Enhancing Intrusion Detection Systems with XGBoost Feature Selection and Deep Learning Approaches

Khalid A.Binsaeed, Prof.Alaaeldin M. Hafez
College of Computer and Information Sciences, King Saud University, Riyadh, KSA

Abstract—**As cyber-attacks evolve in complexity and frequency; the development of effective network intrusion detection systems (NIDS) has become increasingly important. This paper investigates the efficacy of the XGBoost algorithm for feature selection combined with deep learning (DL) techniques, such as ANN, 1DCNN, and BiLSTM, to create accurate intrusion detection systems (IDSs) and evaluating it against NSL-KDD, CIC-IDS2017, and UNSW-NB15 datasets. The high accuracy and low error rate of the classification models demonstrate the potential of the proposed approach in IDS design. The study applied the XGBoost feature extraction technique to obtain a reduced feature vector and addressed data imbalance using the synthetic minority oversampling technique (SMOTE), significantly improving the models' performance in terms of precision and recall for individual attack classes. The ANN + BiLSTM model combined with SMOTE consistently out performed other models within this paper, emphasizing the importance of data balancing techniques and the effectiveness of integrating XGBoost and DL approaches for accurate IDSs. Future research can focus on implementing novel sampling techniques explicitly designed for IDSs to enhance minority class representation in public datasets during training.**

Keywords*—*Intrusion detection system; deep learning (DL); XGBoost; feature extraction; Bidirectional Long Short-Term Memory (BiLSTM); Artificial Neural Networks (ANN); 1D Convolutional Neural Network (1DCNN); Synthetic Minority Oversampling Technique (SMOTE); NSL-KDD dataset; CIC-IDS2017; UNSW-NB15

I. Introduction

As computer networks continue to play an increasingly important role in modern life, ensuring cybersecurity has become a crucial area of research. One method to protect against potential threats is using an Intrusion Detection System (IDS). By continuously monitoring the state of both software and hardware on a network, IDS plays a critical role in maintaining cybersecurity [1]. Many Intrusion Detection Systems (IDSs) generate alerts, even in low-threat situations, straining cybersecurity experts and increasing the risk of actual intrusions going undetected. The literature includes extensive research on the subject, and different approaches to IDSs have been developed. However, current IDSs still face difficulty detecting unknown or novel attacks due to the constantly evolving network configurations. Therefore, it is critical to continue the research on IDSs to identify and detect such attacks [2].

Attackers can initiate attacks by distributing malicious files to devices connected to a network, resulting in potential damage to the device or theft of sensitive information [3]. Various technologies, such as firewalls, anti-virus software, email filters, and virtual private networks (VPN), protect networks from such threats. Intrusion Detection System (IDS) is another commonly used approach, where network traffic is monitored to detect any unusual activities. Network intrusion detection can be categorized into anomaly-based and signature-based [4]. Signature-based methods rely on predetermined criteria to identify attacks and categorize threats, while anomaly-based methods classify threats by studying regular traffic to develop profiles based on available data. Based on the given definitions, we can infer that Signature-based IDS are vulnerable to new undefined attacks as they solely depend on the current rules to generate their alerts. However, anomaly-based IDS are more effective when dealing with new threats as they do not base their alerts on existing rules [5] [6]. That does not mean that Signature-based IDS are less critical; it offers improved detection accuracy and reduced triggers of false alarms while identifying known threats. Although there is a high probability of false positives associated with anomaly detection IDS, the research community has widely accepted it due to its theoretical potential for identifying new threats [7].

A. Feature Engineering Challenges within NIDS

Creating a dependable and flexible NIDS that detects unknown future attacks presents two significant challenges. Selecting appropriate features from internet traffic collection for anomaly detection can present a noticeable challenge [7]. The features selected for one type of threat may not be practical for other attacks due to the ever-changing and evolving nature of attack scenarios. The literature has already proposed potential ways to address the challenge, the most common of which is the use of deep learning, a subset of machine learning techniques that uses hierarchical layers of data processing stages to learn features or representations and classify patterns [8]. Deep learning plays a crucial role in image categorization. It is also frequently utilized in natural language processing, speech recognition, audio, picture, video processing, graphical modeling, pattern identification, and language-related tasks. Improvements in learning algorithms can enhance IDS's ability to achieve a higher detection rate and lower false alarm rate. The use of deep learning-based techniques is anticipated to assist in overcoming the challenges of developing an effective NIDS [8].

Unlabeled network traffic data can be collected from various sources, and deep learning algorithms can be applied to generate a good feature representation of these datasets. These characteristics can then be utilized for supervised classification on a small, labeled traffic dataset consisting of regular and anomalous traffic records. A spoofed, private, and isolated network environment can be used to collect the traffic data for the labeled dataset [9].

B. Aim and Objective

The study aims to provide a reliable and efficient approach for identifying different cyber-attack types using sequence modeling and deep learning by proposing the following:

> Developing an effective intrusion detection model for detecting malicious traffic, leveraging SMOTE, BiLSTM, XGBoost, and 1DCNN/DNN for improved intrusion detection.

To achieve the aim of this study, the following objectives have been created:

- Identify all the model variations that need to be evaluated using SMOTE, BiLSTM, XGBoost, and 1DCNN/DNN.
- Evaluate the identified models against well-known IDS datasets from the literature (NSL-KDD [10], CIC-IDS2017 [11], and UNSW-NB15 [12]).

C. Problem Statement

The increasing use of interconnected computing systems has brought numerous benefits to daily activities and exposed us to vulnerabilities beyond human control. As a result, cybersecurity measures must be included in communication exchanges to ensure secure communication. However, with evolving security risks and threats, there is a constant need to improve security measures, including Intrusion Detection Systems (IDS). Despite the efforts of researchers to develop novel IDS systems, achieving high detection accuracy while reducing false alarm rates remains a challenge.

II. Research Structure

This paper is organized into seven sections, providing a comprehensive examination of the research topic:

- **Section 1: Introduction** - This initial section provides an overview of the paper's structure and objectives, setting the stage for the subsequent discussion and analysis.
- **Section 2: Background and Related Work** - This section explores key concepts and offers some background information to facilitate the understanding of the rest of the paper.
- **Section 3: Proposed Model (Methodology)** - This section explains the methodology of the proposed model and how it works.
- **Section 4: Results** - This section presents the results obtained by applying the newly proposed model to the experimental datasets.
- **Section 5: Evaluation** - This section compares different results using evaluation metrics such as precision and recall.
- **Section 6: Discussion** - This section provides a discussion of the results, offering insights and interpretations.
- **Section 7: Conclusion** - The final section concludes the paper, addressing the limitations encountered during the research and proposing potential directions for future work, aiming to expand upon and build on the current study.

III. Background and Related Work

The field of intrusion detection systems (IDS) has been rapidly evolving in recent years, with a focus on improving the accuracy and efficiency of detection methods. Various techniques have been proposed in this paper, including, IDS data preprocessing (data cleansing - removal of null and duplicate values, data balancing using SMOTE, and data standardization using standard scalar techniques), feature engineering (using XGBoost), and data classification (One-dimensional CNN, DNN Models, and Bidirectional Long-Short-Term Memory (BiDLSTM)). These methods have demonstrated promising results in detecting various network intrusions, including Denial of Service (DoS) attacks, intrusion attempts, and unauthorized access attempts. Furthermore, DNN Models and BiDLSTM have emerged as powerful tools for identifying complex patterns in network traffic data, making them valuable additions to the IDS toolbox. This background section provides an overview of these topics, discussing their underlying principles.

A. Feature Engineering

In machine learning, feature engineering plays a vital role as it converts raw data into a more suitable format, which allows for a better representation of the underlying issue and enhances model performance. XGBoost, a gradient-boosted decision tree algorithm, is renowned for its effectiveness and efficiency in feature engineering. Introduced by Mounika and Rao [13] and embraced by numerous researchers [14] [15], XGBoost is part of the Community for Distributed Machine Learning and excels in optimizing memory and hardware utilization in tree-boosting algorithms. Kasongo and Sun [16] employed XGBoost in their intrusion detection research, using the UNSW-NB15 dataset for model training and evaluation. They implemented a filter-based feature reduction technique alongside the XGBoost algorithm and used specific machine-learning approaches for binary and multi-class classification scenarios. The study showed that the XGBoost-driven feature selection method increased test accuracy for binary classification from 88.13% to 90.85%, demonstrating its effectiveness in boosting the precision of ML-based models. Dhaliwal et al. [17] conducted another study, developing a model to assess various network data attributes such as precision, accuracy, and confusion matrix. They employed the NSL-KDD dataset and XGBoost to achieve their objectives. The primary goal was to better understand data integrity and enhance data detection accuracy. The researchers recommended further investigations to facilitate the deployment of intrusion detection models.

XGBoost supports gradient, regularized, and stochastic gradient boosting techniques and permits the incorporation and adjustment of regularization parameters [18]. The algorithm optimizes memory usage, significantly reduces computation time, and manages missing values. Its sparse-awareness allows for a unified framework in tree structures, improving the trained model with new data [19]. XGBoost constructs a sequential decision tree, assigning a weight to each data value, which influences the likelihood of a decision tree selecting it for analysis [20]. Although challenges exist in network data preprocessing, data classification, and labeling, XGBoost tackles issues such as high-level preprocessing, DDoS attack mitigation, false alarm rates, and semi-supervised techniques necessary for a dependable IDS model [21]. Due to its capability to address most problems encountered in feature selection, XGBoost serves as a potent instrument for developing efficient intrusion detection systems [22].

B. Deep Learning Classification

In this section, we will review the relevant literature on various deep learning classification techniques that have been employed in the proposed model. The methods discussed include XGBoost, BiLSTM, DNN, One-dimensional CNN and Long Short-Term Memory (LSTM). These techniques have been applied in a wide range of intrusion detection systems and have demonstrated their effectiveness in handling complex and large-scale data. We will explore the key aspects of each method, as well as their applications in the field of intrusion detection. By examining the current state looking into the background of these techniques, we aim to provide a functional understanding, ultimately informing the development of a robust and efficient intrusion detection model.

