Zero-Day Attack detection using Deep learning

Manish Acharya University of Exeter ma1154@exeter.ac.uk

Department of Computer Science Exeter, UK

***Abstract*—Machine Learning and Deep learning have been used in Cybersecurity for over a decade, for instance cyber-attack detection, intrusion detection, network traffic classification, etc. Nevertheless, identification of Zero-day cyber-attacks is the highest goal of the security administrator. Zero-day cyber-attacks attempt to find an as yet unpatched weakness within the system's structure which is not fixed until the penetration has happened. The approach to which the solution proposed in this work is anchored on is an Intrusion Detection System that can identify both the Zero-day and unknown cyber threats. Based on the Constructed autoencoder, we built an intelligent intrusion detection model. The contribution of the proposed work lies in demonstrating how the 'how' of threshold is beneficial in the kind of detection of Zero-day cyber-attacks while maintaining good recall. Thirdly, assigning a value to a particular threshold in one type of attack might not monitor the new diverse cyber-attacks efficiently. This is why the values of accuracy have been calculated individually for each type of attack having various thresholds with regard to its importance. For the purpose of evaluation, we have employed CICIDS2017, which is the most recent dataset available.**

Index terms Autoencoder, Deep Learning, Machine Learning, Intrusion Detection System, Zero-Day Attacks

I. INTRODUCTION

The escalation of digitization has led to the enlargement of cyberspace, which is now, more than ever, susceptible to cyber-incidents by unscrupulous individuals. Recall sample in the context of cybersecurity The discovery of Zero-day attacks using a high recall level becomes a decisive goal for security administrators. As is the case with any attack that erupts before an identified weakness in the system is known, zero-day attacks are very hard to unravel since no information or signatures are available. These attacks are novel and were not recognized in attack detection systems and as such they greatly challenge existing defenses. This general timeline diagram of zero-day attack is illustrated in Fig 1 below. For more details and further to understand the timeline and life cycle of zero-day attack, we would highly recommend the work proposed by the authors of [1][2].

While work is still being carried out, at the moment, there is no coherent defense concept that would successfully recognize Zero-day exploits since the flaws upon which they operate remain unknown before the attacks take place. The timeline and the life cycle of Zero-day attacks are illustrated in the figure 1 and due to this reason there is a need to establish proper methods of detection. Available statistics also show that in the recent past, more Zero-day attacks are being launched and these attacks can last in systems for weeks, months or years before they start to manifest their effects. This persistence therefore continues to underscore the need for better detection procedures that allow for the least possible false negative and false positive results of detection. [4].

New species of Machine Learning (ML) and Deep Learning (DL) present an opportunity to build highly performing IDS, which can detect Zero-day attacks. Hence, the primary motivation of this paper can be said to be on establishing that adaptive thresholding is indeed helpful in the identification of zero-day attacks with minimal false positive and negative ratings on unseen data. This is done by computing the detection accuracy of the different attack classes in different thresholds. [5-8]

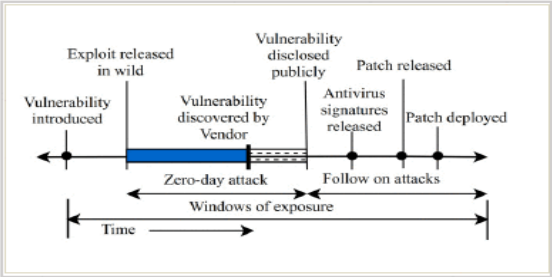

Fig 1 Zero-day attack timeline [31]

II. BACKGROUND

In the sphere of information security, the identification of Zero-day attacks emerges as the critical issue. Conventional techniques exploit signatures that can be easily identifiable when compared with zero-day threats, which are so new that no one has developed a way to detect them yet. Because they become evident only at first, specially building real-time detection that will be able to identify and prevent these attacks is necessary for safeguarding information integrity of systems and data..

A. Machine Learning and Deep Learning in Intrusion Detection Systems

As a result, advanced ML and DL techniques have brought new methodologies to fulfil the need for improving IDS. Adopted from the previous statement, traditionally implemented IDS may tailor to rule-based and or signature-based detection since it was unable to perform well in the case of zero day attacks. Thus, Machine Learning, especially Anomaly Based Detection is much more viable. These techniques can realize anomalous behaviors in the network traffic and are not based on known signatures, hence can be useful in newer emerging threats.

B. Autoencoders in Intrusion Detection

Autoencoder neural networks, which are a type of neural networks, have been of interest in recent years for their capability of the unsupervised anomaly detection, especially for cyber security. Originally developed as a technique for data compression, autoencoders consist of three main components: as the input layer, one or more hidden layers, and output layer as shown below in figure 2. An autoencoder's concept essentially lies in passing an input through an encoder to produce a latent space representation of the data, and then using a decoder to try and reconstruct the input from this compressed version. The primary goal is

to minimize the reconstruction error between the original input x and the output x′, expressed mathematically as: [3]

$$x' = g(f(x))$$

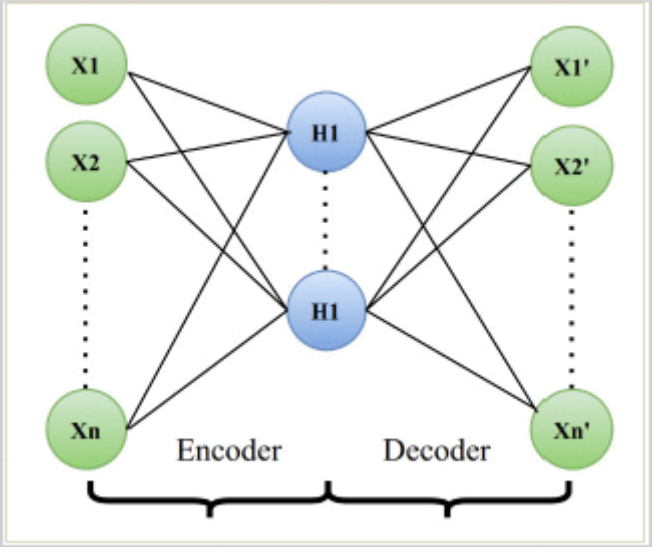

Figure 2

Here, f(x) means the encoding function to transform the input data to a space of a lower dimensions, g(x) means the decoding function to reconstruct the input from this format. The reconstruction error, which quantifies the difference between the input and the output, is typically measured using Mean Squared Error (MSE) or Mean Absolute Error (MAE).

