CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Deep Learning Techniques for Cyber-Attack Predictions in Organizations

A thesis submitted in partial fulfillment of the requirements

For the degree of Master of Science in

Computer Science

By

Nikhila Bikki

May 2024

The thesis of Nikhila Bikki is approved:

Dr. Sevada Isayan, Ph.D.	Date
Dr. Maryam Jalalitabar, Ph.D.	Date
Dr. George Taehyung Wang, Ph.D., Chair	Date

Acknowledgement

I am extremely grateful to Dr. George Taehyung Wang, my thesis advisor, for his generous support, continuous guidance, and meticulous instructions throughout my academic process and for believing in my project and for sustaining my confidence and passion for research.

I extend my heartfelt thanks to Dr. Sevada Isayan and Dr. Maryam Jalalitabar for agreeing to be my thesis committee and sharing their valuable time and thoughts with me despite their busy schedules. Their advice and guidance have made significant contributions to the improvement of this work.

I owe special thanks to my parents, whose boundless love and sacrifices for me and their unshakable confidence in me were the foundation of my success. Their unfailing support, encouragement and sacrifice have made me believe in myself, and this is as much their achievement as it is mine.

Finally, to those of you who contributed – in big ways and small, who believed in me and this project – the support has been instrumental. This endeavor is an achievement, as much as mine, and I am deeply grateful for it.

Dedication

This thesis is dedicated with love to every advisor, mentor and family member who has inspired me along the way. This thesis is an outcome not just of hard work in itself, but of who have supported and helped me throughout the years.

A special thank you to my advisors and supervisors: the teaching, guidance and support have been invaluable. To my teachers and advisers whose ideas and words shaped my thoughts with their words and wisdom, my gratitude abides.

I dedicate this work to my family, who made so many sacrifices so that I could be here. This was all made possible because of the belief in my dreams, which were always a little distant and hard to see. To my friends, for laughter and relaxation during the times of stress, for the moments of joy and sadness shared, you have been my heartstring for coming back into the self. And to those who have inspired me, if I have inspired any of you in return then, in words I cannot write, I thank and salute you all.

I am grateful for the support, encouragement and inspiration that each of you gave to me and cannot thank you enough for teaching me all that you have. This thesis is dedicated to you all, with gratitude.

Table of Contents

List of Figures

List of Tables

Abstract

Deep Learning Techniques for Cyber-Attack Predictions in Organizations

By

Nikhila Bikki

Master of Science in Computer Science

In the modern digital era, organizations across the globe are being assaulted by rapid cyber-attacks. As a result, early intruder detection within organizational networks is essential to alert users and shut down compromised systems. The detection of malicious intruders that have intruded into an organization's network can be predicted using 'intrusion detection system' (IDS). One approach is to deploy an IDS using machine learning (ML) and deep learning (DL) techniques to classify attacks. This thesis traces the development of IDS research, discusses IDS role within modern computer science, emphasizes the need for powerful IDS to combat evolving cyber threats, describes the problem-area and associated difficulties. Using the NSL-KDD dataset which consists of a wide range of network traffic features and attack classes, this study compared the performances of two supervised machine learning classifiers namely Decision Tree and Naïve Bayes with the deep learning models like CNN, VGG11, and ResNet18 to classify the intrusion attacks in the network traces. The NSL-KDD dataset offers a diverse mixture of normal and abnormal network traffic ensuring that it can be used to train IDS solutions that can operate in practical scenarios. The thesis gives details on the system architecture, how the dataset is collected, how the data is preprocessed, how the features are selected, and how the learning models are implemented. Furthermore, it includes feature selection, which is the optimal way of improving

classification accuracy over the state-of-the-art. Statistical performance metrics adopted include accuracy, precision, recall and F1-score. Findings indicate that VGG11 outscores the rest of classifiers with an accuracy of 97.87%, followed very closely by ResNet18 with an accuracy of 97.17%. Moreover, evidence suggests that CNN provides good accuracy of 96.19%. Hence, this work succeeds in performing important and relevant research on machine learning and deep learning techniques for network intrusion detection using the NSL-KDD dataset. In conclusion, the insight of this thesis on the effects of ML and DL algorithms in cyber-attack prediction can help realize that there is fortunately a solution to efficiently tackle modern threats and malicious actions.

Keywords: Cyber-attacks, Intrusion detection system, NSL-KDD, Decision Tree, Naïve Bayes, Deep learning, CNN, VGG11, ResNet18

Chapter 1: Introduction

Living in today's digitalized world where almost everyone depends on the internet in every possible manner, a major threat we all have to face in today's life is cyber-attack and hacking. With the increase in technology in the current world, hackers are finding more advanced ways of intruding in the system, finding, and compromising the system and stealing crucial information. Hacking is a system or computer network which has data that the user wants to access that is forbidden. It is compromised by identifying the weak points in a system, entering the required network, and using or modifying the unauthorized data. They have various ways of intruding on hackers. This could be possible by having physical access to the system or being aware of any software bug, vulnerability or intruding via remote to acquire the required data.

Since more aspects of life have become internet-based, there have been a plethora of anomaly problems and security breach threats. In order to prevent that, several Intrusion detection systems (IDS) have been developed to detect intrusion attacks. By observing and studying records of network [2], IDS monitors the network traffic like a digital guard dog, constantly on the lookout for anything suspicious and alerting the organization if anything goes awry. They tend to watch for malware attacks and unauthorized access request attempts by looking for any anomaly in patterns, and alerting the organization if they sense something strange.

The hackers are still able to get in even with the IDS installed. To make it better you could start by not just recognizing known attacks, but by predicting future new attacks that are not so easy to do. To help a computer to predict and react to these attacks is difficult, but with machine learning and deep learning it is not impossible, and the number of ideas grows every second. The research for cyber-attack prediction is already made and it has shown really good results. But there

is still a lot of research to do to understand different kinds of attacks in each organization. This might include research into data privacy and model generalization between different organizations as well.

This thesis explores Machine Learning and Deep Learning techniques to help predict cyberattacks in organizations. Then, according to the available data, prediction is done using those algorithms with high accuracy. By applying a mathematical model, we can minimize the prediction error and fit it to the available data. Feature selection techniques are used to improve the prediction by selecting only relevant attack features. Since feature selection is an essential task, we should give some caution to choose the right feature selection algorithm and finalize it. To make it work properly, we usually filter out non-related features that have no indication of the feature set size. Feature selection method is categorized into two types, commonly known as filter-based and wrapper-based methods. Filter-based method of feature selection has high accuracy standards [3].

In this Project, machine learning and deep learning methods are implemented on the NSL-KDD dataset for prediction of cyber-attacks [4]. This is a work labelled normal and different instances of intrusion and need to classify whether it is a normal access or suspicious act. Because of its labelled data, machine learning methods of supervised methods is better to use by the model.

In this project model, several algorithms have been used including machine learning algorithms like Decision Trees (DT) and Naive Bayes (NB). Deep learning techniques like Convolutional Neural Networks (CNN), VGG11, and ResNet18 were also implemented because accuracy of intrusion detection can be improved in my design by extending the layers [9].

1.1 Motivation

Today, computer networks are essential to the day-to-day operations of most organizations. However, the networks that enable these operations also leave the organizations vulnerable to a barrage of cyber-attacks, such as malware infections, data breaches and denial-of-service (DoS) attacks. These attacks can lead to serious financial losses, reputational damage, and regulatory fines. As the frequency and sophistication of cyber-attacks continue to rise, it becomes important to understand effective systems for detecting and mitigating these threats.

An Intrusion Detection System (IDS) is an essential component of an organization's network. Used to detect cyber threats that may be taking control of the system network, an IDS analyses network traffic, and the activity on the system to detect any suspicious behavior that might indicate an attack in progress or a potential attack. Traditional IDS, however, sometimes struggle to stay on top of the quickly evolving threat landscape. Most signature-based solutions, for example, rely on the detection of known patterns of misuse or harmful traffic that may be symptomatic of a malicious attack [10]. In particular, these solutions have proved to be ineffective in detecting zero-day attacks and other never-before-seen threats [11]. In addition, given the large number of pieces of information contained in the payloads of network traffic, manual detection and analysis are clearly unfeasible alternatives.

This research is motivated by the desire to address the challenges above and further improve the state-of-the-art in intrusion detection via the application of ML and DL techniques. We can improve the performance of IDSs that are used to identify instances of network intrusions, compromises, and other malicious events by detecting anomalous behavior in network data [12] with machine and deep learning techniques. ML and DL algorithms excel at learning patterns and behaviors within data which can then be applied to predict new instances.

1.2 Background and Context of the Study

This paper is organized around the objective to determine whether a personality test can be developed to identify potential cybersecurity 'insiders' who might become malicious actors. There has been a striking transformation in the arena of cybersecurity threats against organizations across the globe. This study is situated against the backdrop of widespread adoption and application of cutting-edge digital technologies by diverse workforces around the world, which has created a significantly broader attack surface for cyberattacks and malicious actors.

A cybersecurity 'insider' is an employee, contractor, or business partner with legitimate access to an organization, but who has decided or been influenced to misuse this access to harm their employer. They are often motivated by a hunger for greater status and recognition. An insider can be engaged by a nation-state to harm their employer as part of a larger multinational scheme, yet the majority of those compromised by this kind of cyberwarfare are unaware of the deception until it is too late.

Preventing network intrusion by only using rule-based or signature-based methods is no longer practical [13] to defend against increasingly savvy adversaries who alter code behavior and apply other techniques to evade traditional intrusion-detection systems and duck beneath existing firewalls. The sheer volume and complexity of network traffic in today's connected world also presents a challenge for human analysts to remain up to date on threat intelligence and manually investigate and verify each potential breach in real time.

In this context, ML and DL techniques hold the potential to enhance intrusion detection by empowering models to learn from large datasets of network traffic and system logs and subsequently apply this knowledge to make real-time decisions on network traffic in order to

identify potential cyber-attacks. This can allow organizations to identify and respond to cyber threats before any damage occurs, significantly reducing risk.

1.3 Problem Statement

In this domain of computer network defense, ensuring that an organization has an effective capability to detect and respond to cyber-attacks is a challenging task. As malicious actors develop increasingly sophisticated tactics, techniques, and procedures (TTPs) to deliberately detect and violate organizational networks [14], the need for effective and responsive IDS solutions has never been more vital. Unfortunately, legacy IDS solutions are fraught with intractable limitations that prevent an organization from effectively recognizing and responding to adversarial intrusions in a timely manner.

What makes this method of detection so difficult is that traditionally it is signature-based, meaning that it is based on known signatures of malicious activities. So, this IDS will look for a signature that matches an attack that has already been seen. While this approach has proven to be effective against well-defined and mature threats, signature-based IDS is fundamentally a reactive approach and is not able to detect novel or zero-day attacks. As cybercriminals keep building new techniques to avoid IDS and use novel malware variants, the signature-based IDS will typically be behind in its ability to identify new threats, often exposing organizations to exploitation.

Perhaps the most daunting challenge is the sheer volume and velocity of network traffic generated by today's organizational networks – a compounding problem as the number of connected devices (including cloud services, the Internet of Things, etc.), and associated traffic volume, increases exponentially each year. As the amount of data flowing across the enterprise has grown, so has the operational burden of traditional IDS products: today, signature-based IDS

systems experience Juicebox-level false positive rates and miss a significant proportion of detections [15]; security analysts are swamped with an avalanche of meaningless data, and cannot possibly sift through all of it by hand in a timely manner; organizations have invested significant time and money to train staff in audit procedures, but those procedures are often so cumbersome they go unused; likewise, network forensics analysis is hindered by data volumes and burden of noise [16]. As a result, automated, smart network IDS products that can perform real-time analysis of terabytes of traffic in data centers are required.

Moreover, the changing nature of cyber threats and the use of evasion methods, such as polymorphic malware, encryption, and obfuscation, have the potential to expose the vulnerability of IDS to attackers [14]. A skilled adversary who wishes to avoid detention may not exceed the predefined limits for the number of failed logins, nor create levels of activity that are inconsistent with those of other users with similar statistics of failed attempts at logging into a network. In fact, an attacker may not want to draw attention through abnormal behavior, and having been given an account by an insider, might gain access to a network without any indications of malicious intent or apparent need. Traditional IDS may fail to detect such attack methods. Insider threats and, consequently, insider attacks pose further difficulties for IDS systems to detect [17]. For example, should a legitimate user's keystrokes be considered a threat because they might result in data alteration within an accidental or deliberate range deemed malicious?

Also, classical IDS systems are often not capable of scale dynamically or effectively with changing event situations in networks, which is very much the case with an evolving network [16] (such as increasing traffic volume or changing topologies). Legacy IDS therefore exhibit performance bottlenecks and resource constraints in the face of growing network traffic, in turn decreasing their capacity to detect cyberthreats.

Thus, in terms of network organizations, the problem of intrusion detection is a multifaceted problem with its roots in signature-based detection, scalability, its adaptability to the ever-changing network traffic, and evasiveness of attackers using customized toolkits, selecting uncommon attack targets and bypassing traditional IDS tools (instances of dissimilar website instances sharing the same templates are prevalent). However, the issue of intrusion detection and its significance cannot be ignored [13] as it poses a multifaceted problem that necessitates a rethink in IDS solutions to utilize the emerging machine learning and deep learning techniques in network intrusion identification at scale and in real-time. Such endeavors can make online organizations not only safer, but also the citizenry by preserving their privacy and data security.

