National College of Ireland

Enhanced DDoS Attack Detection with Autoencoder Deep Learning Models

MSc Research Project
MSc Data Analytics

Ebin Sujin
Student ID: x23205814

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	EBIN SUJIN
Student ID:	x23205814
Programme:	MSc Data Analytics **Year:** 2024
Module:	MSc Research Project
Supervisor:	Christian Horn
Submission Due Date:	12-12-2024
Project Title:	Enhanced DDoS Attack Detection with Autoencoder Deep Learning Models
Word Count:	7498 **Page Count:** 24

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: EBIN SUJIN

Date: 12-12-2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhanced DDoS Attack Detection with Autoencoder Deep Learning Models

Ebin Sujin
x23205814

Abstract

DDoS (Distributed Denial of Service) is a type of cyberattack where multiple compromised systems are used to flood a target server, service, or network with an overwhelming amount of traffic. This study aims at improving detection of DDoS assaults using the UNSW-NB15 dataset, which comprises a versatile representation of network traffic that is derived from modern normal activities and simulated attacks. To tackle class imbalance, the Synthetic Minority Oversampling Technique (SMOTE) was used and overall results showed an impressive distribution for various attack categories. The proposed models were evaluated based on the following criteria: Machine and Deep Learning models including Decision Tree Classifier, Logistic Regression, Long Short Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM). Notably, the implementation of Autoencoders for feature extraction was a major improvement in the distances used on the model. About all tested models the highest test set accuracy of 86% was obtained with the use of the Bi-LSTM with Autoencoder as it was able to capture longer sequential dependencies in the network traffic data efficiently. The dataset is initially a binary one that is changed to a multi-class to be able to differentiate nine different types of attacks from normal traffic. The findings of this work also establish the possibility of enriching the relational Deep learning architectures to enhance the reliability of DDoS detection systems that can then support the improvement of cybersecurity systems in contemporary networks. Subsequently, the insights call for further study of other Deep Learning approaches so as to counter new threats in network security successfully.

Keywords: DDoS Attack, UNSW-NB15 Dataset, Machine Learning (ML), Deep Learning (DL), Bidirectional LSTM (Bi-LSTM), Autoencoders

1 Introduction

1.1 Background

Distributed Denial of Service (DDoS) attacks remain one of the top available threats to the continuity of online services affecting both the corporate and individual worlds (Wani et al., 2021). They point out that at the same time as attackers advance in their use of the improved methods and massive botnets, traditional detection approaches become ineffective, which gave rise to a need for new and more efficient approaches. Artificial Neural Networks particularly Autoencoders were identified as a useful tool to improving detection of DDoS attacks based on the anomalies in the network traffic patterns (Fouladi et al., 2022). Namely, autoencoders can detect normal traffic and, using a compressed version of a normal traffic model (Boquet et al., 2020), classify and isolate malicious traffic. This capability is particularly valuable in the case when attack signatures can completely differ and gradually evolve. The development of improved Machine Learning methodologies have led to incorporation of autoencoders with other detection strategies creating favorable features for efficient detection methods with reduced false positives. In addition to improving the

robustness of network structures, these activities are valuable for the application of cybersecurity knowledge and analysis of attack behavior to enable timely countermeasures. Given the ever increasing occurrence and sophistication of DDoS attacks, the establishment of Autoencoder based detection systems will remain critical to protecting valuable digital assets and providing uninterrupted services.

1.2 Purpose of the study

The purpose of this work is to propose an intelligent DDoS attack detection system using UNSW-NB15 dataset and applying Autoencoder feature extraction, and Bi-LSTM models. To further counteract these shortcomings of existing methods this research aims to increase the performance of DDoS detection through the prevention of class imbalance, alleviation of the problem of feature redundancy and the handling of network traffic data. Through the usage of Autoencoder models, the study aims at answering the research questions by minimizing the dimensionality of the feature set while further improving classification models' performance, though doing so shall not sacrifice the significant features' quality. For the representation of temporal aspects, that is characteristics that change over a sequence of instances in network traffic data, Bi-LSTM model is employed due to inherent bidirectional sequence analysis capabilities to forward and backward directions to give better representation of attack patterns. Apart from deep learning models, other machine learning models identified as logistic regression and decision tree are also used for the purpose of comparison. The intention is to reach high true positive rate in DDoS detection while having a low number of false positives for its use cases in real-world security systems. The aim of this research is to propose a reliable and easily extendable algorithm for network intrusion detection and thus enhance the general protection of complex current networks against new types of threats.

1.3 Research Questions & Objectives

1.3.1 Research Questions

There are few research questions for this research are as follows:

1. This research proposes the integration of Autoencoder-based feature extraction with Bi-LSTM models in order to build an advanced DDoS attack detection system.

1.3.2 Research Objectives

There are some research objectives which are as follows:

1. How can the integration of Autoencoder-based feature extraction improve the accuracy of DDoS attack detection compared to traditional feature selection methods?
2. What role does Bidirectional Long Short-Term Memory (Bi-LSTM) play in enhancing the detection of complex and evolving DDoS attack patterns in network traffic data?
3. How does the combination of Autoencoders and Bi-LSTM models reduce false positives in DDoS detection while maintaining a high detection rate across different types of attacks?

2. Literature Review

2.1 Understanding DDoS Attacks

Distributed Denial of Service is a form of cybercriminal activity that occurs where attackers send unrealistic volumes of traffic to a server, service or network to exploit its operations (Hyslip, 2020). These attacks employ several controlled computers harbouring malware to create massive traffic, and subsequently, overwhelm the target with such traffic; therefore, depriving its users an opportunity to access its services (Salim et al., 2020).

The DoS attack usually originated from a single source, while the DDoS attack utilises numerous devices, termed as botnet comprising of thousand or even millions of machines (Alahmadi et al., 2023). These devices are usually located all over the world and are manipulated by the attacker unknowns to the owners (Kumari and Jain, 2023). The objective of a DDoS attack is to consume the target's resource, say available bandwidth, processing capabilities or memory space (Bhayo et al., 2021) leading to a service outage or a substantially reduced quality of this service.

DDoS attacks can be categorized as; Bandwidth consumption attacks (Santos et al., 2020). These attacks are responsible of flooding the connection link hence consuming all the available bandwidth of the connection. These type of attacks is responsible of using up all available protocol resources of the connection. Layer 7 or application layer attacks: These attacks are aimed at specific applications such as web sites or databases. It occupies net resources through intensive transmission of data in volume based attack while there are areas in net protocol like the TCP/IP that can easily be exploited.

The most sophisticated attacks target the application layer, where web pages are built, which makes it challenging to identify the threats among standard traffic (Mishra and Pandya, 2021). Since such attacks can be distributed across multiple targets, and attacks can be of varying strength and organization, the identification of such threats and subsequent counteractions is extremely difficult. This makes it significant to advance in the detection and prevention of DDoS since they are on the increase in both frequency and scale, with impacts that may lead to serious financial losses, ruined reputation and disruption of the continuity of services for those who depend on the internet mostly as a business. Therefore, the detection and mitigations of DDoS attacks are among the most important goals in modern cybersecurity.

2.2 ML Models in DDoS Attacks

DDoS attacks are serious threats to networks and systems: they make the targeted node completely unusable for a while, significantly reduce its performance and cause direct and consequential losses. With the increased use of internet based services, prevention of such attacks should be detected early to prevent the worst from happening. Prior DDoS work has attempted at proposing differing methodologies for detecting the DDoS attack, rousing the utilization of several machine learning techniques. Hence in (Tufail et al., 2022), the authors compared the performance of a logistic regression and a shallow neural network (SNN) for DDoS attack prediction. The main finding here is that, although SNN model offered slightly higher accuracy of 99.85% opposed to logistic regression's 98.63%, the amount of time it took to train the model was substantially larger, a sign that there is a relation between accuracy and time taken to train the model. This work indicated directions towards future enhancements including; A multi-class DDoS detection and the incorporation of issues

related to getting the users involved in a DDoS network through social engineering into the system.

Table 2.1: Distinction Between Binary Classification and Multi-Class Classification

Aspect	Binary Classification	Multi-Class Classification
Definition	Predicts between two classes or categories (e.g., Yes/No).	Predicts among more than two classes or categories.
Number of Classes	Only two classes (e.g., Class 0 and Class 1).	More than two classes (e.g., Class 0, Class 1, Class 2, etc.).

A decision tree classifier for a real time DDoS detection was proposed in (Khare and Oak, 2020). In this model feature extraction and information gain is used to develop a decision tree for distinguishing between DDoS and legitimate network traffic. The proposed model had sensitivity and specificity higher than the benchmark algorithms with a success rate of 90.2%. Its key strength is its capability to run in real time in live systems , meaning in addition to protecting data, it can also identify the source of the attack. The small size of the decision tree classifier is more fitting for real-time applications as it is; it is essentially practical.