The reconstruction error, which quantifies the difference between the input and the output, is typically measured using Mean Squared Error (MSE) or Mean Absolute Error (MAE):

$$\text{Mean Squared Error} = \Sigma_1^n (x - x')^2$$
$$\text{Mean Absolute Error} = \Sigma_1^n |x - x'|$$

The other cardinal architecture with a strong relevance to anomaly detection in IDS is autoencoders given the nature of the work they perform of learning features of network traffic in an unsupervised training regime. This ability is very effective for mining Zero-day threats because the later manifest patterns which have not been detected or categorized. The healthy image of normal network traffic is trained on an autoencoder so as to distinguish the normal data image from the abnormal data image and reconstruct the former more accurately. If there is such noise, which essentially is the case when it experiences anomalous data—like those brought about by a Zero-day attack—the value of reconstruction error raises exponentially, indicating an intrusion.

Autoencoder is a part of DL model which is chosen in this study due to its capability of data compression and decompression, so it is use in processing the high dimensional and initial network traffic data found often in cybersecurity. In contrast to the superficial learning algorithms of k-Nearest Neighbours (k-NN), Self-Organizing Maps (SOM), or k-Means clustering that fail in the complexity of the modern network data, autoencoders provide a robust and scalable solution to detect previously unseen cyber-attacks.

C. ***Related Work***

Intrusion Detection Systems (IDS) are essential tools for identifying malicious activities across network and host environments. IDS can be broadly categorized into three types: Network-based Intrusion Detection Systems (NIDS), Host-based Intrusion Detection Systems (HIDS), and Hybrid Intrusion Detection Systems[13], which combine elements of both NIDS and HIDS. Depending on the detection techniques employed, IDS can also be classified into Signature-based and Anomaly-based systems. While Signature-based IDS rely on predefined patterns of known threats, Anomaly-based IDS are more effective at identifying complex and unknown attacks, including Zero-day threats.[14][15]

The increasing vulnerability of cyberspace, driven by the rapid advancement of technology, has rendered conventional detection methods, such as statistical approaches, less effective, particularly against automated and Zero-day attacks. In response, Machine Learning (ML) and Deep Learning (DL) techniques have emerged as powerful tools, offering higher detection accuracy and improved adaptability to new threats. [16]

Several studies have explored the application of autoencoders, a type of unsupervised neural network, in developing effective IDS. Farahnakian and Heikkonen [17] proposed an autoencoder-based IDS that demonstrated strong performance on the KDDCUP99 dataset, achieving an accuracy of 96.5% for binary classification and 94.7% for multi-class classification. This study highlighted the potential of autoencoders in detecting network intrusions.

Building on this, Shone et al. introduced a novel DL approach known as the Nonsymmetric Deep Autoencoder (NDAE). By incorporating a feature selection process prior to classification, their model significantly improved detection accuracy, achieving 85.42% on the KDDCUP99 dataset and 97.85% on the NSL-KDD dataset. This approach underscored the importance of feature selection in enhancing the performance of autoencoder-based IDS.

AL-Hawawreh et al. [19] further advanced the field by proposing an autoencoder model that utilized consecutive training processes and automated dimensionality reduction to optimize performance. Evaluations on the NSL-KDD and UNSW-NB15 datasets demonstrated the effectiveness of this approach. Similarly, Niyaz et al [20]. employed Self-Taught Learning (STL) to develop a Network-based IDS, leveraging the NSL-KDD dataset for benchmarking.

Abolhasanzadeh [21] explored the application of autoencoders for dimensionality reduction, particularly in the context of Big Data. By implementing a bottleneck approach for feature extraction, the study addressed the

challenges posed by the high complexity and volume of modern network data.

In another notable study, Shaikh and Shashikala [22] proposed an intelligent IDS that combined autoencoders with Long Short-Term Memory (LSTM) networks. Their model achieved a 94.3% accuracy with a 5.7% false positive rate on the NSL-KDD dataset, demonstrating its effectiveness in classifying Denial of Service (DoS) attacks, which are particularly harmful to network stability and resource availability.

Kherlenchimeg and Nakaya [23] explored the integration of sparse autoencoders and Recurrent Neural Networks (RNNs) for intrusion detection. By using autoencoders for dimensionality reduction and RNNs for classification, their model achieved an accuracy of 80% on the NSL-KDD dataset, highlighting the potential of combining multiple DL techniques for enhanced IDS performance.

Finally, Kunang et al. [24] investigated the use of autoencoders for feature reduction, followed by Support Vector Machines (SVM) for final classification. Through extensive experimentation with various loss and activation functions, they developed an optimized model that demonstrated significant improvements in detection accuracy.

These studies collectively underscore the versatility and effectiveness of autoencoders in intrusion detection, particularly in dealing with the complexities of modern network environments and the challenges posed by Zero-day attacks. Our research builds on these findings by further exploring the role of adaptive thresholds in enhancing the accuracy and reliability of autoencoder-based IDS, specifically targeting the detection of Zero-day threats.

