Deep Learning-Based Network Intrusion Classification Using Oversampling Methods and Multi-Dataset Validation

By

Mahbubul Haq Bhuiyan and 012 231 0045

Submitted in partial fulfilment of the requirements
of the degree of Masters of Science in Computer Science and Engineering

October 23, 2024

Department of Computer Science and Engineering
United International University

Abstract

Intrusion detection and classification are crucial for network protection against malware incidents and unauthorized access. Machine Learning and Deep Learning contribute significantly to the enhancement of intrusion detection by recognizing patterns in network traffic data. With techniques of Machine Learning (ML) and Deep Learning (DL), most of the attacks can be detected, even stealthy and polymorphic, which conventional security mechanisms often fail to capture. In this work, we approach the class imbalance problem of the network intrusion detection datasets with data sampling techniques like Adaptive Synthetic Sampling (ADASYN), Synthetic Minority Over-sampling Technique (SMOTE), and Borderline-SMOTE. These algorithms enhance the performance of DL models by improving the accuracy of detecting rare intrusion events. We employ, for that purpose, Deep Neural Networks to raw network data, which proves to be a much better option to classify malicious activities, particularly in large and complex datasets. In such a context, this paper proposes an optimized Deep Neural Network (DNN) model that serves effectively to detect stealthy and polymorphic attacks with high precision. Therefore, the model is trained and tested with the NF-ToN-IoT dataset, while its extended multi-datasets validation is done by testing it on datasets such as NF-BoT-IoT, NF-UNSW-NB15, NF-CSE-CIC-IDS2018, NF-UQ-NIDS, and NF-UNSW-NB15-v2. By doing so, this paper comprehensively discusses the robustness of the model in various network environments. We have also compared our architecture with various state-of-the-art architectures like Convolutional Neural Networks (CNN) with Bidirectional Long Short-Term Memory (BiLSTM), DNN, Gated Recurrent Units (GRU) with Recurrent Neural Networks (RNN), and CNN with Long Short-Term Memory (LSTM). The proposed model of DNN has performed exceptionally well, reaching a weighted average of 0.99 for all key metrics including accuracy, precision, recall, and F1-score. Besides, the model outperforms the compared architectures and proves its effectiveness for network intrusion classification tasks. The ablation study conducted in this work also presents the influence of some key parameters in the DL model that could be very informative for further studies.

Acknowledgements

This work would have not been possible without the input and support of many people over the last two trimesters. I would like to express my gratitude to everyone who contributed to it in some way or other.

My sincere gratitude goes to my project Supervisor, Prof. Dewan Md. Farid (PhD). Without his constant support, pulling off this work would have been impossible.

I am also thankful to the Head Examiner and Reviewer, Prof. Dr. Al-Sakib Khan Pathan. He provided some valuable comments in the pre-defense which helped a lot to shape the final version of the work.

I would also like to thank Prof. Dr. Md. Motaharul Islam, Director, MSCSE, UIU and Professor Dr. Mohammad Nurul Huda, Departmental Head of CSE, UIU for their invaluable support.

Last but not the least, I owe to my parents for their unconditional love and immense emotional support.

Publication List

The main contributions of this research are either published or accepted or in preparation in journals and conferences as mentioned in the following list:

Journal Article

1. Mahbubul Haq Bhuiyan, Khorshed Alam and Dewan Md. Farid. Deep Learning-Based Network Intrusion Classification Using Oversampling Methods and Multi-Dataset Validation (Submitted to IEEE Access)

Conference Paper

1. Mahbubul Haq Bhuiyan, Khorshed Alam, Kamrul Islam Shahin, and Dewan Md. Farid. A deep learning approach for network intrusion classification. In The IEEE Region 10 Sym- posium (TENSYMP), pages 1–6, Netaji Subhas University of Technology, New Delhi, India, September 2024

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to Network Intrusion Detection (NID)

Network Intrusion Detection is considered one of the backbones of modern cybersecurity infrastructure, being a very specialized defense mechanism that should protect the integrity, confidentiality, and availability of data in computer networks. With the ever-evolving cyber threats in terms of complexity and sophisticated nature, NID systems act as agile defenders, monitoring the flow of all network traffic in real-time. Their main role is to analyze the data as it moves along the networks for suspicious activities or anomalies. This includes detecting patterns that may indicate the possibility of malware infections, unauthorized access attempts, or other forms of cyber-attacks that may normally bypass traditional security tools like firewalls, intrusion prevention systems, and antivirus software.

NID can accurately detect attacks which probably pass by the traditional security mechanisms such as firewalls and antivirus. Firewalls and antivirus depend on predefined rules or previously detected attack patterns, which is what makes them so vulnerable in cases of new, unknown, or smartly disguised threats. NID systems monitor network activities for unusual events on their part. Examples include unusual traffic patterns that may indicate malware, and even finding suspicious login attempts that could mean someone is trying to access sensitive data unauthorized.

The NID systems protect against internal threats too. Not every threat arises from outside an organization. Disgruntled or unauthorized employees can pose one if they attempt to access systems that they should not. An NID system is able to flag such unusual behavior like trying to access sensitive information. For example, the early warning will enable the organization to take immediate action in the form of blocking access, investigating the situation further or both.

Cyber threats keep on continuously evolving. The NID systems are the same. Next-generation systems will provide real-time alerts and visibility into newly identified threats. By studying how the attacks happen, these systems can help security teams understand the methods of the attackers. This would be critical in trying to improve future security

strategies, thus enabling organizations to protect themselves from emerging threats much better. The outcomes can also be advanced security systems that might more rapidly adapt to new kinds of risks.

Modern cybersecurity requires involvement of an NID system without any doubt. Capabilities from the range of detection and response to cyber threats mean a organization is protected, and that the network is secure. Over time, cyber threats will continuously change, and NID systems will remain part of strategic ways of defense in protecting current and future risks.

1.2 Deep Learning in NID

The application of Deep Learning (DL) has emerged as a game-changer in significantly enhancing the capabilities of NID. In contrast to the conventional approaches of predefined rules and heavy manual feature engineering, DL models can process and analyze huge amounts of high-dimensional data with minimal human intervention. This feature makes DL especially better positioned for modern cybersecurity challenges where the volume, velocity, and variety of network traffic have increased exponentially. Among the most salient attributes of DL, deep neural networks are able to learn autonomously from raw network traffic data, which is complicated in pattern and subtle in feature. These are relationships that traditional methods may not capture, especially in the face of the increasingly sophisticated tactics that modern cyber attackers apply.

The DL models can be trained on comprehensive datasets containing both legitimate and malicious network activity, thus enabling them to deeply learn how different the normal behavior is from various types of network intrusions. This will be through labeled datasets, which have examples of a wide variety of network traffic patterns to show the model's ability in differentiating between benign and harmful activities.

Hence, the opportunity to learn from such large datasets enables the DL-powered NID systems to detect not only conventional cyber threats like malware attacks and DoS attacks but also more advanced and evolving threats that are finding their ways to evade detection and grow more and more sophisticated. The signature of those threats normally involves novel techniques that enable them to bypass the detection capability of traditional security systems, and that is where DL comes into play due to its learning capability. Moreover, the protection afforded by DL-based NID systems is more dynamic and responsive. While other models are static or rule-based and would require constant updating in an effort to attempt to keep pace with new threat vectors, DL models organically improve as they become exposed to more and newer data. This can be done toward those ever-changing cybersecurity landscapes with much higher precision and effectiveness. DL-powered NIDs recognize patterns indicative of emerging threats, so the network infrastructures are well protected against known attacks and will also be in the future against unforeseen challenges. The scalability feature with evolving threats makes DL of particular importance in the ongoing battle to secure these modern, complex network environments. This pro-

vides a much-needed resilience for the organizations to protect sensitive data and critical systems.

1.3 Challenges in NID Research

While NID research has seen tremendous growth, there are some critical issues that still exist and thereby help to restrict the overall performance of such systems. Among these, one big problem is strong dependency on very old datasets both for training and testing. While those datasets served well during the initial phases of NID development, they no longer represent the complex and continually evolving tactics of the modern cyber-attacker. While cyber threats become more advanced, a model's ability to generalize on new types of malicious activities becomes more important. Unfortunately, most of the models that have been trained with older datasets often result in poor performance related to the detection of novel and emerging threats. Besides, although most of the current research has focused on how to achieve high accuracy on controlled environments, it does not necessarily represent reality. The only problem is that, in real-world applications, modern NID systems must be continuously adapted to rapidly evolving network environments and threat landscapes, for which most state-of-the-art models have been designed. In fact, such adaptability offers just that degree of robustness and effectiveness required to confront the dynamic character of cyber-attacks.

The key developing challenges among NID systems is their application to these new, dedicated, and specialized technologies, such as smart grids and other critical infrastructure networks. Those types of environments very often have their particular needs and security gaps, which hardly can be met using traditional NID facilities. A smart grid, for example, relies on a complex combination of operational technologies and decentralized control systems working in a real-time data exchange mode. This automatically presupposes a tailored NID solution that would be able to match such specifications with particular needs and risks. Traditional NID architectures, created with conventional IT networks as the primary focus, are inadequate to deal with threats that are specific and ever-changing in these growing sectors. This, therefore, is an urgent time for specialized intrusion detection solutions because these technologies find increasing applications.

One of the most vital difficulties that has provoked the research of NID is the problem of data imbalance that prevails in the intrusion detection datasets. In most cases, malicious traffic is underrepresented to a large degree compared to normal traffic. The consequence of this may be a biased machine learning model that performs well on detecting normal network activity but fails terribly in intrusion detection. That is, the models will be biased to classify benign traffic, while missing those more critical but infrequent things, malicious activities. This seriously compromises the efficacy of NID systems, especially in the case of sophisticated attacks of stealthy nature, which usually show up only as rare anomalies in network traffic.

Several oversampling techniques are proposed for this problem. This could be ac-

complished through various techniques such as GAN, SMOTE, Borderline SMOTE, and ADASYN, which synthetically augment malicious traffic instances so that the datasets are balanced. In such cases, these models create synthetic samples from the minority under-represented class, allowing machine learning models to see a much balanced dataset during the training phase. This means analyzing and comparing various oversampling techniques, after which we show an in-depth analysis where different performance metrics are shown in a dedicated table contrasting efficiency for the said techniques to enhance NID detection capabilities. With that, have contributed to enhance the robustness of the NID models so that they can identify both frequent and infrequent kinds of network intrusions.

1.4 Contribution

The contribution of this study includes the following:

- The study introduces an innovative DNN model specifically designed for efficient detection of stealthy and polymorphic variants of network intrusions. This model is aimed at mitigating false negatives and positives, a crucial aspect in network security.
- The model's robustness is tested using several datasets, as shown in Table 4.3. It is trained and tested on well-known datasets like NF-BoT-IoT, NF-UNSW-NB15, NF-CSE-CIC-IDS2018, NF-UQ-NIDS, and NF-UNSW-NB15-v2. The model shows strong results on all these datasets.
- The study compares the proposed model with previously established architectures such as CNN+BiLSTM, DHN, GRU+RNN, and CNN+LSTM. Utilizing the NF-ToN-loT dataset as a common ground for comparison, the proposed model outperforms these prior architectures, highlighting its efficacy and advancement in the field shown in Table 4.1.
- Comparison between different oversampling methods like SMOTE, borderline SMOTE, ADASYN has been shown in Table 4.2 that consists of the performance of the model after using the oversampling methods on NF-ToN-IoT dataset.
- An ablation study is conducted to dissect the components of the DNN model shown in Table 3.2. This analysis provides insights into the individual contributions of each component towards detecting malware traffic, offering valuable information for optimizing future NID models in the cybersecurity domain.

1.5 Paper Organization

The paper is organized into several key sections that systematically present the research findings and methodology. In chapter 2, we provide an overview of NID, DNN, the role of DNN in NID, understanding the network flow data of the primary the dataset used, along

with its significance in detecting stealthy attacks. Lastly in this chapter, we have provided literature review of previous related works. In chapter 3, details of the methodology, including dataset formation, preprocessing, and model development are given. Chapter 4 analyses the model's performance using key metrics. In chapter 5, we have discussed about no synthetic vs synthetic data in context of proposed NID, followed by a discussion on its application deployment considerations, the reason behind the successful classification of our NID and lastly about zero-day attack detection in NID. Finally, chapter 6 concludes the research and discusses about the limitations and scope of future works.