1.4 Research Objectives

The overall objective is to develop and assess exploit use ML and DL methods to advance organizations' network cybersecurity posture. To achieve this, the specific objectives are:

1. Investigate the State-of-the-Art in Intrusion Detection Systems (IDS):
    - Survey the State-of-the-Art: Conduct an extensive review of existing research and authoring techniques that have been used in the field of IDS for legacy systems, as well as for modern ML/DL-enabled IDS [18]. This will help in unravelling the existing state-of-the-art and also identify the advantages, drawbacks and the latest directions in IDS research.
    - Understand ML and DL Techniques: Understand and evaluate commonly used ML and DL techniques for intrusion detection, such as decision trees, naive Bayes, convolutional neural networks (CNN), VGG11, and ResNet18. Investigate their capacity to detect different types of cyber threats and assess their performance in the real-world scenario.

2. Design and Implement ML/DL-based IDS Architectures:
    - Architectural Design: Design new IDS architectures based on ML and DL algorithms that can detect and respond to network-based attacks. Draft architectures that can scale across the enterprise, adapt to changing demand, and are energy-efficient enough to keep up with massive volumes of network packets in real time.
    - Implementation: Take the proposed IDS architectures in passive and active modes and implement it in state-of-the-art ML and DL frameworks and libraries, for example, TensorFlow, Keras, and scikit-learn. The role of programmers is so important in the field of machine learning, and researchers often lack this knowledge. Develop a robust and modular code for a variety of data preprocessing techniques, feature extraction methodologies, training diverse and popular ML and DL models, and inferring from various available inference techniques.
3. Collect and Preprocess Network Traffic Data:
    - Dataset Acquisition: Select appropriate datasets and acquire the datasets to train, test and verify the Natural Language Processing-based Intrusion Detection Models (IDSMs). Educate on cyber-attacks and threats: mitigate, evaluate, and prevent network attacks and threats [19]. Keep the system informed: analyze the network traffic and manage any issues. Explore and enhance the tool: come up with different approaches and implement improvements. Identify and resolve issues: collect training and test data with potential remedies. Lexicon-based AIP with CD provides the main tasks.
    - Data Preprocessing: Clean, normalize and transform the collected network traffic data into an input for the ML/DL models. Resolve missing values, outliers, and other forms of noise to ensure the quality and robustness of training data.

4. Train and Evaluate ML/DL Models:
    - Model Training: Trained the ML/DL models with preprocessed network traffic to learn their normal behaviors and anomalous activities indicative of cyber-attack. Also tuned the models' hyperparameters and architecture to achieve best performance and accuracy.
    - Model Evaluation: Evaluate the performance of the trained models on appropriate metrics like precision, recall, accuracy and F1-score. Compare ML/DL models and choose the best approach(s) to detect intrusion.
5. Validate and Benchmark IDS Performance:
    - Validation: Performance analysis of the ML/DL-based IDS architectures on independent large datasets and real-life network traffic trace files. Examine the effectiveness, robustness, reliability, and scalability of models under different settings and attack scenarios.
    - Benchmarking: benchmark the performance of the proposed IDS architectures vs existing signature-based methods and other state of the art intrusion detection systems in detection term rates, false positive rates, response times, and the required resources. In this way, the superior performance of the ML/DL-based methods with respect to these metrics will be highlighted.
6. Contribute to Knowledge Advancement and Practical Application:
    - Knowledge Generation: Generate novel insights, techniques, and methodologies in the domain of intrusion detection and cybersecurity through empirical research and experimentation; this activity supports the broader knowledge construction through diffusion and co-production channels, by publishing research outcomes in well-

recognized peer-reviewed forums, such as journals, conferences, and academic channels.

- Translation: Translate research results into practical tools and solutions that can be leveraged by organizations to strengthen their defenses. Design open-source software implementations, frameworks or libraries that are based on ML/DL for IDS and make them freely available to the security community.

Successful completion of these research tasks will help the study to make contributions in advancing the art of intrusion detection, and to give organizations capabilities to resist the advances of cyber-attacks in the digital world.

1.5 Challenges

Challenges in Implementing Machine Learning and Deep Learning Techniques for Cyber-Attack Prediction. In addition to the increasing pressure of the latest cyber threats and hacking cases, organizations face significant challenges in employing Machine Learning and Deep Learning algorithms for the detection of cyber-attacks, and prediction of intrusions and data breaches. This section will explore some of the challenges related to the use of advanced algorithms in intrusion detection systems (IDS) and prediction.

- Data Quality and Quantity: Collecting enough exploit and attack data (of a high enough quality) from the wild represents one of the biggest hurdles in getting ML and DL techniques used for predicting cyber-attacks. The performance of predictive models tends to be highly dependent on the quantity and quality of the data used to train them, and so representative, high-quality, and high-volume datasets are crucial for good performance [18]. However, datasets containing ground-truth labelled data that accurately represent

real-world cyber-threats can be hard to come by. The NSL-KDD dataset, while one of the most commonly used benchmarks for conducting research into machine-learning approaches for intrusion detection, might not provide enough representative attacks or variations to train an effective model for the purposes of real-world use. Without using data to represent today's real-world threat scenarios, machine-learning models won't learn to generalize effectively – which is critical to a strong defense.

- Feature Selection and Dimensionality: A further major issue is the feature selection and the high dimensionality in the dataset. The feature selection and reduction of dimensionality should be the important issues [21] when the prediction models of cyber-attacks are built. However, the selection of optimal features set with sufficient explain ability to cover the underlying patterns of cyber threats while noise and redundancy is removed remains a difficult task. The investigation indicates that the Chi-squared feature selection algorithm performs well in order to overcome aforementioned problems. Yet we believe there is room for improvement for us to search for more advanced feature selection techniques in order to enhance the performance of the model.
- Model Complexity and Interpretability: The complicated structures of ML and DL models bring interpretability and explain ability issues by nature. Deep Learning models such as CNN, VGG11, and ResNet18 are well known as black-box models. It is difficult to describe why such models make the decisions they make - we only know it works by observing the predicted outcomes [21], which is not an ideal solution. However, people care why a predicted result is made and want to understand it. This criterion for trusting the model is very important not only to give credible evidence for its decision-making process but also to increase the model's transparency and accountability. The trade-off between the model

complexity and interpretability remains a significant bottleneck for employing ML and DL techniques in practice to predict cyber-attacks.

- Scalability and Performance: Scalability and performance issues can also hinder the practicality of ML and DL techniques that go into cyber-attack prediction. Attempting to train complex DL models on sufficiently large datasets can be a computationally complex process, sometimes leading to time-consuming evaluations, and in need of access to high-performance computing [22]. Even so, an essential question not typically considered under algorithmic bias, pertains to the computational-complexity trade-off between models with vast hidden layers and the urgency of delivering real-time prognoses in operational environments to prevent the propagation of threats. A further challenge is the requirement to frequent updates and retraining of the models, as cyber threats evolve over time, to sustain the system's responsiveness and predictive quality.
- Robustness and Generalization: Developing predictive models that are robust enough to generalize across novel attacks and unobserved adversarial threats is another pressing issue in cybersecurity applications. A fundamental challenge is adversarial manipulations and vulnerabilities of ML and DL models that may introduce biases and inaccuracy in predictions. This requires robust evaluation and validation techniques to conduct adversarial robustness tests on the models, which can highlight vulnerabilities and adversarial patterns. The third challenge is how well a predictive model travels from one network domain to another in detecting novel adversarial threats in new domains or environments with new attack types [21].
- Integration with User Interfaces (UI): IDS must be deployed in production environments that need out-of-the-box integration with user interfaces that analysts can work with and

easily interpret model output. Easy-to-use interface/user experience (UI/UX) visualizations and effective interactive decision support by highlighting anomalies and changes need feedback from the user, and this may be difficult to achieve due to a balance of function and form over information overflow.

Although ML and DL techniques can provide a means to improve the prediction rates of cyber-attack and intrusion detection, there are several technical limitations for practical use of these techniques in the area of cybersecurity applications. Issues related to data quality and quantity, identifying proper feature selection and feature reduction methods, striking the right balance between model complexity and interpretability, handling issues of scalability and performance, and ensuring whether models are robust and generalizable need to be resolved. Further research and innovation can tackle these issues and continue to raise the bar of ML and DL techniques, enabling organization's challenges in cybersecurity, in order to better defend against increasingly sophisticated cyberattacks in the near future.

Chapter 2: Related Work

2.1 Introduction

In a digitally transformative period where no one can remember a time when the world was not connected by some form of digital technology, cybersecurity has emerged as a significant organizational challenge for almost just about every industry under the sun. Expectations are that the global network of networked devices, known as the 'Internet of Things' (IoT), along with the burgeoning cloud computing space, will lead to a five-fold increase in the device-based attack surface area during the next the years [23]. Meanwhile, the most sensitive information, intellectual property, critical national and infrastructure systems are migrating into online environments at an accelerating rate. As organizations seek to prevent malicious parties from spying on them, stealing their secrets and infecting their operational technology, they need to develop powerful predictive capabilities to anticipate, identify and counteract cyberattacks.

With deep learning as part of the cyber-attack prediction/detection mechanism, you're applying a type of artificial intelligence (AI) and machine learning (ML) technique that has the potential to provide a greater capability for predicting the occurrence of an attack and acting in ways that enhance the ability to make sense of what's happening with the computers on the networks during an attack. First, you must understand that deep learning is just one kind of machine learning. The way it works is that, in a typical attack prediction/detection system, programmers and analysts can define characteristics of prior attacks, with associated heuristics that define 'dangerous' or 'suspicious' data values or activity that might indicate an attack. This approach is sometimes known as rule-based [24] since there's a predefined list of the attack signs and prescribed actions to stop or slow the attack. In other words, humans define what the system should look for and how it should respond.

In reality, it's much more complicated because of all of the variables and difficult decisions regarding when to alert a human operator and when to respond automatically. Nonetheless, machine learning adds value since human beings can never 'learn' every new manifestation of an attack [24], and the adversaries who write the attacks need an advantage over you. So, what is deep learning? Imagine a system storing the data, analyzing it and reporting it, repeating the process over and over. For example, if shown a bunch of pictures of dogs, some of which are new to the system, the system can learn to detect those pictures and assign a numerical value to its confidence level in its ability to do so. Similarly, if you provide a lot of recordings of spoken language, the system would be able to connect a bunch of odd-looking symbols called vectors that map the soundwaves with the spoken words.

With deep learning techniques, cybersecurity will develop mobile, self-rolling and self-adapting defense schemes, changing from reaction to anticipation [25]. Deep learning techniques will also react to threat changes in real time: current cyber-security response methods rely on various intrusion detection systems (IDS) and security information and event management (SIEM) solutions such as Snort, Bro and Suricata, which work via predefined rules against predefined threats or vulnerabilities [26]. These rules can be simple of 'pace 802.11B or can be based on records of attacks.

Recently, deep learning has been applied to network intrusion, malware and phishing detection, and threat intelligence, to name a few. "LatinApps" The Network Intrusion Detection System by Nathan Shone et al (2018) used a convolutional neural network model [27] to detect 'observed anomalous behaviors on a network that would provide an indication of a cyber-attack'. Alsoufi and Razak (2021) discussed anomaly-based intrusion detection systems in internet of things (IoT) environments [12]. Analyzing and detecting 'anomalies' that may be exploited as a

step towards a cyber-attack is another area where deep learning is widely used for cybersecurity purposes.

Despite the advantages of deep learning for improving cybersecurity, there are a number of practical constraints to consider before achieving the promise envisioned in this review. The scalability and computational complexity of DL models implies that state-of-the-art models cannot be deployed in environments with limited computational and resource constraints [28] (eg, edge computing nodes or resource constrained IoT scenarios). AI model interpretability and human explain ability are other constraints that cast doubt on the reliability and accountability of decision-making processes using DL algorithms. Besides, important implications arise due to data privacy issues, algorithmic bias, and adversarial attacks that require robust safeguards and governance to develop and deploy [9] deep learning-based prediction systems for cyber-attacks in the most challenging critical infrastructure environments in an ethical and fair manner.

Considering these scenarios, it is essential that interdisciplinary research efforts continue with deep learning in cybersecurity. Integrating knowledge from fields such as computer science, data science, cybersecurity and ethics can help researchers and practitioners to develop technically sophisticated and accountable automated responses to cyber threats.

2.2 Analysis of Deep Learning Architectures for Cyber-Attack Prediction

There are multiple related works conducted on preventing cyber-attacks using various deep learning techniques on different network data sets. It includes using a wide array of deep learning architectures like Convolutional neural networks VS Leaderboards Convolution layers within neural networks [29] attempt to track specific patterns and data structures, Deep convolutional

neural networks, Lars, Bayesian Voting systems [30], AdaBoost, Gradient Boosting and many more.

The intrinsic nature of CNNs to find features in sequential data like network traffic because of their inherent pattern-recognition [6] where spatial relationships occur makes such a network well-suited for intrusion detection tasks. B. Cao proposed a system of hybrid CNN combining ADASYN and RENN for IDS that can achieve 99.81 per cent accuracy in recognizing different attack types on the NSL-KDD dataset [8]. Jawad Hussain Kalwar and others used a CNN model for detecting anomalies in Internet of Things (IoT) networks, providing interesting results in detecting malicious traffic [7].