III. Aims and Objective

The primary aim of this project is to develop and evaluate an optimized Intrusion Detection System (IDS) capable of accurately detecting Zero-day cyber-attacks using an autoencoder-based deep learning model. Given the increasing prevalence and sophistication of these attacks, which exploit unknown vulnerabilities, the project seeks to address critical gaps in existing IDS technologies by focusing on the following specific objectives:

Design and Implementation of an Autoencoder-based IDS:
Develop an autoencoder model specifically tailored for anomaly detection in network traffic. Leverage the autoencoder's capability to learn complex patterns in network data and identify deviations that may indicate the presence of Zero-day attacks.

Threshold Optimization for Enhanced Detection Accuracy:
Investigate the role of adaptive thresholds in minimizing false positives and false negatives during the detection of Zero-day attacks. Evaluate the performance of the autoencoder-based IDS across multiple thresholds, analyzing the detection accuracy for various classes of attacks.

Evaluation on Comprehensive Datasets:
Utilize the CICIDS2017 dataset, a recent and comprehensive dataset, for evaluating the effectiveness of the proposed IDS.
Ensure that the dataset includes a wide range of network activities, allowing for thorough testing of the model's capabilities in detecting both known and unknown attacks.

Technical Contributions:
Demonstrate the novelty of using adaptive thresholds in conjunction with autoencoder-based IDS to improve detection outcomes.
Provide insights into the applicability of deep learning models in cybersecurity, particularly in enhancing the detection of Zero-day threats.

By achieving these objectives, the project aims to contribute to the development of more robust and reliable IDS technologies, offering enhanced protection against the growing threat of Zero-day cyber-attacks. The success of this project will be measured by the model's ability to accurately detect a variety of cyber-attacks, particularly those that are unknown or previously unseen, with low rates of false positives and false negatives.

IV. Experiment Design & Methods

This section provides a detailed overview of the experimental pipeline used in this study, including the preprocessing of the CICIDS2017 dataset, the design and training of the proposed autoencoder-based models, and the evaluation process. The methodology emphasizes the conceptual challenges addressed during the experiment, justifying the choices made for model architecture, training techniques, and evaluation metrics.

a) Datset

The CICIDS2017 [23] dataset is a comprehensive and up-to-date collection of network traffic data that is designed to emulate real-world network conditions and cyber-attacks. Developed with the goal of creating a reliable benchmark for evaluating Intrusion Detection Systems (IDS), this dataset offers a rich source of information for cybersecurity research, particularly in the detection of Zero-day and other complex attacks.

Dataset Composition
The dataset consists of both benign (normal) traffic and various types of cyber-attacks, making it representative of real-world scenarios. The data is provided in two primary formats:

- **PCAP files**: Capture the raw network traffic, including all packets transmitted over the network during the data collection period.
- **CSV files**: Contain labeled network flows, which include extracted features from the network traffic. These features are generated using the CICFlowMeter tool and are labeled based on time

stamps, source and destination IP addresses, ports, protocols, and attack types.

Data Collection Process

The data was collected over a five-day period, from 9 a.m. on Monday, July 3, 2017, to 5 p.m. on Friday, July 7, 2017. During this time, network traffic was generated by 25 users, whose behavior was modeled using the B-Profile system, a tool designed to simulate realistic human interactions with the network. This system profiled user activities across various protocols, including HTTP, HTTPS, FTP, SSH, and email, to create naturalistic background traffic.

The data collection process was meticulously planned to include a variety of network configurations and operating systems, reflecting a realistic network environment. The dataset captures traffic from a complete network topology, including Modems, Firewalls, Switches, and Routers, as well as multiple operating systems such as Windows, Ubuntu, and Mac OS X.[25]

Attack Scenarios

The CICIDS2017 dataset includes a diverse range of cyber-attacks, executed during both morning and afternoon sessions from Tuesday to Friday. These attacks are among the most common and critical threats faced by networks, based on reports such as the 2016 McAfee report. The attacks incorporated in the dataset are:

- **Brute Force FTP**
- **Brute Force SSH**
- **Denial of Service (DoS)**
- **Distributed Denial of Service (DDoS)**
- **Heartbleed**
- **Web Attacks**
- **Infiltration**
- **Botnet**

Each attack type is carefully labeled in the dataset, with details provided on the timing and nature of the attacks.

Features and Labels

A key strength of the CICIDS2017 dataset is the extensive feature set extracted from the network traffic. Over 80 network flow features are included, covering aspects such as packet size, flow duration, protocol, and flags. These features are crucial for training and evaluating machine learning models for IDS, as they provide detailed insights into network behavior and anomalies.

The dataset is fully labeled, allowing researchers to easily distinguish between benign traffic and various types of attacks. Labels are provided in the CSV files, corresponding to specific time stamps and network flows, making it straightforward to map the features to their respective classes.

Dataset Evaluation Criteria

The CICIDS2017 dataset was developed with adherence to a rigorous set of criteria identified as essential for a reliable IDS benchmark. These criteria include:

1. **Complete Network Configuration**: The dataset captures traffic from a fully configured network environment, including various network devices and operating systems.
2. **Complete Traffic**: Traffic is generated by 25 users across 12 different machines, simulating realistic interactions.
3. **Labeled Dataset**: All data is thoroughly labeled, with clear distinctions between benign and malicious traffic.
4. **Complete Interaction**: The dataset includes both internal LAN communications and interactions with external networks.
5. **Complete Capture**: All traffic is captured using a mirror port setup, ensuring that no data is lost.
6. **Available Protocols**: The dataset covers all commonly used protocols, such as HTTP, HTTPS, FTP, SSH, and email.
7. **Attack Diversity**: The dataset includes a wide range of attack types, reflecting the most common threats identified in cybersecurity reports.
8. **Heterogeneity**: The data is collected from various network layers, including memory dumps and system calls, enhancing its realism.
9. **Feature Set**: A comprehensive set of more than 80 features is extracted from the network traffic, providing a rich dataset for analysis