Chapter 2

Background

2.1 Network Intrusion Detection (NID)

Network Intrusion Detection is the backbone of network security infrastructure that is supposed to detect and respond to unauthorized or malicious activity because of a computer network. This performs like a passive monitoring system that would observe network traffic continuously for signs that may be related to threats such as hacking attempts, malware infiltrations, or other types of cyber-attacks. The general purpose of NID mainly features analyzing the incoming and outgoing data packets that cross the network. These packets contain data that can be analyzed for suspicious conformance to expected patterns or behavior. NID inspects these packets against a set of predefined rules or baselines of normal network activity by comparing their attributes, referred to as a network signature or statistical model. It is when it detects behaviour that is outside the norm, or finds a pattern matching known attack vectors, that it will alert or notify network administrators.

NID has been found particularly useful because it can enable real-time or near-real-time detection. It can enable an organization to know about the security incident as soon as possible, thus taking whatever response in a very timely manner to reduce any potential harm. Cyber-attacks are still growing in terms of sophistication, hence making the NID a very key component in maintaining sensitive data and network operation integrity. Traditionally, NID uses signature-based techniques whereby specific attack patterns would be matched with known threat signatures. However, traditional methods are limited by their dependence on predefined rules and may struggle to detect new or unknown threats, also called zero-day attacks. However, there are scenarios where zero-day attacks are unlikely, where we can use DNN based NID. In this respect, network intrusion detection nowadays has gained considerable developments especially with the help of machine learning and deep learning methods.

Recently, deep learning has emerged as a key solution in enhancing the capability of NID. Unlike previous traditional methods, which rely on pre programmed rules, deep learning models automatically learn complicated patterns from large datasets. These approaches are really good at picking up known and unseen attacks by spotting complex

relationships in the data that might be intuitive to human analysts. Thus, deep learning-driven NID systems can adapt to the changing landscape of threats, affording a more dynamic and proactive approach toward network defense. The role of NIDS has become increasingly important in modern-day network environments, where things have become increasingly complex with the increase of cloud computing, IoT devices, and distributed systems. This has consequently stirred an increased demand for more robust, flexible, and expandable solutions to secure these networks, which in turn drove deep learning as a way of improving NID effectiveness in detecting and classifying network intrusions.

2.2 Deep Neural Network (DNN)

A DNN is a sophisticated type of artificial neural network designed to model intricate and complex relationships within data, making it one of the most powerful tools in modern machine learning. Structured in multiple layers, a DNN consists of an input layer, several hidden layers, and an output layer, with each layer containing neurons that are interconnected, allowing information to flow from one layer to the next. The information initially enters the network through the input layer, where it is passed through the hidden layers, where extensive computations are performed. Each hidden layer processes the data in such a way that more abstract features are extracted as it progresses through the layers. For example, in tasks like image processing, the first few layers might identify basic patterns such as edges or textures, while deeper layers begin to recognize more complex structures, like specific objects or even entire scenes. Similarly, in natural language processing, early layers might detect individual words or phrases, while the deeper layers can discern the meaning of sentences or even entire paragraphs.

The operations performed at each neuron involve a combination of weights and activation functions, where the weights determine the strength of the signal passed between neurons, and the activation functions introduce non-linearity to help the network learn complex mappings. The learning process of a DNN is guided by backpropagation, a key algorithm that adjusts the weights of the network based on the error between the predicted and actual outcomes. This process continues iteratively during training, refining the network to improve accuracy over time.

DNNs have shown remarkable performance across a wide range of applications, including image classification, speech recognition, natural language processing, and even more complex tasks like game playing and autonomous driving. Their ability to model highly non-linear and complex data relationships stems from their depth, which allows them to learn hierarchical representations of data. However, this depth also comes with challenges. DNNs require vast amounts of data and computational power to train effectively, and they can be prone to overfitting, especially if the training data is not sufficiently diverse or if proper regularization techniques, such as dropout, are not employed. Despite these challenges, advances in computing power, efficient algorithms, and the increasing availability of large datasets have enabled DNNs to become the foundation of many cutting-edge

machine learning technologies, driving progress in artificial intelligence and transforming various industries in the process.

2.3 The Role of DNN in NID

In the field of NID, DNNs have emerged as highly valuable tools for identifying unauthorized access and malicious activities within network environments. Unlike traditional NID approaches, which depend heavily on predefined rules and known signatures, DNNs offer a more dynamic and flexible solution. Conventional techniques, relying on some predefined attack signature, usually cannot keep pace with cyber threats that are continuously evolving and make them less effective against sophisticated and novel or zero-day attacks. DNNs can learn extensively from raw data, which lets them identify intricate underlying features in network traffic that may point to kinds of suspect activity that cannot be previously known. In particular, the ReLU layers within DNNs contribute significantly toward enhancing generalization and thereby enabling the model to capture complex nonlinear relationships without overfitting. This robustness in generalization is very fundamental when it comes to dealing with the dynamic nature of cybersecurity threats because it empowers the DNNs with capabilities to adapt across a wide range of attacks other than those explicitly encountered during training.

The strong points of DNNs include the capability to handle and analyze large and complex datasets, as is usually the case in modern networks. They can handle a wide variety of features such as packet size, IP addresses, protocol types, among others, where they use multiple layers of neural processing to capture both low and high-level patterns in the data. This is layered by nature, where it enhances the capabilities of finding the deviation from the normal network behavior, mostly used by malicious intentions. Compared to traditional NID, DNN-powered systems are more capable of delivering accurate and comprehensive intrusion detection given that modern attackers' tactics keep changing in unforeseen ways.

DNNs have recently turned out to be very powerful for enhancing NID owing to their excellent generalization capability. Once trained, DNNs can detect patterns from both known and previously unseen network behavior with high accuracy, thus enabling the detection of a wide range of threats in complex and dynamic network environments. However, traditional DNNs work in a static mode and thus cannot adaptively learn. For continuous learning to take place in real-time environments, integration methods like online learning, DRL, or incremental learning methods are needed to allow the model to adapt and improve with new network data. Adaptiveness in this perspective will enable the adversary detection system to operate better with emerging threats in rapidly changing environments. Also, DNNs support both anomaly-based detection, focused on finding abnormal or unexpected patterns in network traffic, and misuse-based detection targeting known attack signatures. Together, they provide a strong, holistic defense mechanism. While cyber-attacks become increasingly complex, and networks are, too, DNNs offer powerful

and versatile tools in preventing cyber intrusions.

2.4 Understanding the Network Flow Data of Primary Dataset

We did our base work on the NF-ToN-IoT dataset, a comprehensive dataset provided by University of Queensland, with network traffic information including both benign and malicious traffic. This dataset can be considered as rich and comprehensive foundation for analyzing diversified network behavior; hence, it is very suitable to analyze a wide range of cyber threats. It does have the in-depth traffic data with several attack scenarios, which makes it imperative for ascertaining valuable information regarding potential vulnerabilities in networks. Besides, there is a broad range of transport layer protocols, such as TCP and UDP, supported by the NF-ToN-IoT dataset, which corresponds to a wide range of application-level activities occurring in the network. The dataset will include traffic from different protocols, which ensures that the operation of the network is as close to reality and diverse, simulating both legitimate user activities and malicious attacks. This diversity enables the development of robust models that can effectively detect intrusions across multiple network environments. It contains well-labeled instances of normal and attack traffic that make the dataset very comprehensive, both for training and evaluation; hence, enabling us to capture the important, subtle patterns and anomalies that are useful for improvement in the NID detection.

- IPV4_SRC_ADDR: It stands for the IPv4 address of the source device initiating the traffic. IPv4 addresses are unique identifiers used by a device to communicate over a network.
- L4_SRC_PORT: This is the source port number at an application layer used by an initiating device. In other words, every service or application is differentiated with a distinct port number, such as 80 for HTTP. These can be monitored for flow based on abnormal patterns.
- IPV4_DST_ADDR: The destination IPv4 address of the device that is receiving the data. This forms part of enabling the mapping of traffic flow and thereby helps identify devices that may be targeted or accessed.
- L4_DST_PORT: DST Port number is the port number on the Transport Layer of the target device. This tells which service or application the traffic is intended to reach. Analysis of this data facilitates the detection of security threats that may include 'Service targeted attack'.
- PROTOCOL: This is the application-layer protocol being used, which is identified numerically, such as HTTP, FTP. It serves as a good way of classifying traffic and understanding the different user activities, which is essential in identifying unauthorized or unusual applications.

- L7_PROTO: This is the application layer protocol utilized by traffic; it determines the type of application or service in use with the network. This is essential to identify the nature of the traffic, whether it is related to email, web browsing, or file transfers.
- IN_BYTES: This stores the total bytes transmitted by the source device in a session. This field is very important for consideration when analyzing inbound traffic; it will help mostly in finding traffic anomalies, such as a DoS attack.
- OUT_BYTES: The total amount of bytes transmitted from the source device. It helps to monitor possible data exfiltration since unusual outbound traffic might indicate unauthorized data transfers.
- IN_PKTS: This represents packets received by the source device. High packet counts with relatively small byte size may be indicative of scanning or probing activity.
- OUT_PKTS: Counts all the packets transmitted by the source device. This metric provides information about the amount of data transferred that may help in realizing strange or unusual increases in traffic.
- TCP_FLAGS: This represents the range of flags within the TCP header, for instance SYN, ACK, and FIN, which determine the state of the connection. Being able to keep track of TCP flags will help identify particular types of attacks. A very good example is the SYN flood attack, whereby many SYN requests are sent without completing the handshake.
- FLOW_DURATION_MILLISECONDS: Captures the total duration of the session in milliseconds. Analyzing flow duration can help distinguish between normal communication and malicious traffic, such as very short-lived connections that may indicate reconnaissance activities.
- Label: This is a categorical variable that classifies the traffic flow as either benign or malicious, allowing identification of normal behavior versus any possible security threats.
- Attack: A binary indicator that flags whether the traffic flow is associated with a known attack (1) or is benign (0).

2.5 The Role of DNN to Detect Stealthy Attacks

DNNs play a significant role in detecting stealthy attacks, which are often challenging for traditional NIDs to identify. Stealthy attacks, such as advanced persistent threats (APTs) and zero-day exploits, are designed to bypass conventional rule-based systems by disguising their behavior or mimicking legitimate traffic. Traditional detection methods rely heavily on known signatures and predefined rules, limiting their ability to detect novel or subtle attack patterns. DNNs, on the other hand, can learn from vast amounts of data

and identify patterns that may not have been explicitly programmed into the system, making them highly effective for spotting these elusive threats.

The strength of DNNs lies in their deep hierarchical structure, which allows them to automatically extract features from raw data at various levels of abstraction. This capability enables DNNs to identify hidden or subtle anomalies in network traffic that might indicate the presence of a stealthy attack. Furthermore, DNNs are highly scalable, enabling them to process large datasets and detect attacks in real-time, which is essential for modern NID dealing with massive amounts of network traffic. This adaptability and real-time processing make DNNs a powerful tool for enhancing the detection of stealthy attacks in network security.

2.6 Previous Work

Research works done with the aim of applying traditional machine learning and deep learning to network intrusion detection systems have been presented. Application-specific typical analysis has been performed with typical algorithms such as decision trees, support vector machines, k-nearest neighbors, DNNs, CNNs, and LSTMs to discover regularities that drive network traffic to malicious activities. Whereas classic machine learning has been very successful in finding known attacks, traditional methods tend to be difficult for generalization when the threats are new or evolve over time. Deep learning models show strengths in improving the detection based on automatic complex feature learning from raw data. This especially happens in the case of high-dimensionality data while uncovering latent patterns in it. However, most deep learning methods require substantial computational resources and large volumes of labeled data. This comprehensive survey that has been carried out by [1] presents great interests in GANs ever since the year 2014 in which it was introduced. The authors went on to look at the use of GANs in IDS and how they are currently being employed in this area of the research. Authors of the paper [2] have generated synthetic data using GAN and used DNN, CNN, and LSTM for classification. The results of their tests reveal that GAN+DNN is superior on the UNSWNB15 dataset. Authors from [3] proposed Projection-Based Adversarial Attack Generation Against Network Intrusion Detection where they utilized a traffic space generative adversarial network GAN to approximate distribution mapping between malicious traffic and benign traffic. DNN has become important in the case of classification in recent studies, and it is a fact that DNN can be used for Network Intrusion detection [4, 5].