Long Short-Term Memory (LSTM) Networks have the second most powerful architecture for cyber-attack prediction in log data. LSTMs are particularly good for sequential input such as log-like data. Before diving in, let's create an overview of how LSTMs can be applied to different tasks. LSTMs are a category of recurrent neural networks, capable of 'remembering' past inputs for a useful period of time (hence the name 'long'). They are very powerful at tasks involving temporal dependencies within data streams, such as web log data that is collected over time [7]. For instance, users naturally act over time, which makes them perfectly suited to time-series tasks. Yu presented an LSTM-based anomaly detection approach for web logs and identified various attacks, including SQL injection and cross-site scripting (XSS) attacks [7]. Another interesting idea was presented by Modi Abbasi and others in 2020, leveraged an LSTM autoencoder for traffic anomaly detection in real time [31]. Autoencoders, a type of neural network, tend to discover compact yet powerful representations of the input data, which provides LSTM with an opportunity to uncover anomalous network traffic patterns indicative of attack behavior.

In a research on “Cyberattack prediction with deep learning can be conceptualized as a classification problem”, the model is newly introduced in this paper with using rectified linear units (ReLU) as the activation function in a deep feed forward neural network [4]. The author's concluded that this model shows better performance in comparison with similar models. “Deep learning techniques to detect cybersecurity attacks” is a orderly plotting study by Damiano Torre, did a high quality, systematic and inclusive summary on the already implemented deep learning techniques to identify cybersecurity attacks [5], along with the challenges and limitations of the current research in this domain, has been provided. 14 application domains, 8 cybersecurity attacks, and 93 raw data, publicly available datasets have been identified and categorized [5]. Several open research problems have also been elaborated upon, such as lack of research done in an industrial setting; lack of availability of actual-time datasets; lack of studies based on promising deep learning techniques and relevant cybersecurity attacks.

The exact type of deep learning architecture used for predictions of cyber-attacks will depend on the type of data and the problem you’re trying to solve. For network traffic analysis, CNNs work well. For sequential data such as log files or user behavior, either LSTMs or a more general type of network called a RNN can work. For more complex relationships across diverse data — such as user-agent strings, time-of-day variabilities or specially crafted inputs that reduce noise — a transformer might be the method of choice.

2.3 Advancements in Deep Learning for Intrusion Detection

The recent successes in the field of intrusion detection through application of state-of-the-art deep learning methods, which allow providing more resilient, versatile and accurate means of detecting malicious patterns in network traffic. This section depicts certain aspects of the most

recent research in the area of deep learning for intrusion detection, illustrating and putting the research in proper context.

'A review of recent development in cyber security benchmark datasets for accurate evaluating data-driven based intrusion detection systems' (2015) by Abubakar Al and others [32] is a review of the recent development in cyber security benchmark datasets for the accurate evaluation of machine learning and data mining-based intrusion detection systems. The authors concluded in their review that they found out that the two benchmark datasets of cyber security – KDD and UNM – have issues. They state that the current growth in computer tech renews the need for accurate and up to date benchmarks against cyber security attacks, so that enhancements in IDSs can be evaluated. Furthermore, they add that datasets in these benchmarks can't adhere to the prospects of the current growth in computer technology.

Based on the above analysis, they created a new benchmark, called ADFA Linux (ADFA-LD). It was stated that it can be used for the assessment of ML and data mining techniques based on the intrusion detection systems in 2013 [32].However, the authors also stated that ADFA-LD needs further development, including full descriptions of ADFA-LD attributes.

Following the hierarchical form of human visual cortex that leads us through local features to higher-level spatial features, such as edges and contours, CNNs are considered the highest performing machine learning model for extracting features from raw data. It is no surprise, then, that they have recently been investigated in 'A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection' (2015) by Anna L. Buczak and Erhan Guven Piot describes modern CNN-based techniques [33], aim to handle super high-dimensional (spatial) data by incorporating interval properties. Such technique can be used to obtain spatial dependencies, as well as to identify an object of interest with anomalous behavior within the collected packet data

of the network traffic. To learn to identify discriminative features from packet headers or payload data in packet data flows, a CNN-based IDS could capture such features either from specific fields in packet headers or work on the payload of those packets, after multiple careful steps of data pre-processing. Through convolutional layers, a CNN-based IDS system could effectively detect various threats such as malware propagation and DoS attacks on the system.

GNNs provide a flexible yet powerful framework for representing interactions and functional relationships in network data and attempt to emulate the underlying structural properties of the network traffic. 'Anomaly-Based Intrusion Detection Systems in IoT: A Review (2021)' by Athraa Al Soufi and Alaa G Salem Razak [12] reviews research in this domain, applied to network intrusion detection, characterizing network traffic as if it were a graph structure and training GNNs to learn rich representations of how networks behave. By harnessing neighborhood information from other nodes and edges, GNN-based intrusion detection systems can detect anomalies such as network scanning traffic, botnet communications, or lateral movement by an attacker, with great accuracy and efficiency.

Apart from these, in more recent years, hybrid architectures and ensemble methods such as network intrusion detection and have been introduced based on using a committee of multiple deep learning models that should provide robustness and increase detection accuracy. The use of hybrid architectures, as in the paper 'A Deep Learning Approach to Network Intrusion Detection' by Shone et al, combines features extracted by CNNs, RNNs and GNNs to provide machine learning models [27] with diverse views of network traffic and enhance their overall detection performance. Ensemble methods, whether its model averaging or stacking, can also harness the collective intelligence of multiple models, which provide robustness to generalization and adversarial attacks.

Adversarial learning techniques could improve the robustness of deep learning models against adversarial attacks and evasion strategies used by attackers. The concept of adversarial learning is applied to intrusion detection in ‘CYBER ATTACKS: TRENDS, PATTERNS, AND SECURITY’ by investigating the adversarial training or adversarial regularization techniques that improve the robustness (resilience) and accuracy of the intrusion detection [34]. Using adversarial training or adversarial regularization techniques in the intrusion detection model could make them more resilient to adversarial attacks and also accurate even in the presence of highly advanced adversaries.

Chapter 3: Methodology

3.1 Dataset Description

The project uses the KDD Cup 1999 Dataset [35] which was originally curated for The Third International Knowledge Discovery and Data Mining Tools Competition (KDD-99). The primary task of it is to construct an effective network intrusion detector, which is capable of distinguishing between normal and doubtful connections. This data was captured by Stolfo from DARPA 1998 Intrusion Detection System evaluation program [3], containing 7 weeks of network traffic, with 4 gigabytes of comprised data.

The purpose of the coding for this intrusion concept is solely auditing the intrusions formulated in a military network environment. It contains both a labelled and an unlabeled instance and is a great way of training predictive models. The main objective of this research conducted is to audit the intrusions simulated in a military network environment. It encompasses both labeled and unlabeled instances, which makes it suitable for training and evaluating predictive models.

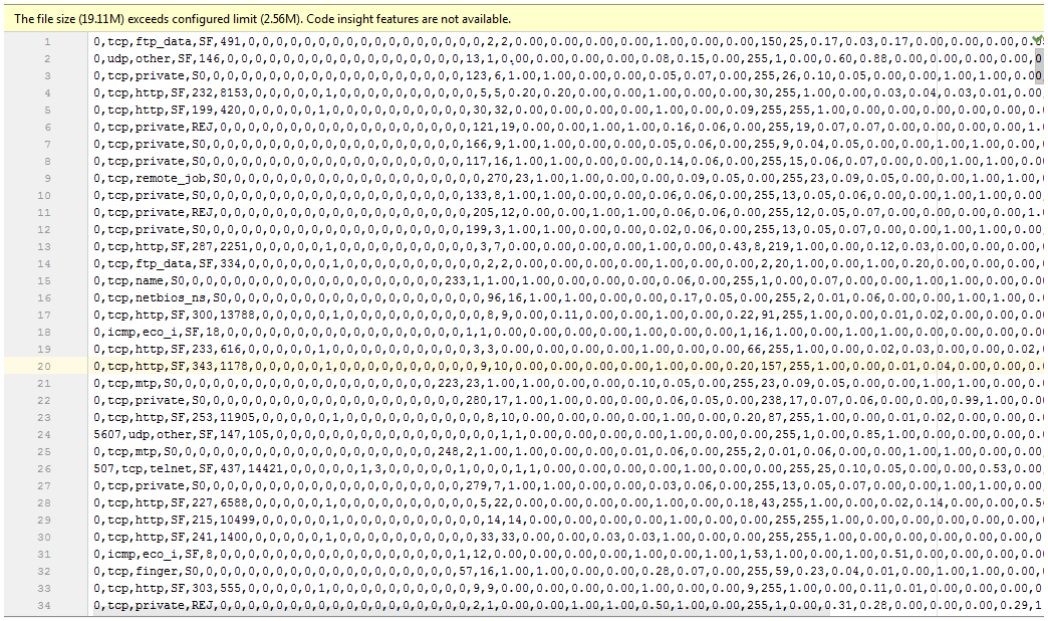

Figure 1 Training dataset

The complete Dataset has 18MB; Uncompressed data 743MB [36]. It will hold 5,000,000 connection records Figure.1, illustrating the data structure of the training dataset (KDDTrain+) that has 41 attributes as shown in Figure 2, which consist of 38 numerical and 3 categorical features and 125973 instances labelled. The training data KDDTest+. CSV consists of 22544 instances for testing purposes [35].

The three categorical attributes are: “protocol_type,” “service,” “flag” which identified from the above-mentioned data columns. The type of network connection is labeled as normal or the type of attack. The attacks are classified into 4 types including Denial of Service (DoS), User to Root Attack (U2R), Remote to Local Attack (R2L) and Probing Attack (PROBE) [36]. The figure below describes the classification of all the attacks into the main 4 types as mentioned.

The below snippet shows the 41 features of the training dataset:

```
datacols = ["duration","protocol_type","service","flag","src_bytes",
"dst_bytes","land","wrong_fragment","urgent","hot","num_failed_logins",
"logged_in","num_compromised","root_shell","su_attempted","num_root",
"num_file_creations","num_shells","num_access_files","num_outbound_cmds",
"is_host_login","is_guest_login","count","srv_count","serror_rate",
"srv_serror_rate","rerror_rate","srv_rerror_rate","same_srv_rate",
"diff_srv_rate","srv_diff_host_rate","dst_host_count","dst_host_srv_count",
"dst_host_same_srv_rate","dst_host_diff_srv_rate","dst_host_same_src_port_rate",
"dst_host_srv_diff_host_rate","dst_host_serror_rate","dst_host_srv_serror_rate",
"dst_host_rerror_rate","dst_host_srv_rerror_rate","attack", "last_flag"]
```

Figure 2 Total 41 features in the KDD-99 dataset.

3.2 Types of Attacks:

In view of the important role cybersecurity plays in today's digital systems, the motivation behind the paper is to provide a categorization of attacks who attempt to gain access to networks, systems and data without authorization. This can take many different shapes, which in turn

determines the objective of an attack, and what tasks the attacker needs to perform to reach their goal. In this section we outline several types of attack, group them into 4 classes [39], and describe their objectives and activities. To make this easier we will consider the classes of attack: Denial-of-Service (DoS) attacks; Probing attacks; User-to-Root (U2R) attacks; and Remote-to-Local (R2L) attacks [37].

Each attack type is defined by an action and target of the attacker and one of the following the classes can be attributed to each different type of cyber-attacks:

3.2.1 Denial-of-Service (DoS) Attacks

Denial-of-Service (DoS) attacks are made in order (or to) make a network or computer system unavailable to its intended users, or to make an existing service unusable. This often involves flooding the target system with traffic or requests to the extent that it cannot cope or respond, and it often causes the victim's system to slow down or crash [38]. DoS attacks are initiated in a number of ways including through malformed packets, exploiting vulnerabilities, or simply sending more traffic than the recipient's network bandwidth can handle. The goal of DoS attacks is generally to deny service, and cause general annoyance, loss of money and reputation to the victim.

Examples of DoS Attacks:

- neptune: floods the victim's network with a large number of TCP SYN packets in order to take control of the system resources and block network traffic.
- back: Exploits weaknesses in protocols or services of a target network in order to overwhelm resources available to the system, thereby resulting in service denial.

- smurf: Spoofs the source IP address and sends ICMP echo requests to broadcast addresses, causing a flood of responses to the victim and overwhelming its network bandwidth.
- pod: Sends malformed or oversized packets to crash the victim's system or network devices.
- land: Parodies the source IP and port so as to create a DoS condition by having the victim's system respond to itself.
- teardrop: Packets are fragmented and have their payloads overlapping in a way that makes the victim's system, from a distance, either cash or become unresponsive.

3.2.2 Probing Attacks

Such attacks normally employ more reconnaissance activity, with a view toward learning details about the target computer system or network. Typical probing attacks involve performing scans to determine the state of vulnerabilities in a target system, including searching for any ports that might be available. Such attacks also seek to map the network topology. This initial phase in a typical cyber-attack campaign allows an adversary to construct a picture of the computer infrastructure and plan the next steps in the attack sequence [39]. Such probing attacks might be considered a reconnaissance process, but they can have serious consequences for their victims if, following the probing, the adversary chooses to launch further attacks based on what they learned in the probing stage.

Examples of Probing Attacks:

- portsweep: Look up a set of port numbers and see which hosts are reachable at some or all of these ports. Can also scan all the ports for a given host and see which are open. Can also specify which services you want checked for.

- satan: Performs an in-depth analysis of network services, identifying potential vulnerabilities and misconfigurations.
- nmap: Pioneering host and service discovery, TCP and UDP scanning, OS and version detection, and security scanning.
- ipsweep: Scans a range of IP addresses to identify active hosts and network devices.