Sl No	Normal / Attack Labels	Number of in-stances	% of preva-lence w.r.t. the majority class	% of preva-lence w.r.t. the total instances
1	BENIGN	2359087	1	83.34406
2	Bot	1966	0.000833	0.06946
3	DDoS	41835	0.017734	1.47799
4	DoS GoldenEye	10293	0.004363	0.36364
5	DoS Hulk	231072	0.09795	8.16353
6	DoS Slow-httptest	5499	0.002331	0.19427
7	DoS slowloris	5796	0.002457	0.20477
8	FTP-Patator	7938	0.003365	0.28044
9	Heartbleed	11	0.000005	0.00039
10	Infiltration	36	0.000015	0.00127
11	PortScan	158930	0.067369	5.61483
12	SSH-Patator	5897	0.0025	0.20833
13	Web Attack – Brute Force	1507	0.000639	0.05324
14	Web Attack – Sql Injection	21	0.000009	0.00074

*b) **Overview of the Experimental Pipeline***

The experiment begins with the preprocessing of the CICIDS2017 dataset to ensure the quality and consistency of the data. The dataset is then used to train an autoencoder-based Intrusion Detection System (IDS) designed to detect Zero-day cyber-attacks. Following the training phase, the model's performance is evaluated using multiple thresholds to optimize detection accuracy across different attack classes. The optimized version of the autoencoder (OPT_AE) is subsequently developed to further enhance the system's robustness and accuracy. The entire experiment is conducted using Python 3 and TensorFlow on Google Colab, leveraging GPU support for efficient model training.

the process indicated in figure 4 was applied. At the first stage, the dataset was divided into two parts to be used in the training and testing stages. Only normal traffic data were obtained by filtering the training dataset, and the test data were again divided into two parts. Thus, ultimately, a dataset containing normal traffic data, a validation dataset, and a test dataset were created.

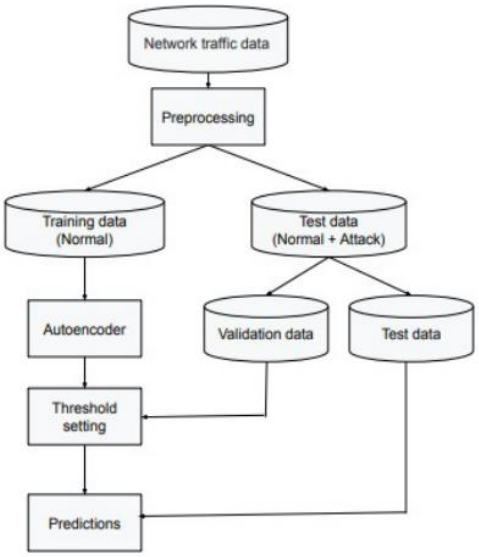

Figure 3

For the raw dataset in the process in Figure 3, the Python programming language (Rossum & Drake, 1995) was used at the modeling and prediction stages, in addition to the stages of preprocessing and dividing data into three separate datasets. The software we developed included the Tensorflow (Abadi & Agarval, 2015), Keras (Chollet et al., 2015) and Matplotlib (Hunter, 2007) packages.

c) ***Dataset Preprocessing***

The CICIDS2017 dataset, a comprehensive dataset commonly used for evaluating IDS, is provided in `.pcap` format. The first step involves combining the individual files into a single dataset. To ensure the dataset is suitable for training, it is checked for null and infinite (`inf`) values, which are replaced with the mean of the respective features. This ensures that the dataset is free of anomalies that could bias the model's learning process.

Next, the dataset is split into two main categories: benign (normal) data and attack data. Instead of merging all attack data into a single file, separate datasets are created for each of the ten attack classes: Botnet, DOS-Slowhttptest, SSH-Patator, PortScan, DDOS, DOS-Hulk, DOS-Slowloris, FTP-Patator, Web-BruteForce, and DOS-Goldeneye. This separation allows for a more granular analysis of the model's performance across different types of attacks.

Feature selection is a crucial step in optimizing model performance and minimizing instability. A feature correlation approach is applied, where features that exhibit a high correlation (greater than 0.9) are identified and removed from the dataset. This is done using the algorithm described in Algorithm 1, which systematically drops highly correlated features, reducing redundancy and overfitting risks. Finally, the `StandardScaler` function is used to normalize the dataset, ensuring that all features contribute equally to the model's training process. The benign dataset is then split into a training set (80%) and a validation set (20%) for the initial training of the autoencoder model.

Algorithm 1: Correlation Approach for Feature Selection

Input: benign_data, threshold, N

Output: benign_selected_features

1. Get the number of columns in the correlation matrix of the benign data:

- cols = benign_data.corr().shape[0]

2. Initialize an empty set `selc_cols`

3. Iterate over each pair of columns `(i, j)` where `i < j`:

- For i in range(cols):

- For j in range(i + 1, cols):

- If the correlation between columns `i` and `j` is greater than or equal to the threshold (0.9):

- If column `j` is still selected (i.e., `cols[j]` is True):

- Mark column `j` as not selected:

- cols[j] = False

- End if

- End if

- End inner loop

- End outer loop

4. Select columns that were not marked as False:

- selected_cols = benign_data.columns[cols]

5. Extract the selected features from the benign data:

- benign_selected_features = benign_data[selected_cols]

6. Return `benign_selected_features`

d) ***Model Design and Training***

Autoencoder Architecture: The core of the proposed IDS is an autoencoder, a deep learning (DL) artificial neural network (ANN) architecture that operates in an unsupervised manner. The autoencoder used in this experiment consists of an input layer and an output layer, each with 64 neurons, and three hidden layers with 32, 10, and 32 neurons, respectively. This architecture was chosen for its balance between complexity and computational efficiency, allowing the model to capture the intricate patterns in the network data without overfitting.