More recently, studies have been conducted which have produced good results in applying AI to NIDS. The early attempts on applying machine learning models were restricted to conventional models, including: decision trees (DTs) [6], support vector machines (SVMs) [7]. In [8], a study was conducted using Naïve Bayes, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN) machine learning algorithms for intrusion detection. Results have been found in various papers for such datasets, such as the NSL-KDD and UNSW-NB15, by evaluating the performance of these algorithms with respect to accuracy and

efficiency in network threat detection. Nowadays, deep learning approaches are becoming more common in NIDS [9]. Zhong et al. [10] proposed a big data-driven deep learning system using tree architecture for intrusion detection. This improved the rate of detection for intrusive attacks compared to the individual learning methods used in earlier times. This proved that a combination of hierarchical data with deep learning could result in a better output. Haghighat et al. [11], introduced a deep learning-based intrusion detection system which, with the help of voting, gathered the best models to come up with more robust and accurate results, hence effectively taking advantage of ensemble learning against the diverse intrusion tactics. Yang et al.[12] presented the anomaly detection system using a tree structure-based approach. They integrated window sliding, detection strategy change, and model update in the locality-sensitive hashing-based iForest model. They illustrated this through extensive experiments on the KDDCUP99 dataset. The advantages of adaptive learning techniques in dynamic network environments were highlighted [13]. Considering that the nature of network data is hierarchical and requires analysis at several levels, authors of [14] proposed the HCA-based decision tree for network traffic data. This paper has not done an extensive comparison of the proposed HC-DTTSVM approach with a range of the latest network intrusion detection approaches, which might limit understanding of how it performs compared to other methods. The authors of [15] have designed a method using GAN to generate adversarial attack data in the context of deep learning and then come up with a robust IDS model, namely RAIDS against such kind of attacks. That clearly explains threats are also getting evolved along with how the defense mechanism also needs to be updated. The model using Convolutional Deep Belief Networks (CDBN) has been proposed by [16] for intrusion detection in wireless networks. CDBN models are computationally expensive, taking into account their complex architecture; much computational resources are in demand when training and inference procedures should be performed, representing one more trade-off between accuracy and computational efficiency. In [17], the authors proposed a novel hybrid model, which combined CNN and BiLSTM networks to extend the previous ideas toward enhancing network intrusion detection using both binary and multiclass classification tasks. The L2 regularizer and dropout technique were used in the training model, which achieved a balance of precision and generalization, showing the great potential of hybrid models with an average accuracy of 84.42% on NF-UNSW-NB15. Conventional deep learning methods are always puzzled by some challenging issues such as data loss or overfitting problems when coming to imbalanced datasets, which can be considerably addressed by GANs. However, GANs may be unstable and suffered from mode collapse during training the model. The authors in [18] designed the ensemble-based multi-layer GANs (EMP-GANs) to address these issues. Thus, maintaining the training stability and minimizing the chances for mode collapse, the proposed method can generate robust, diverse, and realistic synthetic data using multi-layers. Authors in [19] proposed an anomaly-based IDS system on IoT networks using a deep learning architecture, experimenting on the UNSW-NB15 dataset, where they achieved 84% accuracy. They apply GANs to create artificial data from minor-

ity classes, hence tackling the problem of class imbalance and improving model accuracy to 91% on the balanced dataset, which further confirms the efficiency of GANs in model performance improvement using synthetic data augmentation.

The traditional approaches of NIDS, based on either anomaly-based detection methods, face significant challenges when dealing with a high volume of network traffic and the potential novel or complex attacks. Considering these issues, some researchers started working with the integration of DRL, which would let NIDS learn how to handle emerging network threats autonomically. H. Benaddi et al. [20] presented the first substantial work on demonstrating the efficiency of RL in identifying anomalous network activities-a very important perspective of network security. They have focused on how reinforcement learning can help to address the main challenges in correctly identifying threats in complex and dynamic networks. M. He et al. [21] proposed a deep reinforcement learning-based NIDS, namely TA-NIDS. It indeed has enhanced the robustness and adaptability of NIDS through its focus on small-scale datasets that generate various interactions in order to enable systems to pay more attention to outliers with improved detection without relying on large fully labeled datasets. This framework showed the ability to realize high accuracy in detection using various datasets, which actually provided the assurance that the model would indeed transfer learning and stay tuned if applied in new environments. The ability to adapt is important in NIDS; after all, network traffic and attack vectors constantly change and therefore demand models that generalize well to new conditions.

The applications of DRL in NIDS are not restricted to the narrow domain of anomaly detection alone. T. Kim and W. Pak [22] proposed a new application of GAN together with an LSTM-DNN model to improve intrusion detection. Their GANs were used to generate additional training samples from misclassified data to augment model performance. Although that was a promising approach, hardware dependencies need to be tamed, and further improvements in accuracy are needed relative to traditional machine learning methods. All the same, this is a novel step toward developing more robust and adaptable NIDS models through such a combination of GANs and DRL. Another approach, by R. R. d. Santos et al. [23] investigated a machine learning-based NIDS, which, while of high accuracy, suffered from the issue of adapting to network traffic that changed with time. The system needed to be regularly updated and this was likely an issue for practical applications in real networks. By including reinforcement learning and transfer learning, the system improved in terms of long-term reliability and reduced computational requirements. The proposed model used a sliding window mechanism to reduce the need for constant updates, demonstrating that RL could maintain high accuracy without the frequent retraining typically required by conventional models. DRL has also been successfully applied to detecting botnets in IoT environments. For example, M. Al-Fawa'reh et al. [24] introduced MalBoT-DRL, a botnet detection model that adapts dynamically to changing malware patterns. Their model achieved very high detection rates on both MedBIoT and N-BaIoT datasets, especially in early stages of detection. Nevertheless, more stealthy malware like Bashlite remained undetected in this work; further, latency

associated with handling big volumes of network data hindered the completeness of this model. Despite all these challenges, MalBoT-DRL has demonstrated the potentiality of DRL in enhancing the detection capabilities of botnets and other IoT-specific threats where traditional NIDS may fall short.

The authors of [25] conducted a study on the robustness of botnet detectors to adversarial evasion attacks by using DRL. They developed a framework for generating automatically realistic adversarial samples to train more resilient detectors in order to make such systems resistant to unknown attack variants too. Their approach significantly outperformed state-of-the-art methods in particular regarding the maintainability of high detection performance without significant performance penalties even in adversarial settings. However, their work also emphasized the need for more varied datasets with which to train such models and possible computational overheads resulting from their approach. Other works have equally pointed out the advantages of DRL in enhancing the performance of NIDS. Specific work by J. Lansky et al. [26] and those presented in [27] demonstrated the efficiency of DRL against evolving cyber threats, highlighting how DRL algorithms can learn from network data to develop better detection and response against malicious activities. These works underlined the fact that, though traditional machine learning methods, including support vector machines and decision trees, had been helpful in network packet classification and attack detection, they were bound to be inefficient in scaling and adapting to new unknown attack patterns. The current complexity of modern networks, complemented by the unseen attacks, requires even more adaptive and intelligent solutions like DRL to provide timely and accurate intrusion detection [28]. The work in [29] further extended the DRL based approach and proposed a semi-supervised deep reinforcement learning model for abnormal traffic detection. Their model presented a mechanism of DDQN for efficiency in learning and an autoencoder that enabled feature reconstruction. This reduces network data in high volumes with limited information loss relevant to the analysis of essential characteristics. Dimensionality reduction is crucial for NIDS, which makes the model pay more attention to the most relevant features of traffic and gain more from both accuracy and computational efficiency. Their approach [**?**] combines DRL with semisupervised learning, which is considered a key advantage compared to supervised methods that require considerable labeled data, usually the bottleneck in practical applications.

Another contribution in DRL-based NIDS is given by X. Ma and W. Shi [30], who proposed a model that combined reinforcement learning with approaches in handling class imbalance problems in any dataset using techniques such as SMOTE. The methodology was carefully devised to overcome the common problem of unbalanced datasets faced in network intrusion detection, representing far more normal traffic variations compared to malicious variations. The proposed system was then validated on the NSL-KDD dataset and, as expected, had an overall improvement in the accuracy of detection, given that reinforcement learning could concentrate its efforts on the minority class that identifies malicious traffic. However, in their methodologies, the use of the NSL-KDD dataset as a

benchmark does not show the representativeness of network traffic flows, which indicates that multiple-dataset validation is necessary for robustness. DRL-based NIDS applications have gained widespread application since, along with growth in this field, it can handle voluminous data, learn complex flow patterns, and adapt to new cyber threats. DRL-based NIDS solutions significantly improve over traditional methods by improving detection accuracy and resilience from sophisticated attacks. Though the field is still in its development stage, the revolutionary role of DRL in network security is evident [31, 32]. Hence, DRL will play a key role in the future of intrusion detection systems. On the other hand, Study [33] developed two DRL-based techniques to improve the alert prioritization in intrusion detection, namely TD3-AP and SAC-AP, by resolving instability and limited exploration issues of existing DDPG-based techniques. In zero-sum adversarial setting, their models performed the best by reducing the defender loss in datasets such as MQTT-IoT-IDS2020 and CSE-CIC-IDS2018. Notable challenges pertain to the models' sensitivity to hyperparameters and to the high computational costs of their training. Furthermore, it was proposed in C. Rookard and A. Khojandi [34], the RRIoT framework that utilises DDPG reinforcement learning algorithm with LSTM layers to increase the accuracy in attack detection. Their approach was tested against IoT telemetry datasets and demonstrated enhanced interpretability through SAGE, uncovering the importance of features, beating state-of-the-art machine learning and reinforcement learning algorithms. With the emergence of the Industrial Internet of Things, it increased cybersecurity risks, hence more sophisticated IDSs are in urgent need. In this regard, S. Vadigi et al. [35] proposed a Federated DRL-based IDS that employs several agents with DQN logic across the network and maintains data privacy by retaining all data within the local environment, along with using an attention-weighted aggregation process. Their work was trained using the ISOT-CID and NSL-KDD datasets; these showed very promising performance in terms of accuracy, precision, and low false-positive rates, although this method is still facing issues with computational complexity and adaptation to the varied network environment. Another approach was by S. Yu et al. [36], where they proposed a DRL-based DC-IDS framework for IIoT, which utilized DQN with a Conditional Variational Autoencoder to handle the tasks of both known traffic classification and unknown attack detection. Experimental results were made on the TON-IoT dataset, and it showed the effectiveness of the system in detecting unknown attacks. Further scalability and real-time implementation remains to be explored.

Chapter 3

Methodology

In this chapter, we describe the development process of our intrusion detection model, transparently laying out the rationale of this paper, as illustrated in Fig.3.1. The proposed methodology is based on three distinct steps.
Database Formation: In this step, we collected four open source datasets from University of Queensland to build our DNN model.
Data Pre-processing: Some pre-processing is done before model development, in order to fed correct data to model to gain maximum out of it.
Model Development: In this step, we demonstrate the development of DNN model, reason behind choosing this algorithm, parameters and so on.

3.1 Dataset Formation

In this work, we have collected some open-source NIDS datasets provided by the University of Queensland, Australia. These include NF-ToN-loT, NF-BoT-loT, NF-UNSW-NB15, NF-CSE-CIC-IDS2018, NF-UQ-NIDS and NF-UNSW-NB15-v2. NF-UNSW-NB15 has a total of 1,623,118 number of data flows, out of which 4.46% are attack samples. NF-ToN-IoT has 1,379,274 data flows, out of which 80.4% are attack samples. NF-BoT-IoT has a total of 600,100 flows, out of which 97.69% are attacks. NF-CSE-CIC-IDS2018 consists of 8,392,401 data flows, of which 12.14% are attack samples, and NF-UQ-NIDS consists of a total of 11,994,893 records, out of which 23.23% are attack flows. Similarly, NF-UNSW-NB15-v2 includes a total number of 2,390,275 flows, out of which 3.98% are attacks. These datasets cover different kinds of attacks, such as DoS, DDoS, Reconnaissance, and Exploits, to target the security of both traditional and IoT networks. Features extracted from pcap files were used to classify the flows of data into their respective attack categories. We have used the NF-ToN-loT dataset as the primary dataset and the rest of the datasets for the multi-dataset validation strategy.