On this foundation, (Coelho and Filho, 2022) further proposed the possible usage of Random Forest (RF) classifiers in programmable networks and switches/routers to support dynamic DDoS detection. As an RF based approach that employs numerous decision trees for identifying network flows, this method provided both high accuracy, as well as low computational complexity. Although the size of the RF configuration was minimized to 63 match-action table entries, the model demonstrated F1-scores higher than 90 % for DDoS detection in programmable networks and real-time control uses.

So, these works present a transition in the detection of DDoS from generic statistical methodologies such as logistic regression or decision trees, to more refined, yet efficient methods including shallow neural networks and the Random Forest algorithm. The problem still persists, how to obtain a solution that is optimal but also fast enough to be applied quickly in massive real life networks where timeliness is of the essence.

2.3 DL Models in DDoS Attacks

These days, DDoS is common and that is why advanced measures for detecting and countering it are called for. In the study, (Cil et al., 2021) divides the packet samples collected from the network traffic and proposes a DNN as a deep learning model for the early detection and classification of DDoS attacks.. The proposed approach exploits the DNN's feature extraction and classification steps as its integrated steps to enable it work fast and with high accuracy even when dealing with small datasets. Among these was the problem of detection against a background of growing traffic, which increased the vulnerability to cyber attacks. In tests on the CICDDoS2019 dataset, given as a range of different DDoS attacks created in 2019, the model showed excellent results, providing a detection efficiency of DDoS attacks at 99.99% and classification accuracy of attack types at 94.57%. These high accuracy values as shown prove that the DNN model can be used adequately to address DDoS attacks.

Another study of dl model given by, (Zhao et al., 2023) presents an approach of DDoS attack detection which uses self-attention mechanism together with the combination of CNN and BiLSTM networks to address the issues such as high dimensionality the instance data has, multiple feature dimensions, low accuracy of the classification as well as high false

positive rate in raw traffic data. The presented plan, at the initial step, uses the random forest algorithm combined with the Pearson correlation matrix to eliminate unimportant features, thus minimizing the data inputs. Thereafter, one-dimensional CNNs extract spatial features while the Bi-LSTM captures temporal features, and these features are parallelized to form fused representation. This is to help focus on complex features of inputs through giving them appropriate weights in tasks in the model. Last but not the least; the SoftMax classifier is used to determine the classification outcomes. In order to confirm this approach, binary and multi-classification were conducted on CIC-ISD2017 and CIC-DDoS2019 dataset which had remarkable performance with 95.670% accuracy, 95.824% precision and 95.904% recall and 95.864% F1 scores of the proposed model.

Furthermore, a third study given by (Singh and Jayakumar, 2022) proposes a comprehensive framework for DDoS attack detection and mitigation, divided into two stages: detection and mitigation. In the detection phase, an Improved Update-oriented Rider Optimization Algorithm (IU-ROA) is utilized for feature selection from the extracted data while the Deep CNN is used to classify the data. The second phase also includes a bait detection mechanism in which authors achieve 90.06% of the mitigation performance and outperform other techniques examined in this study by 96% in detection performance when tested on the KDD Cup 99 dataset. Taken together, these works paradigm and demonstrate the progress in DDoS detection and mitigation mechanism and reveal how deep learning models and advanced algorithms improve the network security systems.

3. Research Methodology

3.1 Methodology

1. **Business Understanding:** For the Enhanced DDoS Attack Detection project, the main business concern is in building a highly accurate and universally scalable DDoS attack detection and real-time optimization system to reduce the large-scale financial and operational losses that DDoS attacks bring to any businesses and other critical establishments. Due to an overload of traffic, DDoS attacks deny services for a specific period making organizations and businesses lose a lot of money, reputations, and user confidence. Existing IDS today suffers for example false alarms, low true positive rate, and flexibility, that is, it is hard to detect new types of attacks. To overcome these challenges the current project added the deep learning models, Long Short Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), and autoencoder for feature extraction to enhance the accuracy and the invulnerability of the DDoS detector.
2. **Data Understanding:** During the Data Understanding phase of the Enhanced DDoS Attack Detection project, emphasis is placed on a detailed analysis of the UNSW-NB15 dataset which is composed of both real and generated network activities, both legitimate and illicit such as DDoS, Fuzzers, Exploits, amongst others. During the preprocessing step exploratory data analysis, EDA, is carried out in order to analyze the distribution of features and possible relations in the attack traffic and to differentiate between the normal and abnormal behavior. It is useful for understanding the dataset when more features will be created to enrich it, and for developing machine learning models. It is equally important to know the details of the data in

order to apply the appropriate kind of preprocessing methods to make sure that the models are trained using clean meaningful and well-balanced data.

3. **Data Preparation:** The application of data preparation in the Enhanced DDoS Attack Detection project involves several consecutive phases that transform raw data obtained from the UNSW-NB15 dataset to be effectively used in the machine learning models. Due to the fact that the developed dataset have class skewness with DDoS attack instances far less frequent than normal traffic, SMOTE is then employed to balance the instances of each class so that, in training the model, the instances of attack and normal will be almost equal.
4. **Modelling:** In the Modeling phase of the Enhanced DDoS Attack Detection project, several machine learning and deep learning models are employed to classification of DDoS attacks with relatively high accuracy. The models described in this paper are trained on the preprocessed UNSW-NB15 dataset which was transformed from its original high feature space to lower dimension space with the help of autoencoders. To detect DDoS patterns intensified on long-time behavior of traffic flows LSTM models with and without autoencoder feature extraction layers are used. To complement this, self-recurrent structures such as Bi-LSTM is used to improve detection characteristics of the model by processing an input sequence in forward and backward pass improving the recognition of attack patterns in the process. Subsequently, each of the models is fine tuned or hyperparameter tuned to achieve optimal results. To accomplish this, models need to be designed that can generalize over new unseen data points; the use of the autoencoder-enhanced LSTM and Bi-LSTM models used in this paper should provide better performance than conventional techniques in identifying DDoS attacks.
5. **Evaluation:** In the Evaluation phase of the Enhanced DDoS Attack Detection project, the results in terms of certain parameters of conventional machine learning and deep learning algorithms are evaluated and compared to know their efficiency in DDoS attacks detection. The performance of the models is then determined on the test set this was achieved by initially isolating the test set from the training set. Evaluation measures like accuracy, precision, recall value, a technique named F1-score is used in order to see the efficiency of the models to differentiate the normal traffic from attack traffic. A confusion matrix is also used to give detailed information on the classification result; true positive: it is the number of attacks correctly identified, false positive: it is the number of normals falsely accused to be attack, false negative: it is the number of attack he missed and true negative: it is the number of normals that was correctly identified.

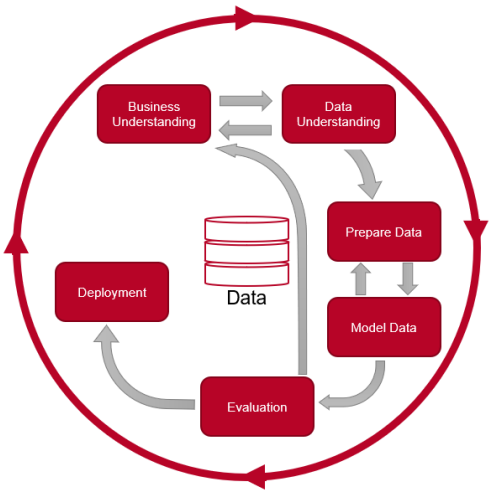

Figure 3.1: CRISP-DM architecture by (https://michael-fuchs-python.netlify.app/2020/08/21/the-data-science-process-crisp-dm/)

3.2 Dataset Description

The UNSW-NB15 dataset was generated using IXIA PerfectStorm in the Cyber Range Lab of ACCS by mimic real world current network traffic and synthetic attack scenarios. The collected dataset is captured by Tcp dump tool, focused with 100 GB of pure raw network traffic in the format Pcap, includes normal and malicious traffic. It includes nine distinct types of attacks: Analysis, Fuzzers and so on of today's threats. The data was analyzed with the help of Argus and Bro-IDS tools, and twelve types of algorithms were applied to compose 49 features. These features involve simple traffic metrics such as packet counts and times along with advanced characteristics features including flags, protocol type, and connectivity states, it is appropriate for IDS studies.

3.3 Data Visualization

This bar chart in Fugure 3.2 shows how various types of attacks look like when they are used in cybersecurity systems. On the x-axis, nine different types of attack have been depicted (Normal, Generic, Exploits, Fuzzers, DoS, Reconnaissance, Backdoor, Shellcode and Worms), and on the y-axis is the frequency. Normal traffic incurs with 35,000 instances, While the Generic attacks are recorded at about 18,000 instances and Exploits at around 10,000 instances. The Fourth one is fuzzers with nearly five thousand cases followed by DoS with roughly four thousand and Reconnaissance attacks, and nearly four thousand cases. The overall Backdoor, Shellcode, and Worms types are the least prevalent, with each type presenting fewer than 2,000 cases. Every attack category has its own colored bar where one colored bars differ from another; at the right top of the chart there is the legend. It also shows how the various types of network traffic can be categorized with 'Normal Traffic' comprising the most significant part of the data, different types of attacks with decreasing numbers to the least.