3.2.3 User-to-Root (U2R) Attacks

In a User-to-Root (U2R) attack, the attacker is an unprivileged user attempting to gain superuser privileges (root or administrative privileges) on a target system. The attack may take advantage of some vulnerability in the system or in an application, thereby allowing the attacker to gain elevated privileges that can then be used to access sensitive information, execute arbitrary commands, or subvert the integrity of the system. U2R attacks typically depend on targeting specific vulnerabilities or weaknesses in the security posture of the system being attacked, and thus may require a deep understanding of the target system and environment and the precise way to go about it [39].

Examples of U2R Attacks:

- buffer_overflow: Exploits a software application's buffer overflow vulnerability to execute a program of the attacker's choosing and then gain root access.
- loadmodule: Tries to load malicious kernel modules or libraries to gain local elevated privileges on the system.
- rootkit: Allows the attacker to remain persistent and maintain control of a system by installing a suite of malware and tools.

- perl: Uses Perl scripts or command injections to elevate privileges and access system resources without authorization.

3.2.4 Remote-to-Local (R2L) Attacks

In a Remote-to-Local (R2L) attack, an adversary tries to compromise a local system from a remote location. In this threat model, an attacker attempts unauthorized access from a remote infrastructure through sequence of attack vectors to eventually compromise a vulnerable local entity. R2L attacks remote systems seek to traverse network protocols, services, or applications to infiltrate an intended vulnerable target, by bypassing authentication processes at each stage of this intricate and sophisticated attack. Such attacks employ weakness in user authentication, software misconfigurations or insecure network connectivity to access a target for unauthorized access. Once inside an intended target, the adversary might attempt further privilege elevation or data exfiltration [39].

Examples of R2L Attacks:

- warezclient: An attempt to link and download or upload software, media, or data from or to a remote system without authorization.
- multihop: Uses more than one intermediate host (including intermediary proxies) to disguise the point of origin and gain access into the target system.
- ftp_write: Exploits unpatched, privileged FTP configurations to gain written access to remote files or directories on a target system.
- imap: This attack leverages a vulnerability in the Internet Message Access Protocol (IMAP) used for email transport, to allow attackers to compromise an email account or disclose confidential data.

- guess_passwd: Brute-force method to try and guess user passwords or to access credentials in order to gain access to the target system.

Through classifying attacks according to their behavioral characteristics and strategic objectives, we can easily build predictive capabilities to better detect, prevent and mitigate attacks on their systems and networks [40].

Attack Group	Attacks
Denial of Service (DoS)	neptune, back, smurf, pod, land, teardrop
User to Root Attack (U2R)	buffer_overflow, loadmodule, rootkit, perl
Remote to Local Attack (R2L)	warezclient, multihop, ftp_write, imap, guess_passwd, warezmaster, spy, phf
Probing Attack (PROBE)	portsweep, satan, nmap, ipsweep

Table 1 Network Attack types categorized into groups.

3.3 System Model

The proposed system architecture, depicted in Figure.3 leveraged both Machine Learning and Deep Learning algorithms to enhance the accuracy of the cyber-attacks prediction model. The model is trained and tested on the NSL-KDD Dataset and the best trained model amongst all is used to predict the network connection of a real time data which can be given as input for detection.

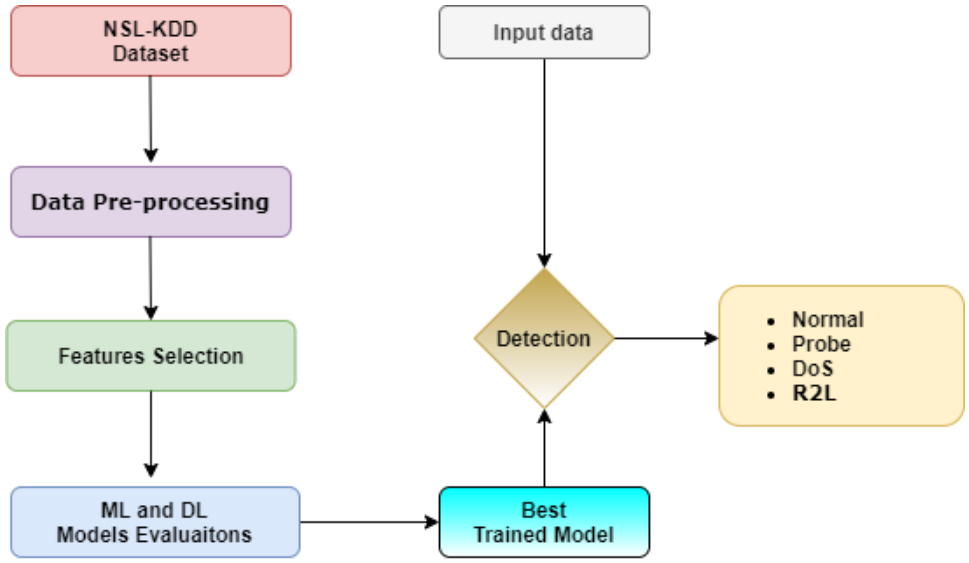

Figure 3 Proposed System Architecture

As illustrated in Figure 3, the focus of the approach lies in the feature selection process, which plays a key role in identifying the most pertinent attributes that can be utilized in the classification of instances. These attributes serve as the building blocks for the predictive models.

Then each ML and DL classifier employs the features selected by the feature selection to create knowledge. Knowledge is a set of algorithms that can be trained from a set of training datasets to learn how to make a decision. These models are then trained from the training dataset before being tested on a testing dataset to make their decisions about the class in any given instance. Once a model is trained, this is fed into the testing dataset which serves as the knowledge that the model will employ to make the appropriate decisions, when applied to new instances for classification. When this is done - we can measure the accuracy between the test dataset and the target dataset by calculating the evaluation metric that measures the extent to which these two

datasets reflect the same information. Using the features that have been carefully selected will significantly impact the model's ability to be able to detect and classify any new instance with a very high accuracy since their knowledge received is 'tuned' with precision.

3.4 Data Preprocessing

Data Preprocessing is a critical step in building robust and accurate predictive models. Several preprocessing tasks are performed on the cyber-attack prediction system to ensure the quality and improve the accuracy of the prediction model. The first step is Data cleaning and transformation. This process deals with missing values and finds a suitable solution in the dataset.

Firstly, after retrieving the whole volume of data and reading the file, then check if there are irrelevant columns in the dataset. The last_flag column has no other value for the model, and we cannot use it in the model, so we drop this column from the dataset.

Based on the information gathered about Feature engineering, a map of attacks was identified based on the Type of an attack and categorized them based on the Type of the attack. This step starts by mapping the attacks based on their types. To do this, all attacks with the same types are identified and assigned to their corresponding attack categories in Table 2. To do this a new column is formed called 'attack_class'. It is assigned the attack categories mapped to its attack type referenced in Table 1.

These attacks presented have now been mapped and placed in their categories. In this step, the attacks need to not only be mapped but also received new types assigned to them. However, it can be revealed from Table 2 that the attack types in this step are now called by their new categories and are given new values as mentioned.

Attack Label	DOS	Normal	Probe	R2L	U2R
Class	0	1	2	3	4

Table 2 Mapping integer value to attack labels.

For the next step, categorical data is preprocessed into numerical values using label encoding technique. Additionally, normalization technique mentioned requires rescaling attributes (numeric real valued) into '0-1' binary space. In this project, after downloading the datasets, for the preprocessing operation where data normalization is applied in the predictive techniques to build the model training less sensitive to the scale of attributes. This preprocessing allows the training model to converge to better weights which leads to getting the most accurate model.

By using the StandardScaler() function which is from the preprocessing library declared above the scaling of the dataset has been implemented in both datasets. Moreover, these huge datasets include lots of features, due to which the training model can affect overfitting problem. To overcome this problem the system uses normalization technique to change all feature values which have linked with similar values. This preprocessing allows the training model to converge to better weights which leads to getting the most accurate model.

3.5 Feature Selection

In the system, the dataset originally comprises of 41 features which makes the data to be sparse and poses challenges for classification tasks, as most of the high dimensional space remains empty [41]. It also results in algorithms taking additional time and resources to process the data and making it computationally expensive. To enhance the model's efficiency and interpretability, feature selection techniques are employed to curate a subset of relevant attributes. Chi-Squared

feature selection algorithm is used to get the top-K features for the models based on the cosine similarity for each independent column with respect to the target column.

The Chi-Squared test is a statistical method used to determine if there is any significant correlation and select the top k most relevant features [42]. So, in the feature selection process, only the correlated attributes will be selected, and the training model will be prepared. This test assesses the relationship among the variables using a contingency table by measuring the association between each feature and the target variable. SelectKBest class is used to extract the top 20 features which are mostly associated with the model based on the highest fit scores with the target variable. Now, these top 20 features are chosen to build the learning model and placed in a new final dataset with only the necessary columns. The top 20 specifications are mentioned below in Figure 4.

```
Top 20 features:
['src_bytes', 'dst_bytes', 'duration', 'count', 'dst_host_srv_count',
'dst_host_count', 'hot', 'srv_count', 'dst_host_srv_serror_rate',
'srv_serror_rate', 'serror_rate', 'dst_host_serror_rate', 'logged_in',
'num_root', 'num_compromised', 'dst_host_same_srv_rate',
'same_srv_rate', 'dst_host_same_src_port_rate',
'dst_host_diff_srv_rate', 'srv_rerror_rate']
```

Figure 4 Top 20 features chosen in Feature Selection

3.6 Building learning model

In the feature selection process, discovering the most suitable subset features for the model is achieved. Now depending on these features, the system learning models are built and implemented. To implement this learning model for the project, selective algorithms are chosen

based on suitability. The selection of algorithms significantly impacts the model's performance. Different algorithms serve different purposes, choosing the right algorithm based on the problem requirement is crucial. It will determine if the model can be released into real time applications to rightfully detect cyber-attacks, where organizations can rely on the output.

Understanding the problem type is the initial step for the decision of the algorithms. Is it a classification, regression, or clustering problem and whether the data is structured, unstructured or image base?

The given data set is clearly structured and well categorized, with a set of relevant features describing features of network traffic. The features consist of some quantitative features (e.g., `src_bytes`, `dst_bytes`, `duration` etc.), which measure the amount of data exchanged, and the duration of the traffic. Another set of features are qualitative (e.g., `attack_class`), describing which network activities belong to which 'attack class'.

The categorization task implicit in the dataset necessitates sophisticated ML and DL models, able to extract such patterns and asymmetries from analogous sea of network traffic data. The choice to utilize decision tree and Naive Bayes classifiers was driven by their capability of functioning well for structured data, where the values of the two dataset attributes (whether binary or categorical) form a structured representation of the activities in the network, and where decision trees emulate how data relationships unfold, and Naive Bayes uses probabilistic inference to distinguish between good and bad network behaviors.

In addition, the use of DL models such as Convolutional Neural Networks (CNNs), VGG11, and ResNet18 marks a new analysis in intrusion detection. Though initially developed for image classification, DL architectures can be used for sequence and tabular data easily.

Therefore, DL models can be valuable tools in assisting the development of intrusion detection systems. We hope to better understand the intricate patterns of network attacks and subtle features in this new attack data with the help of deep learning.

First, the models demonstrated are all examples of ways to convert raw data into insights that are aligned with the structure of the same dataset and relate intimately and tightly to the classification problem posed by intrusion detection. For example, because decision tree and Naive Bayes classifiers are particularly good at interpreting 'structured' data, they are well-suited to classifying records having distinct attributes that reflect the specific nature of intrusion documents. In clear contrast, DL models are especially good at surfacing abstract patterns and representations in complex data. Second, although ML approaches such as the ones we use here can be enhanced, in specific problems and with suitable techniques, to interpret and represent structured data, using DL to structure entity.

They are then implemented by feeding them to the training dataset to train the classifiers with the chosen features. In these algorithms, the last column can be defined as a predicable class defining the attack type. Each algorithm processes the training data, learning from the patterns and relationships within. The classifiers adapt to the specific task, becoming more adept at making predictions. Finally, the system builds the learning model depending on selected classifiers.

Dataset Name	**Number of Instances**	**Number of Features**	**Cyber attacks**	
KDD-99	125973	20	Normal: 67343	Malicious: 58630

Table 3 Information of the Dataset

3.7 Experimental Setup

Hardware:

Local Machine:

• Operating System: Windows 11 Pro

• Processor: 11th Gen Intel(R) Core(TM) i7-1165G7 2.80 GHz

• System Type: 64-bit operating system, x64-based processor

• Integrated GPU: Iris Xe Graphics G7 80EU

• Memory: 16.0 GB RAM

Software

• Python 3

• Sublime Text (IDE for development)

• Pandas

• Numpy

• Scikit-Learn

• Matplotlib

• Keras (Tensorflow for Deep Learning)

• Pyqt5 (Desktop Application GUI)

The experiment was carried out on a local machine with Windows 11 Pro operating system and 64-bit operating system, x64-based processor. The machine was an 11th Gen Intel(R) Core(TM) i7-1165G7 processor with 16.0 GB of DDR4 RAM, able to handle extensive data processing, training of models, and application execution.

A wide range of software tools and libraries were utilized for implementing machine learning, deep learning and developing a desktop application. Python was the primary programming language for building machine learning and deep learning models. A large number of libraries were used to model datasets, build, train and evaluate models, visualize models for insights and validate the performance predicted by the models. Some of the libraries used include pandas and NumPy for data pre-processing and manipulating data; sklearn to build, train and evaluate machine learning algorithms and matplotlib for bar chart visualization.