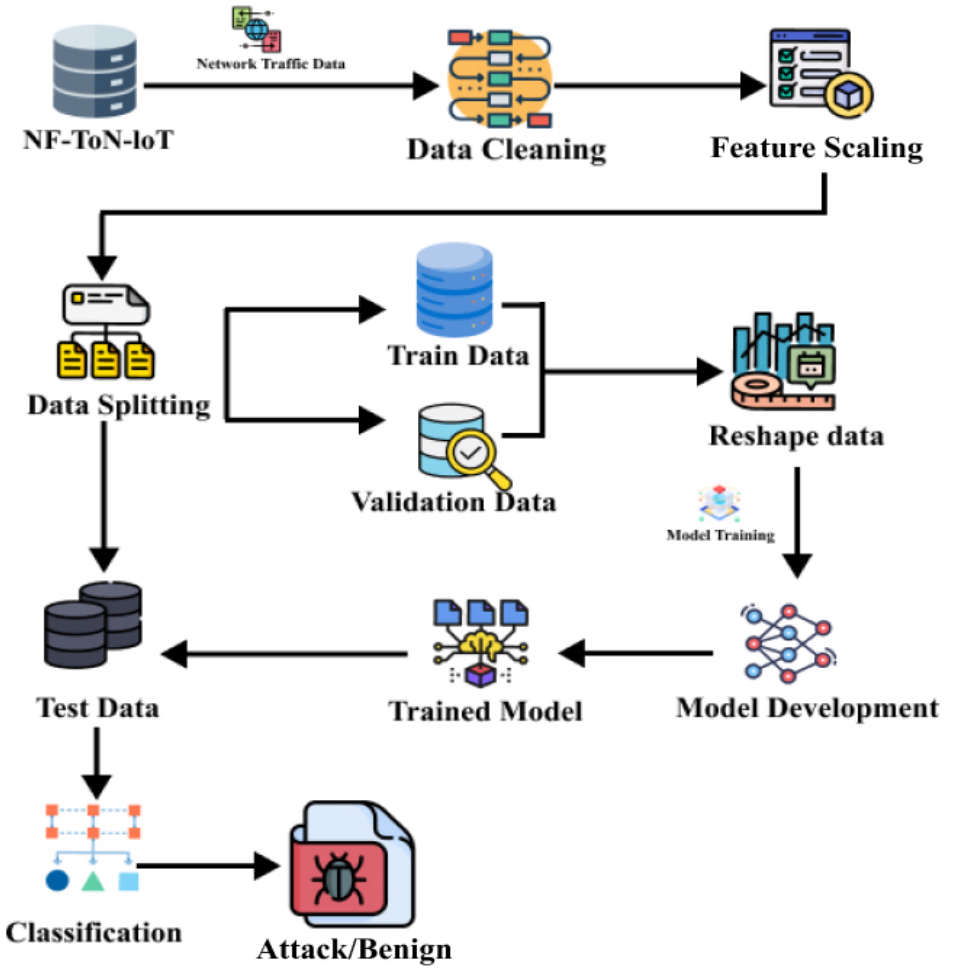


Figure 3.1: Framework of proposed network intrusion classification.

3.2 Data Preprocessing

The preprocessing steps outlined below play a vital role in preparing the NF-ToN-IoT dataset for training a DNN model specifically designed for classification tasks. Initially, the dataset is imported into a pandas DataFrame, which serves as a powerful tool for efficient data manipulation and analysis. Within this dataset, the columns 'IPV4_SRC_ADDR' and 'IPV4_DST_ADDR' are identified and subsequently dropped. These columns likely represent the source and destination IP addresses of the network flows, but they are deemed unnecessary for the classification task at hand. By removing these columns, the focus is shifted toward more relevant features of the network flow, which are critical for the DNN model's ability to accurately identify and classify different types of network intrusions.

Furthermore, the 'Label' column, which contains the target variable representing the classification categories, undergoes one-hot encoding. This transformation converts the

categorical labels into a binary format, making them suitable for training neural networks. One-hot encoding is essential for ensuring that the model can effectively interpret the class labels without imposing any ordinal relationships among them.

In addition to encoding, feature scaling is conducted using the StandardScaler method, which standardizes the features by ensuring they have a mean of 0 and a standard deviation of 1. This scaling technique is particularly important as it improves the convergence speed and overall performance of the DNN model, especially in cases where the features have different scales. By bringing all features to a similar scale, the model can learn more efficiently and make more accurate predictions.

After scaling, the dataset is reshaped to meet the requirements of the DNN model. This reshaping process involves adjusting the dimensions of the data to incorporate the appropriate number of channels, which is crucial for the model's architecture and functionality.

Finally, the reshaped data is split into training and testing sets, with 80% of the data allocated for training purposes and the remaining 20% reserved for testing. This division is critical as it enables the evaluation of the DNN model's performance on unseen data, thus providing a robust measure of its effectiveness and generalizability. Collectively, these preprocessing steps significantly enhance the overall effectiveness of the DNN model by appropriately formatting the data for training, eliminating irrelevant features, and facilitating a comprehensive evaluation of the model's capabilities.

3.3 Model Development

We propose a DNN based deep learning architecture for our network intrustion detection framework which achieves noteworthy performance on NF-ToN-loT, NF-BoT-loT, NF-UNSW-NB15, and NF-UNSW-NB15-v2. In Table 3.1 and Fig. 3.2, we demonstrated the DNN architecture of this work. In Table 3.2, detailed ablation study is shown on the NF-ToN-IoT dataset.

Table 3.1: Proposed DNN architecture.

Layer (type)	**Output Shape**	**Param**
dense (Dense)	(None, 64)	704
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 32)	2080
dropout_1 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 32)	1056
dropout_2 (Dropout)	(None, 32)	0
dense_3 (Dense)	(None, 2)	66
Total Params		**3,906 (15.26 KB)**
Trainable Params		**3,906 (15.26 KB)**
Non-trainable Params		**0 (0.00 Byte)**

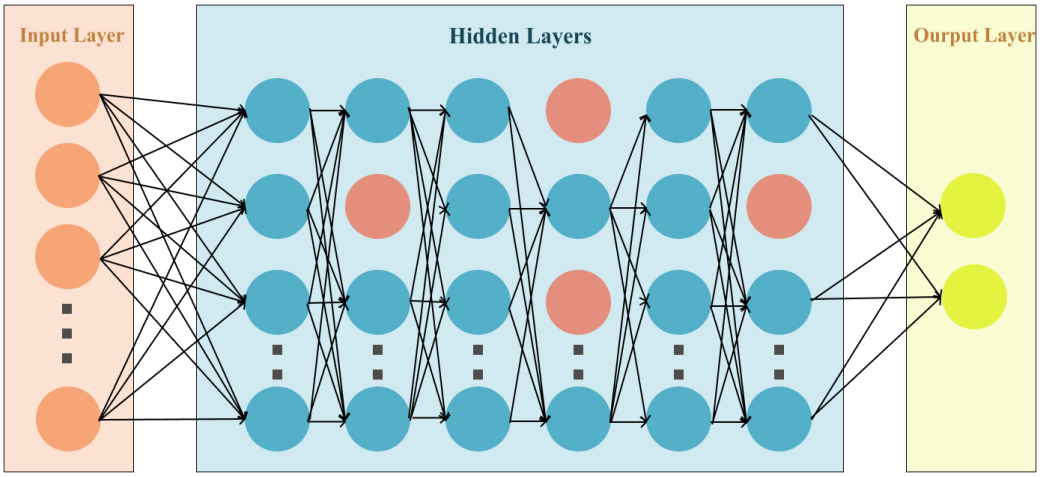


Figure 3.2: Custom DNN architecture.

Table 3.2: Ablation study of our model (30 epochs).

Optimizer	Dataset	Learning Rate	Accuracy	Precision	Recall	F1 Score
Adam	NF-ToN-IoT	0.001	0.9904	0.9894	0.9987	0.9940
Adam	NF-ToN-IoT	0.0001	0.9904	0.9895	0.9987	0.9941
Adam	NF-ToN-IoT	0.00001	0.9873	0.9850	0.9994	0.9921
RMSprop	NF-ToN-IoT	0.001	0.9844	0.9810	0.9999	0.9904
RMSprop	NF-ToN-IoT	0.0001	0.9870	0.9845	0.9994	0.9919
RMSprop	NF-ToN-IoT	0.00001	0.9869	0.9844	0.9995	0.9919
Adagrad	NF-ToN-IoT	0.001	0.9869	0.9844	0.9995	0.9919
Adagrad	NF-ToN-IoT	0.0001	0.9707	0.9649	0.9999	0.9821
Adagrad	NF-ToN-IoT	0.00001	0.8037	0.8037	0.9015	0.8912
SGD	NF-ToN-IoT	0.001	0.9867	0.9854	0.9982	0.9917
SGD	NF-ToN-IoT	0.0001	0.9811	0.9773	0.9997	0.9884
SGD	NF-ToN-IoT	0.00001	0.9217	0.9568	0.9452	0.9510

The DNN architecture for intrusion detection is designed with careful consideration of the number of layers, neurons, optimizer, and learning rate to balance complexity,

efficiency, and performance. The model consists of an input layer tailored to the number of features in the training data, followed by four hidden layers with decreasing neuron counts (64, 32, 32) to progressively condense and distill feature representations. Each hidden layer uses the ReLU activation function (Eq. 3.2) to effectively mitigate the vanishing gradient problem and accelerate learning. To prevent overfitting, dropout layers with a 20% dropout rate are included after each hidden layer, encouraging the model to learn robust features. The output layer euses a softmax activation function to produce a probability distribution over the classes, suitable for multi-class classification tasks.

Fully connected layers, also known as dense layers, are pivotal in processing the flattened output from preceding layers. Their primary function is to conduct high-level feature extraction and classification. Through dense layers, the network integrates the extracted features from earlier layers to make predictions, discerning between malware and benign activities within network traffic. Utilizing multiple dense layers allows the model to grasp intricate relationships between input features and output labels, thus bolstering its discriminatory capabilities. Let x represent the input data, w denote the weight, b stand for the bias, and σ symbolize the activation function that operates element-wise on the summation of weighted inputs and biases shown in Eq. (3.1).

$$\mathbf{y} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{3.1}$$

Dropout layers are a vital component in regularization, as they randomly deactivate a portion of input units during training. This strategy effectively combats overfitting and enhances the model's capacity for generalization. Overfitting arises when the model memorizes noise rather than discerning genuine patterns within the training data. By incorporating dropout layers, the model is compelled to acquire more resilient and comprehensive representations of the data, thereby refining its capability to identify malware activity in novel network traffic.

$$ReLU(x) = max(0, x) \tag{3.2}$$

The output layer produces the final predictions of the model, indicating whether the network traffic is classified as Malware or benign. The output layer employs an activation function (in this case, softmax) to transform the model's raw output into a probability score or binary classification decision. This allows the model to make informed decisions about the presence of malwar activity based on the learned representations from earlier layers.

The Adam optimizer, known for its adaptive learning rate and momentum properties, is selected for its ability to improve convergence speed and stability. A learning rate of 0.0001 is chosen to ensure gradual and precise updates to the model's weights, avoiding the risk of overshooting the minima of the loss function. The model is compiled using categorical cross-entropy as the loss function, ideal for multi-class classification, and accuracy, precision, recall and F1 score as the performance metrics. This combination of

architectural choices aims to create a robust, generalizable model capable of effectively identifying intrusions with high accuracy and minimal overfitting.

Chapter 4

Result Analysis

4.1 Rationale of Choosing Performance Metrics

4.1.1 Accuracy

Accuracy is the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances. It represents the overall correctness of the model.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

While accuracy does stand as a common performance measure for many machine learning models, it is misleading when considering imbalanced datasets such as those found in network intrusion detection. In this case, one class, considering normal network traffic will often dominate the dataset, with the other class, intrusions or malicious activity being very small by comparison. This means that a good model accuracy is achieved by the model when it classifies the instances of the majority class correctly while the model may fail to detect the minority class intrusions. This brings in a misplaced sense of reliability since, while the model may excel for the dominant class, it fails in the case of real threats. It becomes quite important to use other metrics beyond accuracy, including precision, recall, and the F1-score, which give further insight into the performance of a model. These other metrics consider the trade-off between true positives, indicating that intrusions actually exist and low false negatives and false positives, giving a better measure of how well the model will identify malicious activity across a network.