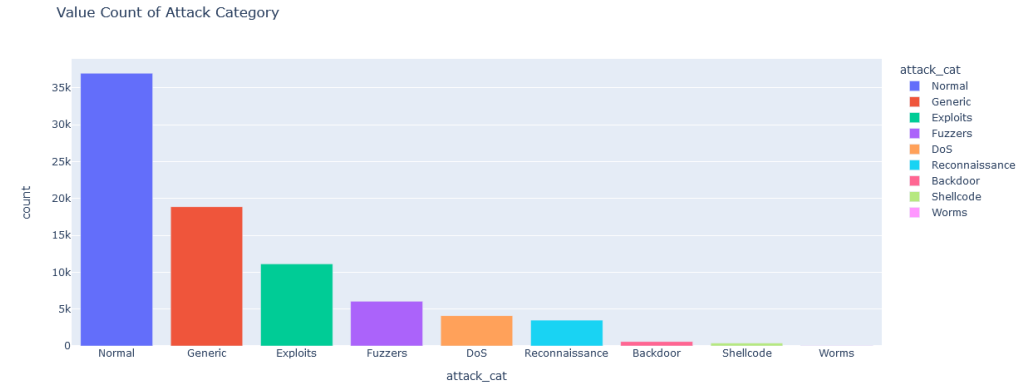

Figure 3.2: Value Count of Attack Category

According to Figure 3.3, the "smean" column looks like a statistical or analytical measurement, so the data is distributed as follows, smean is the mean of the flow packet size transmitted by the source. On the x-axis is the scale of unique enterprise variation found in the "smean" category while the y-axis represents relative strength or count.

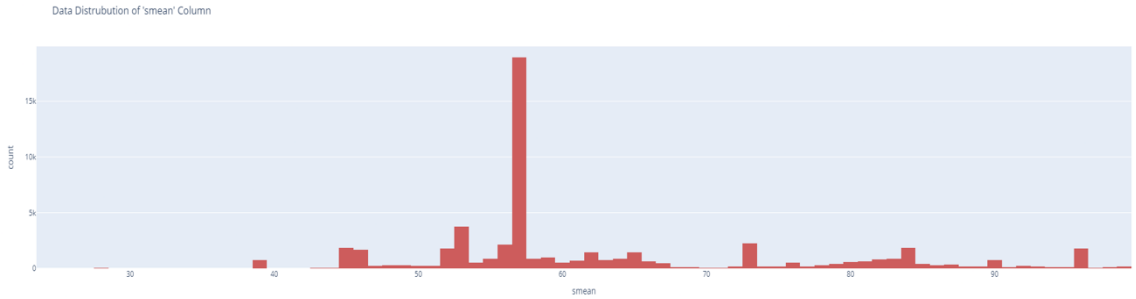

Figure 3.3: Data Distribution of "smean" Column

The second lowest result in discovering "proto" values is shown in table form in Figure 3.4 – the top 10 "proto" value counts. The first row are the categorical variables which stand for the "proto" values mentioned earlier while on the second row we get the count or possibly the frequency of a particular value. The table shows the top 10 "proto" values with the label.

Proto	**tcp**	**udp**	**unas**	**arp**	**ospf**	**sctp**	**any**	**gre**	**rsvp**	**ipv6**
Count	**43037**	**29418**	**3289**	**987**	**668**	**320**	**87**	**84**	**58**	**58**

Figure 3.4: Top 10 "proto" Value Count

The line chart in Figure 3.4 shows the value count of the attack types by state. The x-axis shows the category of the attacks including ACC, CON, FIN, INF, and some others. The y-axis represents the number of values that each kind of attack belongs to. As viewed from the chart, various categories of attacks seen in the dataset are not equally represented with some such as FIN and INF having several folds higher value counts.

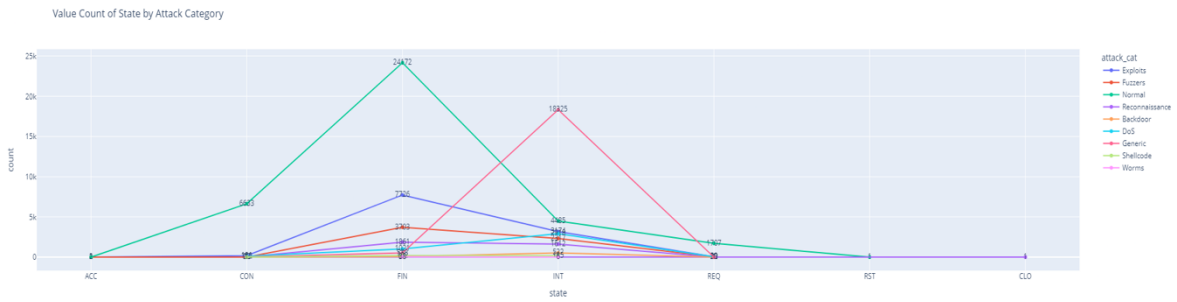

Figure 3.5: Value Count of State by Attack Category

3.4 Libraries Imported

In this project, several libraries are imported to aid in data processing, analysis and building up of several models for machine learning. NumPy is used for using and handling the data while Pandas is used for data manipulation and analysis, Seaborn for the visualization of the data, Plotly Express for the creation of plots in EDA, and lastly Matplotlib is used for creating the plots of the data. Matplotlib is used for basic plots and other specialized plots are done using Interactive Plotting Functions, making it easier in handling large datasets. Scikit-learn provides algorithms and tools for machine learning activities, such as LabelEncoder to transform nominal variables into numeric values as well as normalize the dataset with the help of MinMaxScaler and train_test_split the dataset into sets which will train and test. Balance the class matrix is achieved using SMOTE from the Imbalanced-learn library for oversampling the minority class. These deep learning models are developed using TensorFlow, Keras framework such as layers consist of LSTM, Bidirectional LSTM, Dense, Dropout, BatchNormalization. Some of the optimizers which had been used while training the models includes Adam, SGD, and RMSprop. The general machine-learning models imported are LogisticRegression, DecisionTree Classifier, Random Forest Classifier from the Scikit-learn Library.

3.5 Data Preprocessing

In this project, the data preprocessing phase starts with the categorical variables that are important in transforming the data that is in an unusable format to the one that is understandable by the model. We start first by defining the categorical columns as those that are non-numerical, which in this current dataset includes ‘proto’ and ‘state’. For these, we use Label Encoding in which categorical variables are converted into numbers. This process is essential for the algorithms of machine learning to understand all these fields. After doing the label encoding, the dataset has now got only numeric features, which will be more convenient for the algorithms to understand. Converted categories are presented in the transformed data where each value in the same category is given a different integer. After this, we perform data normalization as all the features are scaled using the MinMaxScaler with a range of 0 to 1 units. Normalization is important in deep learning and machine learning because it accelerates convergence in the training process, prevents a model from ending up in a local minimum, and generally improves the learning process by preventing large valued features from skewing the learning process. In this case, for every attribute in the dataset, every value forms a range and the transformation brings all those values within that range. Furthermore, label encoding is used on the degree of expertise target labels where the LabelBinarizer transform the labels to a binary format for use in multi-class deep learning models.

3.6 Data Balancing with SMOTE

In this project, the SMOTE method is used to solve the problem of imbalance in the numbers of data samples within the dataset. Given that balanced distribution of class instances significantly differs especially when there is a large number of instances in one particular class, this results to a biased model that tends to predict the majority class. Compared to other oversampling methods, SMOTE compensates for the imbalance in the dataset by synthesising data samples for the minority class while training in such a way that all categories become approximately balanced. It is a technique by which new data points belonging to the minority class are generated by operating between the present data points. As it is for SMOTE, it is followed by a resampling of the set, wherein each class contains the same proportion of samples, including both the benign traffic and the different types of the attack. This means that during training the model has enough examples of each category making it easier for it to detect and generalize on the rare type of attacks.

The bar chart below in Figure 3.6 presents the "Value Count of Attack Category After Apply Smote Over Sampling" showing more or less equal distribution of different attack categories in this condition through applying the SMOTE method. As compared to the distribution plot in Figure 3.2, all classes of attack have equal instances of nearly 35000. Normal, Reconnaissance, Backdoor, DoS, Exploits, Fuzzers, Worms, Shellcode, and Generic attacks with their color code respectively. This balanced distribution owes its attainment to the SMOTE technique which is applied in handling the problem of class imbalance in the dataset by synthesizing new instances into the area of the minority classes where the majority occupies. This tilt has been brought about especially when compared with the initial distribution which can be viewed and compared with the one below where the Worms and Shellcode had a very EmpricalDistribution.jpg small representation. This balancing of the dataset is important for machine learning solutions as main tenet is to avoid the training of the model with biased data and the model we build should be equally sensitive to all varieties of attacks rather than observing only those that are most frequently occurring.

