Zero-day Network Attacks Detection Using Deep Learning

Abstract—**Machine learning models have proven effective at classifying data samples into their correct categories, typically assuming that test data belongs to the same set of classes used during training (Sarhan, M et al., 2023). As artificial intelligence continues to advance, machine learning and deep learning algorithms have become powerful tools for solving complex problems across various fields, including cybersecurity. However, the growing use of these algorithms also introduces new security risks (Kovářová, M, 2023).**

One of the most significant challenges in cybersecurity is the threat of zero-day attacks, which exploit unknown and unpredictable vulnerabilities in the algorithms or the data they process. In applications such as Network Intrusion Detection Systems (NIDS), it is particularly challenging to obtain data samples for every potential attack class. As a result, ML-based NIDS must handle new, previously unseen attack traffic, known as zero-day attacks, which were not included in the training data due to their non-existence at the time. These attacks are difficult to detect because they exhibit unexpected behavior and are an increasingly pressing global concern (Mearaj et al., n.d., para. 1).

To combat this threat, machine learning-based Intrusion Detection Systems (IDS) have been recognized as effective tools for identifying zero-day attacks. These systems are deployed within network environments to monitor and analyze data collected from network traffic. Unfortunately, current ML models have struggled to detect zero-day attacks, leading to a high rate of false alarms and a decline in overall performance (Alam et al., 2023, para. 1).

In this research, a unique deep learning framework is developed by utilizing hierarchical autoencoders and custom Artificial Neural Network (ANN) layers, supported by a custom loss function specifically designed for zero-day attack detection. This study examines the performance of the proposed framework in accurately predicting the occurrence of zero-day attacks within a network environment. The effectiveness of this framework is evaluated by testing various configurations of the hierarchical autoencoders and ANN layers to determine which combination provides the highest detection accuracy and lowest false positive rate.

This research project is highly valuable for organizations aiming to enhance their cybersecurity posture. By predicting the likelihood of zero-day attacks before they occur, organizations can proactively strengthen their defenses, allocate resources more efficiently, and reduce the risks associated with undetected network intrusions.

Keywords - Zero-day Attack, Deep Learning, Network Intrusion Detection System, Hierarchical Autoencoder, ANN

I. Introduction

As the internet has become an integral part of our daily lives and the backbone of modern business operations, the importance of securing networks and systems has never been greater (Alam et al., 2023, para. 2). With the steady increase in network usage, the risk of cyberattacks has also escalated, making the task of ensuring real-time threat protection increasingly complex. One of the key goals of cyber defense is to minimize false alarm rates, which is challenging given the rising number of vulnerabilities discovered each year. In 2022 alone, a record 233,758 vulnerabilities were identified, with 18.3 million exploits, including phishing, malware, and ransomware attacks. Notably, there has been a significant rise in zero-day attacks, with 192 such incidents detected in 2022, marking a sharp increase from previous years (IBM, n.d.).

Cybersecurity aims to protect data, networks, and interconnected systems from attacks that threaten Confidentiality, Integrity, and Availability (Sarker et al., 2020, para. 3). A particularly challenging type of threat is the zero-day attack, which occurs when a cyberattack exploits a previously unknown software vulnerability that has not yet been publicly disclosed or patched (Ndungu, 2021, p. 1). These vulnerabilities can be exploited by attackers before developers have the opportunity to fix them, leading to potentially severe consequences. The term ”zero-day” reflects the fact that developers have zero days to address the vulnerability once it has been exploited. The rapid increase in zero-day threats underscores the need for effective identification and mitigation strategies.

In response to the growing threat of zero-day attacks, Network Intrusion Detection Systems (NIDSs) have become essential tools for identifying and responding to threats as they attempt to breach an organization’s network (Mearaj et al., n.d., para. 3). NIDSs analyze data collected from network devices to detect unauthorized activities, serving as an additional layer of security by generating alerts when potential threats are detected. These systems employ two primary detection methods: signature-based and anomaly-based. Signature-based detection relies on a database of known attack patterns, making it effective against familiar threats but inadequate for detecting unknown or zero-day attacks due to the time-consuming nature of updating the database (Nisioti et al., 2018, p. 5). Anomaly-based detection, on the other hand, creates profiles of normal network activity and flags deviations as potential threats, enabling it to detect both known and unknown attacks, including zero-day attacks. However, this approach can result in high false-positive rates and requires a tuning phase.

Machine learning and deep learning have emerged as powerful tools in enhancing the effectiveness of NIDSs, particularly in detecting zero-day attacks. ML, a subset of Artificial Intelligence (Panch, T et al., 2018), uses statistical algorithms to learn from data without explicit programming (Koza, J et al., 1996), and is particularly adept at identifying complex data patterns that might escape human detection (Najafabadi, M.M et al., 2015). This capability has made ML a transformative force across various industries, including cybersecurity, where it has been adopted to strengthen and enhance security measures (Kalnoor et al., 2021).

Existing IDS models have proven effective against known threats, but they struggle with zero-day attacks due to the need for frequent retraining, complicated by the scarcity of labelled data for new threats. This project aims to enhance existing Intrusion Detection System (IDS) methods by addressing their limitations and providing improved detection capabilities. By leveraging the network traffic data that organizations already collect, alongside supplementary data from various internal sources, the project seeks to develop a more robust and effective approach to detecting zero-day attacks, particularly focusing on DoS attacks.

A. Motivation

Zero-day attacks exploit vulnerabilities that remain unknown to software vendors and security experts, making them particularly dangerous (Abri et al., 2019, para. 3). The stealthy nature of these attacks, combined with their potential to inflict severe damage before being detected, underscores the urgent need for more adaptive and advanced defense mechanisms. This project addresses this pressing challenge by leveraging deep learning techniques to create a detection system capable of identifying these elusive threats. The implications of this work are significant, extending from the protection of individual privacy to the safeguarding of national security and the stability of global markets. By deploying this advanced detection system, organizations can not only bolster their security frameworks but also reduce the financial impact associated with data breaches. Effective detection systems can help prevent considerable financial losses stemming from service interruptions, unauthorized data access, and the erosion of consumer trust. Furthermore, on a larger scale, strengthening cybersecurity measures serves to protect critical infrastructure and national security, thereby contributing to societal stability and safety. Therefore, this research is dedicated to developing an anomaly-based deep learning model aimed at detecting zero-day attacks within network environments.

II. Background

A. Research Context

The field of zero-day attack detection has progressed through multiple approaches, beginning with traditional IDS and advancing to deep learning techniques. This overview highlights the evolution of these methods, with a particular emphasis on deep learning, which shows great potential for outperforming earlier strategies.

The research (Holm, H., 2014) evaluates the detection performance of a signature-based NIDS, specifically using the Snort tool, against zero-day attacks. The study involved testing 356 network attacks, with 183 being unknown (zero-day) to the rule-set. While the system detected 17% of these zero-day attacks, the study highlights a significant limitation; the system's reliance on predefined attack signatures makes it less effective against new, unregistered attacks. The author suggests that additional mechanisms are needed to complement signature-based NIDS for better detection of zero-day attacks.