4.1.2 Precision

Precision, also called positive predictive value, is the ratio of true positive predictions to the total number of positive predictions. It tells how many of the predicted positive cases were actually correct.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

Precision is particularly critical in network intrusion classification due to the significant

consequences associated with false positives (FP). In a network security context, a false positive occurs when the system incorrectly flags benign activity as a threat, triggering an unnecessary security response. These false alarms can be highly disruptive, consuming valuable resources such as time, manpower, and computational effort. Moreover, a high rate of false positives can lead to "alert fatigue," where security teams become desensitized to alarms, potentially overlooking genuine threats. Precision, as a metric, reflects the proportion of correctly identified threats among all the alerts raised by the system. Therefore, a system with high precision ensures that the majority of alarms correspond to actual intrusions, minimizing unnecessary interventions and allowing security teams to focus on real, pressing threats. This reduces the overall operational cost and enhances the effectiveness of the intrusion detection system, ensuring a more secure and efficient network environment..

4.1.3 Recall

Recall is the ratio of true positive predictions to the actual number of positive instances. It measures how well the model captures all relevant instances.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}$$

High recall is crucial in intrusion detection systems because it directly impacts the system's ability to detect the vast majority of real threats, ensuring that potential intrusions are not overlooked. In the context of cybersecurity, missing actual threats can have devastating consequences, such as data breaches, unauthorized access, malware propagation, and denial-of-service (DoS) attacks, all of which can compromise the integrity, confidentiality, and availability of critical network systems. A system with low recall would fail to identify many genuine intrusions, leaving the network vulnerable to malicious activities that could go undetected for extended periods. This oversight can cause significant damage, both in terms of operational disruption and financial loss, as well as tarnishing an organization's reputation. Thus, achieving high recall in an intrusion detection system is essential for robust network security, as it minimizes the likelihood of false negatives, thereby enabling proactive responses to cyber threats and ensuring comprehensive protection.

4.1.4 F1 Score

The F1 Score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

The F1 score is an essential metric when evaluating models, especially in scenarios where there is a critical trade-off between precision and recall. In the context of network intrusion detection, this trade-off becomes highly relevant because the goal is not only to detect as

many threats as possible (high recall) but also to minimize the number of false alarms or false positives (high precision). A model with a high recall might catch most intrusions but could also trigger numerous false alarms, overwhelming security teams with benign traffic flagged as malicious. On the other hand, a model with high precision could minimize false positives but might miss several actual threats, compromising security. The F1 score provides a balanced measure by combining both precision and recall, making it a more reliable indicator of the model's performance in real-world applications. A high F1 score means that the model is successfully identifying intrusions while keeping the false positive rate low, thus achieving an optimal balance between threat detection and false alarm reduction. This is crucial for effective intrusion detection systems, where both catching malicious activities and maintaining operational efficiency are equally important.

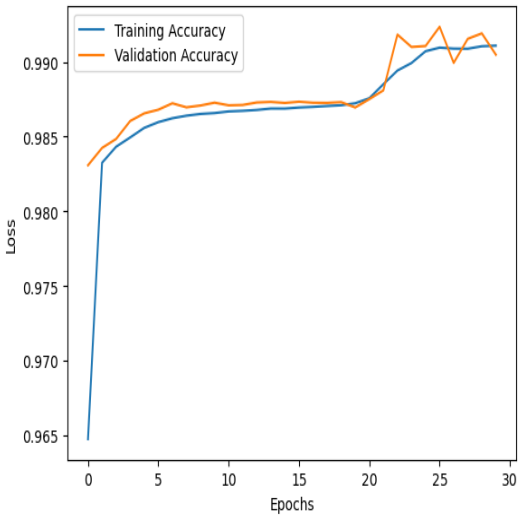
4.1.5 Confusion Matrix

A confusion matrix is a fundamental tool to describe the performance of classification models, giving one a total overview of how well a model differentiates between classes. The confusion matrix, being computed comparing actual labels against the predicted labels, brings all the results in a tabular format that outlines both correct classifications and the types of errors made by the model. This makes it much more valuable in tasks with numerous classes or imbalanced data, such as Intrusion Detection Systems, where it's crucial to identify malicious activity as correctly as possible, while keeping the number of false alarms as low as possible. The matrix is divided into four key components: true positives (correctly identified intrusions), true negatives (correctly identified normal activity), false positives (normal traffic mistakenly marked as intrusion and leading to false alarms), and false negatives (missing intrusions). The confusion matrix gives insight into the strengths and weaknesses of a model by visualizing the components of this process, enabling practitioners to fine tune the model for optimal balance between the detection of true intrusions and the reduction of false positives. This could be essential when trying to determine if an intrusion detection model is not only accurate but indeed practical for real-world deployment, where a proper balance between sensitivity and specificity becomes key in effective cybersecurity.

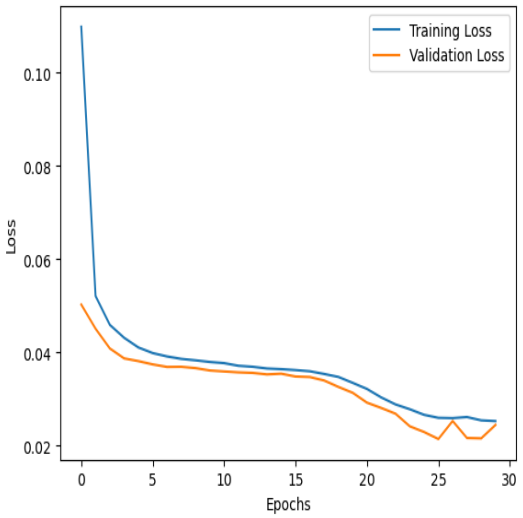
4.2 Performance Analysis

Upon completing model training, it demonstrated a remarkable 99% accuracy on both training and testing datasets shown in Fig. 4.1. Nevertheless, relying solely on accuracy may not provide a comprehensive evaluation of the model's performance. Given the nature of this classification task, it is imperative to consider additional metrics such as precision, recall, F1-score, and confusion matrix.

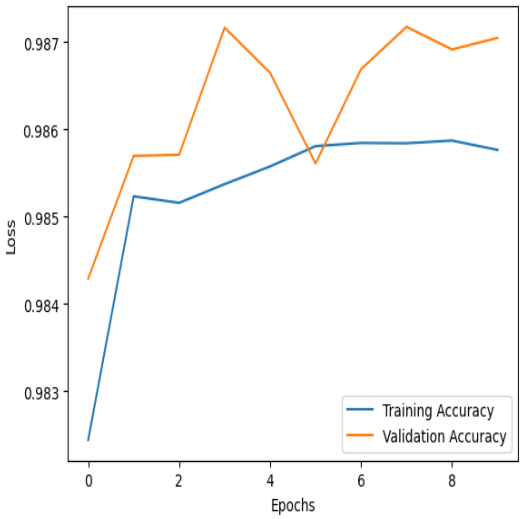
The Table 3.2 provides results from an ablation study conducted on a DNN-based intrusion detection system, highlighting the impact of various optimizers and learning



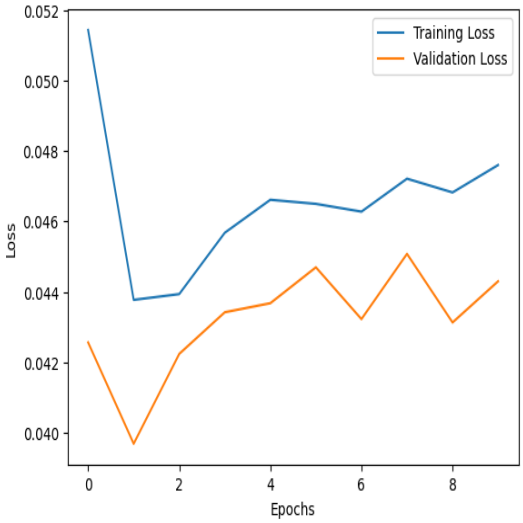
(a) Accuracy of Our Model (lr=0.0001, opt=adam)



(b) Loss of Our Model (lr=0.0001, opt=adam)



(c) Accuracy of Our Model (lr=0.0001, opt=Rmsprop)

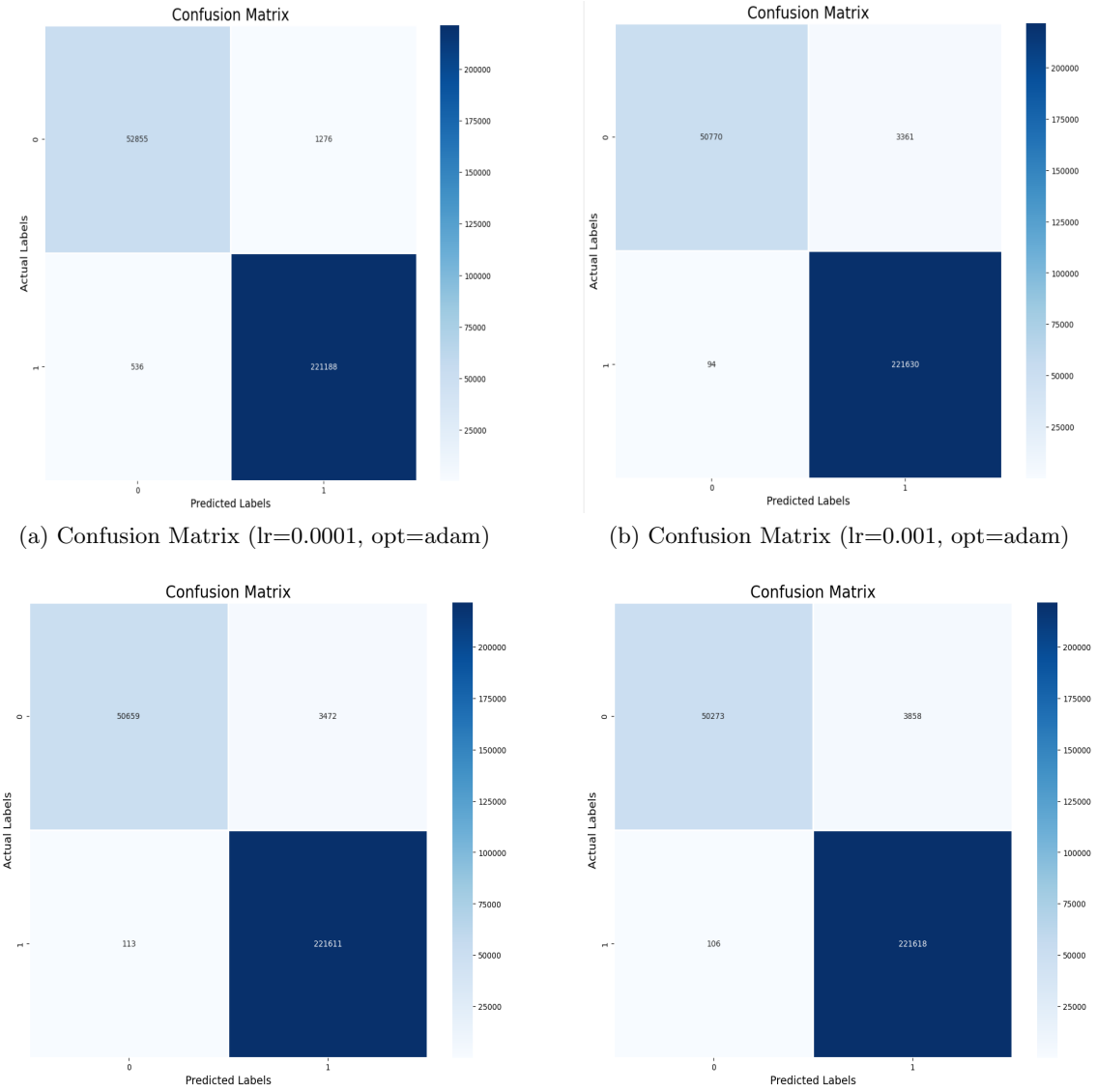


(d) Loss of Our Model (lr=0.0001, opt=Rmsprop)

Figure 4.1: The accuracy and loss metrics result for our model on NF-ToN-IoT dataset.

rates on the NF-ToN-IoT dataset. The optimizers tested include Adam, RMSprop, Adagrad, and SGD (Stochastic Gradient Descent), with learning rates set at 0.001, 0.0001, and 0.00001. Performance metrics evaluated are accuracy, precision, recall, and F1 score. From the table, it's evident that the Adam optimizer yields consistently high results across all metrics, particularly at a learning rate of 0.0001, suggesting an optimal balance between learning speed and model performance. In contrast, the performance drops significantly for Adagrad and SGD when the learning rate is decreased to 0.00001, as seen in the substantial declines in accuracy, precision, recall, and F1 score. This study demonstrates how optimizer choice and learning rate tuning are crucial for optimizing model performance,

with Adam appearing as the most effective optimizer for this specific intrusion detection task on the NF-ToN-IoT dataset.