1) One-dimensional CNN and Long Short-Term Memory (LSTM): Recurrent Neural Networks (RNNs) are a class of neural networks that can generate cycles through links between nodes, allowing the output of certain nodes to influence their future input and enabling temporal dynamic behavior. RNNs, derived from feed-forward neural networks, use their internal state (memory) to handle input sequences of varying length, making them ideal for AI tasks such as speech and handwriting recognition [23]. Long Short-Term Memory (LSTM) networks, a specialized form of RNNs, were developed to address the vanishing gradient problem commonly encountered during the training of conventional RNNs [24]. LSTMs, equipped with memory cells and gating structures, can effectively store information across extended sequences. In one study, the authors of [25] used an RNN with LSTM to recognize threats and normal patterns within IoT traffic. They trained their model using the UMSW-NB15 dataset and found that the LSTM RNN-based IDS was efficient and could detect threats with high accuracy. However, they suggested that further verification with a larger dataset was necessary. Agrawal and Duvey [26] aimed to develop an intrusion detection system using deep learning technology to identify infiltration and malicious activities that could disrupt the network environment. They proposed a hybrid DL-driven method that employs one-dimensional CNN and LSTM to detect attacks on the KDD99 dataset. The proposed model's performance was evaluated on binary and multiclass classifications using the KDD99 datasets. In another study, Qazi et al. [27] proposed a deep learning architecture for network intrusion detection based on a one-dimensional convolutional neural network. The study aimed to identify three types of network attacks, namely PortScan, DoS, and DDoS, using the CICIDS2017 dataset. The results showed an accuracy of 98.96%, but the authors suggested further analysis, such as using Principal Component Analysis (PCA), to investigate the reduction in input characteristics.

2) Bidirectional Long-Short-Term Memory (Bi-LSTM): Bidirectional Long Short-Term Memory (Bi-LSTM) is a variant of Recurrent Neural Network (RNN) that processes input sequences in both forward and backward directions using two hidden layers [28]. It predicts the order of elements based on prior and future context through two LSTMs operating simultaneously in opposite directions, generating a combined output that approximates the target signal. Imrana et al. [1] proposed a BiDLSTM-based intrusion detection system to address challenges faced by IDS. They used the NSL-KDD dataset for training and evaluating the model, a widely recognized dataset in IDS research. Experimental results demonstrated the effectiveness of the BiDLSTM approach, showing superior performance in accuracy, precision, recall, and F-score compared to conventional LSTM and other state-of-the-art models. The false positive rate was also significantly lower. The researchers studied integrated systems by combining cutting-edge feature selection approaches with conventional LSTM and BiDLSTM models. Traditional machine learning methods struggle to effectively identify complex and multidimensional intrusion data in real-world network application environments [29]. In contrast, deep learning-based Network Intrusion Detection Systems (NIDS) have gained interest due to their ability to handle large-scale data and extract essential traffic features. Research by Alghazzawi et al. [30]proposed a hybrid Deep Neural Network (DNN) model that outperformed traditional machine learning classifiers in network and host-level event monitoring. Sun et al. [31] developed the LuNet deep neural network architecture, using RNNs for temporal feature learning and CNNs for spatial characteristic extraction from traffic data. This approach reduced false positives and enhanced validation accuracy. Al-Omari et al. [32] developed an intrusion detection model by combining RNN and LSTM approaches, while Alwan et al. [33] created a Bi-LSTM network using the UNSW-NB15 dataset as a benchmark, achieving an accuracy above 95% . Yu et al. [34] proposed a session detection method using Bi-LSTM, leveraging advancements in Natural Language Processing (NLP) made by LSTMs to represent sessions in a specific language . They based their experiments on the ISCX IDS dataset, grouping packets by IP addresses to create sessions and encoding them using word embedding. They trained an LSTM model to predict anomalous sessions, utilizing a Bi-LSTM model to learn sequence properties in both directions.

In conclusion, deep learning techniques like Bi-LSTM, hybrid models incorporating CNNs and LSTMs, and LSTM -based approaches have shown considerable promise in intrusion detection systems. These methods outperform conventional machine learning approaches in terms of accuracy, recall, and F-score [35]. They provide effective solutions for intrusion detection in real-world network application environments due to their ability to process complex, large-scale data and extract fundamental features from traffic data. Their development and implementation have improved network intrusion detection systems' overall performance and strengthened network security infrastructure.

3) Deep Neural Network (DNN): Deep Neural Networks (DNNs) are a type of artificial neural network (ANN) that consist of multiple layers between their input and output layers. They are composed of biases, synapses, neurons, weights, and functions, which work together to mimic the human brain's processing capabilities. DNNs can be trained like any other machine learning algorithm, making them suitable for various artificial intelligence tasks, such as image and speech recognition [36]. In one study, Devan and Khare [37] proposed an XGBoost-DNN model for network intrusion detection. The XGBoost algorithm was employed for feature selection, while DNNs were used to classify network intrusions. During DNN training, the Adam optimizer was utilized to optimize the learning rate, and the Softmax classifier was employed to categorize network intrusions. To validate their proposed model, cross-validation was performed, and it was compared to other shallow machine learning techniques such as SVM, logistic regression, and naive Bayes. Classification assessment metrics, including accuracy, precision, recall, and F1-score, were computed and contrasted with the existing shallow approaches. In another study, Kumar et al. [38] investigated DNNs to develop a flexible and efficient intrusion detection system (IDS) for identifying and categorizing unanticipated cyber-attacks. The study thoroughly analyzes DNN and other traditional machine learning classifier studies on several freely accessible benchmark malware datasets. However, the study's limitation is that the complex DNN structures have a high computational cost, so they were not trained using benchmark IDS datasets. Other research includes Tang et al. [39], who devised a deep learning-based approach for intrusion detection in software-defined networking (SDN) architecture. Potluri et al. [40] adopted a deep neural technique to manage large volumes of network data for deep-category identification. Kang et al. [41] developed a potential intrusion detection system for vehicular networks using deep neural networks. These studies, among others, demonstrate the potential of DNNs in intrusion detection systems and their ability to effectively classify various types of network intrusions.

IV. Proposed Model (Methodology)

in this paper, we proposed BILSTM and neural network models for classification and XGBoost for feature engineering. Moreover, in order to make the evaluation scientifically accurate, we will be using the datasets from the literature (NSL-KDD [10], CIC-IDS2017 [11], and UNSW-NB15 [12])). Unfortunately, these datasets suffer from significant class imbalance problems between the different categories. Prior researchers have not always addressed this issue, which presents a high risk of failing to detect the minority class target value. While the accuracy of these studies may be high due to the low number of candidates for some target classes, it is essential to note that accuracy alone can be misleading. To overcome this problem, we will incorporate oversampling techniques into the proposed algorithms to improve the detection of target classes in the imbalanced data. In addition to accuracy, we will also focus on precision and recall as performance metrics in this research. A. Modeling Process The proposed study will employ two main modeling processes to train the chosen dataset for intrusion detection. These two processes are combined:

- Xgboost + 1DCNN, BiLSTM
- Xgboost + DNN, BiLSTM.

Furthermore, the hyperparameter tuning process will be applied to both models to ensure the most accurate testing. B. Training and Testing This section will provide an overview of the four models utilized in this study. These models were created to evaluate and compare the performance of the proposed framework under varying circumstances. The defined models are as follows:

- Model 1: Using a standard dataset and applying [1D CNN + BiLSTM]
- Model 2: Using a standard dataset and applying [ANN + BiLSTM]
- Model 3: Using a balanced dataset (created with SMOTE) and applying [1D CNN + BiLSTM].
- Model 4: Using a balanced dataset (created with SMOTE) and applying [ANN + BiLSTM].

The split model framework is illustrated in Fig. 1.

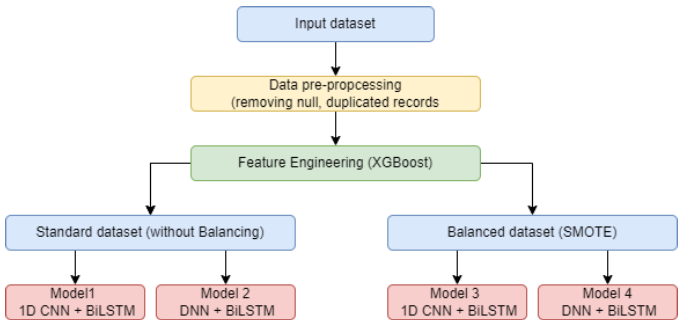
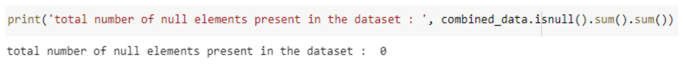


Fig. 1. Proposed model framework.

For this study, three datasets have been utilized, namely NSL-KDD, CIC-IDS2017, and UNSW-NB15. The dataset underwent several preprocessing steps, including removing null elements and duplicate rows, changing data types to use lower memory, and performing label encoding. The presence of any missing values in the selected dataset was analyzed, and the results are presented in Fig. 2.



```
print('total number of null elements present in the dataset : ', combined_data.isnull().sum().sum())

total number of null elements present in the dataset :  0
```

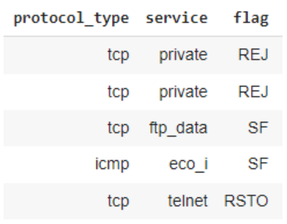
Fig. 2. Null elements.