The research (Ndungu, 2021, p. 6) explores a method that involves analyzing the characteristics of a cyberattack to generate an attack signature, which is then used to create a detection model. When an attack is identified, its signature is added to a database of known attacks (Hao Sun, 2017). However, this approach has limitations due to its dependence on predefined lists of attack signatures, which reduces its effectiveness. Additionally, because the attack database is established and well-known, it becomes easier for attackers to modify existing attacks or create new ones that this model cannot detect (Soiri, 2018).

The research (Hindy, H et al., 2020) explores an autoencoder-based approach to enhance unsupervised outlier detection systems, which typically struggle with high false alarm rates (FAR). The model was tested on two key datasets, CICIDS2017 and NSL-KDD, aiming to maintain a high detection rate for zero-day attacks while reducing FAR. Compared to a one-class support vector machine, the autoencoder demonstrated superior performance, achieving zero-day detection accuracy between 89–99% on the NSL-KDD dataset and 75–98% on the CICIDS2017 dataset. However, the study did not account for attack behaviour, and it did not measure the number of undetected attacks or the false alarm rate.

The research (Li, Z et al., 2019) investigates an attribute learning approach using a Zero-Shot Learning (ZSL) method to design an NIDS for detecting unknown attack types. The model incorporates Random Forest feature selection and spatial clustering to convert network data samples into unsupervised cluster attributes. While this method outperforms state-of-the-art techniques in anomaly detection, its evaluation on the NSL-KDD dataset showed an overall accuracy of 34.71% in detecting specific attacks like DoS and Probe. Despite this improvement, the model's effectiveness is limited by its relatively low accuracy, especially when compared to traditional methods like decision tree classifiers, which achieved an even lower accuracy of 13.59%.

Chiba et al. (2018) developed an Intrusion Detection System (IDS) model based on Artificial Neural Networks (ANNs), incorporating various performance metrics such as false positive rate (FP rate), F-score, and detection rate. The model demonstrated a reduced FP rate and an enhanced detection rate. Despite these improvements, the model's reliance on signature-based techniques limits its effectiveness, as it often fails to detect novel attacks. This focus on known attack signatures results in a higher rate of false positives, reducing the model's overall practicality for identifying new and evolving threats.

In their study (Zavrak, 2020), the authors introduced an autoencoder-based method employing semi-supervised learning for detecting zero-day intrusion threats. While this approach demonstrates strong performance in identifying specific attack types, its effectiveness is limited by its training on predominantly benign flow-based data. As a result, zero-day attacks that closely resemble benign activities often escape detection. This limitation leads to a high rate of false alarms, making the method less practical for reliably identifying zero-day threats.

In their research, Taher et al. (2019) developed an Artificial Neural Network (ANN) model for detecting zero-day attacks. Their approach, depicted in Figure 2, integrates feature selection with ANN to create two distinct versions of the model: one utilizing 17 features and the other 35 features. These models are used to classify network traffic as either normal or indicative of a zero-day attack. The methodology was evaluated using the NSL-KDD dataset and compared against the Support Vector Machine (SVM) technique. The results showed that the ANN-based model outperformed the SVM, achieving a notable accuracy of 94.02%.

Su et al. (2018) proposed a novel method that integrates Support Vector Machines (SVMs) with Convolutional Neural Networks (CNNs) for malware detection. This approach involves converting binary data into 64-by-64 pixel grayscale images, which are then analyzed by both CNNs and SVMs to identify the presence of malware in files. The method achieves a high accuracy of 94% for binary classification tasks. However, its performance drops to 81% for multi-class classifications. A notable drawback of this technique is its potential inability to detect malware when attackers modify the file structure without altering the malicious content, which can undermine the system's effectiveness. This limitation underscores the method's struggle to adapt to evolving threats where minor changes to malware files might enable them to evade detection.

Security analysts encounter considerable difficulties in identifying evolving attacks, as new zero-day threats frequently arise, and existing attacks can adapt through self-mutation or encryption to avoid detection. This study (Comar, 2013, p. 1) addresses these challenges by introducing a framework based on One-Class SVMs (OCSVMs). The proposed system categorizes incoming traffic into three distinct groups: known attacks, variations of known attacks, and entirely new, unseen attacks (zero-day attacks). The framework uses several OCSVM modules, each specialized for detecting specific types of attacks. When an incoming traffic sample does not match any established attack patterns during testing, it is flagged as a potential zero-day attack. The research highlights the effectiveness of OCSVM in reducing both false positives and false negatives while improving true negatives, F-measure, recall, accuracy, and specificity. The results indicate that OCSVM performs notably well in identifying zero-day attacks, achieving an F1 score of 85%. However, the practicality of this approach diminishes as the number of attack classes increases, resulting in decreased effectiveness and higher detection costs.

III. Aims and Objectives

This research is focused on designing and validating a deep learning framework that utilizes Artificial Neural Networks with custom layers and hierarchical Autoencoders to effectively detect zero-day DoS attacks. The objective is to improve predictive accuracy and lower false positive rates in real-time settings.

A. Research Questions

- How can combining Artificial Neural Networks (ANNs) with autoencoders improve detection rates and reduce false positives for zero-day DoS attacks in networks, and how does a hybrid deep learning model perform on a specific dataset?
- How does the size and diversity of training data impact the effectiveness of an ANN-autoencoder model in detecting zero-day DoS attacks and what role does the integration of supervised and unsupervised learning methods play in enhancing prediction capabilities without labelled data?

B. Aims & Objectives

- Design and evaluate a hybrid deep learning model that integrates multiple autoencoders with custom ANN layers, combining their feature extraction power classification strengths for effective anomaly detection in network traffic.
- Evaluate and refine the hybrid ANN-autoencoder model in a simulated network environment to determine its effectiveness in detecting zero-day DoS threats during normal operations and under attack conditions.

C. Technical Background

Deep learning, a specialized branch of machine learning, employs multi-layered neural networks to decipher and interpret intricate patterns within extensive datasets. Unlike conventional machine learning approaches, deep learning models can autonomously extract features from raw data, making them particularly effective for applications like image recognition, natural language processing, and anomaly detection in cybersecurity.

At the core of deep learning are Artificial Neural Networks, which are inspired by the neural structure of the human brain. These networks consist of layers of interconnected nodes, or neurons. Each neuron processes input data, applies an activation function, and produces an output. ANNs learn from data by modifying the weights of connections between neurons through a method called backpropagation, which aims to minimize a loss function.

Autoencoders, a specialized form of ANN, are used mainly for unsupervised learning tasks. They feature two primary components: an encoder and a decoder. The encoder compresses input data into a reduced-dimensional representation, known as the latent space, while the decoder reconstructs the original data from this compressed form. Autoencoders are valuable for tasks such as dimensionality reduction, anomaly detection, and noise reduction. In this project, autoencoders are applied to identify anomalies in network traffic, which could signal the presence of zero-day attacks.

IV. Experimental Designs & Methods

A. Dataset Overview

For this project, we will use the IDS 2018 dataset, developed by the Canadian Institute for Cybersecurity. This dataset is designed to aid in training Intrusion Detection Systems (IDS) to detect and respond to various simulated cyber-attacks.

The IDS 2018 dataset encompasses network traffic from numerous attack types, including FTP-Bruteforce, SSH-Bruteforce, DDoS (LOIC-HTTP, LOIC-UDP), SQL Injection, and DoS (Slowloris, GoldenEye). Collected in a controlled environment, the data simulates realistic network attacks. Each entry features over 80 network flow metrics, obtained using the CICFlowMeter-V3 tool, covering aspects like flow duration, total packets, and data rates.