(a) Confusion Matrix (lr=0.0001, opt=adam)

(b) Confusion Matrix (lr=0.001, opt=adam)

(c) Confusion Matrix (lr=0.0001, opt=rmsprop)

(d) Confusion Matrix (lr=0.001, opt=rmsprop)

Figure 4.2: Confusion matrix on NF-ToN-IoT dataset

A confusion matrix is a performance measurement for machine learning classification problems where output can be two or more classes. The confusion matrix of our proposed model. In Fig. 4.2, we illustrated the confusion matrix of our model on NF-ToN-IoT Dataset. A set of experiments are done to ensure contribution. Our proposed model is evaluated against previously used alternative neural network architectures using NF-ToN-IoT Dataset, and the findings are presented in Table 4.1 and Fig. 4.3. It is to be noted that the Fig.4.3 is nothing but the graphical representation of the Table 4.1.

As seen in Table 4.1, our model demonstrates a crucial 2-3% improvement over state-of-the-art approaches in NID. This enhancement is significant, much like the Six Sigma

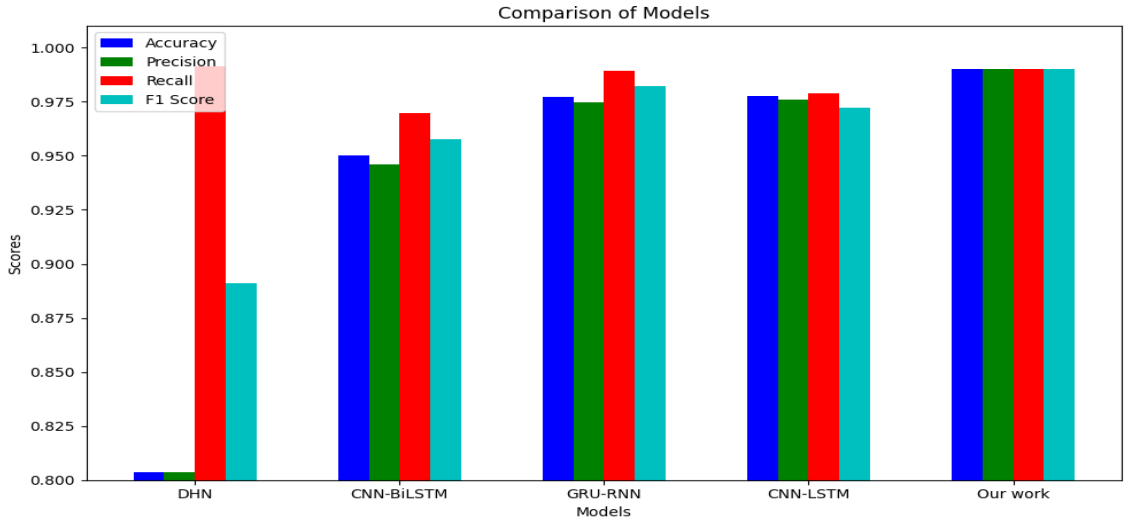


Figure 4.3: Performance comparison with previously used models

Table 4.1: Comparison of the performance among most used models from previous studies.

Model	**Accuracy**	**Precision**	**Recall**	**F1 Score**
DHN [37]	0.8038	0.8038	0.9912	0.8912
CNN-BiLSTM [17, 38, 39]	0.9501	0.9460	0.9698	0.9577
GRU-RNN [40, 41]	0.9772	0.9749	0.9894	0.9821
CNN-LSTM [42], [43]	0.9775	0.9758	0.9788	0.9723
Our work	0.99	0.99	0.99	0.99

methodology's focus on incremental gains leading to near-zero defects. A 2% increase in precision, recall, F1 score, and accuracy means fewer false negatives and positives, drastically improving operational efficiency. It allows security teams to focus on genuine threats and reduces the time spent on false alarms. Moreover, this advancement enhances user trust and data privacy, providing a stronger, more adaptive defense against evolving cyber threats. While working with network security, our main focus is to reduce errors. As per the Table 4.1, we can say that a substantial increase over state of art is a notable contribution.

Table 4.1 presents a comparison of the performance metrics for several deep learning models as used in previous studies, with a learning rate (lr) of 0.0001 and the Adam optimization algorithm using NF-ToN-IoT Dataset. The table lists different models including Deep Hierarchical Network (DHN), CNN-BiLSTM, GRU-RNN, CNN-LSTM, and the propoed model. It shows that our model performs exceptionally well with an accuracy of 0.99, precision of 0.99, recall of 0.99, and F1 score of 0.99, suggesting it may be the most effective among those listed.

Method	Accuracy	Precision	Recall	F1 Score
No Synthetic	0.99	0.99	0.99	0.99
SMOTE	0.98	0.99	0.99	0.98
Borderline SMOTE	0.95	0.96	0.95	0.95
ADASYN	0.95	0.96	0.95	0.95

Table 4.2: Performance metrics for various oversampling methods.

The provided table (Table 4.2) presents a comparative analysis of various oversampling methods applied to the NF-ToN-IoT dataset. The performance of each method is evaluated using four metrics: Accuracy, Precision, Recall, and F1 Score. The results indicate that the "No Synthetic" method, which does not employ any oversampling technique, achieves the highest scores across all metrics, suggesting that the dataset may not have a significant class imbalance issue. However, the performance of the oversampling methods, SMOTE, Borderline SMOTE, and ADASYN, is comparable to the "No Synthetic" method, indicating that they are effective in handling potential class imbalance and improving model performance. Among the oversampling methods, SMOTE and ADASYN demonstrate slightly better performance than Borderline SMOTE in terms of Recall and F1 Score, suggesting their effectiveness in addressing class imbalance and improving the model's ability to correctly identify instances from the minority class.

Table 4.3: Comparison of the performance among other benchmark datasets

Dataset	Accuracy	Precision	Recall	F1 Score
NF-ToN-loT [44]	0.99	0.99	0.99	0.99
NF-BoT-loT [44]	0.9902	0.9911	0.9989	0.9950
NF-UNSW-NB15 [44]	0.9861	0.9776	0.9727	0.9847
NF-CSE-CIC-IDS2018 [44]	0.9915	0.9907	0.9394	0.9643
NF-UQ-NIDS [44]	0.9832	0.9799	0.9473	0.9633
NF-UNSW-NB15-v2 [44]	0.9968	0.9284	0.9786	0.9528

Achieving high accuracy on a single dataset is not sufficient to demonstrate the robustness of a model. Robustness entails the model's ability to maintain its performance across various datasets and under different conditions. To thoroughly evaluate the robustness of our model, we retrained it using multiple diverse datasets: NF-BoT-IoT, NF-UNSW-NB15, NF-CSE-CIC-IDS2018, NF-UQ-NIDS and NF-UNSW-NB15-v2. The results of this comprehensive evaluation are presented in Table 4.3, showcasing our model's noteworthy performance across these datasets. This approach ensures that our model is not only performing well in the primary dataset, but is generalizable and reliable in different scenarios.

Chapter 5

Discussion

5.1 No Synthetic vs Synthetic in the Context of Proposed NID

In Table 4.3, the performance of our proposed DNN model stands out across all key performance metrics such as accuracy, precision, recall, and F1 score, achieving a notable weighted value of 0.99 for each. This remarkable result highlights the model's effectiveness in detecting both normal and malicious network traffic. However, it is crucial to acknowledge that these results are based on an ideal scenario, where the model was trained on benchmark datasets that provided high-quality and large volumes of data. While these datasets enabled robust model training, the situation may be drastically different in real-world environments, where the availability of large, well-labeled datasets is often limited.

In practical scenarios, organizations may not always have access to vast amounts of high-quality data to train a model, and this can introduce several challenges. DNNs are known to excel when trained on large datasets because their layers are designed to capture complex patterns in the data. However, when the data is limited or of low quality, the performance of these models can degrade significantly. One major issue that arises from this is data imbalance, where certain classes such as malicious traffic are underrepresented in the dataset. This imbalance can cause the model to become biased toward the majority class (normal traffic/ benign), reducing its ability to detect less frequent but critical intrusions.

In our study, the benchmark datasets provided sufficient data for training, and as a result, our model performed exceptionally well without needing specific techniques for class balancing. However, this might not always be the case in real-world applications, where the presence of minority classes (such as various forms of cyber-attacks) may be much lower than normal traffic. In such cases, the performance of the model could hamper, potentially leading to misclassification of intrusions, especially when the data imbalance is not addressed.

To address the data imbalance issue in more realistic conditions, we incorporated oversampling methods such as SMOTE, borderline SMOTE, and ADASYN to generate

synthetic data for the minority class. These techniques are widely used to mitigate the effects of class imbalance by creating additional synthetic samples of underrepresented classes. Upon examining the results in Table 4.2, we observe that for our primary dataset, the performance of the model after applying SMOTE remained nearly identical to the scenario where no synthetic data was used, maintaining the high performance across all metrics. This indicates that for our dataset, SMOTE was effective in addressing the class imbalance without negatively impacting the model's ability to distinguish between normal and malicious traffic.

However, the results for both borderline SMOTE and ADASYN reveal a slight downgrade in performance. The accuracy, precision, recall, and F1 score dropped to 0.95, 0.96, 0.95, and 0.95 respectively. Although these values still indicate strong performance, the decline suggests that these oversampling techniques may have introduced some noise or overfitting, particularly in cases where the synthetic data created by these methods did not align well with the actual distribution of the minority class in the dataset. This highlights the complexity of applying oversampling methods, as they may not always result in performance improvements and can sometimes even degrade the model's accuracy, especially when the data does not match real-world conditions.

In real-world NID deployments, the quality of the dataset becomes even more critical. The ability to accurately detect intrusions relies heavily on the quality of the training data. Therefore, when using DNNs for NID, it is essential to take the minority classes into account during the training process. If the minority classes, such as malicious traffic, are not properly represented or are overshadowed by the majority class, the NID could become biased, favoring the detection of normal traffic while missing critical attack patterns. This underlines the importance of using appropriate techniques to balance the dataset and ensure that the model remains sensitive to all types of traffic, even in the face of limited or imbalanced data.

In summary, while our proposed model demonstrates outstanding performance under ideal conditions, practical deployment in real-world environments must consider the limitations posed by data quality and imbalance. Methods like SMOTE can be useful, but their effectiveness may vary depending on the dataset. Continuous attention to the distribution of minority classes and careful selection of oversampling techniques are key to ensuring that NID systems remain effective in diverse, real-world network environments.

5.2 Potential Applications of the Proposed System

Our lightweight NID stands out for being a practical, high-performance solution to securing network infrastructures in real-world environments. Some concrete scenarios in which the proposed system can be an optimal choice for cybersecurity needs are described below.

5.2.1 Application in Startup Networks

Starting from just about 10 and up to 250 employees, early-stage startups may often have stretched resources. The startups necessarily got to be focused on growth and innovation, which leaves very little scope for bringing in a dedicated IT security team. Many start-ups by necessity operate within cloud environments or from a hybrid version owing to the cost.

Our lightweight NID is particularly suited to startups since its computational overhead is very low, along with easy deployment. For instance, a sensitive customer data-handling tech startup might leverage our NID in hardening its network against rapid scaling by swiftly detecting new vulnerabilities introduced by additional services or more traffic. This makes sure that our system ensures threat detection metrics to an assurance level of 99% accuracy, maintained by the firm for startups not looking to invest highly in infrastructure securities, therefore making the NID a perfect fit where every second and dollar counts.

5.2.2 Deployment in Small to Medium-Sized Enterprises (SMEs)

Small to medium-sized enterprises (SMEs), which often lack the dedicated cybersecurity teams, indicate a different challenge. The SME may fall victim to ransomware attacks for which timely detection of malicious traffic can significantly reduce the impact. Now, considering a medium-sized e-commerce company that processes a lot of customer transactions, this may be done with the use of our NID by which its network will be analyzed for suspicious patterns showing possible phishing or DDoS attacks.

In this case, our NID ensures that it produces minimal false positives and false negatives to ensure continuity of valid transactions. At the same time, it effectively catches threats in real time. For SMEs, it is simple and automated enough not to require much human intervention, hence freeing employees to attend to the core activities of the business.