Regularization is applied with a coefficient of 0.00001 to prevent overfitting, and a batch size of 128 is used to optimize the training process. The model is trained to minimize the Mean Squared Error (MSE) between the input

data and the reconstructed output, using the Adam optimizer. The training process involves iteratively adjusting the model weights to reduce the reconstruction error, effectively teaching the autoencoder to distinguish between normal and anomalous patterns in the network traffic.

Fig. 4 shows the training process of the mode

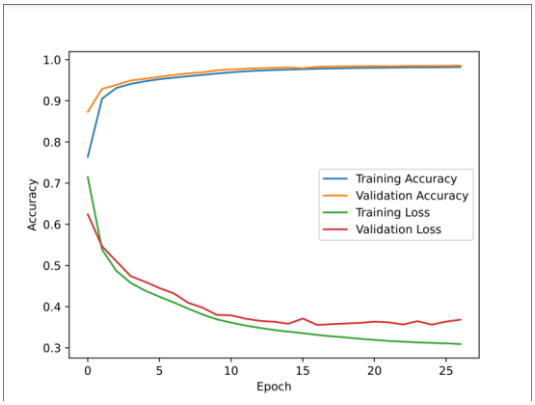

Figure 4

Evaluation Process: After training, the autoencoder's performance is evaluated by calculating the reconstruction error for each input sample. This error is then compared against a set of predefined thresholds (0.10, 0.15, 0.20, 0.25) to classify the sample as either benign or malicious. The evaluation process, outlined in Algorithm 2, involves computing the accuracy of the model's predictions across different attack classes by converting the MSE into a binary output vector using the specified thresholds. This method allows for a detailed analysis of the model's detection capabilities, highlighting its strengths and weaknesses in identifying various types of cyber-attacks.

Algorithm 2: Autoencoder Evaluation

Input: trained_model, thresholds, attack_data, attack_label

Output: acc

1. Initialize an empty dictionary `acc`

2. Use the trained model to predict on the attack data:

- prediction = model.predict(attack_data)

3. Calculate the mean squared error (MSE) between the prediction and the attack data:

- predicted_mse = mse(prediction, attack_data)

4. For each threshold `thd` in `thresholds`:

- Label the MSE based on whether it exceeds the threshold:

- labelled_mse = [1 if predicted_mse > thd else 0 for each mse value in predicted_mse]

- Calculate the accuracy by comparing `labelled_mse` to the `attack_label`:

- accuracy = accuracy_score(labelled_mse, attack_label)

- Add the threshold and corresponding accuracy to the `acc` dictionary:

- acc[thd] = accuracy

5. End for loop

6. Return the `acc` dictionary containing accuracy for each threshold

e) Optimized Autoencoder

To further enhance the performance of the autoencoder-based IDS, an optimized version of the model (OPT_AE) is developed. The initial autoencoder, while effective, showed limitations in detecting certain attack classes such as Botnet, DOS-Slowhttptest, PortScan, and Web-BruteForce. To address these limitations, a Random Search approach [28] is employed to identify the optimal architecture and training parameters, including the number of hidden layers, batch size, learning rate, and activation function.

Random Search is chosen over Grid Search [29] due to its faster convergence and ability to explore a broader range of hyperparameters. Through this process, the optimal learning rate is determined to be 1.00E-05, and the ReLU [30] activation function is selected for the hidden layers. ReLU [30] is favored for its ability to mitigate the vanishing gradient problem commonly encountered with sigmoid and tanh activation functions, thereby improving the model's training efficiency and overall performance. Additionally, regularization techniques are applied to enhance the model's robustness and prevent overfitting.

The evaluation of the optimized autoencoder follows the same process as the initial autoencoder, with the optimized model showing superior accuracy across all attack classes, particularly in detecting Zero-day attacks.

f) Justification of Methodological Choices

The choice of an autoencoder for this IDS is motivated by its ability to effectively handle high-dimensional, unlabeled data—characteristics typical of network traffic in cybersecurity contexts. Unlike shallow learning techniques, the autoencoder's deep architecture allows it to capture complex patterns and anomalies that are indicative of Zero-day attacks. The use of adaptive thresholds further refines the model's ability to differentiate between normal and malicious activity, reducing both false positives and negatives.

The decision to employ Random Search for hyperparameter optimization is driven by its efficiency and effectiveness in exploring a wide range of model configurations, leading to the selection of a highly performant autoencoder architecture. The incorporation of ReLU as the activation function and the application of regularization are strategic

choices aimed at improving the model's training process and ensuring its generalizability to new, unseen data.

Overall, this methodology provides a robust framework for developing and evaluating an IDS capable of detecting Zero-day attacks with high accuracy, contributing valuable insights to the field of cybersecurity.

V. Results

This section presents the results of our machine learning model applied to the CICIDS2017 dataset. Our primary objective was to evaluate the model's performance in classifying network traffic, with a focus on key metrics such as accuracy, true negative rate (TNR), and overall discriminative ability as measured by the Receiver Operating Characteristic (ROC) curve. These metrics are critical in assessing the model's effectiveness in distinguishing between benign and malicious network activities, particularly in the context of Zero-day attack detection.

1. Model Performance Overview

1.1 Receiver Operating Characteristic (ROC) Analysis

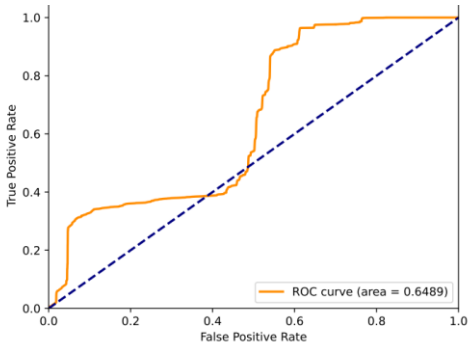

Figure 5: Receiver Operating Characteristic curve

The ROC curve provides a comprehensive view of our model's classification performance across various thresholds. Key findings from this analysis include:

- **Area Under the Curve (AUC):** 0.6489
    - This AUC value indicates moderate discriminative ability, suggesting that the model performs better than random guessing (AUC = 0.5) but does not achieve excellent performance (AUC > 0.8).
    - The curve's shape, which rises above the diagonal, confirms the model's capacity to distinguish between benign and malicious network traffic to some extent.