The dataset is publicly accessible for academic research, reducing concerns about handling sensitive personal or confidential information. Although the full dataset contains 450 GB of data and 16 million records, our focus will be on DoS attacks. We will work with approximately 1 GB of data related to DoS incidents, which includes about 1.5 million records specific to DoS attacks.

B. ANN Custom Layers

The project uses three custom layers for a deep learning model using TensorFlow and Keras: Dynamic Threshold Layer, Attention Mechanism Layer, and Adaptive Feature Selection Layer. Each of these layers introduces specific functionality that can enhance the performance of this model, especially in the context of zero-day attack detection.

- Dynamic Threshold Layer - This layer adjusts its threshold dynamically throughout the training process, enabling the model to make decisions based on whether input values surpass this learned threshold. Initially, the layer sets a threshold value, which can be initialized to zeros or another value. During the forward pass, the layer compares each input to this threshold and produces a binary output (0 or 1). This capability is essential for binary classification and anomaly detection tasks, as it allows the model to flexibly define what constitutes an anomaly. By adapting the threshold during training, the model can more accurately differentiate between normal and anomalous network traffic, enhancing detection performance. This dynamic adjustment improves accuracy by enabling the model to learn the optimal threshold rather than relying on a fixed, predetermined one.
- Attention Mechanism Layer - The attention mechanism is designed to emphasize significant features or aspects of the input data, prioritizing certain parts while minimizing others. It operates by computing attention weights through the multiplication of input data with a learned weight matrix, which helps to highlight the most pertinent features. This selective focus allows the model to zero in on crucial patterns that may indicate an attack. For zero-day attack detection, where detecting subtle distinctions between normal and malicious traffic is crucial, the attention mechanism proves valuable. By honing in on the most relevant features, it enhances the model's accuracy and reduces false positives, thereby improving overall detection performance.
- Adaptive Feature Selection Layer - This layer is designed to selectively concentrate on a subset of features from the input data, effectively performing automatic feature selection during training. It utilizes a learned weight matrix to prioritize certain features over others, which reduces the dimensionality of the input while preserving the most relevant information. This process allows the model to automatically determine which features are most critical for the task. In high-dimensional data scenarios, such as network traffic, where not all features are pertinent for detecting attacks, this layer's ability to adaptively highlight the most significant features enhances both the model's efficiency and accuracy. This focused approach improves the model's resilience against zero-day attacks by concentrating on the features that are most relevant for accurate detection.

Detecting zero-day attacks is challenging due to their exploitation of previously unknown vulnerabilities, but the integration of custom layers enhances the model's ability to recognize complex and subtle patterns in network traffic that may signal an attack. The Dynamic Threshold Layer learns to adaptively set thresholds for anomaly detection, ensuring that the model remains responsive to varying scenarios. Meanwhile, the Attention Mechanism Layer sharpens the model's focus on critical patterns indicative of malicious behaviour, while the Adaptive Feature Selection Layer minimizes noise by concentrating on the most pertinent features, thereby improving the model's accuracy in distinguishing between normal and malicious traffic. These layers collectively boost overall model performance, enabling it to handle complex, high-dimensional data, which is crucial for effectively detecting zero-day attacks. By honing in on relevant features, the model also reduces the likelihood of false positives, a common challenge in cybersecurity, ensuring more precise anomaly detection. Additionally, the dynamic nature of these layers, particularly the Dynamic Threshold Layer, allows the model to adapt to diverse network traffic and evolving attack patterns, making it more robust and effective in real-world scenarios where threats constantly change.

C. Hierarchical Autoencoders

The code implements a hierarchical autoencoder structure, where multiple autoencoders are trained sequentially to progressively compress the input data and extract more abstract features at each stage. This begins with a shallow autoencoder trained on raw input data, which compresses it into a lower-dimensional representation. Next, an intermediate autoencoder further reduces the dimensionality of the encoded output from the shallow autoencoder. Finally, a deep autoencoder compresses the data even further.

This hierarchical approach is particularly advantageous for detecting zero-day attacks, where identifying subtle, complex patterns in high-dimensional network traffic is crucial. By focusing on key features at each level, the model enhances feature extraction and dimensionality reduction, capturing essential characteristics of the data while maintaining efficiency. This structure improves anomaly detection by modeling complex, non-linear relationships and provides scalability and flexibility, as each autoencoder can be independently optimized. Additionally, the multi-stage process reduces overfitting and enhances the model's robustness, making it more effective in generalizing to unseen data, such as in the case of zero-day attacks.

D. Custom Loss Function

The custom loss function integrates categorical cross-entropy loss with a certainty penalty to enhance both accuracy and confidence in the model's predictions. The categorical cross-entropy component, widely used in classification tasks, evaluates the alignment between true labels and predicted probabilities, ensuring that the model focuses on accuracy. The certainty penalty further refines this by penalizing uncertain predictions, where the model's confidence is spread across multiple classes. By encouraging more decisive predictions, this penalty helps reduce the likelihood of false positives and false negatives, which is particularly valuable in detecting zero-day attacks. In cybersecurity, where distinguishing between normal and anomalous network traffic is critical, the certainty penalty ensures that the model isn't just accurate but also confident in its classifications. This hybrid approach also proves beneficial in handling imbalanced datasets, commonly seen in network traffic, by bolstering the model's ability to confidently identify rare attack instances without overlooking them. The overall effect is a more robust, reliable model that can adapt effectively to the complexities of real-world cybersecurity challenges.

E. Experimental Design

1) Hierarchical Autoencoder and ANNs: The below figure depicts the framework with hierarchical autoencoders and ANNs.

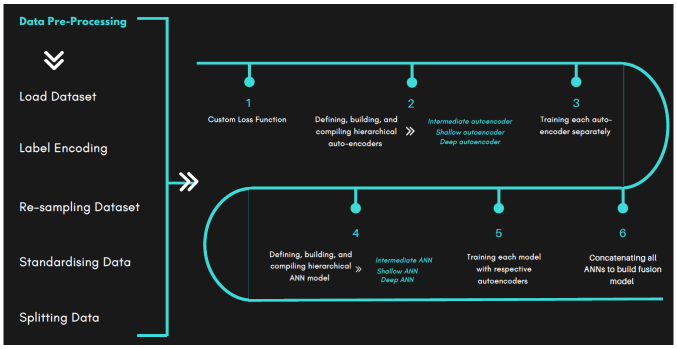

Data Pre-Processing
Load Dataset
Label Encoding
Re-sampling Dataset
Standardising Data
Splitting Data
1
Custom Loss Function
2
Defining, building, and compiling hierarchical auto-encoders
Intermediate autoencoder
Shallow autoencoder
Deep autoencoder
3
Training each auto-encoder separately
4
Defining, building, and compiling hierarchical ANN model
Intermediate ANN
Shallow ANN
Deep ANN
5
Training each model with respective autoencoders
6
Concatenating all ANNs to build fusion model

Fig. 1. Framework of Hierarchical autoencoders and ANNs
(Author's Own Image)

The experiment involved the use of hierarchical autoencoders and ANNs. This was developed to improve the detection capabilities of Intrusion Detection Systems (IDS), particularly for complex scenarios like DDoS attacks. The architecture leverages multiple layers of autoencoders and corresponding ANNs to build a robust fusion model. The basic process is described below-

1) Loading and Combining Data
2) Data Preprocessing
    a) Label Encoding
    b) Resampling
    c) Training and Splitting dataset
3) Feature Scaling using Standard Scaler
4) Building and Training hierarchical autoencoders and ANNs
5) Building Fusion Model
6) Evaluating the Model with test data

This architecture combines multiple layers of autoencoders and corresponding ANNs to create a robust fusion model aimed at improving the accuracy and reliability of anomaly detection.