5.2.3 Integration with Cloud Service Providers (IaaS)

Cloud service providers have to juggle a delicate balance, ensuring client security and scalability while managing resources, especially in IaaS. Our NID has an exclusive advantage in these environments due to its lightweight design that guarantees scalability without compromising performance. Think of a cloud provider managing sensitive data from the government or financial institutions of different clients. This uniform yet flexible application of security across infrastructures becomes crucial, especially in balancing such a scale with the client-specific security needs.

For this, our NID can be deployed at different entry points to ensure it covers all the entry and exit points. Once there is the need for a provider to increase the scale of the services for a new client who may have very firm high-security standards in the shortest time possible, then our system with low resource utilization allows for quick deployment without increasing the overhead significantly. The 99% accuracy across all the performance metrics further ensures that even at the provider level, sophisticated threats will not cause much damage.

5.2.4 Securing Telecommunications Companies and ISPs

Telecommunications companies and Internet Service Providers (ISPs), which manage extensive infrastructure and serve millions of users, are prime targets for cyberattacks. These companies need a system that can handle high volumes of data without degrading network performance. Our NID's low-resource footprint makes it ideal for such environments.

For instance, an ISP might face a large-scale DDoS attack targeting its DNS servers. In this scenario, the speed and efficiency of our NID in detecting and mitigating the threat would be crucial in preventing service outages for its customers. Unlike bulkier systems that might struggle with the influx of data, our lightweight model can efficiently process large-scale traffic, ensuring the continued smooth operation of essential services. The system's ability to scale to meet the demands of such large networks further cements its role as a critical security tool in this domain.

5.2.5 Protection in High-Security Sectors

Industries like healthcare, finance, and critical infrastructure are particularly susceptible to advanced cyber threats, such as polymorphic malware and stealthy attacks that target multiple points in the network. A hospital handling sensitive patient data, for instance, might be a prime target for ransomware, where threat detection and mitigation speed can make the difference between a contained incident and widespread data compromise.

In such cases, our NID can be deployed across various layers of the hospital's network, from servers storing patient records to devices managing critical equipment. The system's 99% accuracy rate is essential in detecting threats that evolve to evade traditional security measures, ensuring that even the most sophisticated attacks are caught in real-time. Its lightweight nature also means that it can be deployed on devices with limited computational resources, such as IoT devices that manage patient monitoring systems, without impacting the performance of life-critical applications.

5.3 Deployment Consideration for the NID

The deployment of the proposed lightweight DNN model within a network infrastructure, particularly in an IoT-rich environment, is crucial to achieving real-time network intrusion detection without placing a heavy burden on system resources. Given the increased production of millions of IoT devices, the proposed model is designed to operate efficiently in edge-computing environments. Its lightweight nature ensures that it can be deployed on devices with limited computational power, such as routers or IoT gateways, while maintaining high accuracy. For cloud-based deployments, the model can be integrated into network monitoring systems that leverage scalable infrastructure, such as Google Cloud or AWS, to handle large-scale network traffic. Based on literature and domain knowledge, in Table 5.1, we have presented a hypothesis for hardware requirement for successful deployment of NID which includes edge devices, server deployment and cloud infrastructure.

Table 5.1 provides an overview of the deployment strategies and hardware requirements for the proposed lightweight DNN model, highlighting its flexibility in various network environments. The table categorises the deployment into three types: edge devices, server deployment, and cloud infrastructure, each with its recommended hardware, suitable use cases, and key advantages.

Table 5.1: Deployment strategy and hardware requirements (approx.).

Deployment Type	**Recommended Hardware**	**Use Case**	**Advantages**
Edge Devices	2GB RAM, Multi-core processor (e.g., ARM Cortex-A53)	IoT gateways, Routers, Local traffic analysis	Low latency, Local processing, Minimal resource usage
Server Deployment	8GB RAM, Intel i5 or equivalent processor, basic GPU	Medium-sized networks, Higher traffic volume	Improved throughput, Moderate latency, Suitable for larger networks
Cloud Infrastructure	16GB RAM, High-performance GPU (e.g., NVIDIA T4)	Large-scale IoT networks, Enterprise-level, Real-time detection	Scalable, High-performance, Continuous model updates

For IoT gateways and routers, the model can be deployed on devices with minimal computational power, such as those equipped with 2GB RAM and a multi-core processor (e.g., ARM Cortex-A53). This makes it well-suited for local traffic analysis in IoT environments where low latency and minimal resource usage are critical. The advantage of deploying on edge devices is that it allows for local processing, which reduces the dependency on external servers and ensures faster response times for detecting network intrusions. In medium-sized networks with higher traffic volumes, server deployment becomes more appropriate. Here, the model requires a system with at least 8GB of RAM, an Intel i5 (or equivalent) processor, and potentially a basic GPU to handle increased network traffic. This deployment type offers improved throughput and moderate latency, making it suitable for larger networks where faster, more robust processing is needed, but without the need for high-end infrastructure. For enterprise-level and large-scale IoT networks, cloud deployment is recommended. This option leverages high-performance hardware, such as systems with 16GB of RAM and a powerful GPU like the NVIDIA T4, to handle real-time detection across vast network traffic. Cloud infrastructure allows for scalability, continuous model updates, and high performance, making it the most capable solution for environments with extensive network traffic and the need for real-time monitoring. Each deployment type is designed to meet specific needs based on the scale of the network and available resources, ensuring that the proposed model can be efficiently integrated across different network infrastructures. For our model, In table 5.2, we have showed the minimum hardware requirement to compile the NID model. In order to run GANs or bigger

Table 5.2: Minimum hardware requirements

Component	**Specification**	**Remarks**
Processor	AMD Ryzen 5 3500U (4 cores, 8 threads)	Suitable for both training and inference, though faster CPUs can help decrease training duration.
RAM	8GB	Sufficient for handling network data and training small to medium-sized models. Additional memory is advised for larger datasets.
Graphics	Radeon Vega 8 GPU	Integrated GPU provides basic functionality, but a dedicated GPU is recommended for enhanced performance.
Total Graphics Memory	Can use up to 2 GB of the system's RAM	Memory is dynamically allocated from the system RAM for graphical processing, shared between tasks and display.

datasets, better hardware setup is needed.

5.4 Reason Behind Successful Classification of our NID

Our proposed DNN model for Network Intrusion Detection achieves its success mainly because of its lightweight and highly effective architecture, designed for precision and efficiency in the detection of sophisticated and stealthy cyber threats. This varies from further complicated models, which may involve high computational complexity. With a simple structure, this model comprises the necessary dense, dropout, and activation layers. These components work together in an optimization of model performance, along with computational demands, balancing accuracy and efficiency in their use.

Underlying the architecture is the core position of dense layers that allow the model to learn complex relationships and patterns from raw network traffic data. These layers make it possible to detect subtle indicators of intrusion, mostly not spotted by traditional systems set up for its detection. The nature of the attacks have been changed, where an attacker would use different sophisticated means to get past a standard defense. It will be worth applying our model to find the latent and complex pattern of the data represented by dense layers, often indicative of emerging threats such as polymorphic malware and stealthy intrusions. This capability is very key in today's cybersecurity landscape, where the ability to adapt to new forms of attacks is crucial.

Adding dropout layers to the network essentially introduces regularization to make this model more robust. This technique prevents overfitting, which can easily occur with big and complex data. Overfitting is a well-known problem in machine learning where a model becomes overly specialized in its training data and fails to generalize well on new, unseen data. In that, dropout ensures that during training a model does not depend too much on any one set of neurons but rather makes the model learn more diverse and general features. This regularization mechanism definitely plays an important role in maintaining the effectiveness of the model across different network environments and various types of cyber threats that have been facing reliable performance in real-world deployment.

Another important layer in our model is the activation layer, relying on special functions such as ReLU. ReLU is an example of an activation function providing nonlinearities to the model which is crucial to deal with real-world data that involves complex and nonlinear relationships. But in the domain of intrusion detection on networks, attacks do not proceed according to a simple, linear pattern of behavior; traditional models try hardly to recognize every advanced threat evolving over time. Since we have used ReLU, our DNN model will be good at learning nonlinear patterns and hence detect even the most complex and evolved attack vectors-for example, polymorphic threats that continuously change their characteristics in order to outrun the detection. That is to say, the model would maintain its adaptability and efficiency in countering a wide range of cyber threats.

The salient feature of our DNN model is its high accuracy and precision without the use of heavy computational burdens. Its simplicity in architecture enables the network

to turn in excellent performance in intrusion detection tasks without necessitating very high processing power. This makes the model quite suitable for real-time network environments where fast decision-making is needed to reduce the impact of any potential attack. In other words, its lightweight nature ensures the model can be swiftly deployed in time-sensitive environments, whether it is a large-scale industrial network or even a small enterprise setup, through the rapid detection of threats without throttling the overall network operations.

This again makes the model scalable to a wide variety of network infrastructures. Be it a small enterprise network or a large industrial system, the lightweight design of the model ensures that it can be applied in different environments with minimum adjustments. Its ease of computation easily scales up the model to be practical in organizations that may have variable network complexities and resources. This will be particularly important in the dynamic cybersecurity landscape of today, where no network is unpenetrable to attacks, some even sophisticated, irrespective of its size from the smallest to the largest.

We made relative comparisons with other state-of-the-art models such as CNN+BiLSTM and GRU+RNN, DHN, CNN+LSTM in Table 4.1 and Fig. 4.3, where our DNN model was found outperforming all, thus being robust and efficient. While these models may possess their own effectiveness, they do need many computational resources for processing and generalization capability on new, unseen data. Meanwhile, our model with optimized architecture not only grants a better detection rate but also much lower computational overhead, thus making it more practical and efficient for any resource-limited real network environment.

We have finally applied advanced oversampling methods, such as ADASYN and SMOTE during model validation to handle the challenge of unbalanced datasets which is considered as a common problem in intrusion detection. It is actually the architecture that truly makes our model powerful. These oversampling techniques contribute to enhancing the performance of the model in real-world scenarios regarding unbalanced data distributions, where malicious traffic is usually much less frequent than normal traffic. It is, however, in the underlying architecture that the difference actually comes. Based on a lightweight, efficient, and scalable architecture, our DNN model presents an innovative approach to intrusion detection within the network with an emphasis on high detection rates, low computational overhead, and adaptability to modern cybersecurity challenges.

5.5 Zero Day Attack Detection in NID

Our model is based on Deep Neural Network-based Network Intrusion Detection. Our proposed method follows the traditional deep learning approach for the classification of incoming and outgoing traffic in a network into attack and non-attack categories. Although DNNs generalize to some degree and work well regarding identification of known attack patterns, finding completely new unseen attacks called zero-day attacks are challenging for traditional deep learning approaches, especially when those were not part of the training

data. One of our main goals was to develop a solution that consumes fewer computational resources, but this becomes a challenge because the majority of deep learning approaches depend heavily on patterns learned from labeled data.

Some recent works [45, 46, 47] provide evidence that machine learning and, in particular, deep learning may detect zero-day attacks in some cases when the latter is very similar to the characteristics of the training data. In our work, ReLU activation introduces non-linearity into our system, which might help generalize it at least partially. It is still hard to confidently claim that our traditional deep learning approach can effectively detect zero-day attacks in NID as per the literature. For accurate detection of zero-day attacks, DRL based models are suggested.

Hence, our approach is practical for SMEs where we trade off between effectiveness and lower computational demands. The SME sector cannot afford high-powered computational systems; therefore, our solution has to provide robust security in the context of low computational capacity. Though it will not catch all zero-day attacks, the likelihood of zero-day attacks hitting SMEs is low since cyber criminals are focused on known vulnerabilities and attack patterns most of the time. With this approach, the cost of protection against common threats is affordable, making it an asset to businesses with smaller budgets.

Chapter 6

Conclusion

The current research study is a significant improvement in the network intrusion detection method as it focuses on the development of a lightweight DNN model for rapid and efficient detection of stealthy, hidden intrusion attempts. It performed well with a weighted accuracy of 99%, whereas precision, recall, and F1-score were also at 99%. These results are showing that this model is able to catch all various kinds of sophisticated threats while minimizing the common mistakes, such as false positives and negatives, that always seem problematic in traditional systems. This improvement, by a wide margin over prior models, is indicative of its more sophisticated capability for catching intrusions that conventionally bypass detection. Excellence in these key performance areas ensures that the model will be reliable and robust in the real network environment where the correct identification of normal versus harmful behavior is of prime importance. This makes the proposed DNN model one major step toward the improvement of network intrusion detection effectiveness.