A. Converting Data to Numerical Values

To utilize the categorical features in deep learning algorithms, the data needs to be converted into numerical values. One way to achieve this is through one hot encoding, a technique that represents categorical variables as numerical values in a machine learning model. One hot encoding offers several advantages, such as enabling categorical variables in models requiring numerical input and improving model performance by providing more information about the categorical variable. For this study, the categorical variables identified in the features selection stage have been used, and one hot encoding is performed on these variables. An example of the resulting output is shown below using the NSL-KDD dataset after performing the one hot encoding function. The pre-processed data can now be used for training the deep learning models.

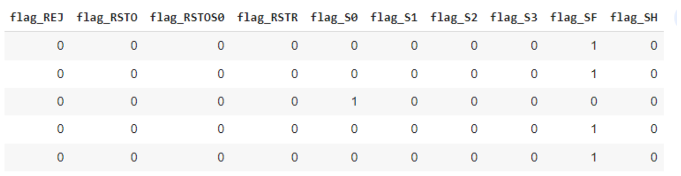
In their study, Mohammed [42] designed a fully connected network structure consisting of input, hidden, and output layers. To define these layers, the author utilized the Dense class, which allows them to specify the number of nodes or neurons in the layer as the first argument and the activation function to use the activation argument. In their architecture, the ReLU activation function was applied to the first two layers to introduce non-linearity. In contrast, the Sigmoid activation function was used in the output layer to ensure the network output is confined between 0 and 1. Moreover, the author implemented a dropout regularization technique to mitigate overfitting during training.

In the proposed model, one-hot encoding is crucial since both BiLSTM and XGBoost have been shown to work better with numerical values rather than categorical values [43], [44]. By applying one-hot encoding to categorical variables such as Protocol type, service, and Flag, a more appropriate input format is achieved for our machine learning algorithms. This approach ensures enhanced performance and more accurate results. One-hot encoding is applied to variables such as Protocol type, service, and Flag [44], as shown in Fig. 3 (before encoding) and Fig. 4 (after encoding).



protocol_type	service	flag
tcp	private	REJ
tcp	private	REJ
tcp	ftp_data	SF
icmp	eco_i	SF
tcp	telnet	RSTO

Fig. 3. Data before One Hot Encoding.



flag_REJ	flag_RSTO	flag_RSTOS0	flag_RSTR	flag_S0	flag_S1	flag_S2	flag_S3	flag_SF	flag_SH
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0

Fig. 4. Data after One Hot Encoding.

B. Data Standardization

To bring the numeric columns in the dataset to a standard scale without distorting their differences in ranges of values, the data needs to be standardized. Standardization is a process that rescales the distribution of values so that the mean of the observed values becomes 0 and the standard deviation becomes 1. To achieve this, the standard scalar method was used for standardization. A value is standardized as follows:

$$y = \frac{x - \text{mean}}{\text{standard\_deviation}} \tag{1}$$

Where the mean is calculated as:

$$\text{mean} = \frac{\sum x}{\text{count}(x)} \tag{2}$$

And the standard deviation is calculated as:

$$\text{standard\_deviation} = \sqrt{\frac{\sum (x - \text{mean})^2}{\text{count}(x)}} \tag{3}$$

C. Oversampling Technique

The data preprocessing and standardization steps are applied uniformly to all three datasets. However, since the datasets are highly imbalanced, it is necessary to use data-balancing techniques to improve performance. Imbalanced data refers to datasets in which the target class has an uneven distribution of observations, with one class label having a significantly higher number of observations than the other. To address this issue, we will use an oversampling technique called SMOTE (Synthetic Minority Oversampling Technique) to balance the data. The SMOTE technique uses a hybrid method called SMOTE+TOMEK [45] to remove overlapping data points for each class dispersed in the sample space. Once SMOTE has finished its oversampling, the class clusters may encroach on each other's space, causing the classifier model to overfit. The pairs of samples from opposing classes closely related are called Tomek linkages. Most of the class observations from these linkages are eliminated to improve class separation near decision boundaries. The links are applied to oversampled minority class samples from SMOTE to obtain better class clusters. Therefore, both class observations from the Tomek linkages are typically deleted, rather than just the observations from the majority class. In their study, Mohammed [42] designed a fully connected network structure consisting of input, hidden, and output layers. To define these layers, the author utilized the Dense class, which allows them to specify the number of nodes or neurons in the layer as the first argument and the activation function to use the activation argument. In their architecture, the ReLU activation function was applied to the first two layers to introduce non-linearity. In contrast, the Sigmoid activation function was used in the output layer to ensure the network output is confined between 0 and 1. Moreover, the author implemented a dropout regularization technique to mitigate overfitting during training. The hyper parameters for the four models were tuned based on their performance on selected performance metrics, which include accuracy, precision, recall, and F1 score. Tables 1 and 2 list the hyper parameters chosen for each model.

D. Feature Engineering and DL Classifications

The next step involves performing feature engineering to develop a deep learning model. In this study, XGBoost, an optimized gradient boosting algorithm, will be used for feature engineering as it performs better on numerical datasets for both Model 3 and Model 4. The top 20 essential features from the used dataset were identified using the XGBoost classifier algorithm to perform feature extraction. This was achieved by tuning hyperparameters such as the number of estimators, leaves, and the maximum depth of trees in the algorithm.

The literature describes that sequence modeling algorithms have shown promising results in handling numerical data in recent years. Various algorithms such as RNN, LSTM, MLP, DNN, and BISLTM have been used as classification models for Intrusion detection problems. For instance, an unsupervised deep learning technique called Autoencoder takes a vector as input and produces the same dimension vector as output. The primary process involves taking input data, reducing its dimensionality, reconstructing it into a lower dimension, and then attempting to reconstruct it back to its original dimension. During this process, the noise in the data is removed, and only the essential features are retained as input data shape.

The hyper-parameters tuned for these four models are outlined in Tables I and II. These were selected based on the performance metrics chosen for evaluation. The considered performance metrics include Accuracy, Precision, Recall, and F1 Score.

TABLE I. HYPER PARAMETER USED FOR MODELS 1 AND 3

Hyper parameters	**1D CNN (Model 1 and 3)**
Epochs	150
Optimizer	Adam
Batch Size	128
# of Layers (1D)	2
Maxpooling Layers	3
Batch Normalization	3
BiLSTM Layers	2

TABLE II. HYPER-PARAMETER USED FOR MODELS 2 AND 4

Hyper parameters	DNN (Model 2 and 4)
Epochs	150
Optimizer	Adam
Batch Size	256
# of Layers (NN)	2
Maxpooling Layers	1
Batch Normalization	1
BiLSTM Layers	2

The dataset has been divided into training, and testing data using the Train Test Split function from Sci-Kit Learn to prepare for training the model. For the model an optimal ratio of 80% has been selected for training data and 20% for testing data.

V. RESULTS

this section present the results of the proposed module aimed at enhancing the performance of an Intrusion Detection System (IDS). After running the module, we have reached some critical conclusions regarding the efficacy of the proposed approach. Specifically, we have utilized four different models to assess the system's performance and determine which model performs better based on various metrics. In this section, we will discuss the results of the evaluations and highlight the strengths and weaknesses of each model.

Various performance metrics determine which of the four models performs better. The metrics used to evaluate the models include precision and recall. Precision measures the percentage of correctly predicted positive outcomes out of all the predicted positive outcomes. It can be expressed as the ratio of true positives (TP) to the sum of true and false positives (TP + FP). Mathematically, precision can be defined as TP / (TP + FP). Precision primarily focuses on the positive class rather than the negative class. Recall, also known as sensitivity, measures the percentage of correctly predicted positive outcomes out of all the actual positive outcomes. It can be expressed as the ratio of true positives (TP) to the sum of true positives and false negatives (TP + FN). Mathematically, recall can be defined as TP / (TP + FN). The definitions of precision and recall are:

$$Precision = \frac{TP}{(TP + FP)}$$

Another metric is The F1-score, which is a weighted harmonic mean of precision and recall and measures the overall performance of a classifier model. The highest possible F1 score is 1.0, while the lowest is 0.0. As the F1-score is based on precision and recall, it is always lower than accuracy measures, which incorporate only one of these factors. The weighted average of F1-scores should be used to compare classifier models rather than a global accuracy measure.

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

In this paper, we will explain how we implemented the four modules on the NSL-KDD datasets. The performance metrics described earlier were used to evaluate the three datasets using the four models previously defined. All four models were implemented for each dataset, and the results were compared to determine which model performed best. Subsequently, we will present a summarized table of the results obtained from the other two datasets without delving into the implementation details since we followed the same approach as the first dataset.

1) Model 1: imbalanced dataset: 1D CNN + BILSTM: In the case of the NDL-KDD dataset, model 1 achieved better accuracy, which is noteworthy given the imbalance in the data. Specifically, the training accuracy was 98.3%, and the testing accuracy was 97.9%. The accuracy plot in Fig. 5 provides additional insight into the model's behavior and can be used to identify any inconsistencies that may have occurred during training.

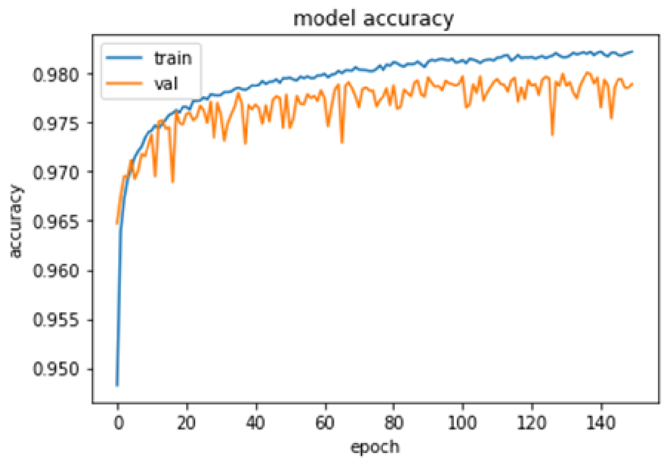


Fig. 5. Model 1 accuracy-epoch.