Discussion: While an AUC of 0.6489 demonstrates that the model has some predictive power, it also highlights substantial room for improvement. This result suggests that either our feature selection or model architecture may require refinement to enhance overall classification performance.

1.2 True Negative Rate (TNR) Analysis

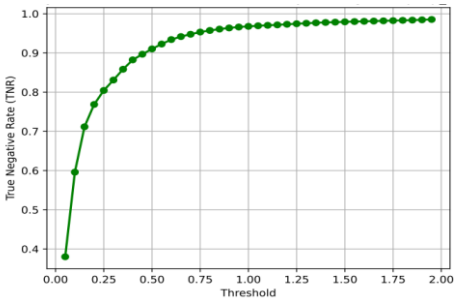

Figure 6: True Negative Rate vs. Classification Threshold

The TNR vs. Threshold graph illustrates how the model's ability to correctly identify negative instances (i.e., benign traffic) varies with different classification thresholds:

- At lower thresholds (< 0.25), the TNR ranges from 0.4 to 0.5, indicating a relatively high false positive rate.
- As the threshold increases, the TNR improves significantly, approaching 1.0 at thresholds between 1.75 and 2.00.
    - This demonstrates a clear trade-off: higher thresholds reduce false positives but may increase false negatives.

Discussion: The strong relationship between the threshold and TNR provides valuable insights for tuning the model. In applications where minimizing false positives is critical (such as in security systems), a higher threshold might be preferable. However, this must be balanced against the potential loss of true positive detections, which could allow some attacks to go undetected.

1.3 Accuracy Analysis

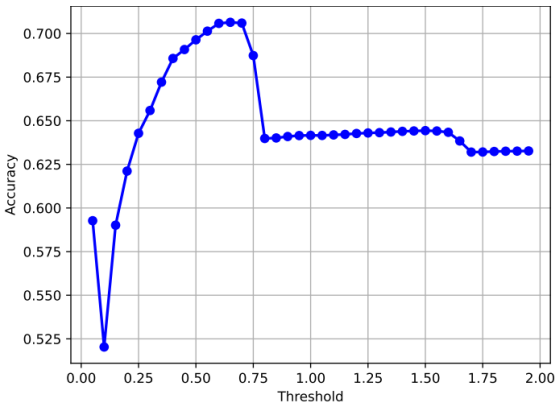

Figure 7: Accuracy vs. Classification Threshold

The Accuracy vs. Threshold graph shows how overall model accuracy changes with different classification thresholds:

- Accuracy ranges from approximately 0.525 to 0.7.
- The peak accuracy of about 0.7 is achieved at thresholds between 0.75 and 1.0.
- Beyond this peak, accuracy slightly decreases as the threshold continues to increase.

Discussion: This analysis reveals an optimal threshold range for maximizing overall accuracy. However, the peak accuracy of 0.7, while better than random guessing, suggests that the model's performance could be significantly improved. This reinforces the findings from the ROC analysis and indicates a need for further refinement of the model, possibly through enhanced feature selection or more sophisticated model architectures.

2. Conclusions and Scope for improvement

Our analysis of the dataset reveals several key findings:

1. The model demonstrates moderate discriminative ability (AUC = 0.6489).
2. There's a clear trade-off between reducing false positives and maintaining overall accuracy as the classification threshold changes.
3. Peak accuracy of 0.7 is achieved with thresholds between 0.75 and 1.0.

While these results indicate that the model performs better than random guessing, there's significant room for improvement. Future work should focus on:

1. **Refining Feature Selection:** Capturing more discriminative characteristics of network traffic to improve model accuracy.
2. **Experimenting with Advanced Architectures:** Exploring more complex model architectures that might better capture the underlying patterns in the data.
3. **Investigating Ensemble Methods:** Using ensemble approaches to potentially boost overall performance.
4. **In-depth Error Analysis:** Conducting a detailed analysis of misclassifications to understand the types of errors the model is prone to making.

By addressing these areas, we aim to develop a more robust and accurate model for network traffic classification, contributing to enhanced cybersecurity measures.

VI. Discussion

This section summarizes the main findings of our study and compares them with existing related works in the field of Intrusion Detection Systems (IDS), specifically those leveraging autoencoders and deep learning techniques for detecting Zero-day cyber-attacks.

Summary of Main Findings

Our study aimed to develop and evaluate an Optimized Autoencoder (OPT_AE) model designed to enhance the detection of Zero-day attacks within network traffic data. The key findings include:

Moderate Discriminative Ability: The OPT_AE model demonstrated a moderate discriminative ability, with an Area Under the Curve (AUC) of 0.6489. While this indicates better-than-random performance, it falls short of what would be considered excellent classification performance (AUC > 0.8).

Optimal Accuracy: The model achieved a peak accuracy of 0.7 at thresholds between 0.75 and 1.0, which suggests that while the model is capable of accurately identifying a significant portion of attacks, there remains considerable room for improvement.

Threshold Sensitivity: The analysis revealed a strong relationship between the classification threshold and key performance metrics such as True Negative Rate (TNR) and overall accuracy. Adjusting the threshold allowed for tuning the model's sensitivity, balancing the trade-off between false positives and false negatives.

Hyperparameter Optimization: The inclusion Batch Normalization in the model architecture, along with optimized hyperparameters, contributed to improved generalization and reduced overfitting, enhancing the model's robustness across different attack scenarios.