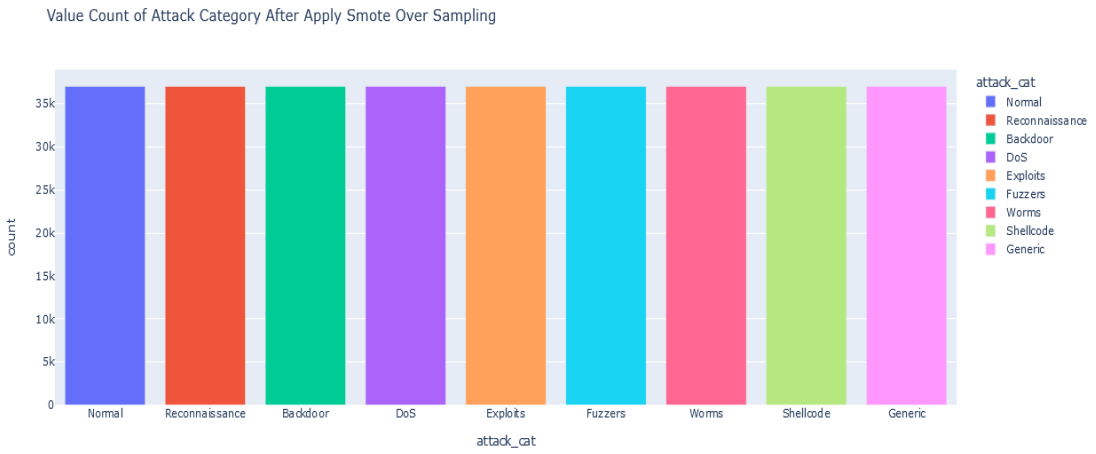

Figure 3.6: Value Count of Attack Category After Apply Smote Over Sampling

3.7 Data Splitting (Training and Testing the Model)

Typically in this step, the dataset is divided into two or more portions for use in training, testing and validation of the model as it is trained on a good portion of the data but is checked for performance on a small unseen subset of the data. Firstly, matrix of features X =

df.drop(['attack_cat', 'label'], axis=1) where as the target variable y contains the attack categories. To guarantee the generalization of the results and to prevent the model from based on similar examples the data is divided into training and testing in the ratio of 9: 1 using the train_test_split function from sklearn. This split ratio is important because, on the hand it gives enough data for the model to learn to pick up patterns, yet it leaves a portion of data that has not been exposed to the model with which to test the model's ability to predict new unseen data. The training set (X_train and y_train) has 299700 records and 41 features which are used by the model to interpret both normal and attack traffic. On the other hand the test set samples a small subset as the focus here is to have a standard by which the result of the model can be accurately judged by its performance on unseen data which is shown below. Through doing this, this step makes it possible to test the accuracy of the model on data it has not been exposed to in a bid to understand how it will possible to contain the DDoS attacks.

4. Design Specification

This paper presents the design specification of an IDS, which describes the architecture, methods, and tools required for effectively designing the IDS solution. The project based on the UNSW-NB15 dataset which consists of normal and attack traffic and capable of using modern machine learning including deep learning to detect DDoS attacks. Some of the components of the system under consideration are the pre-processing of data, the features' extraction, selection of the models, and assessment of performances. Missing data is handled by converting missing values into the mean of data already existing; features containing categories are encoded The features are normalized while utilizing MinMaxScaler(). Data preprocessing is performed to observe that the distribution of the data is balanced using the SMOTE technique to address the imbalance of classes for minority classes. This task involves feature extraction and dimensionality reduction via an Autoencoder – which will select the most optimized features from the large feature set improving the efficiency and accuracy of the models. Using deep learning model, the LSTMs along with Bi-LSTMs do include Autoencoder feature extraction. The LSTM models are designed to learn the temporal dependencies inherent, in the network traffic data while the Bi-LSTM models enable learning of context for temporal sequences that are processed in both directions. When define the structure of the model we use the batch normalization layer, dropout layer to avoid the overfitting problem, and the full connected layer for output the final result of classification.

The system is built in the Python language and utilizes such libraries as TensorFlow and Keras, as well as Pandas for handling data and Plotly for data visualization, and Keras for modelling. The logistic regression and the decision tree classifiers are also trained as a comparison. Performance measurement includes accuracy, confusion matrix, and classification report to achieve valid assessment of the structured system's capacity to identify and sort DDoS threats appropriately. This design combines conventional and advanced machine learning as a way of boosting detection efficacy, while at the same time, minimizing wrong detections.

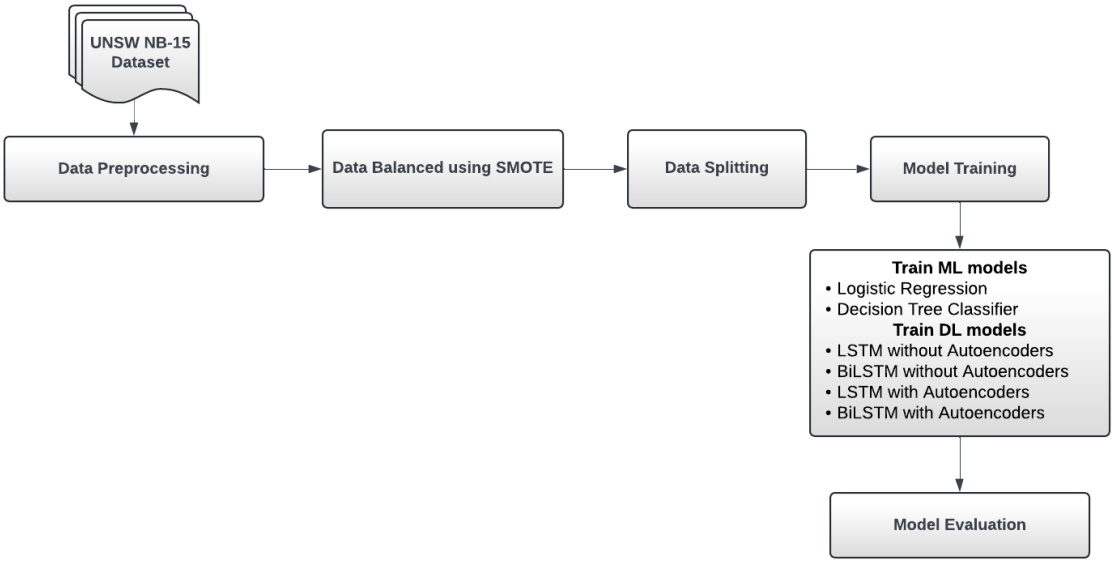

Figure 4.1: Proposed Workflow

5. Implementation

5.1 Machine Learning Models

5.1.1 Case Study 1: Logistic Regression Results

This Figure 5.1 visualises how the logistic regression model performs in terms of confusion matrix heatmap. In the matrix with dimensionality 9 by 9 actual labels are in rows (from 0 to 8) and predicted labels are in columns (also from 0 to 8) and the colors represents frequency of predictions. In the heatmap, the color scale assigned runs from dark purple which suggests the lowest values to yellow which suggests high values; while the intermediate colors like pink suggest moderate values. Several patterns are clearly distinguished; the diagonal line suggests that the model's predictive accuracy for these classes is high, especially in positions (0,0), (4,4), as well as (8,8).

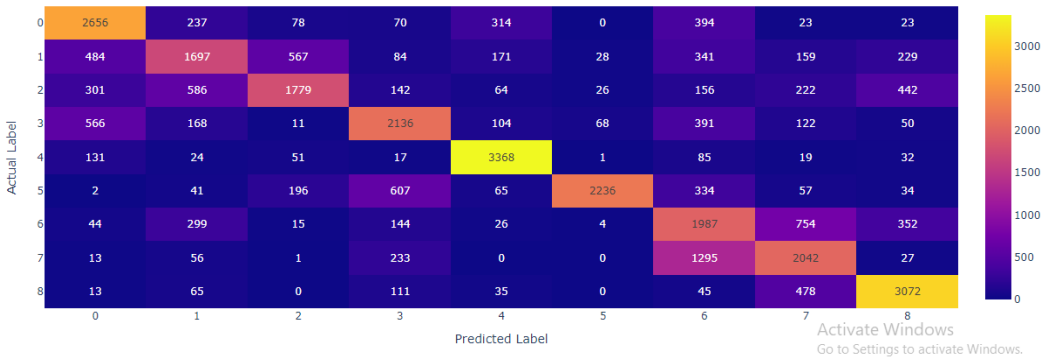

Figure 5.1: Confusion Matrix

Figure 5.2 illustrate a detailed performance matrices of a logistic regression model for traffic classification that is, different type of network activities or attacks. The figure

summarises four performance indices - precision, recall, F1-score and support – for backdoor, DoS, Exploits, Fuzzers, Generic, Normal, Reconnaissance, Shellcode and Worms. In comparing the results obtained for Generic traffic with those obtained for Normal traffic, the former exhibits the highest both precision of 0.81 and recall of 0.90 with an F1-score of 0.86 while the latter has a high precision of 0.95 but a moderate recall of 0.63. At the same time, Reconnaissance also indicate the lowest accuracy as 0.40 while DoS demonstrates the lowest recall as 0.45 only. The last row of the table contains mean model performance measures; accuracy – labelled as accuracy, macro-average – labelled as macro avg, and weighted average – labelled as weighted avg ; all these parameters are within the range of 0.63 to 0.65; total support of instances:33,300.