Fig. 5 demonstrates a typical trend, with a gradual decrease in loss observed during the training phase and some fluctuations across all epochs for the testing dataset. These fluctuations are also visible in the accuracy plot, indicating that the model tries to learn from the loss experienced in the previous epoch.

TABLE III. Model 1: Confusion Matrix

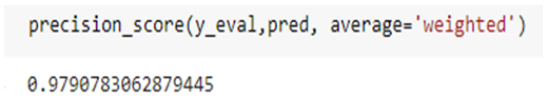
Attacks	Normal	Dos	Probe	R2L	U2R
Normal	10609	49	1	3	0
Dos	52	15208	97	167	6
Probe	4	64	2683	10	0
R2L	0	158	1	569	1
U2R	0	9	0	3	10

The confusion matrix in Table III indicates that the model performed exceptionally well for more class variables. In contrast, the last two classes, R2L and U2R, had very few correctly classified target variables. This could be due to the imbalance in the dataset, which contains only a few classes for these categories. Table 4 shows each class's false positive rate, recall, and F1 score to understand better which class had the lowest correct classification rate.

TABLE IV. Model 1: Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Normal	0.99	1.00	0.99	10662
Dos	0.99	0.98	0.98	15530
Probe	0.96	0.97	0.97	2671
R2L	0.76	0.78	0.77	729
U2R	0.59	0.45	0.51	22

The output from the classification report in Table IV indicates that the U2R class has the lowest precision and recall values, with a precision value of 59%, recall of 45%, and F1 score of 51%. This is likely due to the imbalance in the dataset. However, the precision and recall values for the overall model are high, at 97.9% and 97.8%, respectively.

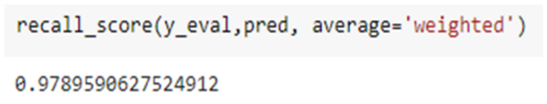


```
precision_score(y_eval,pred, average='weighted')

0.9790783062879445
```

Fig. 6. Model 1 precision score.

Although the total recall and precision values in Fig. 6 and 7 provide an overview of the model's performance, they do not fully capture the degree of imbalance in the data when classifying the testing dataset. To gain a more nuanced understanding of the model's performance, it is necessary to examine each class's precision and recall values separately.



```
recall_score(y_eval,pred, average='weighted')

0.9789590627524912
```

Fig. 7. Model 1 recall score.

2) Model 2: Imbalanced dataset: ANN + BILSTM: accuracy of nearly 99%. Fig. 8 show that accuracy has consistently increased with each epoch. However, some epochs have lower validation accuracy, possibly because the model encountered new images that it had not seen in previous epochs. The same trend is observed in the loss plot.

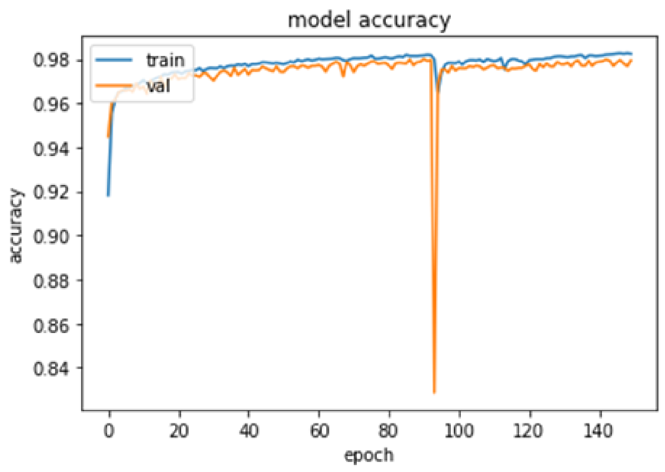


Fig. 8. Model 1 accuracy-epoch.

TABLE V. Model 2: Confusion Matrix

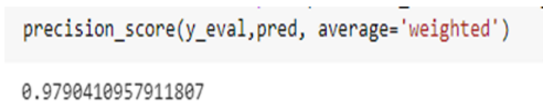
Attacks	Normal	Dos	Probe	R2L	U2R
Normal	10583	36	8	0	0
Dos	64	15136	88	135	3
Probe	12	52	2805	10	0
R2L	2	182	2	557	1
U2R	1	12	0	4	11

The confusion matrix in Table V reveals that the normal class outperforms all other target classes. The U2R and R2L classes demonstrate the weakest performance in accurately classifying the test dataset, which may also be attributed to class imbalance in the dataset.

TABLE VI. Model 2: Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Normal	0.99	1.00	0.99	10627
Dos	0.98	0.98	0.98	15426
Probe	0.97	0.97	0.97	2879
R2L	0.79	0.75	0.77	744
U2R	0.73	0.39	0.51	28

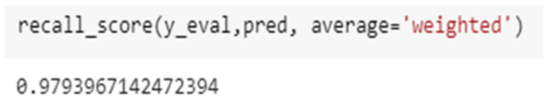
Examining the classification report in Table VI, it can be concluded that precision is improved for all classes when compared to Model 1. However, the recall for the U2R class in Model 2 is the lowest among all classes. This indicates that the ratio of correctly predicted positive elements to the total positive elements is low. As a result, the model struggles to predict the U2R class accurately and often misclassifies it as another class.



```
precision_score(y_eval,pred, average='weighted')

0.9790410957911807
```

Fig. 9. Model 2 precision score.



```
recall_score(y_eval,pred, average='weighted')

0.9793967142472394
```

Fig. 10. Model 2 recall score.

The precision and recall values for Model 2 shown in Fig. 9 and 10 are 97.9% and 97.9%, respectively. Although the overall performance of the model is quite remarkable, it is crucial to evaluate the model's performance for individual classes, particularly those with fewer samples in the dataset. The U2R and R2L classes, for example, exhibit lower performance metrics, which could be ascribed to the class imbalance in the dataset. The total recall and precision do not fully reveal the extent of data imbalance when classifying the testing dataset. It is only by examining each class individually that one can fully understand the model's performance.

3) Model 3: Balanced dataset with SMOTE: 1D CNN + BILSTM: For models 3 and 4, the dataset will be prepossessed using SMOTE to oversample the least represented target classes and increase their count to match the highest represented target classes. This oversampling process balances each category in the dataset and eliminates any class imbalance. The resulting model achieved an accuracy of 95%, indicating that it did not perform better than the first two models that used the imbalanced dataset. The accuracy plot for this model are shown in Fig. 11.

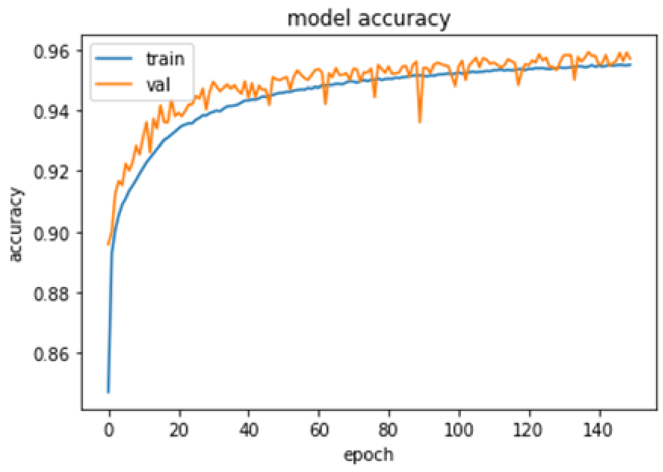


Fig. 11. Model 3 accuracy-epoch.

The confusion matrix presented in Table VII reveals that almost all the target classes have performed better than in the previous two models. The reason for the lower accuracy is that the number of incorrectly classified instances is higher, as the dataset size increased due to the oversampling of the data. This increase in dataset size led to more opportunities for misclassification, resulting in a lower overall accuracy.

TABLE VII. Model 3: Confusion Matrix

Attacks	Normal	Dos	Probe	R2L	U2R
Normal	15377	18	37	6	0
Dos	28	14255	115	814	180
Probe	1	32	15308	83	7
R2L	0	278	10	14972	144
U2R	0	206	0	1361	13870

TABLE VIII. Model 3: Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Normal	1.00	1.00	1.00	15438
Dos	0.96	0.93	0.94	15392
Probe	0.99	0.99	0.99	15431
R2L	0.87	0.97	0.92	15404
U2R	0.98	0.90	0.94	15437

The classification report in given in Table VIII. The precision and recall of Model 3 for all the target classes were better than those of the previous two models. When comparing these metrics for Models 1 and 2, the precision of the U2R class for Model 3 was 98%, while for the others, it was 73% and 59%. For the U2R class in recall, Model 3 had 90%, whereas Model 1 had 45% and Model 2 had 39%. This demonstrated that Model 3 performed significantly better than the other models and had a higher prediction rate for less frequent target classes. This performance improvement was solely due to the balancing of the data, which allowed the model to learn more about the less frequent target variable classes. The classification report was provided as well. The average weighted precision and recall values for Model 3 were 95.9% and 95.6%, respectively.

4) Model 4: Balanced dataset with SMOTE: ANN + BILSTM: The accuracy plot shown in Fig. 12 displays a significant amount of fluctuation in the validation dataset, which can be attributed to the fact that the model may have encountered a different set of data during testing on that epoch.

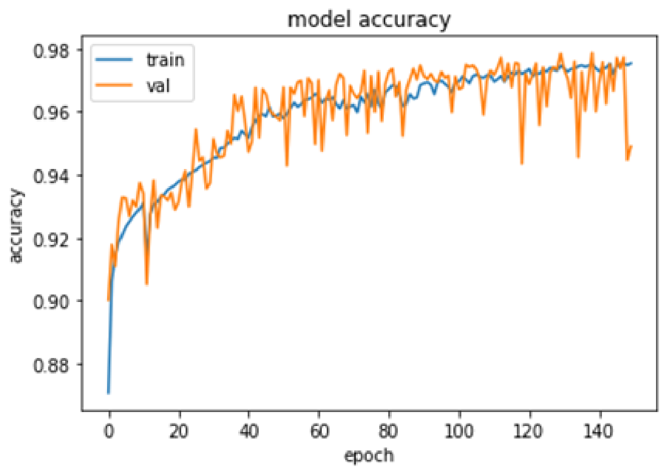


Fig. 12. Model 4 Accuracy-Epoch.