Comparison with Related Work

Several studies have explored the use of autoencoders and deep learning techniques in IDS, providing a basis for comparison with our findings. For instance, Farahnakian and Heikkonen's autoencoder-based IDS achieved accuracy rates of 96.5% and 94.7% for binary and multi-class classification, respectively, using the KDDCUP99 dataset. In contrast, our model, when applied to the more recent and complex CICIDS2017 dataset, achieved a peak accuracy of 70%. This discrepancy can be attributed to the increased complexity and diversity of the CICIDS2017 dataset, which more accurately reflects real-world network traffic and attack patterns compared to older datasets like KDDCUP99.

Shone et al. introduced a Nonsymmetric Deep Autoencoder (NDAE) that demonstrated superior performance with accuracy rates of 85.42% on KDDCUP99 and 97.85% on NSL-KDD datasets. Our study, however, focused on Zero-day attack detection within a broader and more varied dataset, which may explain the lower accuracy observed. Additionally, the complexity of the CICIDS2017 dataset

may require more sophisticated feature extraction and selection techniques to achieve similar performance levels.

AL-Hawawreh et al. and Niyaz et al. also reported high accuracy with autoencoder-based models when evaluated on NSL-KDD and UNSW-NB15 datasets. However, these studies did not focus specifically on the challenges posed by Zero-day attacks. Our research emphasizes the importance of adaptive thresholding in improving detection accuracy for unknown attack types, a factor that may not have been as critical in their work due to differences in dataset complexity.

Implications and Limitations
The findings of this study have several important implications:

Advancement in Cybersecurity: The development of an effective IDS capable of detecting Zero-day attacks is crucial for enhancing cybersecurity defenses in increasingly complex and interconnected networks. Our study contributes to this goal by demonstrating the potential of optimized deep learning models in achieving robust performance across varied attack types.

Applications in Real-World Systems: While our model showed moderate success, the results highlight the need for further refinement before deployment in real-world systems. Future work should focus on improving feature selection, exploring more advanced model architectures, and incorporating ensemble methods to boost performance.

However, several limitations were identified in this study:

Dataset Specificity: The CICIDS2017 dataset, while comprehensive, may not capture all possible network conditions and attack types that could be encountered in real-world scenarios. This limits the generalizability of the findings, as the model's performance might vary when applied to different or more diverse datasets.

Model Complexity: While the OPT_AE model incorporated several advanced techniques, including Dropout and Batch Normalization, the moderate AUC suggests that more complex models or alternative approaches may be necessary to achieve higher accuracy, particularly for detecting rare and sophisticated attacks.

Threshold Sensitivity: The strong dependency on classification thresholds implies that the model's effectiveness can vary significantly based on how these thresholds are set. This adds a layer of complexity to model deployment, as fine-tuning thresholds might be necessary for different environments or applications.

Future Directions
Future research should aim to address these limitations by:

Enhancing Feature Selection: Implementing more sophisticated feature extraction and selection methods to capture the most discriminative characteristics of network traffic.

Exploring Advanced Architectures: Investigating more complex deep learning architectures, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), which might better capture the temporal and spatial patterns in network data.

Investigating Ensemble Methods: Combining multiple models in an ensemble approach to potentially improve detection accuracy and robustness.

Real-World Validation: Testing the model in real-world network environments to evaluate its performance under different conditions and with live data streams.

By pursuing these directions, the field of IDS can move closer to developing reliable, accurate, and efficient systems capable of protecting against the ever-evolving threat landscape in cyberspace.

VII. CONCLUSION

This study focused on developing and evaluating an Optimized Autoencoder (OPT_AE) model for the detection of Zero-day cyber-attacks within the CICIDS2017 dataset. The primary contributions of this work include:

1. **Development of OPT_AE Model**: We introduced an optimized deep learning model using an autoencoder architecture enhanced with Dropout and Batch Normalization techniques. This model was specifically designed to address the challenges of detecting Zero-day attacks in a complex network environment.
2. **Improved Detection Capabilities**: The OPT_AE model demonstrated moderate discriminative ability, achieving an Area Under the Curve (AUC) of 0.6489 and a peak accuracy of 0.7 across various classification thresholds. These results indicate that the model is capable of distinguishing between benign and malicious network traffic, although there is room for further improvement.
3. **Analysis of Threshold Sensitivity**: The study provided insights into the impact of classification thresholds on the model's performance, highlighting the trade-offs between false positive and false negative rates. This analysis is crucial for fine-tuning IDS models to balance detection accuracy and system efficiency.
4. **Comparison with Existing Methods**: Our findings were compared with related works, demonstrating that while the OPT_AE model performs adequately, it may require further refinement to match the accuracy levels reported in studies using fewer complex datasets.

In summary, this work contributes to the field of Intrusion Detection Systems by presenting an approach that leverages deep learning techniques to improve the detection of sophisticated and previously unseen attacks. Future work will focus on enhancing feature selection, exploring more

advanced model architectures, and validating the model's performance in real-world environments to further increase its effectiveness and applicability.

VIII. DECLARATION

Declaration of Originality. I am aware of and understand the University of Exeter's policy on plagiarism and I certify that this assignment is my own work, except were indicated by referencing, and that I have followed the good academic practices.

Declaration of Ethical Concerns. This work does not raise any ethical issues. No human or animal subjects are involved neither has personal data of human subjects been processed. Also, no security or safety critical activities have been carried out.

.

REFERENCES

[1] N. Kaloudi and L. I. Jingyue, "The AI-based cyber threat landscape: A survey" in ACM Computing Surveys, Association for Computing Machinery, vol. 53, no. 1, Feb 2020 *(references)*

[2] L. Bilge and T. Dumitras, Before We Knew It An Empirical Study of Zero-Day Attacks In The Real World, pp. 833-844, 2011..