6.1 Limitation

Our current work is grounded in traditional deep learning methods, which, while effective for certain tasks, does not lend itself to adaptive modeling. Adaptive modeling is a dynamic approach that can adjust to changing environments and threats in real-time, making it particularly well-suited for frameworks based on DRL. However, implementing DRL comes with significant challenges, including the need for extensive training time and substantial computational resources, which are beyond the scope of our current work. In contrast, our focus has been on developing a lightweight model that offers efficient performance for NID without the heavy computational overhead typically associated with adaptive systems. In future research, we intend to explore DRL-based approaches for NID, aiming to harness their adaptability while addressing the resource constraints. DRL based NID is suitable for detecting zero-day attacks because of adaptive learning feature which is a limitation of our deep learning based NID. Additionally, based on prior studies, we have observed that GAN-based oversampling techniques are particularly effective in addressing

data imbalance issues. However, due to the hardware limitations in our current setup, we were unable to conduct a comparative study using GAN-based methods for data balancing.

6.2 Future Work

Looking ahead, the future scope of this research is promising and multifaceted, particularly concerning the integration of the proposed DNN model with real-time intrusion detection systems. Such integration could significantly enhance the model's effectiveness in responding to threats as they arise, thereby improving overall network security. Additionally, further optimization of the DNN model can be achieved through advanced techniques such as transfer learning and ensemble methods, which have the potential to boost model performance by leveraging knowledge from pre-trained models and combining the strengths of multiple algorithms. Furthermore, the literature highlights GAN as a powerful oversampling technique that could outperform the methods explored in this research. However, the choice not to include GAN was primarily due to hardware limitations; future researchers who are interested in employing GAN for oversampling must ensure they have access to sufficient computational resources, as GANs require substantial processing power, especially when working with large benchmark datasets. In addition to this, expanding the dataset to incorporate a wider array of diverse and complex attack scenarios will be essential for enhancing the model's robustness and adaptability to the ever-evolving landscape of cyber threats. By laying this groundwork, the research paves the way for the development of more resilient cybersecurity solutions that can effectively safeguard critical data and infrastructure within increasingly sophisticated digital environments. This holistic approach not only addresses current limitations but also prepares for the challenges of tomorrow, ensuring that cybersecurity measures remain effective in a dynamic threat landscape. Also, deploying the NID in real life systems can be done in future by researchers and necessary information regarding the deployment has been discussed here in this work.

Dataset Obtained

1. https://staff.itee.uq.edu.au/marius/NIDS_datasets/

References

[1] A. Dunmore, J. Jang-Jaccard, F. Sabrina, and J. Kwak. A Comprehensive Survey of Generative Adversarial Networks (GANs) in Cybersecurity Intrusion Detection. *IEEE Access*, 11:76071–76094, 2023.

[2] C. Park, J. Lee, Y. Kim, J.-G. Park, H. Kim, and D. Hong. An Enhanced AI-Based Network Intrusion Detection System Using Generative Adversarial Networks. *IEEE Internet of Things Journal*, 10(3):2330–2345, 2023.

[3] M. Wang, N. Yang, N. J. Forcade-Perkins, and N. Weng. ProGen: Projection-Based Adversarial Attack Generation Against Network Intrusion Detection. *IEEE Transactions on Information Forensics and Security*, 19:5476–5491, 2024.

[4] W. Wang et al. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection. *IEEE Access*, 6:1792–1806, 2018.

[5] S. Elsayed, K. Mohamed, and M. A. Madkour. A Comparative Study of Using Deep Learning Algorithms in Network Intrusion Detection. *IEEE Access*, 12:58851–58870, 2024.

[6] A. B. Mohammed, L. C. Fourati, and A. M. Fakhrudeen. Comprehensive systematic review of intelligent approaches in UAV-based intrusion detection, blockchain, and network security. *Computer Networks*, 239:110140, Feb. 2024.

[7] M. Samantaray, R. C. Barik, and A. K. Biswal. A comparative assessment of machine learning algorithms in the iot-based network intrusion detection systems. *Decision Analytics Journal*, 11:100478, June 2024.

[8] R. Tahri, Y. Balouki, A. Jarrar, and A. Lasbahani. Intrusion Detection System Using Machine Learning Algorithms. In *ITM Web of Conferences*, volume 46, page 02003, 2022.

[9] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016.

[10] W. Zhong, N. Yu, and C. Ai. Applying Big Data Based Deep Learning System to Intrusion Detection. *Big Data Min. Anal.*, 3(3):181–195, 2020.

[11] M. H. Haghighat and J. Li. Intrusion Detection System Using Voting-Based Neural Network. *Tsinghua Sci. Technol.*, 26(4):484–495, 2021.

[12] Y. Yang et al. ASTREAM: Data-Stream-Driven Scalable Anomaly Detection with Accuracy Guarantee in IIoT Environment. *IEEE Trans. Netw. Sci. Eng.*, 2022. early access.

[13] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation-Based Anomaly Detection. *ACM Trans. Knowl. Discov. Data*, 6(1):1–39, 2012.

[14] L. Zou, X. Luo, Y. Zhang, X. Yang, and X. Wang. HC-DTTSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering. *IEEE Access*, 11:21404–21416, 2023.

[15] A. Sarıkaya, B. G. Kılıç, and M. Demirci. RAIDS: Robust Autoencoder-Based Intrusion Detection System Model Against Adversarial Attacks. *Computers & Security*, 135:103483, 2023.

[16] L. Yang, J. Li, L. Yin, Z. Sun, Y. Zhao, and Z. Li. Real-Time Intrusion Detection in Wireless Network: A Deep Learning-Based Intelligent Mechanism. *IEEE Access*, 8:170128–170139, 2020.

[17] R. Ben Said, Z. Sabir, and I. Askerzade. CNN-BiLSTM: A Hybrid Deep Learning Approach for Network Intrusion Detection System in Software-Defined Networking with Hybrid Feature Selection. *IEEE Access*, 11:138732–138747, 2023.

[18] R. Soleymanzadeh and R. Kashef. Efficient Intrusion Detection Using Multi-Player Generative Adversarial Networks (GANs): An Ensemble-Based Deep Learning Architecture. *Neural Computing and Applications*, 35(17):12545–12563, 2023.

[19] B. Sharma, L. Sharma, C. Lal, and S. Roy. Anomaly Based Network Intrusion Detection for IoT Attacks Using Deep Learning Technique. *Computers and Electrical Engineering*, 107:108626, 2023.

[20] H. Benaddi, K. Ibrahimi, A. Benslimane, M. Jouhari, and J. Qadir. Robust Enhancement of Intrusion Detection Systems Using Deep Reinforcement Learning and Stochastic Game. *IEEE Transactions on Vehicular Technology*, 71(10):11089–11102, 2022.

[21] M. He, X. Wang, P. Wei, L. Yang, Y. Teng, and R. Lyu. Reinforcement Learning Meets Network Intrusion Detection: A Transferable and Adaptable Framework for Anomaly Behavior Identification. *IEEE Transactions on Network and Service Management*, 21(2):2477–2492, 2024.

[22] T. Kim and W. Pak. Early Detection of Network Intrusions Using a GAN-Based One-Class Classifier. *IEEE Access*, 10:119357–119367, 2022.

[23] R. R. d. Santos, E. K. Viegas, A. O. Santin, and V. V. Cogo. Reinforcement Learning for Intrusion Detection: More Model Longness and Fewer Updates. *IEEE Transactions on Network and Service Management*, 20(2):2040–2055, 2023.

[24] M. Al-Fawa'reh, J. Abu-Khalaf, P. Szewczyk, and J. J. Kang. MalBoT-DRL: Malware Botnet Detection Using Deep Reinforcement Learning in IoT Networks. *IEEE Internet of Things Journal*, 11(6):9610–9629, 2024.

[25] G. Apruzzese, M. Andreolini, M. Marchetti, A. Venturi, and M. Colajanni. Deep Reinforcement Adversarial Learning Against Botnet Evasion Attacks. *IEEE Transactions on Network and Service Management*, 17(4):1975–1987, 2020.

[26] J. Lansky et al. Deep Learning-Based Intrusion Detection Systems: A Systematic Review. *IEEE Access*, 9:101574–101599, 2021.

[27] Z. Wang, D. Jiang, Z. Lv, and H. Song. A Deep Reinforcement Learning Based Intrusion Detection Strategy for Smart Vehicular Networks. In *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1–6, 2022.

[28] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli. A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection. *IEEE Communications Surveys & Tutorials*, 21(1):686–728, 2019.

[29] S. Dong, Y. Xia, and T. Peng. Network Abnormal Traffic Detection Model Based on Semi-Supervised Deep Reinforcement Learning. *IEEE Transactions on Network and Service Management*, 18(4):4197–4212, 2021.

[30] X. Ma and W. Shi. AESMOTE: Adversarial Reinforcement Learning With SMOTE for Anomaly Detection. *IEEE Transactions on Network Science and Engineering*, 8(2):943–956, 2021.

[31] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas. Application of Deep Reinforcement Learning to Intrusion Detection for Supervised Problems. *Expert Systems with Applications*, 141:112963, 2020.

[32] N. Mishra and S. Pandya. Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review. *IEEE Access*, 9:59353–59377, 2021.

[33] L. Chavali, A. Krishnan, P. Saxena, B. Mitra, and A. Sreevallabh Chivukula. Off-Policy Actor-Critic Deep Reinforcement Learning Methods for Alert Prioritization in Intrusion Detection Systems. *Computers & Security*, 142:103854, 2024.

[34] C. Rookard and A. Khojandi. RRIoT: Recurrent Reinforcement Learning for Cyber Threat Detection on IoT Devices. *Computers & Security*, 140:103786, 2024.

[35] S. Vadigi, K. Sethi, D. Mohanty, S. P. Das, and P. Bera. Federated Reinforcement Learning Based Intrusion Detection System Using Dynamic Attention Mechanism. *Journal of Information Security and Applications*, 78:103608, 2023.

[36] S. Yu et al. Deep Q-Network-Based Open-Set Intrusion Detection Solution for Industrial Internet of Things. *IEEE Internet of Things Journal*, 11(7):12536–12550, 2024.

[37] K. Jiang, W. Wang, A. Wang, and H. Wu. Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network. *IEEE Access*, 8:32464–32476, 2020.

[38] W. Dai, X. Li, W. Ji, and S. He. Network Intrusion Detection Method Based on CNN-BiLSTM-Attention Model. *IEEE Access*, 12:53099–53111, 2024.

[39] X. Liang, H. Xing, and T. Hou. Network Intrusion Detection Method Based on CGAN and CNN-BiLSTM. In *2023 IEEE 16th International Conference on Electronic Measurement & Instruments (ICEMI)*, pages 396–400, 2023.

[40] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho. Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 202–206, 2018.

[41] I. I. Kurochkin and S. S. Volkov. Using GRU Based Deep Neural Network for Intrusion Detection in Software-Defined Networks. *IOP Conference Series: Materials Science and Engineering*, 927(1):012035, 2020.

[42] M. Abdallah, N. An Le Khac, H. Jahromi, and A. Delia Jurcut. A Hybrid CNN-LSTM Based Approach for Anomaly Detection Systems in SDNs. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*. ACM, 2021.

[43] L. Karanam, K. K. Pattanaik, and R. Aldmour. Intrusion Detection Mechanism for Large Scale Networks using CNN-LSTM. In *2020 13th International Conference on Developments in eSystems Engineering (DeSE)*, pages 323–328, Liverpool, United Kingdom, 2020.

[44] Marius Portmann. NIDS Datasets. `https://staff.itee.uq.edu.au/marius/NIDS_datasets/`. Accessed: 2024-10-19.

[45] F. Deldar and M. Abadi. Deep learning for zero-day malware detection and classification: A survey. *ACM Computing Surveys*, 56(2):1–37, 2023.

[46] Y. Guo. A review of machine learning-based zero-day attack detection: Challenges and future directions. *Computer Communications*, 198:175–185, 2023.

[47] J. Zhao, S. Shetty, J.W. Pan, C. Kamhoua, and K. Kwiat. Transfer learning for detecting unknown network attacks. *EURASIP Journal on Information Security*, 2019(1), 2019.