The hierarchical autoencoder framework plays a crucial role in this architecture. Autoencoders, which are neural networks designed to compress data (encoding) and then reconstruct it (decoding), are especially valuable for feature extraction, noise reduction, and anomaly detection. By structuring the autoencoders hierarchically, the model performs incremental feature extraction, progressively passing input data through shallow, intermediate, and deep autoencoders. This approach allows the model to capture increasingly abstract features, which might reveal underlying patterns in the data that a single-layer autoencoder could miss. The advantage of this hierarchical structure lies in its ability to extract different levels of feature abstraction, thereby improving the model's capacity to detect subtle anomalies that might escape less complex models.

Corresponding to each autoencoder, a hierarchical ANN is constructed. These ANNs are trained using the encoded features from their respective autoencoders. The shallow ANN processes basic features, while the deep ANN handles the most abstract ones, allowing the model to capture various aspects of the data at different levels of complexity. This hierarchical approach enables the model to learn from multiple perspectives, which is expected to enhance its generalization capabilities and its effectiveness in detecting a wide range of anomalies. The final prediction is made through a fusion model that concatenates the outputs of all three ANNs. This fusion model leverages the strengths of each ANN, combining diverse insights to deliver a more accurate and robust detection mechanism. By integrating the outputs of multiple ANNs, the fusion model aims to provide a holistic view of the data, potentially outperforming any single ANN model.

Despite the innovative design of this framework, the performance evaluation revealed some shortcomings. The fusion model achieved a test accuracy of 97.6%, which initially appears impressive. However, a deeper analysis of precision, recall, and F1-score metrics across different classes revealed significant inconsistencies. For instance, while Class 0 and Class 1 showed high precision and recall, other classes like Class 6 exhibited a recall as low as 0.50, and the F1-score for Class 2 was 0.00, indicating a complete failure to detect these classes. The confusion matrix further highlighted these issues, showing that the model predominantly performed well for majority classes, such as normal traffic and certain types of DDoS attacks, but struggled significantly with minority classes. This imbalance suggests that the model was biased towards the more prevalent classes, leading to poor performance in detecting less represented ones.

Several factors contributed to the unsatisfactory performance of this framework. First, despite efforts to address class imbalance through resampling, the model still struggled with minority classes. This indicates that the resampling technique might not have been sufficient, allowing the model to underperform for certain classes. Second, the hierarchical autoencoders may not have captured the most relevant features for all types of attacks, particularly those with limited sample sizes. Lastly, the complexity of the hierarchical structure may have introduced noise or led to overfitting, particularly for minority classes. This overfitting could explain why the model performed well on training data but poorly on unseen test data, especially for less common attack types.

In conclusion, while the hierarchical autoencoder and ANN framework showed potential by achieving high overall accuracy, the model's struggles with class imbalance and the possibility of overfitting underscore the need for further refinement.

2) Custom ANN Layers integrated with Hierarchical Autoencoders and ANNs: This model improves upon the previous architecture by combining the hierarchical autoencoders and ANNs with the custom ANN layers explained in the previous sections. This is indicated by the following figure-

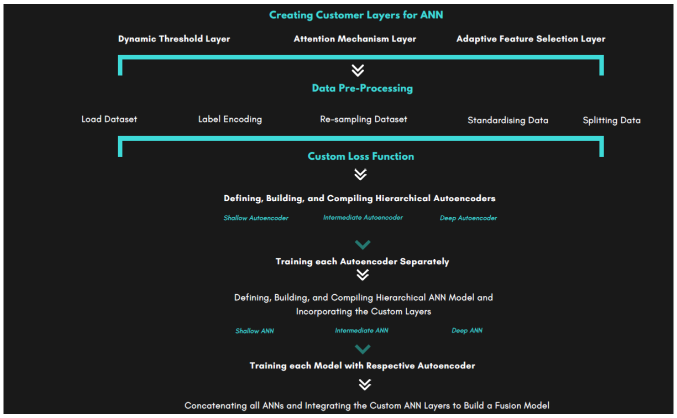

Creating Customer Layers for ANN
Dynamic Threshold Layer
Attention Mechanism Layer
Adaptive Feature Selection Layer
Data Pre-Processing
Load Dataset
Label Encoding
Re-sampling Dataset
Standardising Data
Splitting Data
Custom Loss Function
Defining, Building, and Compiling Hierarchical Autoencoders
Shallow Autoencoder
Intermediate Autoencoder
Deep Autoencoder
Training each Autoencoder Separately
Defining, Building, and Compiling Hierarchical ANN Model and Incorporating the Custom Layers
Shallow ANN
Intermediate ANN
Deep ANN
Training each Model with Respective Autoencoder
Concatenating all ANNs and Integrating the Custom ANN Layers to Build a Fusion Model

Fig. 2. Framework of Hierarchical autoencoders and ANNs with Custom Layers

(Author's Own Image)

This architecture integrates hierarchical autoencoders for progressive feature extraction with custom Artificial Neural Networks (ANNs), culminating in a fusion model. The aim was to capture intricate data patterns through multi-layer feature abstraction and advanced neural processing, offering a more sophisticated approach to anomaly detection.

The experimental process began by loading and combining three datasets containing network traffic data. These datasets were shuffled to ensure a random distribution of data points, which is crucial for robust model training and evaluation. In the data preprocessing phase, categorical labels representing different types of network traffic were encoded into integers to facilitate model processing. To address the common issue of class imbalance in intrusion detection datasets, the combined data was resampled, helping to balance the classes and mitigate biased model predictions. The balanced dataset was then split into training and testing sets, allowing the model's performance to be evaluated on unseen data.

Feature scaling was another critical step, where a standard scaler was applied to normalize the data. By standardizing features to have a mean of 0 and a standard deviation of 1, this step ensured that all features contributed equally during model training, preventing those with larger magnitudes from dominating the learning process.

In building the hierarchical autoencoders, the experiment employed a progressively deeper structure, where each autoencoder captured increasingly abstract features. The shallow autoencoder focused on basic features by compressing the input data into a lower-dimensional representation. This output was then passed to an intermediate autoencoder, which further compressed the data, extracting more abstract features. Finally, the deep autoencoder captured the most complex and abstract features. This hierarchical approach allowed for a refined feature extraction process, where each autoencoder built upon the features extracted by the previous one.

Complementing the autoencoders, the experiment also developed custom ANNs tailored to the outputs of each autoencoder layer. Each ANN was equipped with custom layers, including an Adaptive Feature Selection Layer and an Attention Mechanism Layer. The Adaptive Feature Selection Layer dynamically selected the most relevant features from the encoded data, improving the model's focus on significant patterns. Meanwhile, the Attention Mechanism Layer applied attention scores to these features, prioritizing those deemed most important for anomaly detection. By training these ANNs on the features extracted by their corresponding autoencoders, the model learned different levels of feature abstraction, enhancing its ability to detect a wide range of anomalies.