	precision	recall	f1-score	support
Backdoor	0.63	0.70	0.66	3795
DoS	0.53	0.45	0.49	3760
Exploits	0.66	0.48	0.55	3718
Fuzzers	0.60	0.59	0.60	3616
Generic	0.81	0.90	0.86	3728
Normal	0.95	0.63	0.75	3572
Reconnaissance	0.40	0.55	0.46	3625
Shellcode	0.53	0.56	0.54	3667
Worms	0.72	0.80	0.76	3819
accuracy			0.63	33300
macro avg	0.65	0.63	0.63	33300
weighted avg	0.65	0.63	0.63	33300

Figure 5.2: Classification Report

5.1.2 Case Study 2: Decision Tree Classifier Result

The matrix in Figure 5.3 presented in a form of color scale where colors range from dark purple which can be associated with low prediction frequencies to yellow with high frequencies. The x-axis is always made of predicted labels while grouped on the y-axis are actual labels. High diagonal values especially at index positions (0,0), (4,4), and (8,8) of 3101, 3534, 2972 respectively suggest improved class discrimination.

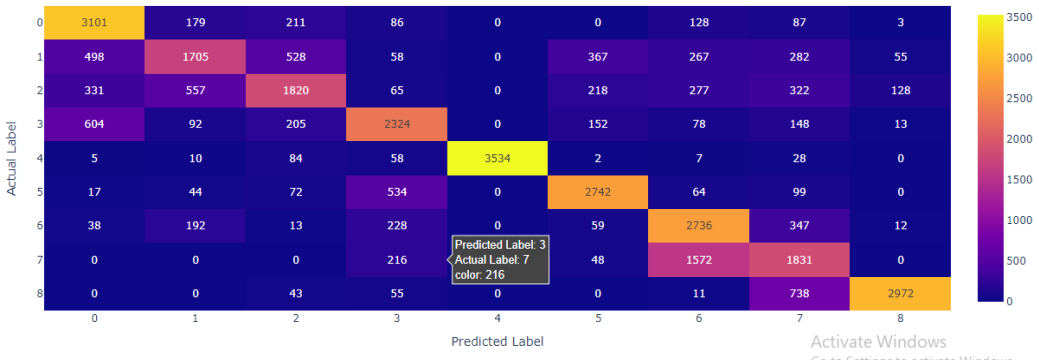

Figure 5.3: Confusion Matrix

In figure 5.4 below, metrics of a decision tree classifier model for classifying network attacks are presented in terms of precision, recall, F1-score, and support for nine classes of attacks. The best results are achieved by the Generic class, know shedding light on its high accuracy, with the value of precision equals 1.00, as well as a high value of recall equals to 0.95 resulting in an F1-score that is the highest and makes 0.97. The Worms and Backdoor classes get the second and fifth highest F1-scores with scores of 0.85 and 0.74 respectively

while Shellcode gives the lowest F1-scores with a score of 0.49. A high, reasonably good, performance is noted on most of the classes and the accuracy score of 0.68 is realised. The two other performance metrics of macro average & weighted average are in 0.68-0.70 which is fair for all the classes. It is evident from support column that the sample size of instances is almost equal for all classes with variation between 3572 and 3819.

	precision	recall	f1-score	support
Backdoor	0.68	0.82	0.74	3795
DoS	0.61	0.45	0.52	3760
Exploits	0.61	0.49	0.54	3718
Fuzzers	0.64	0.64	0.64	3616
Generic	1.00	0.95	0.97	3728
Normal	0.76	0.77	0.77	3572
Reconnaissance	0.53	0.75	0.62	3625
Shellcode	0.47	0.50	0.49	3667
Worms	0.93	0.78	0.85	3819
accuracy			0.68	33300
macro avg	0.69	0.68	0.68	33300
weighted avg	0.70	0.68	0.68	33300

Figure 5.4: Classification Report

5.2 Deep Learning Models

5.2.1 Case Study 1: LSTM without Autoencoders

The performance metrics of a model are depicted in the Figure 5.5 using two sub-plots of the training and validation curves at 30 epoch iterations. The first subplot contains the accuracy curves, where training accuracy (yellow) and validation accuracy (green) are gradually increased from about the 40-45 percent to the 80 percent level, and in the latter 5 epochs validation accuracy is marginally higher than training accuracy. The second subplot presents an analogy to the loss function where it is seen that the values are decreasing from approximately 1.4-1.5 down to 0.6 which reflects proper model learning.

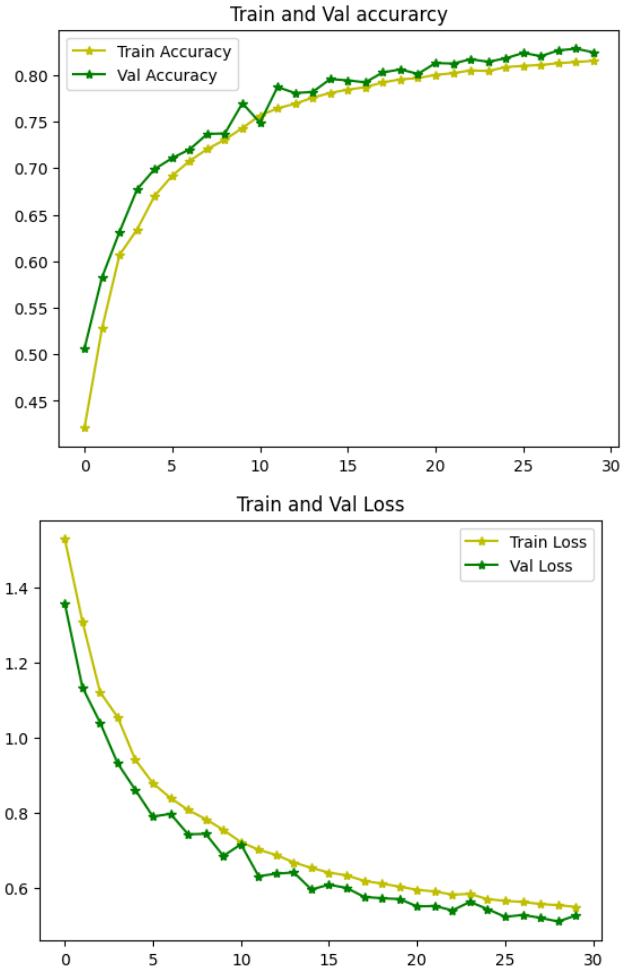

Figure 5.5: Accuracy and Loss Graph

Figure 5.6 is presented with the LSTM model classification performance on nine classes (0-8) in form of Confusion matrix. For this reason, the heatmap indicates a high diagonal elements where there are high values highly displayed at position (0,0), (1,1), (6,6), (7,7) and (8,8) of 3459, 2642, 2995, 3422 and 3653 respectively which refers to good classification accuracy of the classes. This is manifested by a relatively low level of accuracy in classifying, for example, between classes 2 and 1, which involved 927 instances, as well as between classes 5 and 2, 1947 instances, depicted by the pink and orange coloured cells in the off diagonal. The ideal classifier works significantly well in classifying classes 0, 6, 7, and 8 where the bright yellow squares on the diagonal correspond to these classes.