As seen in the confusion matrix in Table IX, the values appear similar to those of Model 3, with the ratio of correctly predicted classes remaining consistent between the two models. To gain a deeper understanding of the model's performance, it is necessary to examine the precision, recall, and F1 score mentioned in Table X. When compared to Model 3, Model 4 exhibits weaker performance, as the precision for the U2R class is lower, and the recall for the R2L class is significantly lower than the previous model. The precision and recall values obtained for this model are 95.3% and 94.8%, respectively.

TABLE IX. MODEL 4: CONFUSION MATRIX

Attacks	Normal	Dos	Probe	R2L	U2R
Normal	15336	17	81	2	2
Dos	24	14374	128	555	311
Probe	3	37	15320	28	43
R2L	0	164	13	12766	246
U2R	0	20	0	44	15373

TABLE X. MODEL 4: CLASSIFICATION REPORT

Attacks\metrics	Precision	Recall	F1-Score	Support
Normal	1.00	0.99	1.00	15438
Dos	0.98	0.93	0.96	15392
Probe	0.99	0.99	0.99	15431
R2L	0.95	0.83	0.89	15404
U2R	0.85	1.00	0.91	15437

VI. EVALUATION

A. Evaluation of the Models Against the NSL KDD Dataset

In the NSL-KDD dataset, when considering accuracy, Model 1 and Model 2 perform better than Model 3 and Model 4, achieving a high accuracy of approximately 99%. When taking average precision and recall into account, Models 1 and 2 still outperform Models 3 and 4. However, Models 1 and 2 predict poorly for two attack classes, namely R2L and U2R, which can be clearly observed in both the confusion matrix and classification report. The number of incorrect predictions is higher in Models 1 and 2 compared to Models 3 and 4.

In Models 3 and 4, the precision and recall for U2R and R2L are above 85%, and for other attack classes, they are above 93%. Model 3 demonstrates better precision and recall rate performance compared to Model 4, as Model 3 combines Bi-LSTM and 1D-CNN with balanced data, whereas Models 1 and 2 were trained on imbalanced data.

Additionally, Model 3 excels at correctly detecting most of the classes in the dataset. This model incorporates SMOTE for balancing data, XGBoost for feature engineering, and a combination of 1D-CNN and Bi-LSTM as the classification algorithm. Although the training and testing accuracy of Model 3 is lower than that of Models 1 and 2, this is due to the significantly smaller number of elements in the target classes with the least representation in Models 1 and 2 (U2R - 52, R2L - 995). Consequently, even if these classes were entirely misclassified, only 1,047 out of 125,973 would be incorrectly classified. However, if these classes were misclassified in real-time, they would be detected as normal or other attacks, undermining the goal of detecting intrusions.

To address this issue, precision and recall were calculated for each target class in the dataset. This approach allows to evaluate how well the model has predicted each target class individually, rather than. Models 1, 2, and 4 exhibit lower precision and recall for the U2R and R2L classes. In contrast, Model 3 demonstrates a precision of 98% for U2R and 87% for R2L, along with a recall of 90% for U2R and 97% for R2L. Table XI provide a summery of NSL-KDD dataset performance. While Table XII and XIII provides a comparison of the performance of the best model against similar research in the literature.

TABLE XI. SUMMARY OF THE NSL – KDD DATASET PERFORMANCE

Models	Precision	Recall
1	97.9	97.8
2	97.9	97.9
3	95.9	96.9
4	95.3	94.8

TABLE XII. PRECISION PERFORMANCE METRIC - COMPARISON OF OTHER PAPERS

Target class	Best model	Chongzhen	Mohammed	Yakubu Imrana
Normal	100%	71%	61%	75%
DoS	96%	96%	94%	97%
Probe	99%	86%	97%	84%
R2L	87%	81%	99%	98%
U2R	98%	73%	100%	77%

TABLE XIII. RECALL PERFORMANCE METRIC - COMPARISON OF OTHER PAPERS

Target class	Best model	Chongzhen	Mohammed	Yakubu Imrana
Normal	100%	71%	61%	75%
DoS	96%	96%	94%	97%
Probe	99%	86%	97%	84%
R2L	87%	81%	99%	98%
U2R	98%	73%	100%	77%

B. Evaluation of the models against the CIC-IDS2017 dataset

For the CIC-IDS2017 dataset, Model 1 exhibits better accuracy, considering the data imbalance, with a training accuracy of 97.4% and a testing accuracy of 97.3%. The confusion matrix and classification report can be found in Tables XIV and XV.

TABLE XIV. MODEL 1: CIC-IDS2017'S - CONFUSION MATRIX

Attacks	Benign	Port Scan	DDos	WEbattack	Bot
Benign	82423	35	1599	1	3
PortScan	64	12308	0	0	0
DDos	818	0	11340	0	0
Web Attack	120	0	0	64	0
Bot	194	0	0	0	13

TABLE XV. MODEL 1: CIC-IDS2017'S - CLASSIFICATION REPORT

Attacks\metrics	Precision	Recall	F1-Score	Support
Benign	0.99	0.98	0.98	84061
PortScan	1.00	0.99	1.00	12376
DDos	0.88	0.93	0.90	12158
Web Attack	0.81	0.06	0.12	207
Bot	0.98	0.35	0.52	185

Based on the confusion matrix in Table XIV, it is evident that the model performs exceptionally well for the classes with a higher number of instances. The last two classes, DDOS and Web Attack, have very few correctly classified target instances. This can be attributed to the imbalance in the dataset, as there are very few instances for those categories. To understand which class has performed the poorest in terms of correct classification, the false positive rate, recall, and F1 score of the model can be examined in Table XV.

TABLE XVI. Model 2: CIC-IDS2017's - Confusion Matrix

Attacks	Benign	Web Attack	PortScan	DDos	Bot
Benign	83824	1	36	157	43
Web attack	127	58	0	0	0
PortScan	18	0	12358	0	0
DDos	114	0	9	12033	2
Bot	145	0	0	0	62

TABLE XVII. Model 2: CIC-IDS2017's - Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Benign	1.00	1.00	1.00	84061
Web attack	0.98	0.31	0.48	185
PortScan	1.00	1.00	1.00	12376
DDos	0.99	0.99	0.99	12158
Bot	0.58	0.30	0.39	207

From the confusion matrix XVI, it can be observed that model 2 performs exceptionally well for the classes with a higher number of instances. Similar to Model 1, the last two classes, DDOS and Web Attack, have very few correctly classified target instances. This can be attributed to the imbalance in the dataset, as there are very few instances for those categories. The false positive rate, recall, and F1 score of the model can be seen in Table XVII to understand which class has performed the poorest in terms of correct classification.

For Models 3 and 4, a new dataset will be created using SMOTE, which over samples the least represented target classes and matches their count to that of the most represented target classes.

TABLE XVIII. Model 3: CIC-IDS2017's - Confusion Matrix

Attacks	Benign	Web Attack	PortScan	DDos	Bot
Benign	23314	553	16	997	312
Web attack	24	25234	0	0	0
PortScan	395	0	24861	0	0
DDos	1	0	0	25176	21
Bot	2	0	0	6	25251

TABLE XIX. Model 3: CIC-IDS2017's - Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Benign	0.98	0.93	0.95	25192
Web attack	0.98	1.00	0.99	25258
Port Scan	1.00	0.98	0.99	25256
DDoS	0.96	1.00	0.98	25198
Bot	0.99	1.00	0.99	25259

The confusion matrix in Table XVIII demonstrates that almost all the target classes in model 3 have performed better than in the previous two models. The lower accuracy can be attributed to the increased number of mis-classifications due to the over-sampled data, which increased the dataset size.

The precision and recall of this model for all the target classes are better than the previous two models. For instance, the precision of the Web Attack class for Model 3 is 99%, while the other models have 81% and 58%. In terms of recall for the Web Attack class, Model 3 has 100%, whereas Model 1 has 6% and Model 2 has 30%. This indicates that Model 3 has performed significantly better than the other models and has a higher prediction rate for less frequent target classes. This performance improvement is primarily due to the balancing of the data, allowing the model to learn more about the underrepresented target classes. The classification report is provided in Table XIX. The weighted precision and recall values for this model are 98.1% and 98.1%.

TABLE XX. Model 4: CIC-IDS2017's - Confusion Matrix

Attacks	Benign	Web Attack	PortScan	DDos	Bot
Benign	24550	470	2	70	100
Web attack	0	25258	0	0	0
PortScan	23	0	25233	0	0
DDos	21	0	0	25173	4
Bot	16	0	0	6	25237

TABLE XXI. Model 4: CIC-IDS2017's - Classification Report

Attacks\metrics	Precision	Recall	F1-Score	Support
Benign	1.00	0.97	0.99	25192
Web attack	0.98	1.00	0.99	25258
PortScan	1.00	1.00	1.00	25256
DDos	1.00	1.00	1.00	25198
Bot	1.00	1.00	1.00	25259

In the CIC-IDS2017 dataset, and based on Tables XX and XXI. When evaluating average precision and recall for all models, each one performs well. However, when examining individual precision and recall, Models 1 and 2 under-perform in two classes, namely Web Attack and Bot, which might be due to the imbalance in the dataset. The number of incorrect predictions is higher in Models 1 and 2 compared to Models 3 and 4 for the attack classes. After balancing the data, Model 4 outperforms all other models we have built, with an accuracy of 99% and individual precision and recall for the two classes that performed poorly in Models 1 and 2. For these classes, Model 4 achieves a recall of 100% and precision of 99%.