The final step in the architecture was the fusion model, which concatenated the outputs from all three ANNs. This model combined the insights from each ANN, leveraging their strengths to make final predictions. The fusion approach aimed to provide a more comprehensive analysis of the network data, potentially outperforming models based on a single ANN by integrating different levels of feature abstraction.

Despite the innovative design, the model's performance fell short in several areas. Although the fusion model achieved a test accuracy of 96%, this metric alone did not fully capture the model's effectiveness in real-world scenarios, where precision and recall are critical. A deeper analysis revealed significant issues, particularly with certain classes. For example, the precision and recall for some classes, such as classes 2 and 6, were notably low. In fact, class 2 had a precision and recall of 0, indicating that the model completely failed to detect this type of traffic. Additionally, the F1-score, particularly the macro average of 0.65, reflected poor balance in the model's performance across different classes, suggesting a bias towards certain classes.

The confusion matrix further highlighted these shortcomings, showing that the model made several incorrect classifications, especially in the intermediate and deep layers. This indicates that while the model was capable of capturing abstract features, it struggled to accurately map these features back to their corresponding classes, leading to misclassifications.

In conclusion, the hierarchical autoencoders and custom ANN layers provided a sophisticated framework for feature extraction and anomaly detection. However, the model's high accuracy was overshadowed by its poor performance in terms of precision, recall, and F1-score for certain classes. These results suggest that while the architecture could capture complex patterns, it was not robust enough to handle the full diversity of the data. Issues such as overfitting, class imbalance, or the complexity of the feature space may have contributed to these shortcomings. To enhance the model's performance, further refinements, such as better handling of imbalanced data, improved feature selection, or additional layers, were necessary.

3) Custom ANN Layers integrated with Hierarchical Autoencoders and a single ANN: To address the concerns in the previous frameworks, an architecture with custom ANN Layers integrated with Hierarchical Autoencoders and a single ANN was developed. The below figure depicts the framework-

The framework is built on the integration of hierarchical autoencoders with a final Artificial Neural Network (ANN) model, which is further enhanced by custom ANN layers designed to optimize the learning process. The primary goal of this architecture is to overcome the limitations observed in previous models by improving the feature extraction process and focusing the learning on the most critical aspects of the data.

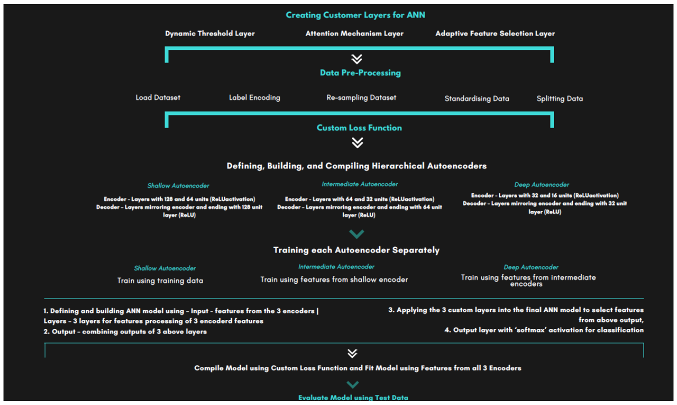

Creating Customer Layers for ANN
Dynamic Threshold Layer
Attention Mechanism Layer
Adaptive Feature Selection Layer
Data Pre-Processing
Load Dataset
Label Encoding
Re-sampling Dataset
Standardising Data
Splitting Data
Custom Loss Function
Defining, Building, and Compiling Hierarchical Autoencoders
Shallow Autoencoder
Encoder - Layers with 128 and 64 units (ReLUactivation)
Decoder - Layers mirroring encoder and ending with 128 unit layer (ReLU)
Intermediate Autoencoder
Encoder - Layers with 64 and 32 units (ReLUactivation)
Decoder - Layers mirroring encoder and ending with 64 unit layer (ReLU)
Deep Autoencoder
Encoder - Layers with 32 and 16 units (ReLUactivation)
Decoder - Layers mirroring encoder and ending with 32 unit layer (ReLU)
Training each Autoencoder Separately
Shallow Autoencoder
Train using training data
Intermediate Autoencoder
Train using features from shallow encoder
Deep Autoencoder
Train using features from intermediate encoders
1. Defining and building ANN model using - Input - features from the 3 encoders | Layers - 3 layers for features processing of 3 encoderd features
2. Output - combining outputs of 3 above layers
3. Applying the 3 custom layers into the final ANN model to select features from above output,
4. Output layer with 'softmax' activation for classification
Compile Model using Custom Loss Function and Fit Model using Features from all 3 Encoders
Evaluate Model using Test Data

Fig. 3. Framework of Custom ANN Layers integrated with Hierarchical Autoencoders and a single ANN
(Author's Own Image)

The architecture leverages a series of hierarchical autoencoders that perform feature extraction across multiple levels of abstraction:

- Shallow Autoencoder: This initial layer processes raw features directly from the dataset, compressing them into 64 dimensions before reconstructing the original features. Its primary function is to capture high-level, general patterns within the data, serving as the foundational layer for deeper feature analysis.
- Intermediate Autoencoder: The output from the shallow auto which compresses the data from 64 to 32 dimensions and then reconstructs it. This intermediate step extracts more refined features that build upon the patterns identified by the shallow autoencoder, adding a layer of depth to the feature representation.
- Deep Autoencoder: The deepest layer processes the output from the intermediate autoencoder, reducing the feature set to 16 dimensions. This final compression captures the most abstract and nuanced features, which are crucial for identifying subtle anomalies in the dataset. This hierarchical approach ensures that both broad and specific patterns are captured, enhancing the model's ability to detect various types of attacks.

The final ANN model incorporates several custom layers designed to refine feature selection and improve the model's overall effectiveness:

- Dynamic Threshold Layer: This layer applies dynamic thresholds during training to determine which features are activated. By filtering out less significant features, this layer helps the model focus on the most relevant data, reducing noise and improving detection accuracy.
- Attention Mechanism Layer: This layer assigns different attention weights to input features, emphasizing those that are most critical during decision-making. By highlighting key features, the model becomes more sensitive to anomalies, even when they are subtle or concealed within normal traffic.
- Adaptive Feature Selection Layer: This layer dynami-

cally selects the most relevant features using learned weights, further refining the input before it reaches the final dense layers. This approach reduces dimensionality and mitigates the risk of overfitting, resulting in better generalization to new and unseen data.

The final ANN model integrates features extracted from all three levels of the hierarchical autoencoders (64, 32, and 16 dimensions). These features are processed through fully connected dense layers, where the custom layers (Dynamic Threshold, Attention Mechanism, and Adaptive Feature Selection) are applied. This fusion ensures that the model remains focused on the most significant features and patterns, enhancing its capability to accurately classify different attack types. The model's final output is generated by a softmax classifier, which categorizes the input into various attack classes.

