay Kumar Mishra, Priyank Singhal, and Amit Kumar. An overview of supervised machine learning algorithm. In *2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART)*, pages 87–92. IEEE, 2022.

[49] Nils F Jansson, Rodney L Allen, Göran Skogsmo, and Saman Tavakoli. Principal component analysis and k-means clustering as tools during exploration for zn skarn deposits and industrial carbonates, sala area, sweden. *Journal of Geochemical Exploration*, 233:106909, 2022.

[50] Ferath Kherif and Adeliya Latypova. Principal component analysis. In *Machine learning*, pages 209–225. Elsevier, 2020.

[51] IBM. https://www.ibm.com/docs/en/db2oc?topic=procedures-principal-component-analysis-pca. In *PCA*, pages 209–225. Elsevier, 2023.

[52] Pedro Casas, Johan Mazel, and Philippe Owezarski. Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. *Computer Communications*, 35(7):772–783, 2012.

[53] Miel Verkerken, Laurens D'hooge, Tim Wauters, Bruno Volckaert, and Filip De Turck. Unsupervised machine learning techniques for network intrusion detection on modern data. In *2020 4th Cyber Security in Networking Conference (CSNet)*, pages 1–8. IEEE, 2020.

[54] YCAP Reddy, P Viswanath, and B Eswara Reddy. Semi-supervised learning: A brief review. *Int. J. Eng. Technol*, 7(1.8):81, 2018.

[55] Hye-Woo Lee, Noo-ri Kim, and Jee-Hyong Lee. Deep neural network self-training based on unsupervised learning and dropout. *International Journal of Fuzzy Logic and Intelligent Systems*, 17(1):1–9, 2017.

[56] Kyle Weston. *Improving image classification by co-training with multi-modal features*. McGill University (Canada), 2011.

[57] Guo-Jun Qi and Jiebo Luo. Small data challenges in big data era: A survey of recent progress on unsupervised and semi-supervised methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2168–2187, 2020.

[58] Idah Pindai Zengeni and Mohamad fadli Zolkipli. Zero-day exploits and vulnerability management. *Borneo International Journal eISSN 2636-9826*, 7(3):26–33, 2024.

[59] Kaspersky. Wannacry ransomware: Everything you need to know. kaspersky daily. 2021.

[60] CISA. Cybersecurity amp; infrastructure security agency (cisa). (2021). alert (aa20-352a). 2021.

[61] Alexander Grakovski, Aleksandr Krivchenkov, and Boriss Misnevs. Feature selection method for ml/dl classification of network attacks in digital forensics. *Transport and Telecommunication Journal*, 23(2):131–141, 2022.

[62] Zhipeng Li, Zheng Qin, Pengbo Shen, and Liu Jiang. Zero-shot learning for intrusion detection via attribute representation. In *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part I 26*, pages 352–364. Springer, 2019.

[63] Alok Kumar Shukla. An efficient hybrid evolutionary approach for identification of zero-day attacks on wired/wireless network system. *Wireless Personal Communications*, 123(1):1–29, 2022.

[64] Priya Pitre, Arya Gandhi, Vaishnavi Konde, Rahul Adhao, and Vinod Pachghare. An intrusion detection system for zero-day attacks to reduce false positive rates. In *2022 International Conference for Advancement in Technology (ICONAT)*, pages 1–6. IEEE, 2022.

[65] Nerella Sameera and M Shashi. Deep transductive transfer learning framework for zero-day attack detection. *ICT Express*, 6(4):361–367, 2020.

[66] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8256–8265, 2019.

[67] Zhun Zhang, Qihe Liu, Shilin Qiu, Shijie Zhou, and Cheng Zhang. Unknown attack detection based on zero-shot learning. 8:193981–193991.

[68] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. Zero-shot learning with semantic output codes. *Advances in neural information processing systems*, 22, 2009.

[69] Vikash Kumar and Ditipriya Sinha. A robust intelligent zero-day cyber-attack detection technique. *Complex & Intelligent Systems*, 7(5):2211–2234, 2021.

[70] Shiyi Jin, Jin-Gyun Chung, and Yinan Xu. Signature-based intrusion detection system (ids) for in-vehicle can bus network. In *2021 IEEE international symposium on circuits and systems (ISCAS)*, pages 1–5. IEEE, 2021.

[71] Hannes Holm. Signature based intrusion detection for zero-day attacks:(not) a closed chapter? In *2014 47th Hawaii international conference on system sciences*, pages 4895–4904. IEEE, 2014.

[72] Veeramreddy Jyothsna and Koneti Munivara Prasad. Anomaly-based intrusion detection system. *Computer and Network Security*, 10, 2019.

[73] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[74] Pedro Garcia-Teodoro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1-2):18–28, 2009.

[75] Shafiq Hussain and Tabinda Shehzadi. Machine learning-powered intrusion detection: Safeguarding networks in the digital era. *MZ Journal of Artificial Intelligence*, 1(1):6–15, 2024.

[76] Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. Netflow datasets for machine learning-based network intrusion detection systems. In *Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, December 11, 2020, Proceedings 10*, pages 117–135. Springer, 2021.

[77] Ravi Vinayakumar, Mamoun Alazab, K Padannayil Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[78] Amr Attia, Miad Faezipour, and Abdelshakour Abuzneid. Network intrusion detection with xgboost and deep learning algorithms: an evaluation study. In *2020 international conference on computational science and computational intelligence (CSCI)*, pages 138–143. IEEE, 2020.

[79] Aitor Belenguer, Javier Navaridas, and Jose A Pascual. A review of federated learning in intrusion detection systems for iot. *arXiv preprint arXiv:2204.12443*, 2022.

[80] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.

[81] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. Towards a standard feature set for network intrusion detection system datasets. *Mobile networks and applications*, pages 1–14, 2022.

[82] Lijuan Ren, Tao Wang, Aicha Sekhari Seklouli, Haiqing Zhang, and Abdelaziz Bouras. A review on missing values for main challenges and methods. *Information Systems*, page 102268, 2023.

[83] G Gowthami and S Silvia Priscila. Classification of intrusion using cnn with iqr (inter quartile range) approach. In *International Conference on Advancements in Smart Computing and Information Security*, pages 259–269, 2023.

[84] Emad Obaid Merza and Nashaat Jasim Mohammed. Fast ways to detect outliers. *Journal of Techniques*, 3(1):66–73, 2021.

[85] OpenAI. Chatgpt: A conversational ai model. `https://chat.openai.com/`, 2024. Accessed: 2024-10-24.

[86] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[87] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, 8(1):014008, 2015.

[88] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427–437, 2009.

[89] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[90] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

Vita

Candidate's full name:

Akinwale Mayomi Aisida

University attended (with dates and degrees obtained):

Master of Computer Science under the supervision of Dr. Saqib Hakak, University of New Brunswick (UNB), Fredericton, New Brunswick, Canada, Expected May 2025.

Master of Science in Information Technology, National Open University, Nigeria, December 2021.

Bachelor of Science in Computer Science, Achievers University, Owo, Ondo State, June 2011.

Publications: Predicting Students' Academic Performance Using Artificial Neural Network (www.isteams.net/ghanabespoke2021).

Conference Presentations: N/A