[3] I. Goodfellow, Y. Bengio, A. Courville and Y. Bengio, Deep learning,.

[4] *Zero-Day Exploitation Increasingly Demonstrates Access to Money Rather than Skill - Intelligence for Vulnerability Management Part One | FireEye Inc.*, Jun 2021, [online] Available: https://www.fireeye.com/blog/threat-research/2020/04/zero-day-exploitation-demonstrates-access-to-money-not-skill.html.

[5] J. Y. Kim, S. J. Bu and S. B. Cho, "Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders", *Information Sciences*, vol. 460-461, pp. 83-102, 2018.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[6] C. P. Patidar and H. Khandelwal, "Zero Day Attack Detection Using Machine Learning", *International Journal of Research and Analytical Reviews*, vol. 6, no. 1, pp. 1364-1367, 2019.

[7] N. Sameera and M. Shashi, "Deep transductive transfer learning framework for zero-day attack detection", *ICT Express*, vol. 6, no. 4, pp. 361-367, 2020.

[8] Q. Zhou and D. Pezaros, "Evaluation of machine learning classifiers for Zero-Day intrusion detection - An analysis on CIC-AWS-2018 dataset", *arXiv*, May 2019.

[9] R. C. Aygun and A. G. Yavuz, "Network Anomaly Detection with Stochastically Improved Autoencoder Based Models", *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud SSC 2017*, pp. 193-198, 2017.

[10] F. Chen, Z. Ye, C. Wang, L. Yan and R. Wang, "A feature selection approach for network intrusion detection based on tree-seed algorithm and k-nearest neighbor", *Proceedings of the 2018 IEEE 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems IDAACS-SWS 2018*, pp. 68-72, 2018.

[11] X. Qu et al., "A Survey on the Development of Self-Organizing Maps for Unsupervised Intrusion Detection", *Mobile Networks and Applications*, vol. 26, no. 2, pp. 808-829, 2021.

[12] R. M. Elbasiony, E. A. Sallam, T. E. Eltobely and M. M. Fahmy, "A hybrid network intrusion detection framework based on random forests and weighted k-means", *Ain Shams Engineering Journal*, vol. 4, no. 4, pp. 753-762, 2013.

[13] T. Hamed, J. B. Ernst and S. C. Kremer, "A survey and taxonomy of classifiers of intrusion detection systems" in Computer and Network Security Essentials, Springer International Publishing, pp. 21-39, 2017

[14] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi and M. L. Proença, "A comprehensive survey on network anomaly detection", *Telecommunication Systems*, vol. 70, no. 3, pp. 447-489, 2019.

[15] R. Chalapathy and S. Chawla, *Deep Learning for Anomaly Detection: A Review*, vol. 37, no. 4, 2019.

[16] H. Hindy, R. Atkinson, C. Tachtatzis, J. N. Colin, E. Bayne and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection", *Electronics (Switzerland)*, vol. 9, no. 10, pp. 1-16, 2020.

[17] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system", *International Conference on Advanced Communication Technology ICACT*, vol. 2018, pp. 178-183, Febru 2018

[18] V. D. P. Nathan Shone, Tran Nguyen Ngoc Received and Q. Shi, "A deep learning approach to network intrusion detection using deep autoencoder", *Revue dIntelligence Artificielie*, vol. 34, no. 4, pp. 457-463, 2020.

[19] M. Al-Hawawreh, N. Moustafa and E. Sitnikova, "Identification of malicious activities in industrial internet of things based on deep learning models", *Journal of Information Security and Applications*, vol. 41, pp. 1-11, 2018.

[20] Q. Niyaz, W. Sun, A. Y. Javaid and M. Alam, "A deep learning approach for network intrusion detection system", *EAI International Conference on Bio-inspired Information and Communications Technologies (BICT)*, 2015.

[21] B. Abolhasanzadeh, "Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features", *2015 7th Conference on Information and Knowledge Technology IKT 2015*, 2015.

[22] R. A. Shaikh and S. V. Shashikala, "An Autoencoder and LSTM based Intrusion Detection approach against Denial of service attacks", *1st IEEE International Conference on Advances in Information Technology ICAIT 2019 - Proceedings*, pp. 406-410, 2019.

[23] Z. Kherlenchimeg and N. Nakaya, Network Intrusion Classifier Using Autoencoder with Recurrent Neural Network, pp. 94-100, 2018.

[24] Y. N. Kunang, S. Nurmaini, D. Stiawan, A. Zarkasi and F. Jasmir, "Automatic Features Extraction Using Autoencoder in Intrusion Detection System", *Proceedings of 2018 International Conference on Electrical Engineering and Computer Science ICECOS 2018*, vol. 17, pp. 219-224, 2019.

[25] *IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB*, Jan 2021, [online] Available: https://www.unb.ca/cic/datasets/ids-2017.html.

[26] I. Sharafaldin, A. H. Lashkari and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization", *ICISSP 2018 - Proceedings of the 4th International Conference on Information Systems Security and Privacy*, vol. 2018, pp. 108-116, January 2018.

[27] R. Panigrahi and S. Borah, *A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems*, vol. 7, pp. 479-482, 2018.

[28] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization", *Journal of Machine Learning Research*, vol. 13, pp. 281-305, 2012.

[29] . Liashchynskyi and P. Liashchynskyi, *Grid Search Random Search Genetic Algorithm: A Big Comparison for NAS*, no. 2017, pp. 1-11, 2019.

[30] G. Adam and P. Josh, Deep Learning: A Practitioners Approach, 2017.

[31] Khushnaseeb Roshan, Aasim Zafar, Shiekh Burhan Ul Haque, "A Novel Deep Learning based Model to Defend Network Intrusion Detection System against Adversarial Attacks", *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp.386-391, 2023.