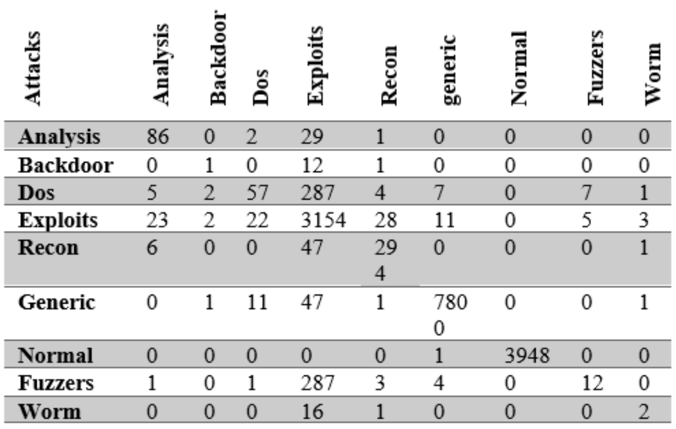
TABLE XXII. Summary of the CIC-IDS2017 Dataset Performance

Models	Precision	Recall
1	97.4	97.3
2	99.3	99.3
3	98.2	98.2
4	99.4	99.4

Table XXII presents the precision and recall results of four different models for the CIC-IDS2017 dataset. Model 1 achieves a precision of 97.4% and a recall of 97.3%, indicating good performance, but not the best among the tested models. Model 2 performs exceptionally well, achieving a precision and recall of 99.3% each, which suggests a high degree of accuracy in identifying true positives and avoiding false negatives. Model 3 shows slightly lower results, with a precision and recall

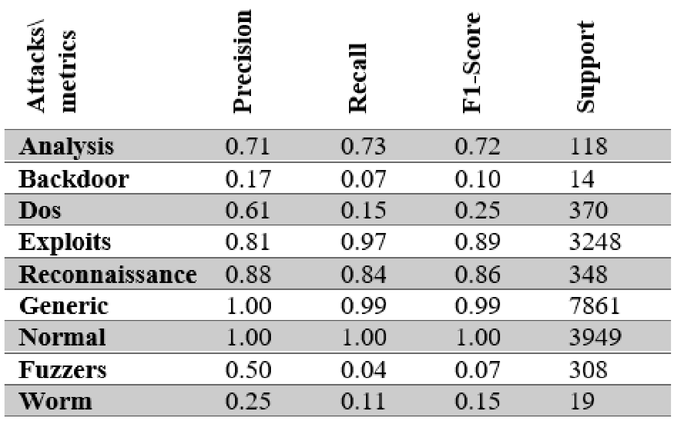
C. Evaluation of the Models Against the UNSW-NB15 Dataset

For the UNSW-NB15 dataset, Model 1 exhibits better accuracy considering the data imbalance, with a training accuracy of 94.6% and a testing accuracy of 94.5%. The confusion matrix and classification report can be found in Fig. 13 and 14



Attacks	Analysis	Backdoor	Dos	Exploits	Recon	generic	Normal	Fuzzers	Worm
Analysis	86	0	2	29	1	0	0	0	0
Backdoor	0	1	0	12	1	0	0	0	0
Dos	5	2	57	287	4	7	0	7	1
Exploits	23	2	22	3154	28	11	0	5	3
Recon	6	0	0	47	294	0	0	0	1
Generic	0	1	11	47	1	7800	0	0	1
Normal	0	0	0	0	0	1	3948	0	0
Fuzzers	1	0	1	287	3	4	0	12	0
Worm	0	0	0	16	1	0	0	0	2

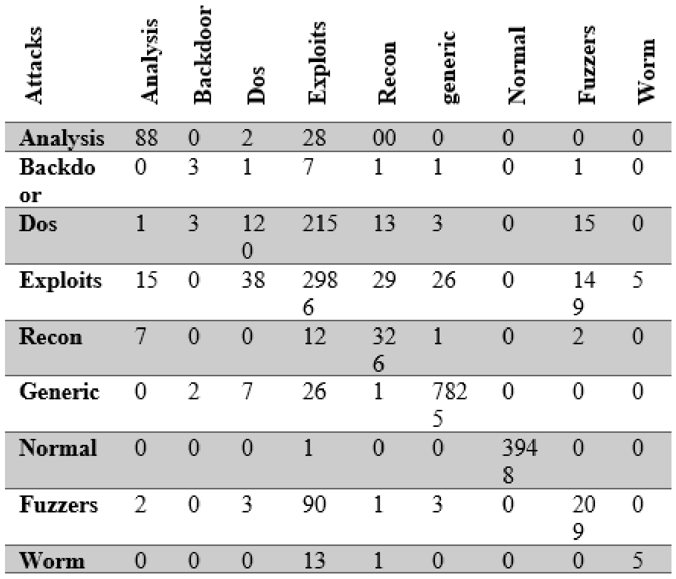
Fig. 13. Model 1 -UNSW-NB15 - confusion matrix.



Attacks\ metrics	Precision	Recall	F1-Score	Support
Analysis	0.71	0.73	0.72	118
Backdoor	0.17	0.07	0.10	14
Dos	0.61	0.15	0.25	370
Exploits	0.81	0.97	0.89	3248
Reconnaissance	0.88	0.84	0.86	348
Generic	1.00	0.99	0.99	7861
Normal	1.00	1.00	1.00	3949
Fuzzers	0.50	0.04	0.07	308
Worm	0.25	0.11	0.15	19

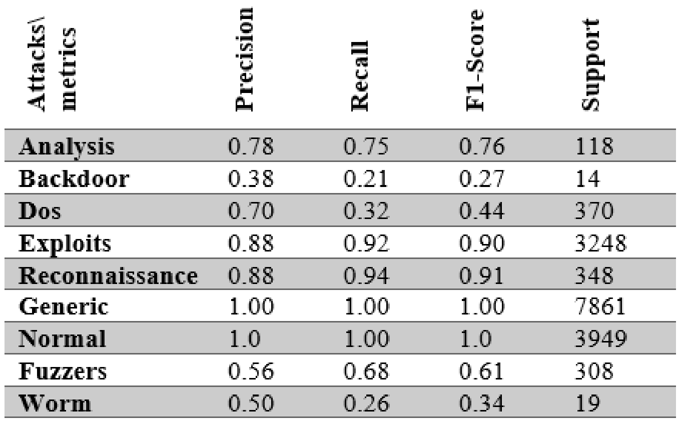
Fig. 14. Model 1 -UNSW-NB15 - classification report.

Fig. 15 and 16 presents the confusion matrix and the classification report for Model 2 on the UNSW-NB15 dataset, showing precision, recall, F1-score, and support for each attack category. The best precision and recall scores of 1.00 are seen in the Generic and Normal classes, indicating perfect classification for these categories. Exploits and Reconnaissance classes also perform well, with precision scores of 0.88 and 0.88, and recall scores of 0.92 and 0.94, respectively. These scores suggest accurate and comprehensive classification in these categories. However, the Backdoor and Worm classes exhibit lower performance, with Backdoor having a precision of 0.38 and a recall of 0.21, and Worm having a precision of 0.50 and a recall of 0.26. The F1-scores for these classes are also relatively low at 0.27 for Backdoor and 0.34 for Worm, indicating weaker classification performance for these categories.



Attacks	Analysis	Backdoor	Dos	Exploits	Recon	generic	Normal	Fuzzers	Worm
Analysis	88	0	2	28	00	0	0	0	0
Backdoor	0	3	1	7	1	1	0	1	0
Dos	1	3	120	215	13	3	0	15	0
Exploits	15	0	38	2986	29	26	0	149	5
Recon	7	0	0	12	326	1	0	2	0
Generic	0	2	7	26	1	7825	0	0	0
Normal	0	0	0	1	0	0	3948	0	0
Fuzzers	2	0	3	90	1	3	0	209	0
Worm	0	0	0	13	1	0	0	0	5

Fig. 15. Model 2 -UNSW-NB15 - confusion matrix.

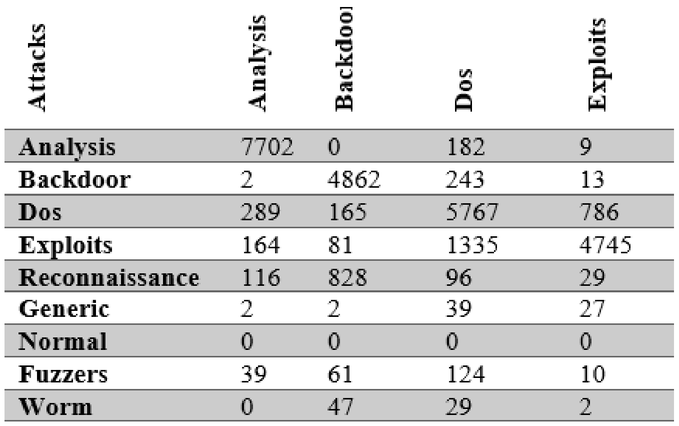


Attacks\ metrics	Precision	Recall	F1-Score	Support
Analysis	0.78	0.75	0.76	118
Backdoor	0.38	0.21	0.27	14
Dos	0.70	0.32	0.44	370
Exploits	0.88	0.92	0.90	3248
Reconnaissance	0.88	0.94	0.91	348
Generic	1.00	1.00	1.00	7861
Normal	1.0	1.00	1.0	3949
Fuzzers	0.56	0.68	0.61	308
Worm	0.50	0.26	0.34	19

Fig. 16. Model 2 -UNSW-NB15 - classification report.