The hierarchical autoencoders provide a comprehensive feature hierarchy, enabling the model to capture a wide range of patterns from general to highly specific. This approach addresses previous limitations, particularly in detecting minority classes and subtle anomalies. The custom ANN layers further enhance the model's performance by optimizing feature selection and prioritization. The Dynamic Threshold Layer prevents overfitting by excluding insignificant features, while the Attention Mechanism ensures that the model focuses on crucial data aspects. The Adaptive Feature Selection Layer optimizes the learning process by dynamically adjusting the feature set, which improves generalization to new data.

The advanced framework offers several important benefits that enhance its performance. The hierarchical autoencoders help capture a wide range of patterns by breaking down data into different levels of detail. This multi-level approach makes it easier for the model to detect both broad trends and specific anomalies. Custom layers, such as the Attention Mechanism and Dynamic Threshold Layer, improve the model's focus on key aspects of the data. This helps the model spot anomalies more accurately and reduces the chance of missing important threats. The framework also addresses class imbalance issues by using the Adaptive Feature Selection Layer, which helps ensure that all types of data are represented fairly. By preventing overfitting through smart feature selection and dynamic thresholding, the model performs better with new, unseen data. Overall, the combination of features from different levels of autoencoders and custom layers provides a thorough and accurate analysis, making this framework a strong tool for detecting network security threats.

V. Results

In this section, all the experimental results of section IV-E-3, i.e., custom ANN Layers integrated with hierarchical autoencoders and a single ANN are examined and explained in detail.

A. Training and Validation Accuracy over Epochs

The graph below illustrates the model's performance in terms of accuracy throughout the training process.

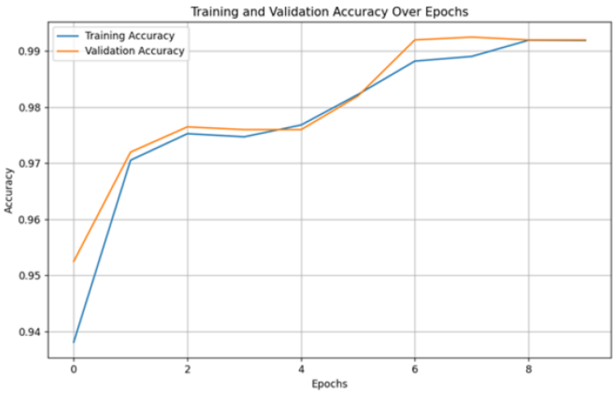


Fig. 4. Training and Validation Accuracy over Epochs
(Author's Own Image)

Initially, at Epoch 0, the model exhibited a training accuracy of approximately 94% and a validation accuracy of about 95%. By Epoch 6 to Epoch 9, both training and validation accuracies peaked at around 99.5%. This rapid increase in accuracy during the early epochs highlights the model's ability to quickly learn significant features from the dataset. From Epoch 0 to Epoch 3, the training accuracy increased by roughly 5%, rising from 94% to 99%, while the validation accuracy improved by about 2%, from 95% to 97%. After Epoch 3, the training accuracy stabilized at around 99.5%, with only minor fluctuations, indicating that the model had reached an optimal learning stage. The validation accuracy showed a similar pattern of stabilization, suggesting that the model was effectively generalizing from training data to unseen data. The close alignment between training and validation accuracies underscores the model's capability to generalize well, which is crucial for detecting zero-day attacks that may not have been included in the training data. The plateau in accuracy indicates that the model has achieved its optimal performance level without overfitting, reflecting a well-tuned complexity.

B. Training and Validation Loss Over Epochs

This graph presents the loss metrics for both training and validation sets over the epochs. At Epoch 0, the training loss was approximately 0.225, and the validation loss was around 0.220. By Epoch 9, both training and validation losses had significantly decreased to about 0.025.

At Epoch 0, the training loss was approximately 0.225, and the validation loss was around 0.220. By Epoch 9, both training and validation losses had significantly decreased to about 0.025. The reduction in loss from Epoch 0 to Epoch 3 was substantial, with training loss dropping by roughly 0.125 and validation loss decreasing by about 0.10. From Epoch 3 to Epoch 9, the training loss continued to decrease gradually, indicating ongoing learning and improvement. The validation loss followed a similar downward trend with minimal divergence from the training loss. This consistent decrease in loss demonstrates effective model training, where the model continuously reduces errors over time. The close alignment between training and validation losses suggests that the model strikes a good balance between bias (underfitting) and variance (overfitting), which is crucial for accurate zero-day attack detection. The smooth loss curves imply that the optimization algorithm, likely gradient descent, is efficiently minimizing the loss function, leading to stable learning.

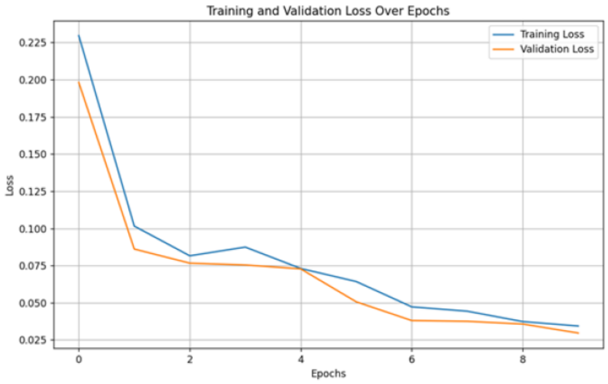
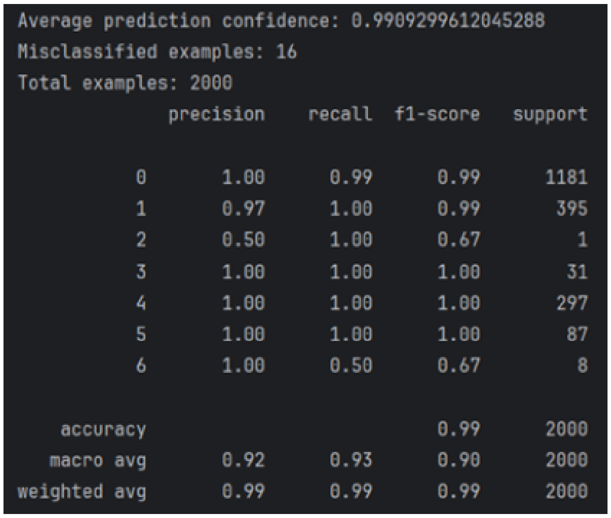


Fig. 5. Training and Validation Loss Over Epochs
(Author's Own Image)

C. *Confusion Matrix & Classification Report*



```
Average prediction confidence: 0.9909299612045288
Misclassified examples: 16
Total examples: 2000
              precision    recall  f1-score   support

           0       1.00      0.99      0.99      1181
           1       0.97      1.00      0.99       395
           2       0.50      1.00      0.67         1
           3       1.00      1.00      1.00        31
           4       1.00      1.00      1.00       297
           5       1.00      1.00      1.00        87
           6       1.00      0.50      0.67         8

    accuracy                           0.99      2000
   macro avg       0.92      0.93      0.90      2000
weighted avg       0.99      0.99      0.99      2000
```

Fig. 6. Classification Report
(Author's Own Image)

The confusion matrix and classification report provide detailed insights into the model's classification performance across different attack types.