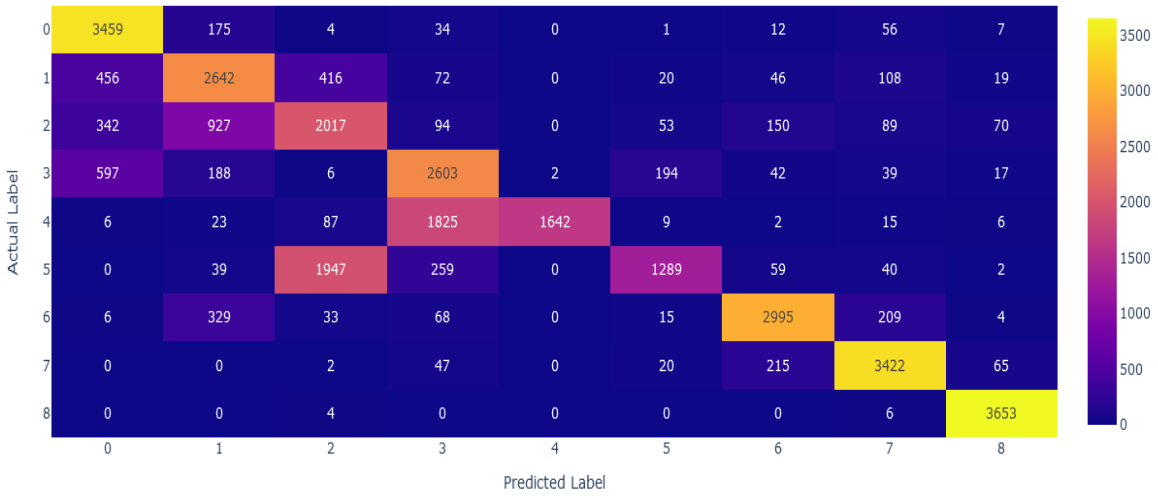

Figure 5.6: Confusion Matrix

Figure 5.7 displays performance of an LSTM model without autoencoders as follows: precision, raw recall, F1-score, and the number of instances correctly classified are shown for nine classes (0 to 8). Class 8 has the best performance score with a recall of 1.00 and precision of 0.95, and a total index of F1= 0.97. Class 4 has perfect precision but, relatively low recall at 0.45. As it is observed from the graph, accuracies differ across classes, and average accuracy is approximately 0.71. The macro avg and the weighted avg are 0.75 for precision, 0.71 for recall and F1-score. Support column reveals a nearly 3,600-3,800 samples distribution per class, and a total of 33,300 instances are available.

	precision	recall	f1-score	support
0	0.71	0.92	0.80	3748
1	0.61	0.70	0.65	3779
2	0.45	0.54	0.49	3742
3	0.52	0.71	0.60	3688
4	1.00	0.45	0.62	3615
5	0.81	0.35	0.49	3635
6	0.85	0.82	0.83	3659
7	0.86	0.91	0.88	3771
8	0.95	1.00	0.97	3663
accuracy			0.71	33300
macro avg	0.75	0.71	0.71	33300
weighted avg	0.75	0.71	0.71	33300

Figure 5.7: Classification Report

5.2.2 Case Study 2: Bi-LSTM without Autoencoders

The performance during training of a Bidirectional LSTM (Bi-LSTM) model without the use autoencoders trained up to 30 epochs is shown in Figure 5.8 in two ways. The accuracy graph indicates fairly good learning evolution of the two metrics, whereby the training and validation accuracy are initiated at between 50% and 58% before rising to about 82%, with the validation accuracy being higher than the training accuracy at each iteration of training.

The loss plot shows a steep decline in the initial epochs, from 1.4 (training), and 1.2 (validation) to 0.8, after which it gradually decreases to 0.5 at epoch 30.

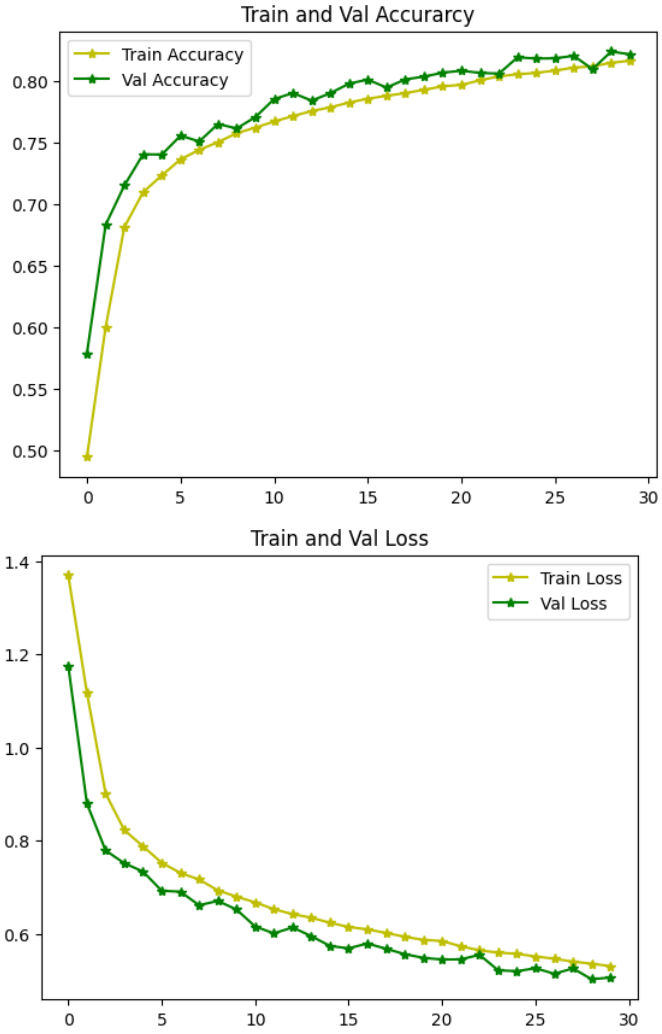

Figure 5.8: Accuracy and Loss Graph

The classification of the bidirectional LSTM model without autoencoders is summarized in terms of confusion matrix heatmap using 9 classes 0-8 in Fig 5.9. The diagonal elements suggest the model has high predictive accuracy as demonstrated in Fig 4 with bright yellow cells at the coordinates (0,0), (4,4), (5,5) and (8,8) of the accuracy score 3563, 3482, 3316, 3727 respectively. The proposed model proves to perform well in these classes though some confusions are evident between classes 1 and 2 derived from off-diagonal matrix values.

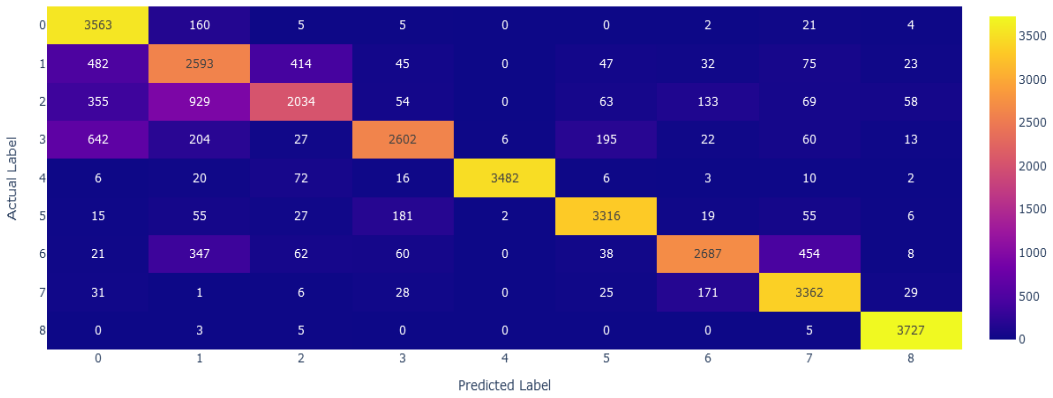

Figure 5.9: Confusion Matrix

The performance metrics of the Bidirectional LSTM with no autoencoder are presented in figure 5.10 in nine classes. According to the model, recall is the highest at 1.00 for class 8 and precision at 1.00 for class 4, with slightly lower F1-scores of 0.98. As we can see F1 scores of all classes are above average and range from 0.64 to 0.98.

	precision	recall	f1-score	support
0	0.70	0.95	0.80	3760
1	0.60	0.70	0.65	3711
2	0.77	0.55	0.64	3695
3	0.87	0.69	0.77	3771
4	1.00	0.96	0.98	3617
5	0.90	0.90	0.90	3676
6	0.88	0.73	0.80	3677
7	0.82	0.92	0.87	3653
8	0.96	1.00	0.98	3740
accuracy			0.82	33300
macro avg	0.83	0.82	0.82	33300
weighted avg	0.83	0.82	0.82	33300

Figure 5.10: Classification Report

5.2.3 Case Study 3: LSTM using Autoencoders

The epochs vs accuracy plot upper line in Figure 5.11 represents the training accuracy for the training set; the lower line represents the validation accuracy for the set of the so named; both increase from initial 55% to approximately 82% showing successful learning without serious overfitting. The corresponding loss plot (bottom) shows a continuous variable reduction in both train and validation loss from initial.13 and.10 respectively to approximately.55 signifying Optimal model tuning and feature extraction ability of the LSTM-Autoencoder architecture.