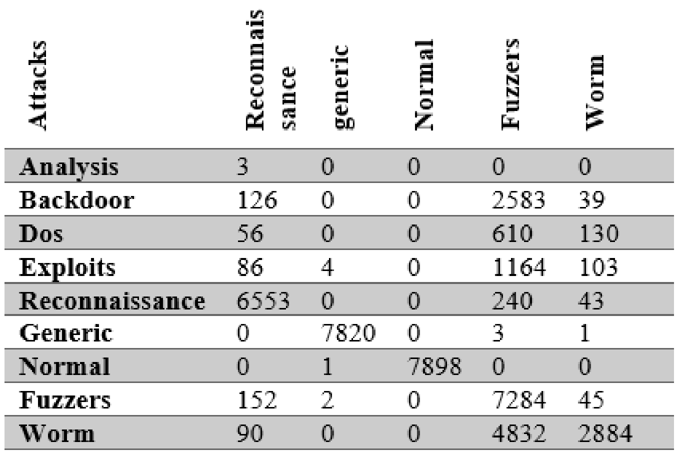
The classification report for Model 3 in Fig. 17 includes metrics such as precision, recall, and F1-score for each class in the UNSW-NB15 dataset. These metrics allow us to assess how well the model is able to identify each individual class. The model shows remarkable performance for “Generic” and “Normal” classes, achieving nearly perfect precision, recall, and F1-scores. Additionally, “Analysis” and “Reconnaissance” classes exhibit strong performance with high precision and recall values resulting in impressive F1-scores. However, there are some classes that have weaker performance. For example, the “Backdoor” and “Exploits” classes have a notable discrepancy between their precision and recall values, leading to lower F1-scores. Additionally, the “Dos” class has balanced precision and recall values, but they are still lower than those of other classes. The “Fuzzers” class has a low precision of 0.44 but a high recall of 0.94, resulting in a moderate F1-score. Lastly, the “Worm” class has a high precision of 0.89 but a low recall of 0.37, leading to a relatively low F1-score of 0.52.

The evaluation of Model 3’s classification report and confusion matrix shown in Fig. 19 and 18 weights revealed that the model can accurately identify positive instances for each class with a precision score of 84.15%. However, it is only able to capture 78.6% of the actual positive instances in the dataset, indicating room for improvement in recall performance. Despite the model’s varied performance across different classes, the higher precision score suggests that the model is relatively reliable when it makes predictions for a specific class. In summary, these results suggest that the model can benefit from further optimization to improve its overall performance, especially in the under performing classes.



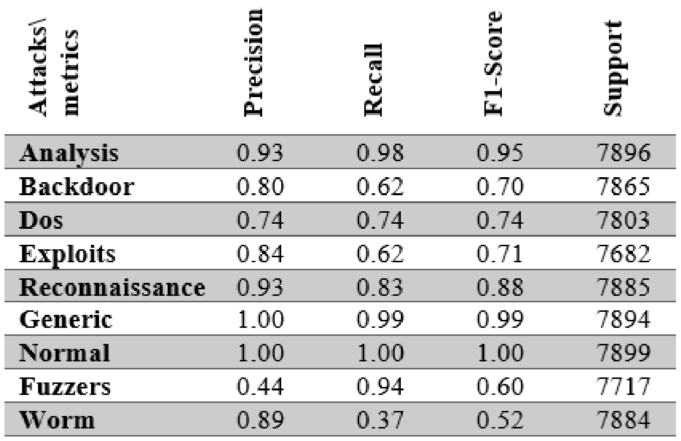
Attacks	Analysis	Backdoo	Dos	Exploits
Analysis	7702	0	182	9
Backdoor	2	4862	243	13
Dos	289	165	5767	786
Exploits	164	81	1335	4745
Reconnaissance	116	828	96	29
Generic	2	2	39	27
Normal	0	0	0	0
Fuzzers	39	61	124	10
Worm	0	47	29	2

Fig. 17. Model 3 -UNSW-NB15 - confusion report (Part1).



Attacks	Reconnais sance	generic	Normal	Fuzzers	Worm
Analysis	3	0	0	0	0
Backdoor	126	0	0	2583	39
Dos	56	0	0	610	130
Exploits	86	4	0	1164	103
Reconnaissance	6553	0	0	240	43
Generic	0	7820	0	3	1
Normal	0	1	7898	0	0
Fuzzers	152	2	0	7284	45
Worm	90	0	0	4832	2884

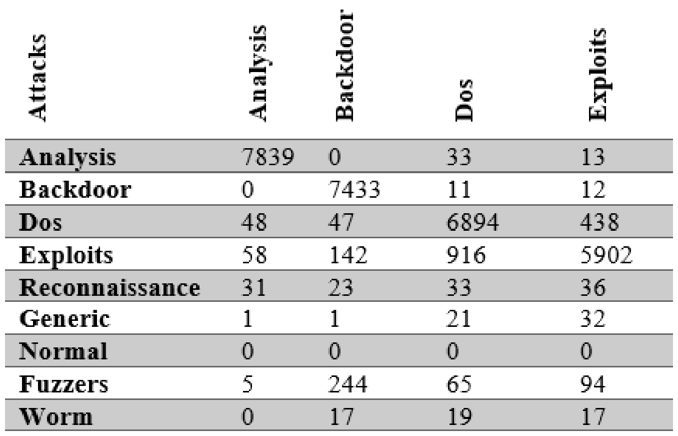
Fig. 18. Model 3 -UNSW-NB15 - confusion report (Part 2).



Attacks\ metrics	Precision	Recall	F1-Score	Support
Analysis	0.93	0.98	0.95	7896
Backdoor	0.80	0.62	0.70	7865
Dos	0.74	0.74	0.74	7803
Exploits	0.84	0.62	0.71	7682
Reconnaissance	0.93	0.83	0.88	7885
Generic	1.00	0.99	0.99	7894
Normal	1.00	1.00	1.00	7899
Fuzzers	0.44	0.94	0.60	7717
Worm	0.89	0.37	0.52	7884

Fig. 19. Model 3 -UNSW-NB15 - classification report.

Model 4’s performance is similar to that of Model 3, with accuracy improving gradually with each epoch. However, there are some fluctuations in the validation accuracy that could be due to the model being exposed to new, unseen data during specific epochs. The same is true for the loss plot.

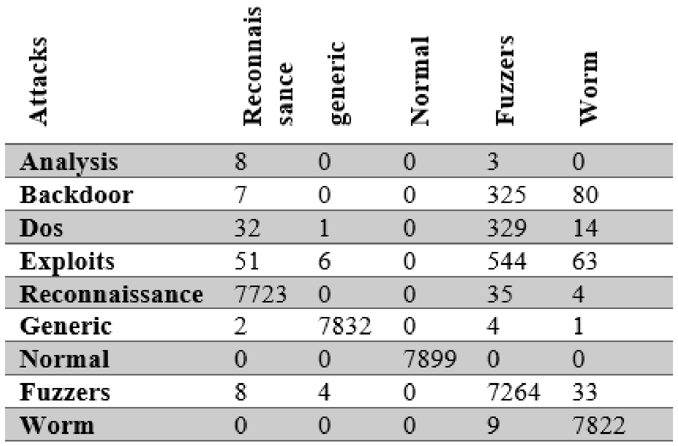


Attacks	Analysis	Backdoor	Dos	Exploits
Analysis	7839	0	33	13
Backdoor	0	7433	11	12
Dos	48	47	6894	438
Exploits	58	142	916	5902
Reconnaissance	31	23	33	36
Generic	1	1	21	32
Normal	0	0	0	0
Fuzzers	5	244	65	94
Worm	0	17	19	17

Fig. 20. Model 4 -UNSW-NB15 - confusion matrix (Part1).

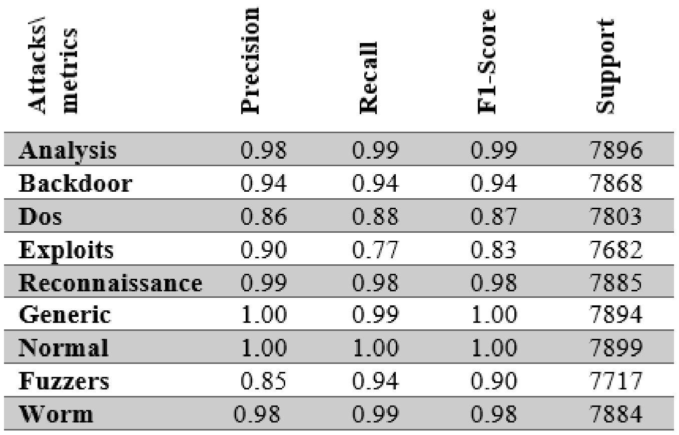
The classification report and confusion matrix for Model 4, presented in the Fig. 20 and 21 and 22, indicates that the model performs exceptionally well in most categories. Precision and recall scores are notably high for Analysis, Reconnaissance, Generic, Normal, and Worm attack types, with values near or at 1.00, indicating excellent performance. The performance for Backdoor, Dos, Exploits, and Fuzzers is also quite good, with precision and recall scores ranging between 0.85 and 0.99. Overall, the high scores across the board suggest that Model 4 is highly effective at identifying various attack types in the UNSW-NB15 dataset.

In the UNSW dataset, a total of nine attack classes have been used, with a higher number of attacks features present in Generic, Normal, and Exploits. Other attack classes have fewer rows, resulting in data imbalance. Models 1, 2, and 4 achieve an accuracy of around 95%, while Model 3 has an accuracy of 78%. This clearly demonstrates that Model 3



Attacks	Reconnais sance	generic	Normal	Fuzzers	Worm
Analysis	8	0	0	3	0
Backdoor	7	0	0	325	80
Dos	32	1	0	329	14
Exploits	51	6	0	544	63
Reconnaissance	7723	0	0	35	4
Generic	2	7832	0	4	1
Normal	0	0	7899	0	0
Fuzzers	8	4	0	7264	33
Worm	0	0	0	9	7822

Fig. 21. Model 4 -UNSW-NB15 - confusion matrix (Part2).