For deep learning models, Keras and TensorFlow were used. Keras is a neural networks API that was built using TensorFlow and is used to develop and train deep learning models easily and quickly. An interface called PyQt5 was used to create a graphical user interface (GUI) for the desktop application [43]. This interface makes it easier for users to interact with the application. The desktop application is written using Sublime Text which is an IDE used as the integrated development environment (IDE) used to write, test and debug application code. The client-side implementation of the application utilizes dialog boxes for output, ensuring a seamless user experience.

Chapter 4: Models

This section demonstrates the performance of the machine learning (ML) based models and the deep learning (DL) based models in detection and classification of cyber-attack. In this section, I first introduce the various ML models, followed by DL models used in the procurement of experimental setup.

4.1 Machine Learning Models

Machine learning models which are trained by algorithms by learning based on large datasets are widely used in cyber security and their power has been proven to allow reasonable decision making. The Decision Tree and Naive Bayes Classifier are two classical ML algorithms which are explored in this study.

4.1.1 Decision Tree

Decision Tree is a supervised learning algorithm typically used to solve classification and regression problems. It recursively splits the dataset into disjoint subsets by the values of the input variable which maximizes the purity of the output variable in each of the subset [44]. Decision Tree constitutes an understandable, interpretable, and human-readable algorithm capable of operating with both numerical and categorical data, excelling in extracting collaboration order from cybersecurity data and identifying the most indicative features of different types of cyber-attacks.

4.1.2 Naive Bayes Classifier

Naive Bayes Classifier is a probabilistic ML algorithm derived by combining Bayes' theorem and the conditional independence assumption, an assumption that all the features are independent of each other given the class label. Naive Bayes classifiers work surprisingly well in

practice despite their simple assumptions. In fact, they excel at text classification and spam filtering tasks. They are widely used in the cybersecurity world for both efficiency and scalability. Moreover, since they can deal effectively with high-dimensional data, they are suitable for detecting outliers and for identifying odd patterns within network traffic or system logs.

4.2 Deep Learning Models

Unlike the conventional ML approaches, in deep learning models, intricate neural architecture exposes intricate structures and patterns in data. Although the study of DL models in intrusion detection is yet to be finished, hopefully we can further include CNNs, VGG11, ResNet18, etc.

As an added benefit, deep learning models provide great flexibility in processing different data modalities, such as sequential, tabular, and image-based data. CNNs are known for their image classification strengths and get some preference as representatives in detecting various visual anomalies in network traffic data. The use of monitoring and instrumentation tools along with data collection helps in the smooth implementation of the models [45]. Both the models of VGG11 and ResNet18 are very deep and architecturally sophisticated and are known for their strong features of extracting and learning representations, which are crucial for intrusion detection.

4.2.1 CNN

The Convolutional Neural Network (CNN) is a deep learning architecture that is primarily used for training image processors for Object Detection tasks. However, it is also possible to apply it to one-dimensional data if we make some modifications to the architecture. In this article, we are going to explore the data inputs and modifications made to the CNN architecture to work with one-dimension data [46].

```
def cnn_evaluation():

    x, y = training_features()
    x = np.array(x, dtype=np.float16)
    x = x.reshape(x.shape[0], x.shape[1], 1)
    xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.30,random_state=27)

    model = Sequential()
    model.add(Conv1D(32, 2, activation="relu", input_shape=(20, 1)))
    model.add(Dense(16, activation="relu"))
    model.add(MaxPooling1D())
    model.add(Flatten())
    model.add(Dense(5, activation='softmax'))
    model.compile(loss='binary_crossentropy',optimizer="adam",
                  metrics=['accuracy'])
    #model.summary()
    model.fit(xtrain, ytrain, batch_size=64, epochs=5, verbose=1)
```

Figure 5 Algorithm Implementation for Training and Evaluating CNN Model

Components are broken down into multiple layers as mentioned above in Figure 5. Let's go more into detail on these convolution layers.

Input Layer: A one-dimensional signal is fed to the system, such as a temporal signal with a sequence of data points as input, as is the case in the input shape specified in the code snippet: input_shape = (20, 1) This specifies that the length of each input sequence (the second 20 in the pair) is 20 data points, while its dimensionality (the first 20 in the pair) is one.

Convolutional Layer: The first layer of the CNN consists of a convolutional layer which takes the input and applies filters on top. In the code, we can see that we are first using a Conv1D layer with 32 filters and with kernel size 2. This layer convolves each filter in the input sequence to extract local features or local patterns.

Activation Function: A convolution layer which does not include an activation function as part of its process is performed next, followed by an activation function (AF), and so on. The AF allows the network to learn increasingly complex relationships in the data.

Dense Layer: From this convolutional layer, we then pass the features to a dense layer with 16 units (fully connected layer) with ReLU activation function for further processing. In this layer, a linear transformation with 3×3 kernel size is calculated on the input features and then passed to the ReLU activation function.

MaxPooling Layer: With the MaxPooling1D layer we sample the feature maps along the time axis, reducing the dimensions of the feature-maps by retaining the most significant features – by default a window of size 2 is used in the code below.

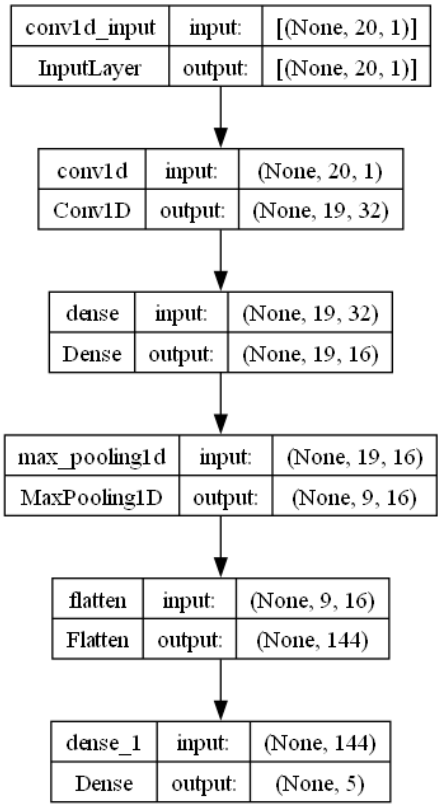

Figure 6 CNN Model Architecture

Flatten Layer: At the end of the pooling layer, we add another layer called the Flatten layer, where the feature maps are flattened into a one-dimensional vector for the next fully connected layers to take as input.

Output Layer: The last layer of the CNN is a dense layer with 5 units according to the number of output classes. Here we use the softmax activation function that produces the probability distributions over the classes, which will let the model make predictions.

Compilation and Training: Model is fitted with the binary cross-entropy loss function, the Adam optimizer and accuracy metric, and trained on the training data for 5 epochs with a batch size of 64.

This version of the CNN architecture as shown in Figure 6 with Conv1D layers is well suited to analyzing one-dimensional data, e.g., for time series classification or for sequence prediction applications.

4.2.2 VGG-11

VGG11 is a convolutional neural network architecture that was originally designed for image classification tasks. However, when you mention "conv1d," you're likely referring to a modification of the original VGG architecture to accommodate 1-dimensional convolutions instead of 2-dimensional convolutions. The input data might be 2-dimensional but if it’s 1-dimensional (e.g., an audio file or text), then the process is to swap the filters and receptive fields into a 1-dimensional setting. This modification can be useful for tasks such as time series analysis or sequence modeling, where the input data is one-dimensional.

```python
def VGG11(self):
    inputs = tf.keras.Input((self.length, self.num_channel))  # The input tensor
    # Block 1
    x = Conv_1D_Block(inputs, self.num_filters * (2 ** 0), 3)
    if x.shape[1] <= 2:
        x = tf.keras.layers.MaxPooling1D(pool_size=1, strides=2, padding="valid")(x)
    else:
        x = tf.keras.layers.MaxPooling1D(pool_size=2, strides=2, padding="valid")(x)

    # Block 2
    x = Conv_1D_Block(x, self.num_filters * (2 ** 1), 3)
    if x.shape[1] <= 2:
        x = tf.keras.layers.MaxPooling1D(pool_size=1, strides=2, padding="valid")(x)
    else:
        x = tf.keras.layers.MaxPooling1D(pool_size=2, strides=2, padding="valid")(x)

    # Block 3
    x = Conv_1D_Block(x, self.num_filters * (2 ** 2), 3)
    x = Conv_1D_Block(x, self.num_filters * (2 ** 2), 3)
    if x.shape[1] <= 2:
        x = tf.keras.layers.MaxPooling1D(pool_size=1, strides=2, padding="valid")(x)
    else:
        x = tf.keras.layers.MaxPooling1D(pool_size=2, strides=2, padding="valid")(x)

    # Block 4
    x = Conv_1D_Block(x, self.num_filters * (2 ** 3), 3)
    x = Conv_1D_Block(x, self.num_filters * (2 ** 3), 3)
    if x.shape[1] <= 2:
        x = tf.keras.layers.MaxPooling1D(pool_size=1, strides=2, padding="valid")(x)
    else:
        x = tf.keras.layers.MaxPooling1D(pool_size=2, strides=2, padding="valid")(x)

    # Block 5
    x = Conv_1D_Block(x, self.num_filters * (2 ** 3), 3)
    x = Conv_1D_Block(x, self.num_filters * (2 ** 3), 3)
    if x.shape[1] <= 2:
        x = tf.keras.layers.MaxPooling1D(pool_size=1, strides=2, padding="valid")(x)
    else:
        x = tf.keras.layers.MaxPooling1D(pool_size=2, strides=2, padding="valid")(x)

    # Fully Connected (MLP) block
    x = tf.keras.layers.Flatten(name='flatten')(x)
    x = tf.keras.layers.Dense(4096, activation='relu')(x)
    x = tf.keras.layers.Dense(4096, activation='relu')(x)
    if self.dropout_rate:
        x = tf.keras.layers.Dropout(self.dropout_rate, name='Dropout')(x)
    outputs = tf.keras.layers.Dense(self.output_nums, activation='linear')(x)
    if self.problem_type == 'Classification':
        outputs = tf.keras.layers.Dense(self.output_nums, activation='softmax')(x)

    # Create model.
    model = tf.keras.Model(inputs=inputs, outputs=outputs)

    return model
```

Figure 7 Implementation of VGG Models for Cyber Attack Detection

Let's break down the components of VGG11 and how they might be adapted for 1D convolutions as shown in Figure 6

Input Layer: The input to the network would be a 1-dimensional signal, such as a time series or sequence of text embeddings.

Convolutional Layers: VGG11 is composed of multiple convolutional layers. As stated in the name of this layer that the model uses convolution to perform operations on the input data and

extract features. Each convolutional layer utilizes a set of filters to extract features. In the case of conv1d, these filters slide across the input sequence in one dimension, capturing patterns in the data. For example, a moving filter might detect short-term patterns in a time series or sequential dependencies in text data.

Activation Functions: These usually take place at the output of each convolutional operation, these act as a non-linearity to again maximize the network's ability to discover complicated patterns, for example the ReLU (Rectified Linear Unit) applies a discontinuous activation function.

Pooling Layers: After convolutional layers, pooling layers are often applied to down sample the features and reduce the dimensionality of the data. In VGG11, max-pooling layers are commonly used. Max-pooling involves taking the maximum value from each window of the feature map. In 1D max pooling, the operation is performed along the time axis.

Fully Connected Layers: After the convolutional and pooling layers in the hierarchy, any learned features are flattened and passed through one or more fully connected layers that can perform classification given the features the network learned.

After declaring the convolutional and pooling layers, the extracted features are inserted and passed through one or more fully connected layers.These layers perform classification based on the learned features.

Output Layer: The output layer used is softmax activation function, especially for classification tasks, which produces probabilities for each class.

4.2.3 ResNet-18

ResNet-18 is a popular convolutional neural network architecture that is well-known for its effectiveness in image classification tasks. However, similar to VGG11, it can be adapted for

1-dimensional convolutions (conv1d) to handle one-dimensional data such as time series or sequences. ResNet-18 is a popular convolutional neural network architecture that is well-known for its effectiveness in image classification tasks. However, similar to VGG11, it can be adapted for 1-dimensional convolutions (conv1d) to handle one-dimensional data such as time series or sequences.

Through substitution of 2-D convolutional layers with 1-D convolutions [46], what started out as a promising image classifier emerges as a method of processing data in thick-slices of time while retaining the benefits of the original ResNet-18: residual connections and a hierarchical feature-extraction architecture. The model can be trained on a range of time series data, from images to signals such as heart rate and financial price fluctuations One of the advantages of training the otherwise generic ResNet-18 with limited datasets is that the complex nonlinearity of the architecture works well in a surprising range of time series contexts.

Let's break down how ResNet-18 can be adapted for conv1d as shown in Figure 7.

Input Layer: The input to the network would be a 1-dimensional signal, such as a time series or sequence of text embeddings.