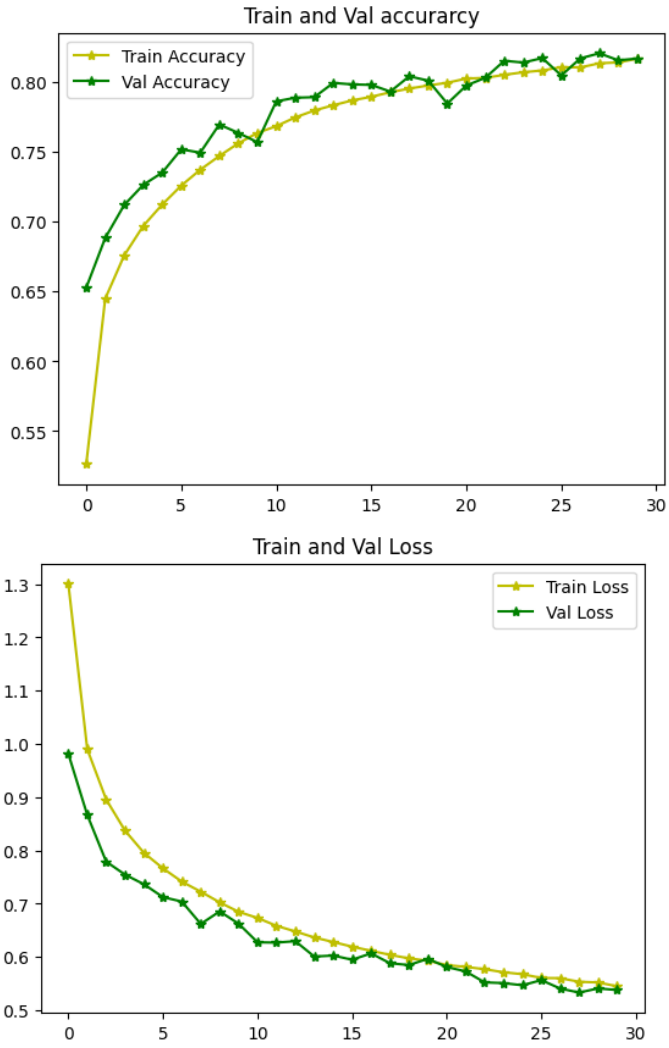

Figure 5.11: Accuracy and Loss Graph

The confusion matrix for the LSTM with Autoencoder model's classification performance given in figure 5.12 focuses on 9 classes of response ranging from 0 to 8. When considering the diagonal, most classes have very good predication accuracy characterized by high values such as 3553, 2572, 1438, 2340, 3401, 3149, 2934, 3570 and 3602.

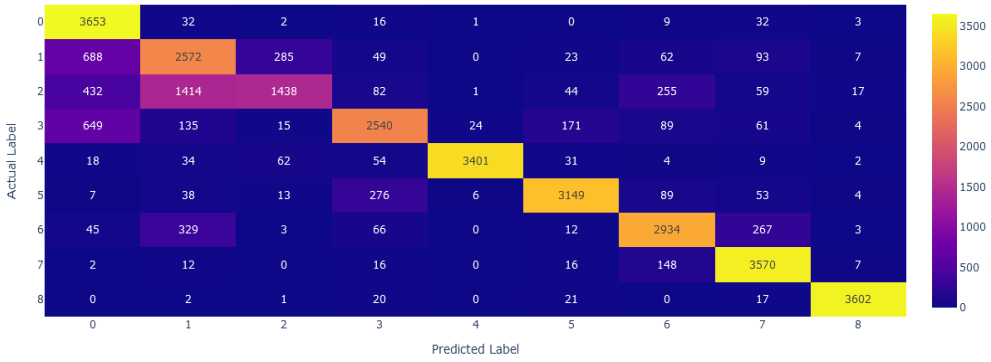

Figure 5.12: Confusion Matrix

Figure 5.13 below shows the classification metrics of the LSTM model without Autoencoders across the nine classes 0-8. The model obtains an average accuracy estimate of 81% with a macro and weighted average of 0.82 and 0.81 for precision and recall respectively. Class-wise performance varies significantly, with class 8 showing exceptional performance (precision: 0.Same kind of result observed in class 3 (precision: 0.99, recall: 0.98 F1 score: 0.99) and class 4 (precision: 0.99, recall: 0.94 F1 score: 0.97). However, class 2 demonstrates relatively weaker performance (precision: 0.79, recall: 0.38, F1: 0.52).

	precision	recall	f1-score	support
0	0.66	0.97	0.79	3748
1	0.56	0.68	0.62	3779
2	0.79	0.38	0.52	3742
3	0.81	0.69	0.75	3688
4	0.99	0.94	0.97	3615
5	0.91	0.87	0.89	3635
6	0.82	0.80	0.81	3659
7	0.86	0.95	0.90	3771
8	0.99	0.98	0.99	3663
accuracy			0.81	33300
macro avg	0.82	0.81	0.80	33300
weighted avg	0.82	0.81	0.80	33300

Figure 5.13: Classification Report

5.2.4 Case Study 4: Bi-LSTM using Autoencoders

The accuracy and loss graphs illustrating the performance of a training and validation set of a machine learning model are illustrated in Figure 5.14 The first plot on top represents the training and validation accuracy over the epochs which increases slowly but steadily for the training set and shows a similar rising but relatively steep line for validation set, confirming the performance of the learning process. The second graph below shows the corresponding train and validation loss, which are initially high but reduce continually as the training advances.

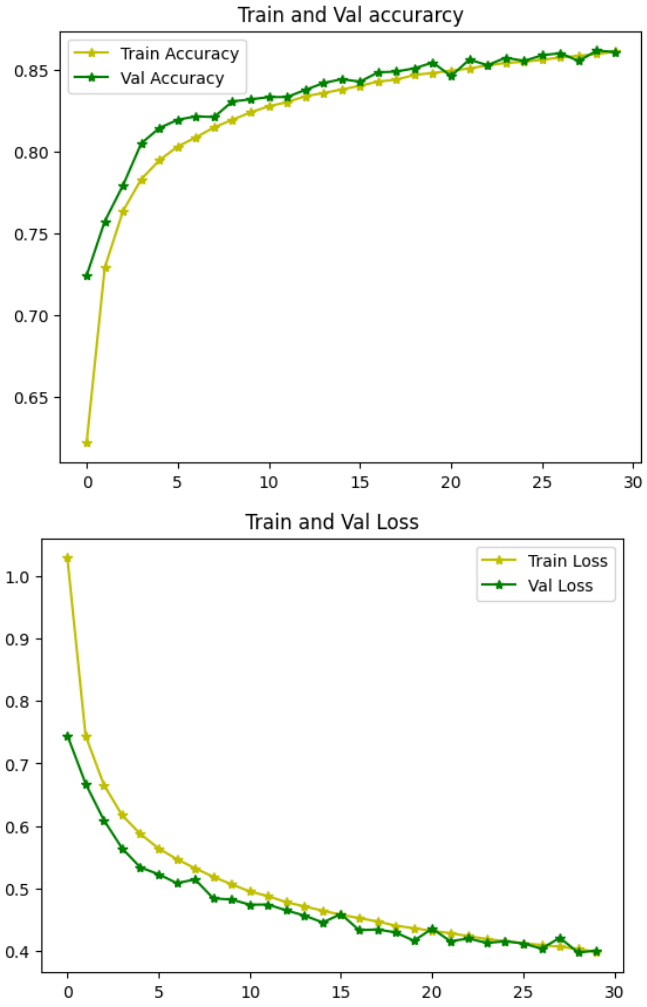

Figure 5.14: Accuracy and Loss Graph

The following confusion matrix corresponds to a Bi-LSTM model with an autoencoder for feature extraction, as shown in Figure 5.15.

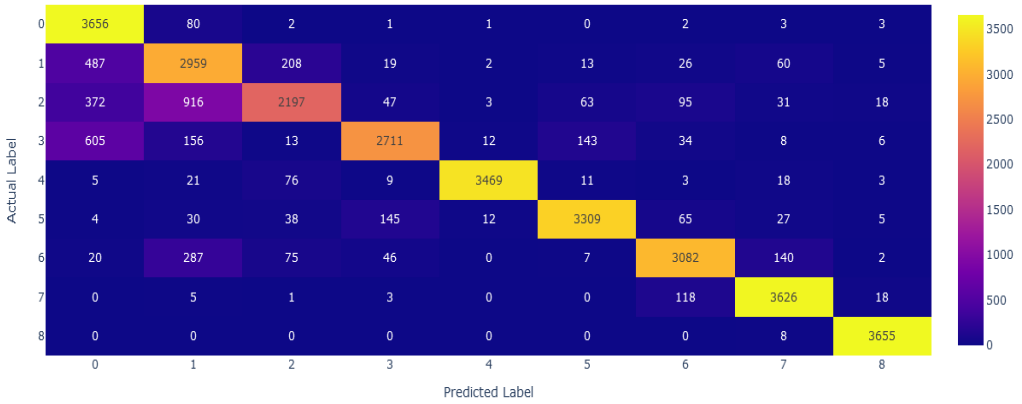

Figure 5.15: Confusion Matrix

Figure 5.16 shows the results in a classification report of the Bi-LSTM model with autoencoder for feature extraction. The results of the test achieved is evidenced by the recall, precision, F1-score, and support values presented in the report for each of the class labels

from 0 to 8. Analyzing the micro-level classification based on the class labels, it can be observed that classes 5, 4, and 3 have achieved highest F1 scores of 0.91, 0.96, 0, 81 respectively. The model also has reasonable performance on the other class labels the F1-score of which varies from 0.59 to 0.98. The first column represents support, which shows the number of samples of each class to give context to the model's performance on samples of different types.