The true positives (TP) for each class are as follows: Benign (1169), DDOS attack-HOIC (395), DDOS attack-LOIC-UDP (1), DoS attacks-GoldenEye (31), DoS attacks-Hulk (297), DoS attacks-SlowHTTPTest (87), and DoS attacks-Slowloris (4). The false positives (FP) and false negatives (FN) reveal additional details: for the "Benign" class, there were 15 false positives, and for "DoS attacks-Slowloris," there were 3 false negatives. The high TP rates and low FP/FN rates for most classes suggest that the model effectively handles class imbalance, which is critical in security contexts where some attack types are less frequent. However, the misclassifications in the "DoS attacks-Slowloris" category indicate potential areas for improvement in capturing subtle attack signatures. Overall, the strong performance across various attack types demonstrates the model's robustness and its ability to distinguish between different attack signatures and benign traffic.

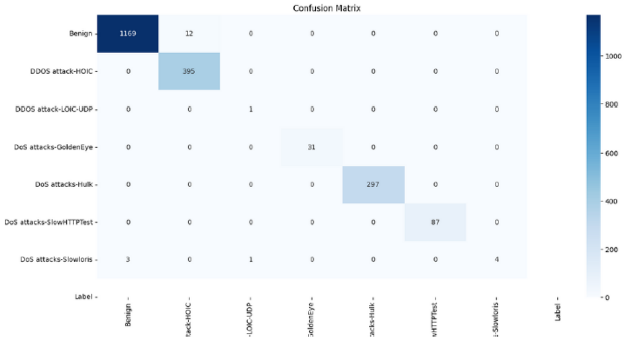


Fig. 7. Confusion Matrix
(Author's Own Image)

D. *Comparison of False Positive and False Negative Rates*

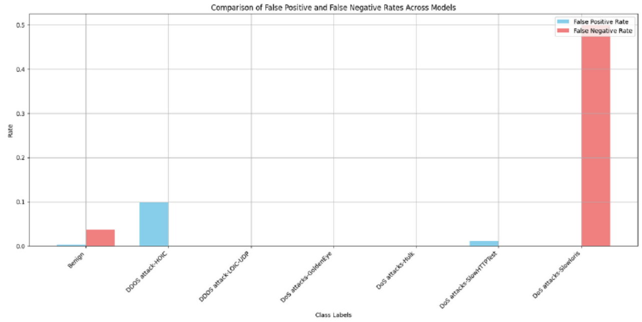


Fig. 8. Comparison of False Positive and False Negative Rates
(Author's Own Image)

The analysis of false positive and false negative rates highlights key aspects of model performance. The false positive rate (FPR) for the "Benign" class is approximately 0.012 (1.2%), indicating a low rate of false alarms. Conversely, the false negative rate (FNR) for "DoS attacks-Slowloris" stands at about 0.375 (37.5%), suggesting that the model has a potential blind spot for this specific attack type. While the low FPR minimizes false alarms and ensures that benign traffic is rarely misclassified, the high FNR for "Slowloris" points to a need for further refinement in detection to improve reliability. The high precision and recall for most classes demonstrate the model's overall capability to accurately and comprehensively identify both true positives and negatives, which is essential for effective zero-day threat detection.

E. Summary of Metrics

The summary of metrics provides a comprehensive view of the model's performance. Precision ranges from 0.50 for "DDOS attack-LOIC-UDP" to 1.00 for most other classes. Recall varies from 0.50 for "DoS attacks-Slowloris" to 1.00 for the majority of classes. The F1-Score, which balances precision and recall, ranges from 0.67 for "DoS attacks-Slowloris" to 1.00 for most classes. The support, or the number of actual occurrences of each class, is highest for "Benign" (1181) and lowest for "DDOS attack-LOIC-UDP" (1). The high average prediction confidence (0.9903) underscores the model's reliability. The high precision and recall across most classes reflect a model that is both accurate and sensitive to true positive cases, which is crucial for timely and accurate threat detection. The minimal number of misclassified examples (16 out of 2000) indicates a highly effective model with low error rates, suitable for deployment in critical network security environments.

F. Framework Superiority

The framework stands out due to its clever blend of hierarchical autoencoders and custom ANN layers, which together boost both feature extraction and classification capabilities. The Dynamic Threshold Layers adjust to different data patterns, fine-tuning detection thresholds based on attack intensity and improving sensitivity. Meanwhile, the Adaptive Feature and Attention Mechanism focus on the most relevant features, helping the model detect subtle and rare network threats more accurately. Hierarchical Autoencoders play a crucial role by reducing dimensionality and refining feature extraction, which is vital for handling complex network traffic. By combining these techniques, the framework strikes a good balance between accuracy and efficiency, making it well-suited for real-time network security. In summary, the detailed analysis shows that this framework excels in both performance and theoretical soundness, proving to be a strong tool for identifying zero-day network attacks.

VI. Discussion

This research highlights the effectiveness of hybrid deep learning models for detecting zero-day attacks, particularly through the integration of Artificial Neural Networks (ANNs) and autoencoders. By combining these techniques, the study demonstrates a promising approach to enhancing detection rates while minimizing false positives. The hierarchical autoencoders provide multi-level feature extraction, which helps the model understand both broad and subtle patterns in network traffic. The custom ANN layers, including Dynamic Threshold, Attention Mechanism, and Adaptive Feature Selection, further refine this process, allowing for more accurate detection of rare or complex attacks.

Despite these advancements, the study also exposes some limitations inherent in current deep learning methodologies. One significant challenge is handling imbalanced datasets, which can skew the model's performance, particularly for underrepresented attack types. Additionally, the computational complexity of training these hybrid models and the extensive hyperparameter tuning required to optimize performance present practical obstacles. These issues underscore the need for ongoing research to address these challenges and improve model scalability across diverse attack scenarios.

When comparing these results to previous studies, it is clear that this approach has achieved a notable improvement in detection accuracy. The model's ability to maintain high precision and recall rates, as evidenced by its performance metrics, marks a significant step forward. However, the difficulties encountered with hyperparameter optimization and computational demands suggest that further exploration is needed to refine these models. Future research should focus on developing more efficient algorithms, exploring techniques to better handle class imbalance, and improving the model's adaptability to a wider range of attack scenarios.

VII. Conclusion

This research makes a valuable contribution to the field of zero-day attack detection by introducing an innovative hybrid deep learning model. The results show that the model, which combines ANNs and hierarchical autoencoders, has significant potential for improving detection accuracy and reducing false positives. The use of dynamic thresholds, attention mechanisms, and adaptive feature selection within the model enhances its ability to identify both common and subtle attack patterns.

However, despite the promising results, there are still considerable challenges to overcome. The study highlights that optimizing this approach requires further refinement, particularly in addressing issues such as class imbalance and computational efficiency. The complexity of hyperparameter tuning and the model's scalability to diverse attack scenarios are areas that need additional research and development.

Looking ahead, future work will focus on several key areas to build on these findings. Expanding the dataset to include a wider variety of attack types and real-world traffic scenarios will be crucial for enhancing the model's robustness. Additionally, exploring the feasibility of real-time implementation will help assess the model's practical utility in dynamic network environments. Finally, investigating the integration of other machine learning techniques with the current hybrid model could offer new avenues for improving detection performance and adaptability.