Convolutional Layers: In the original ResNet-18 architecture, the initial layers consist of 2D convolutional operations with 3x3 filters. For the adaptation to conv1d, these convolutional layers would be modified to perform 1D convolutions. Instead of sliding 2D filters across the image, 1D filters would slide along the input sequence, capturing patterns and features in one dimension.

```
def resnet_18(input_shape, num_classes):
    inputs = Input(shape=input_shape)

    # Initial convolution layer
    x = Conv1D(64, kernel_size=7, strides=2, padding='same')(inputs)
    x = BatchNormalization()(x)
    x = Activation('relu')(x)
    x = MaxPooling1D(pool_size=3, strides=2, padding='same')(x)

    # Residual blocks
    x = basic_block(x, filters=64, stride=1)
    x = basic_block(x, filters=64, stride=1)

    x = basic_block(x, filters=128, stride=2)
    x = basic_block(x, filters=128, stride=1)

    x = basic_block(x, filters=256, stride=2)
    x = basic_block(x, filters=256, stride=1)

    x = basic_block(x, filters=512, stride=2)
    x = basic_block(x, filters=512, stride=1)

    # Global average pooling and fully connected layer
    x = AveragePooling1D(pool_size=x.shape[1])(x)
    x = Flatten()(x)
    outputs = Dense(num_classes, activation='softmax')(x)

    model = Model(inputs=inputs, outputs=outputs)

    return model
```

Figure 8 Algorithm Implementation for Training and Evaluating ResNet-18 Model

Residual Blocks: ResNet architecture consists of residual blocks as the main units. Each residual block is comprised of two times convolution with batch normalization and ReLU activation, and a 'shortcut' that allows input of a particular feature map to jump directly to the output of convolution and ReLU combination, without going through the convolution with batch

normalization and ReLU units. When the conv1d adaptation is performed, the convolutional units inside the residual blocks are changed into 1D convolutions.

Pooling Layers: In the original ResNet architecture, max pooling is used to down sample feature maps spatially. However, in the context of conv1d, max pooling can be used to down sample along the time axis.

Fully Connected Layers: After many convolutional and pooling layers, the features are flattened and passed through some number of fully connected layers to distinguish classes.

Output Layer: The output layer is usually made up of a softmax activation, in which case it outputs probabilities for each class, given the unusual nature of the problem.

Adapting ResNet-18 for conv1d involves modifying the convolutional layers to operate in one dimension instead of two while retaining the residual connections that facilitate the training of deeper networks.

Chapter 5: Results

5.1 Evaluation Metrics

The performance predicted by the machine learning and deep learning models is generally evaluated in terms of widely utilized metrics such as precision, recall, the F1 score and accuracy. Precision and recall evaluate how well models distinguish normal from malicious network traffic, and the F1 score balances accuracy in these assessments [47].

Precision: The accuracy of a model's predictions is determined by the precision metric. It is the proportion of the true positive instances that models predicted as positive among all the instances that models predicted as positive. It can be calculated using the formula mentioned below, as the number of predictions made by the system which were correct (True Positives) divided by the total number of the system's positive predictions, correct (True Positives) and incorrect (False Positives).

$$Precison = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Precision is a very relevant metric in cybersecurity, especially in the context of intrusion detection and prediction of cyber-attacks. In cybersecurity, precision is important because it measures the capability of the model to make as few false positives as possible. False positives are instances in which normal network traffic is tagged as malicious traffic. In a wide area network, this can lead to large amounts of normal but misidentified traffic slowing down network performance. Suppose the network identifies 20% of normal network traffic as malicious; this slows down the network by 20%. Additionally, such alerts can lead to unnecessary actions from security officers on the ground. Due to this possibility, high precision in predictive models is crucial so that the firms can trust the alerts generated by the IDS technology, avoiding false alarms,

and preventing undue attention to insignificant issues, thereby allowing them to work on genuine breaches.

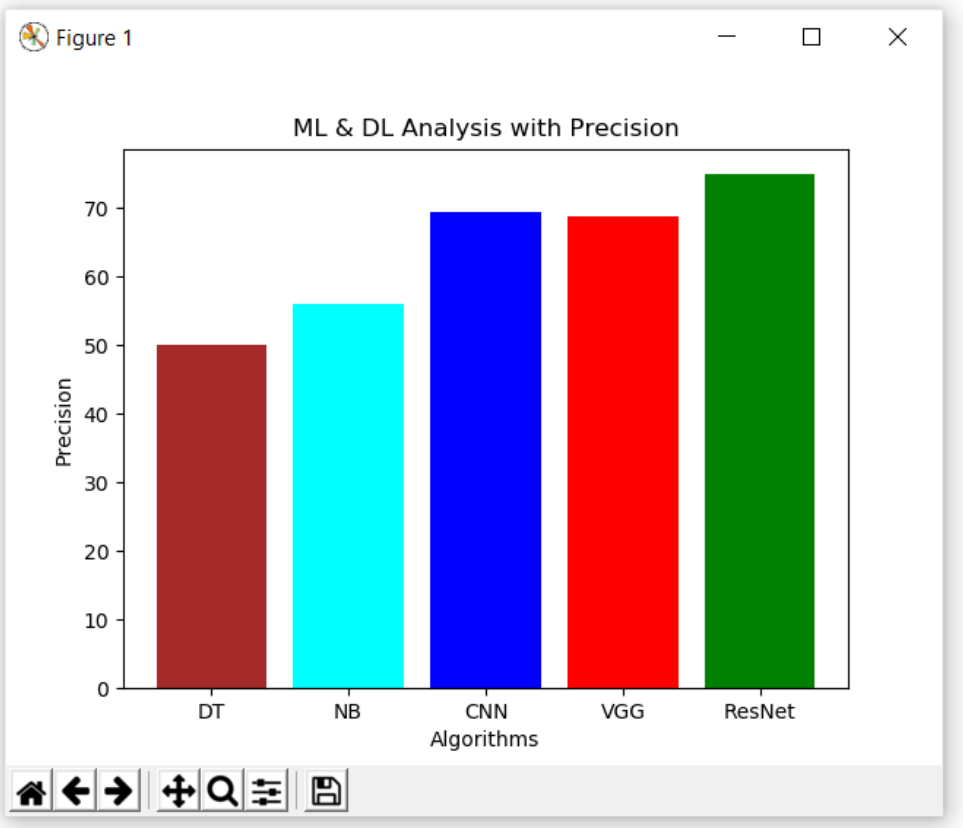

Figure 9 Precision of ML and DL Classifiers

From figure.8, this system will be displayed in the bar chart with the precision of classifiers such as DT is 50.12 percent, NB is 56.08 percent, CNN is 69.40 percent, VGG11 is 68.80 percent, and ResNet18 is 74.93 percent.

Recall: Recall or sensitivity is the ratio of actual test results generated by the system that resulted in positive reads that actually correlated with positive observations called True Positives, to all observations in the actual malicious class called Actual Positives. In general, it evaluates the extent to which trained models can detect malicious traffic.

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

It's equally critical that the model has a high recall when applied to cyber-attacks as evaluated by real-world performance. False negatives, where malicious traffic is not detected by an intrusion detection system, can cause security breaches and data compromises. High recall helps to ensure that all instances of intrusive traffic are identified and flagged by the system – potentially reducing the risk of latent threats and improving the organization's security posture. In short, it is used to check - of all the total attacks that are malicious, how many did the system flag as attacks?

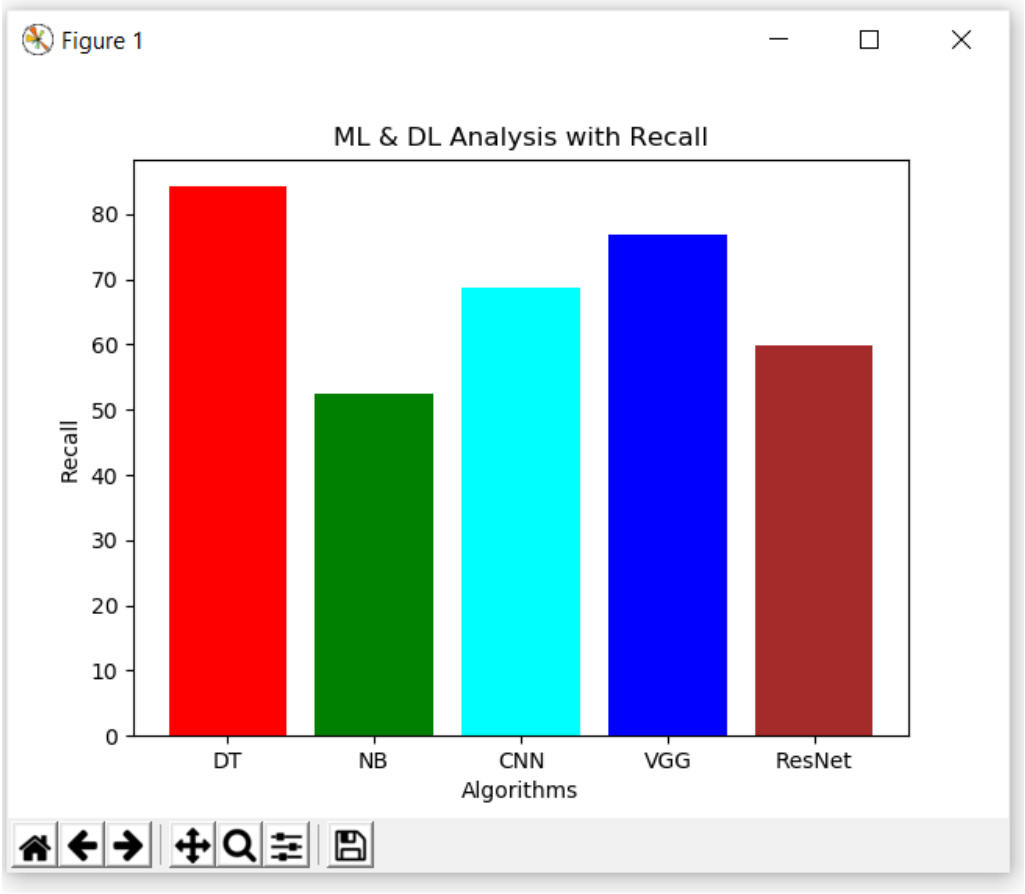

Figure 10 Recall of ML and DL Classifiers

From figure.9, this system will be displayed in the bar chart with the recall of classifiers such as DT is 84.21percent, NB is 52.47 percent, CNN is 68.82 percent, VGG11 is 76.77 percent, and ResNet18 is 59.91 percent.

Accuracy: Accuracy measures the overall correctness of predictions, which indicates the correct classification over the total instances evaluated. Accuracy provides a high-level description of the correctness of the model. It is simply the ratio of the correct classifications (True Positives + True Negatives) to the total Test Dataset. It indicates how many instances the system correctly classified for the network intrusion model (both True Positives + True Negatives) over the total of the dataset.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + False\ Positives + False\ Negatives + True\ Negatives}$$

From the below Figure.10, this system will be displayed in the bar chart with an accuracy of classifiers such as DT is 73.24 percent, NB is 80.28 percent, CNN is 96.38 percent, VGG11 is 97.87 percent, and ResNet18 is 97.17 percent.

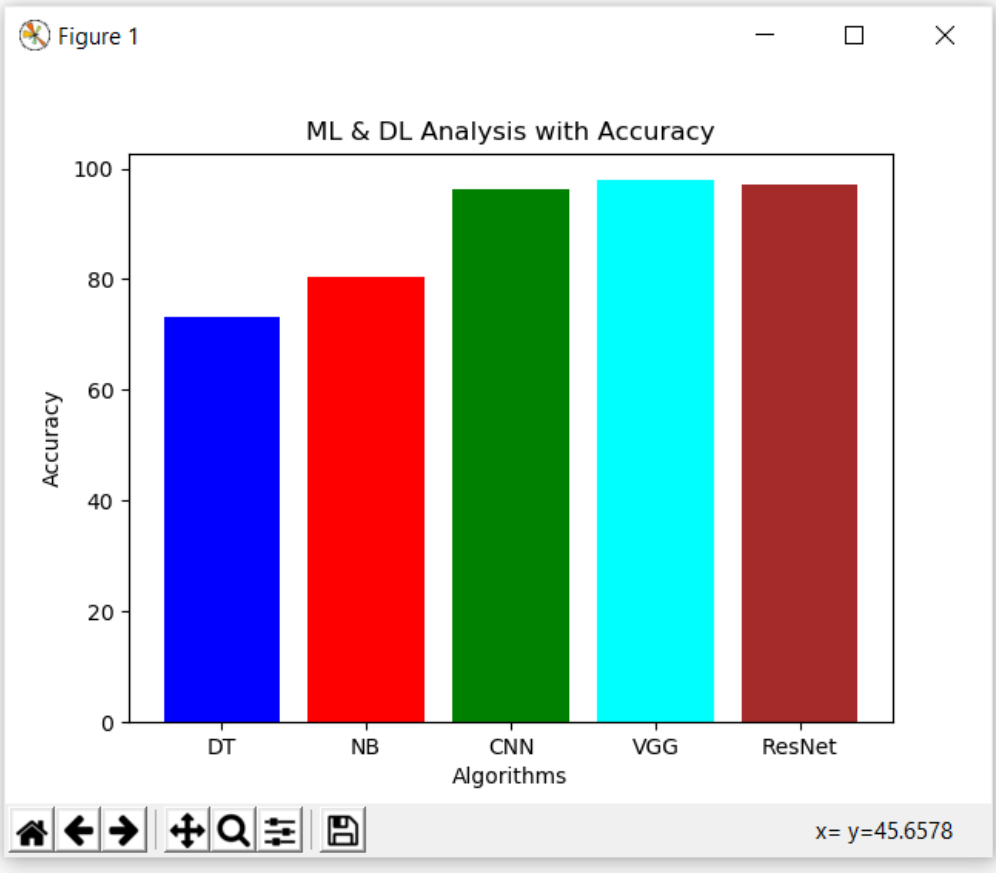

Figure 11 Accuracy of ML and DL Classifiers

F1 score: F1 score is the harmonic mean of precision and recall, which balances between extremes of precision and recall. It is the favored overall performance assessment in statistics amongst other evaluation metrics. This is because F1 score is the harmonic or weighted average of Precision and Recall. It gives a balance between how well the model is performing. Therefore, this measure balances the cost of False Positives and False Negatives between precision and Recall.