	precision	recall	f1-score	support
0	0.71	0.98	0.82	3748
1	0.66	0.78	0.72	3779
2	0.84	0.59	0.69	3742
3	0.91	0.74	0.81	3688
4	0.99	0.96	0.98	3615
5	0.93	0.91	0.92	3635
6	0.90	0.84	0.87	3659
7	0.92	0.96	0.94	3771
8	0.98	1.00	0.99	3663
accuracy			0.86	33300
macro avg	0.87	0.86	0.86	33300
weighted avg	0.87	0.86	0.86	33300

Figure 5.16: Classification Report

6. Results and Evaluation

6.1 Performance of ML and DL Models Classifications

As shown in Table 6.1, the accuracy of different ML and DL models is presented, which makes obvious distinctions and shows a wide variety of the overall performance between these two types of models. The conventional ML models which include the Decision Tree Classifier and Logistic Regression had accuracy levels of 63% and 68% respectively. These models, although useful for simple classification problem, are insufficient and unable to model the non-linear patterns and dependencies in the network traffic data to distinguish between a normal and a DDoS attack. The reason why their accuracy is comparatively low is because most of them cannot deal with high-dimensionality and sequential feature that belongs to the project.

On the other hand, the DL models, namely LSTM and Bi-LSTM, outperform the remaining models. By using only LSTM without Autoencoder for feature extraction, the model attains an accuracy of 71%, while Bi-LSTM without Autoencoders enjoy an accuracy of 82%. These enhancements we have made since the Bi-LSTM has more capabilities to learn sequential dependency information both in forward and backward direction, so it is well suited for identifying the network anomalies and attack signatures.

Autoencoders as used in the case of feature extraction also yield a better performance of the DL models. He explained that the Autoencoders pre-processing technique is of importance in enhancing efficiency of the learning models since they are capable of eliminating the unnecessary data and focusing on meaningful features which define the dimensionality of the data. By testing this method, LSTM with Autoencoders get 81% accuracy, and therefore this work prove that this method have enhanced standard LSTM. The highest accuracy of 86% is obtained using the Bi-LSTM model along with Autoencoder as

feature extraction for this research, thus proving the goodness of integrating sophisticated feature extraction methods with strong temporal models. It also outperforms the other model at identifying detailed features in the data, and in filtering out unnecessary noise for a higher degree of classification. Therefore, Bi-LSTM connected with autoencoders is the most promising model in DDoS attack detection process and is more successful than ML and other DL types.

Table 6.1: Table of Comparison for ML and DL Models

Model	Accuracy of models in (%)
Logistic Regression	63%
Decision Tree Classifier	68%
LSTM Without Autoencoders For Feature Extraction	71%
BILSTM Without Autoencoders For Feature Extraction	82%
LSTM With Autoencoders For Feature Extraction	81%
BILSTM With Autoencoders For Feature Extraction	86%

7. Conclusion and Future Works

7.1 Conclusions

Specifically, for this study, various features of ML and DL models to identify the DDoS attacks were investigated using the UNSW-NB15 dataset. The outcomes shown were that using DL models especially the ones using feature extraction with autoencoders was greatly superior to traditional ML models. Overall, from the results obtained by all models implemented, the impact of the implementation of the Bi-LSTM with Autoencoder-based feature extraction gave the best results, with an accuracy of 86% indicating how beneficial advanced sequential learning methods fused with dimensionality reduction. The paper validates the increased relevance of DL models for cybersecurity solutions, particularly given the large data size and high data dimensionality characteristic of most network traffic. The study agrees with the apprehensions that the generic structures of M, architectures are suboptimal in capturing complex attacking patterns insisting on the use of better structures like Bi-LSTM for enhancing the detection rates and decreasing false positives.

7.2 Discussion

From the results of this study, it is clear that even though ML models including Logistic Regression and Decision Tree are good for the basic level of accuracies, these models are inadequate for the complexity of current day DDoS attack detection. The DL models, especially Bi-LSTM, can learn long term dependencies of the temporal patterns inherent with the network traffic data sequence which is so essential for anomaly identification. Autoencoders employed for feature extractor posed a very effective factor for increasing performance by minimizing data noise and in getting to only the central and important features. However, the training of these DL models incurred more computational cost and the real-time implementation is an added factor. Additionally, although there is extreme performance enhancement in the highest Bi-LSTM model, this research still provides insight for more improvement options, especially for time and scaling up for practical applications.

7.3 Future Works and Limitations

The limitations of the study include that more time was spent on training the DL models and that the models could be more lightweight in future work to accomplish this. The real-time deployment can be built up through the edge computing that makes the models to detect the attack within the network boundary. One can also look into other scenarios, which are the combination of both the ML and DL methodologies to minimize on their failings. Also, premium feature selection or extraction techniques over and above those used by Autoencoder could be contemplated to improve performance. Problems of this study encompass high risk of overfitting taking into account the complexity of the models used and limited data set which was fixed in this analysis and may not be applicable well in different network environment. This approach can be applied with more attack scenarios and realistic traffic data to enhance the performance of the models and their applicability to different cyber security context.

References

1. Wani, S., Imthiyas, M., Almohamedh, H., Alhamed, K.M., Almotairi, S. and Gulzar, Y., 2021. Distributed denial of service (DDoS) mitigation using blockchain—A comprehensive insight. *Symmetry*, *13*(2), p.227.
2. Fouladi, R.F., Ermiş, O. and Anarim, E., 2022. A DDoS attack detection and countermeasure scheme based on DWT and auto-encoder neural network for SDN. *Computer Networks*, *214*, p.109140.
3. Boquet, G., Morell, A., Serrano, J. and Vicario, J.L., 2020. A variational autoencoder solution for road traffic forecasting systems: Missing data imputation, dimension reduction, model selection and anomaly detection. *Transportation Research Part C: Emerging Technologies*, *115*, p.102622.
4. Hyslip, T.S., 2020. Cybercrime-as-a-service operations. *The palgrave handbook of international cybercrime and cyberdeviance*, pp.815-846.
5. Salim, M.M., Rathore, S. and Park, J.H., 2020. Distributed denial of service attacks and its defenses in IoT: a survey. *The Journal of Supercomputing*, *76*, pp.5320-5363.
6. Alahmadi, A.A., Aljabri, M., Alhaidari, F., Alharthi, D.J., Rayani, G.E., Marghalani, L.A., Alotaibi, O.B. and Bajandouh, S.A., 2023. DDoS attack detection in IoT-based networks using machine learning models: a survey and research directions. *Electronics*, *12*(14), p.3103.
7. Kumari, P. and Jain, A.K., 2023. A comprehensive study of DDoS attacks over IoT network and their countermeasures. *Computers & Security*, *127*, p.103096.
8. Bhayo, J., Jafaq, R., Ahmed, A., Hameed, S. and Shah, S.A., 2021. A time-efficient approach toward DDoS attack detection in IoT network using SDN. *IEEE Internet of Things Journal*, *9*(5), pp.3612-3630.
9. Santos, R., Souza, D., Santo, W., Ribeiro, A. and Moreno, E., 2020. Machine learning algorithms to detect DDoS attacks in SDN. *Concurrency and Computation: Practice and Experience*, *32*(16), p.e5402.
10. Mishra, N. and Pandya, S., 2021. Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. *IEEE Access*, *9*, pp.59353-59377.
11. Tufail, S., Batool, S. and Sarwat, A.I., 2022, March. A comparative study of binary class logistic regression and shallow neural network for ddos attack prediction. In *SoutheastCon 2022* (pp. 310-315). IEEE.
12. Khare, M. and Oak, R., 2020. Real-Time distributed denial-of-service (DDoS) attack detection using decision trees for server performance maintenance. *Performance Management of Integrated Systems and its Applications in Software Engineering*, pp.1-9.
13. Coelho, B. and Schaeffer-Filho, A., 2022, December. BACKORDERS: using random forests to detect DDoS attacks in programmable data planes. In *Proceedings of the 5th International Workshop on P4 in Europe* (pp. 1-7).
14. Cil, A.E., Yildiz, K. and Buldu, A., 2021. Detection of DDoS attacks with feed forward based deep neural network model. *Expert Systems with Applications*, *169*, p.114520.
15. Zhao, J., Liu, Y., Zhang, Q. and Zheng, X., 2023. CNN-AttBiLSTM Mechanism: A DDoS Attack Detection Method Based on Attention Mechanism and CNN-BiLSTM. *IEEE Access*, *11*, pp.136308-136317.
16. Singh, S. and Jayakumar, S.K.V., 2022. DDoS attack detection in SDN: optimized deep convolutional neural network with optimal feature set. *Wireless Personal Communications*, *125*(3), pp.2781-2797.