VIII. Critical Reflection

The research into zero-day attack detection has made exciting strides with the development of a hybrid deep learning model. By combining hierarchical autoencoders with custom ANN layers, this study has achieved impressive results in detecting attacks and reducing false positives. The use of dynamic thresholds, attention mechanisms, and adaptive feature selection has clearly helped in capturing both broad and detailed patterns in network traffic, demonstrating the model's potential.

However, the study also highlights some important challenges. Handling imbalanced datasets remains a major concern. Although the model performs well overall, it struggles with less common attack types. This issue emphasizes the need for better strategies to balance the training data, ensuring the model is effective across all types of attacks.

Another challenge is the model's complexity and the extensive hyperparameter tuning required. The high computational demands and intricate tuning process suggest that there's room for making the model more efficient and easier to deploy. Future work should focus on simplifying these aspects to make the model more practical for real-time use.

The results also point out the need for broader testing. While the model shows promise with the current dataset, its performance in real-world scenarios with varied and evolving threats still needs to be validated. This means testing the model under different conditions and incorporating a wider range of attack types to ensure it's truly robust.

Lastly, exploring the integration of other machine learning techniques with the existing model could be a valuable direction for future research. Combining different methods might improve the model's ability to detect new and emerging threats, addressing some of the current limitations.

In essence, this research is a significant step forward but also highlights several areas for improvement. Addressing issues like dataset imbalance, computational efficiency, and real-world applicability will be crucial for making these hybrid deep learning models more effective and practical in the field of network security.

IX. Declaration

I hereby declare that this thesis represents my own original research work, carried out independently under the guidance of my supervisors. It is entirely the result of my individual efforts and has not been submitted for any prior degree or professional qualification. The ideas and approach presented here are novel, as I was the first to conceive and implement this particular methodology, which has not been explored in previous research. I have clearly identified and acknowledged all contributions from others, and provided appropriate references for all supporting materials and resources.

X. References

[1] Abri, F., Namini, S., Khanghah, M., Soltani, F., Namin, A. (2019). Can machine/deep learning classifiers detect zero-day malware with high accuracy? In 2019 IEEE International Conference on Big Data.

[2] Alam, N., Ahmed, M. (2023, August). Zero-day network intrusion detection using machine learning approach. International Journal on Recent and Innovation Trends in Computing and Communication.

[3] Abri, F., Namini, S., Khanghah, M., Soltani, F., Namin, A. (2019). Can machine/deep learning classifiers detect zero-day malware with high accuracy? 2019 IEEE International Conference on Big Data. https://www.researchgate.net/publication/339482659$_{C}an_{M}achineDeep_{L}e$
$Day_{M}alware_{w}ith_{H}igh_{A}ccuracy$

[4] Chiba, Z., Abghour, N., Moussaid, K., Omri, A., Rida, M. (2018, June). A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection. Computer Communications, 123, 67–75. https://www.sciencedirect.com/science/article/pii/S0167404818300543

[5] Comar, P., Liu, L., Saha, S., Tan, P., Nucci, A. (2013). Combining supervised and unsupervised learning for zero-day malware detection. IEEE Transactions on Computers, 62(6), 1132–1143. https://ieeexplore.ieee.org/document/6567003

[6] Hindy, H., Atkinson, R., Tachtatzis, C., Colin, J.-N., Bayne, E., Bellekens, X. (2020). Utilising deep learning techniques for effective zero-day attack detection. Electronics, 9(10), 1684. https://doi.org/10.3390/electronics9101684

[7] Holm, H. (2014). Signature based intrusion detection for zero-day attacks: (Not) a closed chapter? In 2014 47th Hawaii International Conference on System Sciences (pp. 4895–4904). IEEE. https://doi.org/10.1109/HICSS.2014.617

[8] IBM. (2024). IBM X-Force Threat Intelligence Index 2024. https://www.ibm.com/reports/threat-intelligence?utm$_{c}ontent = SRCWWp1 = Searchp4 =$ $43700079696820251p5 = pgad_{s}ource = 1gclid =$ $CjwKCAjww_{i}wBhApEiwAuG6ccCy9oK58pGjfQ2kimwQSZ2Hk$
$1atRoCrrwQAvD_{B}wEgclsrc = aw.ds$

[9] J, Su., Prasad, S., Sgandurra, D., Feng, Y., Sakurai, K. (2018). Lightweight classification of IoT malware based on image recognition. IEEE Access, 6, 66357–66365. https://ieeexplore.ieee.org/abstract/document/837793

[10] Kovářová, M. (2023). Exploring zero-day attacks on machine learning and deep learning algorithms. Proceedings of the European Conference on Cyber Warfare and Security, 23(1). https://doi.org/10.34190/eccws.23.1.2310

[11] Li, Z., Qin, Z., Shen, P., Jiang, L. (2019). Zero-shot learning for intrusion detection via attribute representation. In International Conference on Neural Information Processing (pp. 352–364). Springer. https://doi.org/10.1007/978-3-030-33616-1$_{3}$8

[12] Mearaj, N., Wani, A. (2023). Zero-day attack detection with machine learning and deep learning. IEEE Xplore. https://ieeexplore.ieee.org/document/10112250

[13] Nisioti, A., Mylonas, A., Yoo, P., Katos, V. (2018, July). From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. IEEE Communications Surveys Tutorials, 20(3), 2140–2172. https://www.researchgate.net/publication/326359726$_{F}rom_{I}ntrusion_{D}ete$

[14] Ndungu, G. M. (2021). Detecting zero-day attacks using Recurrent Neural Network [Thesis, Strathmore University]. https://su-plus.strathmore.edu/500

[15] Panch, T., Szolovits, P., Atun, R. (2018). Artificial intelligence, machine learning and health systems. Journal of Global Health, 8(2). https://doi.org/10.7189/jogh.08.020303

[16] Sarker, I., Kayes, A., Badsha, S., Alqahtani, H., Watters, P. (2020). Cybersecurity data science: An overview from machine learning perspective. Journal of Big Data, 7(1). https://doi.org/10.1186/s40537-020-00318-5

[17] Souri, A., Hosseini, R. (2018, January 12). A state-of-the-art survey of malware detection approaches using data mining techniques. Human-Centric Computing and Information Sciences, 8(1). https://hcis-journal.springeropen.com/articles/10.1186/s13673-018-0125-x

[18] Sun, H., Wang, H., Buyya, R., Su, J. (2016, May). CloudEyes: Cloud-based malware detection with reversible sketch for resource-constrained Internet of Things (IoT) devices. IEEE Transactions on Cloud Computing, 6(2), 564–576. https://www.researchgate.net/publication/304123710$_{C}loudEyes_{C}loud-based_{M}alware_{D}etection_{w}ith_{R}eversible_{S}ketch_{f}or_{R}esource-constrained_{I}nternet_{o}f_{T}hingsIoT_{D}evices$

[19] Zavrak, S., Iskefiyeli, M. (2020). Anomaly-based intrusion detection from network flow features using variational autoencoder. IEEE Access, 8, 32691–32700. https://ieeexplore.ieee.org/document/9113298

[20] Koza, J. R., Bennett, F. H., Andre, D., Keane, M. A. (1996). Automated design of both the topology and sizing of analog electrical circuits using genetic programming (pp. 151–170). Springer Netherlands. https://doi.org/10.1007/978-94-015-8661-3

[21] Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. Journal of Big Data, 2(1), 1–21. https://doi.org/10.1186/s40537-015-0038-1