$$F1\ Score = \frac{2\ X\ (Precision\ X\ Recall)}{Precision + Recall}$$

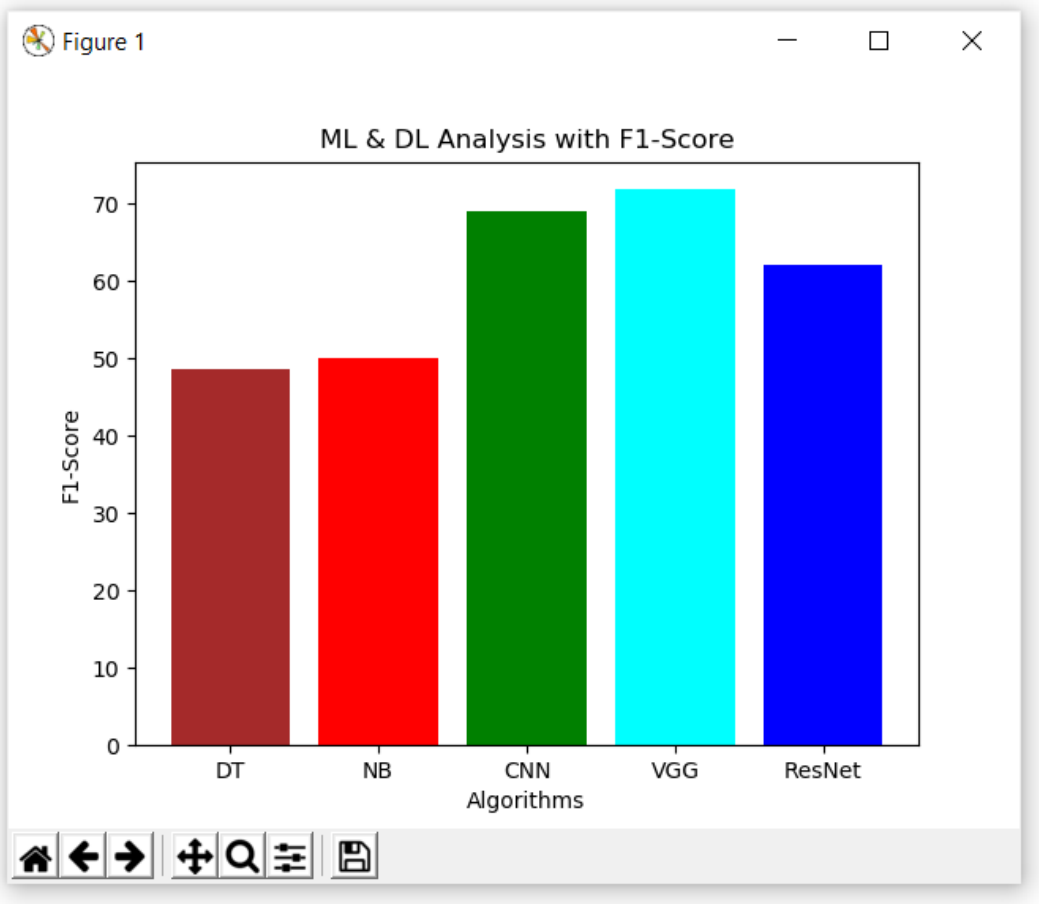

Figure 12 F1-Score of ML and DL Classifiers

From Figure.11, this system will be displayed in the bar chart with the f1-score of classifiers such as DT is 48.65 percent, NB is 50.09 percent, CNN is 69.05 percent, VGG11 is 71.86 percent, and ResNet18 is 62.17 percent.

5.2 Model Results

Table 3 reveals the numerical scores from the prediction of ML and DL approaches according to their accuracy, precision, recall and F1-score.

ML & DL Models	Accuracy	Precision	Recall	F1-Score
DT	73.24	50.12	84.21	48.65
NB	80.28	56.08	52.47	50.09
CNN	96.19	70.11	59.05	69.34
VGG11	**97.87**	**68.80**	**76.77**	**71.86**
ResNet18	97.17	74.93	59.91	62.17

Table 4 ML and DL model performance evaluations

Decision Tree:

The Decision Tree model had an accuracy of 73.24%, which is relatively acceptable, therefore it is able to classify correctly the network traffic. The precision and F1-score values though, which represent the proportion of the correctly classified instances out of the instances that were labeled by the model as being in a specific category (e.g., malicious), is really poor (precision of 50.12%, F1-score of 48.65%). This can clearly be noticed from the high recall rate of 84.21% and low precision (Figure 8). It seems that the model detects the highest rate of malicious traffic but those identified instances (recall) also incorporate a large number of false positives, which is indicated by the relatively low precision. The relatively low precision indicates that out of the traffic detected by the model as malicious, a large percentage can be benign.

Naive Bayes :

The accuracy and recall values of the Naive Bayes model slightly improved compared to DT, with 80.28% and 66.17%, respectively. However, similarly to DT, the precision and F1-score values were relatively low, at 56.08% and 50.09%, respectively. Here, the recall value (52.47%) was lower compared to DT because of more false negatives. This means that although NB is a simple model with speedy computation, it did not perform well in classifying network traffic to identify benign or malicious instances accurately.

Convolutional Neural Networks (CNN):

The CNN model had an accuracy of 96.19%, which was the best out of all the ML models, with the precision and recall values at 70.11% and 59.05%, respectively, and achieving F1-score of 69.34%. This shows that CNN is able to have a robust capability to identify and classify cyber-attacks on network traffic with high accuracy. Higher classification accuracy was achieved because of the spatial covariates embedded in the data and the way that CNN learned presumptive spatial organization and hierarchical representations to internally model the covariate-response relationship.

VGG11:

Compared to CNN, VGG11 performed a little better with an accuracy of 97.87%. Similarly, the precision rose to 68.80% and the recall rose to 76.77%, yielding an impressive F1-score of 71.86%. And the improvement in performance can reasonably be attributed to the increased depth of the architecture, with more complex image features and representations learned from the data. The pretrained weights we used also made a significant contribution to this remarkable improvement.

ResNet18:

Another modern and robust deep convolutional neural network architecture – the ResNet18 – was trained on the same network. The training process for ResNet18 lasted for 48 epochs and resulted in the accuracy of 97.17 per cent. The precision and recall were a bit better than VGG11 – 74.93 per cent and 59.91 per cent, respectively, giving us the F1-score of 62.17 per cent. These results show that ResNet18 has slightly better precision than VGG11 but worse recall, which is to say that it allows for a lower number of false negatives. ResNet18 proved especially effective when it comes to predicting cyber-attacks using residual connections to cope with the issue of the vanishing gradient and train deeper networks.

Looking at the results we can infer that Deep Learning models outperform traditional Machine Learning models as far as accuracy and F1-score are concerned. Specifically, VGG11 and ResNet.

Chapter 6: Conclusion and Future Work

The approach is to examine if ML and DL techniques are an appropriate use case for Intrusion Detection Systems (IDSs) in terms of their ability to distinguish and classify diverse types of cyber-attacks; and if further improvements are possible using ML and DL techniques. With real world datasets, I give my own approach for use and evaluation of the performance and robustness of various types of ML and DL techniques using a variation of different machine learning algorithms; the weakness and strengths of target intrusion detection models; and propose an engineering perspective for future researches in the cybersecurity field.

The last part of the research plan is the conclusion section, where we finalize the research by summarizing the results and the useful outcome of the research the author got like how good the ML algorithms like are, Decision Tree, Naive Bayes Classifiers to detect cyberthreats or the DL models like Convolutional Neural Networks (CNN), VGG11, ResNet18 to detect cyberthreats. Then, we also discuss the limitation of this research or challenges that we face during this research so that people can figure out what is the situation of this research and what were the challenges that the author faced in order to overcome those challenges or mitigate those limitations.

Besides, the discussion section concludes with insights into possible future works, such as further exploration on the subject of anomaly detection, the development of hybrid ML-DL models, or the implementation of adaptive learning systems that could counter the complexities brought about by the continuous evolution of cyber threats, to name a few. Another important aspect touched upon in the last part of the paper is the ethicality of AI-assisted security systems, where it is suggested that core values of justice, fairness and accountability should predominate in any type of algorithm-driven decision-making processes [3].

6.1 Summary

In this research study, we take an initiative to investigate intrusion detection systems (IDS) employing machine learning (ML) and deep learning (DL) techniques based on the idea to investigate the capacity of ML and DL models in detecting and classifying cyber-attacks on organization networks. Through experimentation and analysis, we attempt to highlight the potential of these models in securing cybersecurity practices and preventing malware activities from various hackers.

We explored several classical ML algorithms, such as Decision Tree and Naive Bayes Classifier, and more involved state-of-the-art DL architectures as well. The results were encouraging and revealed that all models studied handled structured data insightfully. These models succeeded in identifying suspicious activities by extracting rules and features that support suspicious and malicious behavior. It is why they played a prominent role in intrusion detection.

Secondly, we also applied the findings to the DL models by adding Convolutional Neural Networks (CNNs), VGG11 and ResNet18. These architectures have shown powerful capability in dealing with data of various modalities for building DL models, especially the CNNs models specially designed for image data workload to extract high-level features in images. They may be an exciting strategy in the race against malware.

These results point to the value of a hybrid detection approach that takes advantage of the complementary strengths of ML- and DL-based methods – previous attempts had combined the two approaches to train neural networks, but the findings show that combining the interpretability of classical ML solutions with the representational power of DL architectures is key. In this way, organizations can better defend their networks from evolving cyberattacks.

But precision and recall are a difficult balance – high precision minimizes false positives, but must be balanced with recall, both of which are sensitive to the choice of model and its parameters. As the results suggest, practitioners should be thoughtful and conscientious in navigating this space.

Feature engineering is key. The features we can extract from the raw data – whether network traffic logs, system logs, user behavior or other kinds of data – will feed into the knowledge the models accumulate over time. The question we then face is: what are the most pertinent features to extract, such that they explain the 'cyber threat' we are after? Research in the future might also examine automatically learning which features best explain what we are after, relieving analysts of the burden of doing this manually.

An interpretation dilemma is some models can be understood, but the more accurate the predictions they deliver, the more difficult they are to comprehend. While Deep Learning may be quite accurate, Decision Trees are at the opposite end of the spectrum, highly interpretable. Deep neural networks, though, are sometimes best described as a black box: humankind has no real understanding of how they produce their results. Finding a good balance between model fit and understandability is a continuing objective and there's some promise in the explainable AI techniques that offer to translate the output of these complex models into something humans can appreciate.

Another issue highlighted by the results is that of the persistent skew of datasets. In cybersecurity, instances of good behavior outweigh instances of bad behavior. There are several approaches for mitigating skews (oversampling the minority class; under sampling the majority class; synthetic data generation). We also need improvement in methodologies relating to outlier detection – perhaps the next most useful frontier for new threats.

In short, the work highlights not only how effective machine learning and deep learning models can be in cybersecurity, but also the fine line between being effective and causing more harm in real-world deployment. Addressing this line will make cybersecurity more robust in the future.

6.2 Limitations

While the work makes an important contribution to the field of cybersecurity through machine learning (ML) and deep learning (DL) models, it is imperative to recognize the limitations which limit the extent of the work and its consequences. A major concern about the datasets used for the experiments is the difficulty of obtaining high-quality data sets in the first place. In that case, I tried to consider reputable data sets such as the NSL-KDD. However, biases or incompleteness in the data could have, unknowingly, affected how well the models perform.

Another important limitation, for example, is the inherent bias in the data we use in training and testing the models. It is necessary to create datasets containing representative online threats that are as unbiased as possible – reflecting what is currently happening online. However, the way the data was used to train the model may have contributed bias through the source of the data, the timing of its collection, the sampling technique used and so on. If we cannot generalize the past to future cyber events through any of these factors, we will be convinced that the ML models work when in fact they do not. Dataset bias is a known challenge in ML research that will require attention and vigilance on an ongoing basis.

Moreover, the ability to scale the models to large-scale enterprise networks and diverse environments will be limited. Notably, experiments conducted in the work do not capture the nuanced and complex dynamics of real-world cybersecurity environments, in which we observe various heterogeneous network architectures, different human usage patterns and advanced adversarial techniques. Scaling the models to generalize across diverse and dynamic modern IT infrastructures poses a significant challenge for future work. A further challenge is related to practicalities. We would need to consider the constraints on computational and resource infrastructure requirements for training and deployment of DL models. For example, organizations lacking capital investment and/or specialist expertise in computer science may struggle to accommodate to the high computational costs associated with training the DL model on large datasets and/or utilizing hardware accelerators for training.

Additionally, its lack of explain ability makes it difficult to decipher how the DL model made its decision and behaves, although DL models are sometimes found to outperform their counterparts in a wide variety of complex tasks. Autonomous technology is difficult to trust given concerns about accountability. The explain ability and transparency problem is a subject of ongoing research to enhance the trustworthiness of DL models in many high-stakes areas including cybersecurity.

Despite these limitations, the research can help to provide better insights and set the path for future advancements in ML and DL-based cybersecurity solutions. Addressing these challenges and striving for better approaches can help to build more resilient, adaptive and trustworthy ML-based cybersecurity to mitigate new emerging threats and protect the critical digital assets.

6.3 Future Work

Moving forward, there are several potential paths for research in intrusion detection, including the use of anomaly detection techniques in conjunction with ML or DL modelling methods to detect strategic cyber-attacks that may otherwise go unnoticed. Unsupervised learning methods may help organizations identify and mitigate zero-day exploits and new forms of attacks.

Furthermore, hybrid models that have the best of the ML and the DL architectures could be developed through ensemble methods or hybrid architectures [48]. Ensemble methods or hybrid architectures could be built to help researchers with real-world intrusion detection deployment that can benefit from the complementary effectiveness of multiple models.

Future work in the Adaptive Learning Systems is showing promising results for effectively countering evolving cyber threats. Does the models have the capacity to evolve on the fly to new attack vectors? One key area for further exploration is investigating models that grow alongside threats, known as adaptive learning systems.

Since we're talking about human lives, however, as well as computer-mediated lives, and computer-automated life-and-death decisions, the 'beyond algorithms' element also applies. Think human-in-the-loop and human-centric approaches. If we give the appropriate tools to security analysts in the field of such automated systems, they can often deliver more useful, more useable, more interpretable information. It's a brave new world of explainable AI, the challenges of which are often addressed when the analysis can be better and more human-centered.

As AI takes on security tasks, ethics emerges as a catch-all category that poses key questions. In what ways can AI be biased? What are the relevant privacy and accountability issues? In conclusion, it can be stated that recent advancements in disruptive technologies of AI, ML and

DL have paved the way for proactive, reactive and preventive intrusion detection and adaptive learning systems. However, it is important to adopt a collaborative, holistic, ethical, transparent and human-centric approach so that the world of data can be effectively learned, understood and utilized in a manner.

References

[1] J. Jabez and B. Muthukumar, “Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach,” *Procedia Computer Science*, vol. 48, pp. 338–346, 2015, doi: https://doi.org/10.1016/j.procs.2015.04.191.

[2] M. Shardlow, “An Analysis of Feature Selection Techniques.” Available: http://syllabus.cs.manchester.ac.uk/pgt/2019/COMP61011/goodProjects/Shardlow.pdf

[3] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Jul. 2009, doi: https://doi.org/10.1109/cisda.2009.5356528.

[4] A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and O. O. Ekabua, “Conceptualisation of Cyberattack prediction with deep learning,” *Cybersecurity*, vol. 3, no. 1, Jun. 2020, doi: https://doi.org/10.1186/s42400-020-00053-7.

[5] D. Torre, F. Mesadieu, and Anitha Chennamaneni, “Deep learning techniques to detect cybersecurity attacks: a systematic mapping study,” *Empirical Software Engineering*, vol. 28, no. 3, May 2023, doi: https://doi.org/10.1007/s10664-023-10302-1.

[6] O. Ben Fredj, A. Mihoub, M. Krichen, O. Cheikhrouhou, and A. Derhab, “CyberSecurity Attack Prediction: A Deep Learning Approach,” *13th International Conference on Security of Information and Networks*, Nov. 2020, doi: https://doi.org/10.1145/3433174.3433614.

[7] J. Hussain Kalwar and S. Bhatti, “Deep Learning Approaches for Network Traffic Classification in the Internet of Things (IoT): A Survey.” Accessed: Apr. 08, 2024. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/2402/2402.00920.pdf

[8] B. Cao, C. Li, Y. Song, and X. Fan, “Network Intrusion Detection Technology Based on Convolutional Neural Network and BiGRU,” *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–20, Apr. 2022, doi: https://doi.org/10.1155/2022/1942847.

[9] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *Journal of Big Data*, vol. 8, no. 1, Mar. 2021, Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8

[10] H. Hammouchi, O. Cherqi, G. Mezzour, M. Ghogho, and M. E. Koutbi, “Digging deeper into data breaches: An exploratory data analysis of hacking breaches over time,” *Procedia Computer Science*, vol. 151, pp. 1004–1009, 2019, doi: https://doi.org/10.1016/j.procs.2019.04.141.

[11] A. Kuehn and M. Mueller, “Shifts in the Cybersecurity Paradigm,” *Proceedings of the 2014 workshop on New Security Paradigms Workshop - NSPW ’14*, 2014, doi: https://doi.org/10.1145/2683467.2683473.

[12] M. A. Alsoufi *et al.*, “Anomaly-Based Intrusion Detection Systems in IoT Using Deep Learning: A Systematic Literature Review,” *Applied Sciences*, vol. 11, no. 18, p. 8383, Sep. 2021, doi: https://doi.org/10.3390/app11188383.

[13] Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, “Rule Generation for Signature Based Detection Systems of Cyber Attacks in IoT Environments,” *Bulletin of Networking,*

Computing, Systems, and Software, vol. 8, no. 2, pp. 93–97, Jul. 2019, Available: http://ww.bncss.org/index.php/bncss/article/view/113

[14] W. Tounsi and H. Rais, "A survey on technical threat intelligence in the age of sophisticated cyber attacks," *Computers & Security*, vol. 72, pp. 212–233, Jan. 2018, doi: https://doi.org/10.1016/j.cose.2017.09.001.

[15] A. Hussain and P. Kumar Sharma, "Efficient Working of Signature Based Intrusion Detection Technique in Computer Networks," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 60–64, Mar. 2019, doi: https://doi.org/10.32628/cseit195215.

[16] A. Wagholi, "Detection of DDoS Attacks based on Network Traffic Prediction and Chaos Theory," 2014. Accessed: Apr. 08, 2024. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=bf5d7b4b84818d084f8d047a99a555e88036e6ac

[17] I. A. Gheyas and A. E. Abdallah, "Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis," *Big Data Analytics*, vol. 1, no. 1, Aug. 2016, doi: https://doi.org/10.1186/s41044-016-0006-0.

[18] H. Albasheer *et al.*, "Cyber-Attack Prediction Based on Network Intrusion Detection Systems for Alert Correlation Techniques: A Survey," *Sensors*, vol. 22, no. 4, p. 1494, Feb. 2022, doi: https://doi.org/10.3390/s22041494.

[19] H. Ahmetoglu and R. Das, "A comprehensive review on detection of cyber-attacks: Data sets, methods, challenges, and future research directions," *Internet of Things*, p. 100615, Sep. 2022, doi: https://doi.org/10.1016/j.iot.2022.100615.

[20] M. Husak, J. Komarkova, E. Bou-Harb, and P. Celeda, "Survey of Attack Projection, Prediction, and Forecasting in Cyber Security," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 640–660, 2019, doi: https://doi.org/10.1109/comst.2018.2871866.

[21] A. B. de Neira, B. Kantarci, and M. Nogueira, "Distributed denial of service attack prediction: Challenges, open issues and opportunities," *Computer Networks*, vol. 222, p. 109553, Feb. 2023, doi: https://doi.org/10.1016/j.comnet.2022.109553.

[22] P. Datta, N. Lodinger, A. S. Namin, and K. S. Jones, "Cyber-Attack Consequence Prediction," *arXiv:2012.00648 [cs]*, Dec. 2020, Available: https://arxiv.org/abs/2012.00648

[23] P. Goyal *et al.*, "Discovering Signals from Web Sources to Predict Cyber Attacks," *arXiv.org*, Jun. 08, 2018. https://arxiv.org/abs/1806.03342

[24] K. Goztepe, "Designing Fuzzy Rule Based Expert System for Cyber Security," *International Journal of Information Security Science*, vol. 1, no. 1, pp. 13–19, Apr. 2012, Available: https://dergipark.org.tr/en/pub/ijiss/issue/16057/167848

[25] P. Radanliev *et al.*, "Design of a dynamic and self-adapting system, supported with artificial intelligence, machine learning and real-time intelligence for predictive cyber risk analytics in extreme environments – cyber risk in the colonisation of Mars," *Safety in Extreme Environments*, vol. 2, no. 3, pp. 219–230, Oct. 2020, doi: https://doi.org/10.1007/s42797-021-00025-1.

[26] B. R. Murphy, “Comparing the Performance of Intrusion Detection Systems: Snort and Suricata - ProQuest,” *www.proquest.com*, 2019. https://www.proquest.com/openview/885ab9a9d8f5c1b92d177780fbe81699/1?pq-origsite=gscholar&cbl=18750&diss=y

[27] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A Deep Learning Approach to Network Intrusion Detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, Feb. 2018, doi: https://doi.org/10.1109/tetci.2017.2772792.

[28] “Deep Learning for Edge Computing Applications: A State-of-the-Art Survey | IEEE Journals & Magazine | IEEE Xplore,” *ieeexplore.ieee.org*. https://ieeexplore.ieee.org/abstract/document/9044329

[29] K. Vivekanandan and N. Praveena, “Hybrid convolutional neural network (CNN) and long-short term memory (LSTM) based deep learning model for detecting shilling attack in the social-aware network,” *Journal of Ambient Intelligence and Humanized Computing*, Jun. 2020, doi: https://doi.org/10.1007/s12652-020-02164-y.

[30] R. Golchha, A. Joshi, and G. P. Gupta, “Voting-based Ensemble Learning approach for Cyber Attacks Detection in Industrial Internet of Things,” *Procedia Computer Science*, vol. 218, pp. 1752–1759, Jan. 2023, doi: https://doi.org/10.1016/j.procs.2023.01.153.

[31] M. Abbasi, A. Shahraki, and A. Taherkordi, “Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey,” *Computer Communications*, vol. 170, pp. 19–41, 2021, doi: https://doi.org/10.1016/j.comcom.2021.01.021.

[32] A. I. Abubakar, H. Chiroma, S. A. Muaz, and L. B. Ila, “A Review of the Advances in Cyber Security Benchmark Datasets for Evaluating Data-Driven Based Intrusion Detection Systems,” *Procedia Computer Science*, vol. 62, pp. 221–227, 2015, doi: https://doi.org/10.1016/j.procs.2015.08.443.

[33] A. L. Buczak and E. Guven, “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016, doi: https://doi.org/10.1109/comst.2015.2494502.

[34] A. Bendovschi, “Cyber-Attacks – Trends, Patterns and Security Countermeasures,” *Procedia Economics and Finance*, vol. 28, no. 28, pp. 24–31, 2015, doi: https://doi.org/10.1016/s2212-5671(15)01077-1.

[35] “NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB,” *www.unb.ca*. https://www.unb.ca/cic/datasets/nsl.html

[36] K. Nalavade and B. B. Meshram, *Imanagerpublications.com*, 2024. https://imanagerpublications.com/assets/htmlfiles/JIT3(2)Mar-May20142780.html (accessed Apr. 08, 2024).

[37] S. R, S. S, A. A, V. V, R. E, and G. T, “Data scientific approach to detect the DoS attack, Probe attack, R2L attack and U2R attack,” *IEEE Xplore*, May 01, 2023. https://ieeexplore.ieee.org/abstract/document/10199636

[38] Srinivas Mukkamala and A. H. Sung, “Computational Intelligent Techniques for Detecting Denial of Service Attacks,” *Lecture notes in computer science*, pp. 616–624, Jan. 2004, doi: https://doi.org/10.1007/978-3-540-24677-0_63.

[39] S. Kaushik, “Detection of Attacks in an Intrusion Detection System.” Accessed: Feb. 11, 2023. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=20f0adc524e835d921c631e8d778f656e6cdeb6b

[40] “Intrusion Detection Systems,” *learn.saylor.org*. https://learn.saylor.org/mod/book/tool/print/index.php?id=29755&chapterid=5449 (accessed Apr. 08, 2024).

[41] H. G. Kayacik, Z. Heywood, and M. I. Heywood, “SELECTING FEATURES FOR INTRUSION DETECTION: A FEATURE RELEVANCE ANALYSIS ON KDD 99 INTRUSION DETECTION DATASETS,” no. 3, Jan. 2005.

[42] Huan Liu and R. Setiono, “Chi2: feature selection and discretization of numeric attributes,” *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, doi: https://doi.org/10.1109/tai.1995.479783.

[43] J. Willman, “Overview of PyQt5,” *Modern PyQt*, pp. 1–42, Dec. 2020, doi: https://doi.org/10.1007/978-1-4842-6603-8_1.

[44] B. Charbuty and A. Abdulazeez, “Classification Based on Decision Tree Algorithm for Machine Learning,” *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp. 20–28, Mar. 2021.

[45] G. Lăzăroiu, M. Andronie, M. Iatagan, M. Geamănu, R. Ștefănescu, and I. Dijmărescu, “Deep Learning-Assisted Smart Process Planning, Robotic Wireless Sensor Networks, and Geospatial Big Data Management Algorithms in the Internet of Manufacturing Things,” *ISPRS International Journal of Geo-Information*, vol. 11, no. 5, p. 277, Apr. 2022, doi: https://doi.org/10.3390/ijgi11050277.

[46] G. Sai, R. Kiran Kumar, K. Parish, N. Raghavendra Sai, and Madamachi Brahmaiah, “Deep residual convolutional neural Network: An efficient technique for intrusion detection system,” *Expert Systems with Applications*, vol. 238, pp. 121912–121912, Mar. 2024, doi: https://doi.org/10.1016/j.eswa.2023.121912.

[47] “4 things you need to know about AI: accuracy, precision, recall and F1 scores,” *lawtomated*, Oct. 10, 2019. https://lawtomated.com/accuracy-precision-recall-and-f1-scores-for-lawyers/

[48] X. Fang, M. Xu, S. Xu, and P. Zhao, “A deep learning framework for predicting cyber attacks rates,” *EURASIP Journal on Information Security*, vol. 2019, no. 1, May 2019, doi: https://doi.org/10.1186/s13635-019-0090-6.

[49] A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and C. O. Uwadia, “Novel adaptive cyberattack prediction model using an enhanced genetic algorithm and deep learning (AdacDeep),” *Information Security Journal: A Global Perspective*, pp. 1–20, Mar. 2021, doi: https://doi.org/10.1080/19393555.2021.1883777.