Attacks\ metrics	Precision	Recall	F1-Score	Support
Analysis	0.98	0.99	0.99	7896
Backdoor	0.94	0.94	0.94	7868
Dos	0.86	0.88	0.87	7803
Exploits	0.90	0.77	0.83	7682
Reconnaissance	0.99	0.98	0.98	7885
Generic	1.00	0.99	1.00	7894
Normal	1.00	1.00	1.00	7899
Fuzzers	0.85	0.94	0.90	7717
Worm	0.98	0.99	0.98	7884

Fig. 22. Model 4 -UNSW-NB15 - classification report

significantly under performs compared to the other models. When examining the individual precision and recall of the attack classes in Models 1 and 2, only the classes with a higher number of data points perform well, while others, aside from Generic, Normal, and Exploits attack classes, perform poorly. The least performing class is Backdoor, where the recall in Model 2 is around 7%. After oversampling the dataset using SMOTE, Model 4 performs better than the other models, with an accuracy of 95% and improved individual precision and recall for each of the classes. Classes with fewer data points, such as Analysis, Backdoor, DoS, and Exploits, perform better in Model 4, which incorporates ANN + Bi-LSTM models after data balancing.

D. Evaluation Summary for the Three Datasets

In summary, for the NSL-KDD, CIC-IDS2017, and UNSW-NB15 datasets, Models 1 and 2 generally perform well in terms of accuracy, while Model 3 under performs. Model 4, which employs ANN + Bi-LSTM models after data balancing using SMOTE, consistently outperforms the other models, especially when considering individual precision and recall for each attack class. Model 4 improves performance for underrepresented attack classes, indicating that balancing the dataset and using ANN + Bi-LSTM models contribute to better overall performance in intrusion detection. Table XXIII demonstrate the results of the best preforming model.

TABLE XXIII. Best Model (Model 4) - Results

Dataset	Accuracy	Precision	Recall
NSL-KDD	95.8	95.3	94.8
CIC-IDS2017	99.0	99.3	99.3
UNSW-NB15	95.3	94.4	94.2

VII. Discussion

The results presented in this study provide valuable insights into the performance of four different intrusion detection models across three distinct cybersecurity datasets. Through a rigorous evaluation process, we have demonstrated the effectiveness of the proposed model, which combines BiLSTM, XGBoost, and 1DCNN/DNN, in detecting intrusions across diverse cybersecurity environments.

The findings show that Model 4, which utilizes ANN + Bi-LSTM architecture with data balancing using SMOTE, consistently outperforms the other models in terms of precision and recall. This indicates the model's capability to effectively detect intrusion attempts, even in underrepresented attack classes. These results suggest that data balancing techniques combined with deep learning architectures offer a promising approach to improving the performance of intrusion detection models.

Additionally, the study shows that the proposed model is adaptable to different cybersecurity contexts, as demonstrated by its strong performance across the NSL-KDD, CIC-IDS2017, and UNSW-NB15 datasets. The diversity of these datasets highlights the need for intrusion detection models that can effectively handle varying cybersecurity landscapes. the results suggest that the proposed model has the potential to offer an effective solution to the ongoing challenge of intrusion detection in such environments.

It is worth noting that the study is not without limitations. While we have evaluated the proposed model across three distinct datasets, there exist many other cybersecurity datasets that could provide a more comprehensive evaluation of the model's performance. Additionally, the study has focused solely on the effectiveness of intrusion detection models and has not explored other potential applications of deep learning in cybersecurity, such as anomaly detection or threat intelligence.

Overall, our study provides evidence that combining deep learning architectures with data balancing techniques can lead to improved performance in intrusion detection. The proposed model's adaptability to diverse cybersecurity contexts offers promise for the development of effective and robust intrusion detection systems. Future research could explore the use of the proposed model on other cybersecurity datasets and investigate the potential applications of deep learning in other areas of cybersecurity.

VIII. Conclusion

This study investigated the effectiveness of using the XGBoost algorithm for feature selection in combination with different deep learning (DL) approaches, such as ANN, 1DCNN, and BiLSTM, to build accurate intrusion detection systems (IDSs) in both binary and multiclass classification settings. Three datasets were used to evaluate the proposed methods: NSL-KDD, CIC-IDS2017, and UNSW-NB15. The results demonstrate the classification models' high accuracy and low error rate, indicating that the proposed methods are a viable and promising approach for designing IDS.

Initially, the suggested DL techniques were used to test the datasets across the entire feature space, followed by the implementation of the XGBoost feature extraction technique presented in this study to obtain a reduced feature vector. The study also analyzed the performance results of researchers who utilized various classifiers. The experimental results showed that using a reduced feature vector can help reduce the model's complexity while improving detection accuracy on test data.

The datasets used in this study all contained data imbalance, which can lead to biased models over the larger categorical class. This issue was observed in models 1 and 2 on all three datasets. The SMOTE data balancing technique was introduced to address this issue, resulting in better performance in models 3 and 4, as indicated by the individual class's precision and recall. Specifically, for the NSL-KDD dataset, models 1 and 2 had low U2R class performance, with precision and recall values of 59

Overall, the performance of Models 1 and 2 is generally good in terms of accuracy for the NSL-KDD, CIC-IDS2017, and UNSW-NB15 datasets, while Model 3 does not perform as well. However, Model 4, which involves the use of ANN + Bi-LSTM models after applying SMOTE for data balancing, consistently outperforms the other models, particularly when looking at the precision and recall of each individual attack class. By improving the performance of underrepresented attack classes, Model 4 suggests that balancing the dataset and utilizing ANN + Bi-LSTM models contribute to overall improvement in intrusion detection performance.

Therefore, future work could involve implementing a novel sampling technique designed explicitly for IDSs to boost the prevalence of the minority classes in other public datasets during the training phase. Overall, this study's findings highlight the importance of data balancing techniques in addressing data imbalance issues and the effectiveness of using XGBoost and DL approaches in building accurate IDSs.

Acknowledgment

We are appreciative of our colleagues and other researchers for their aid with editing, feedback meetings, and moral support. The university's research assistants and study participants, who impacted and inspired us should also be thanked.

References

[1] Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, "A bidirectional lstm deep learning approach for intrusion detection," *Expert Systems with Applications*, vol. 185, p. 115524, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417421009337

[2] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "Lstm-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185 489–185 502, 2020.

[3] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, and D. Liu, "An optimization method for intrusion detection classification model based on deep belief network," *IEEE Access*, vol. 7, pp. 87 593–87 605, 2019.

[4] Y. Xiao, C. Xing, T. Zhang, and Z. Zhao, "An intrusion detection model based on feature reduction and convolutional neural networks," *IEEE Access*, vol. 7, pp. 42 210–42 219, 2019.

[5] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020.

[6] Y. Zhong, W. Chen, Z. Wang, Y. Chen, K. Wang, Y. Li, X. Yin, X. Shi, J. Yang, and K. Li, "Helad: A novel network anomaly detection model based on heterogeneous ensemble learning," *Computer Networks*, vol. 169, p. 107049, 2020.

[7] Z. A. A. Alyasseri, M. A. Al-Betar, I. A. Doush, M. A. Awadallah, A. K. Abasi, S. N. Makhadmeh, O. A. Alomari, K. H. Abdulkareem, A. Adam, R. Damasevicius *et al.*, "Review on covid-19 diagnosis models based on machine learning and deep learning approaches," *Expert systems*, vol. 39, no. 3, p. e12759, 2022.

[8] R. Magán-Carrión, D. Urda, I. Díaz-Cano, and B. Dorronsoro, "Towards a reliable comparison and evaluation of network intrusion detection systems based on machine learning approaches," *Applied Sciences*, vol. 10, no. 5, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/5/1775

[9] H. M. Song, J. Woo, and H. K. Kim, "In-vehicle network intrusion detection using deep convolutional neural network," *Vehicular Communications*, vol. 21, p. 100198, 2020.

[10] MIT Lincoln Laboratory, "1998 DARPA Intrusion Detection Evaluation Dataset," https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset, 1998, accessed: April 2023.

[11] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1976–1999, 2018.

[12] N. Moustafa, J. Slay, and G. Creech, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," *Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, 2015.

[13] K. Mounika and P. V. Rao, "Idcsnet: Intrusion detection and classification system using unified gradient-boosted decision tree classifier," in *2022 International Conference on Automation, Computing and Renewable Systems (ICACRS)*, Dec 2022, pp. 1159–1164.

[14] J. Hancock and T. M. Khoshgoftaar, "Performance of catboost and xgboost in medicare fraud detection," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2020, pp. 572–579.

[15] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Computer Networks*, vol. 174, p. 107247, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619314203

[16] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset," *Journal of Big Data*, vol. 7, pp. 1–20, 2020.

[17] S. S. Dhaliwal, A.-A. Nahid, and R. Abbas, "Effective intrusion detection system using xgboost," *Information*, vol. 9, no. 7, 2018. [Online]. Available: https://www.mdpi.com/2078-2489/9/7/149

[18] N. Qu, Z. Li, X. Li, S. Zhang, and T. Zheng, "Multi-parameter fire detection method based on feature depth extraction and stacking ensemble learning model," *Fire Safety Journal*, vol. 128, p. 103541, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0379711222000194

[19] A. Asselman, M. Khaldi, and S. Aammou, "Enhancing the prediction of student performance based on the machine learning xgboost algorithm," *Interactive Learning Environments*, vol. 0, no. 0, pp. 1–20, 2021. [Online]. Available: https://doi.org/10.1080/10494820.2021.1928235

[20] I. F. Kilincer, F. Ertam, and A. Sengur, "A comprehensive intrusion detection framework using boosting algorithms," *Computers and Electrical Engineering*, vol. 100, p. 107869, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790622001598

[21] Z. Liu, N. Thapa, A. Shaver, K. Roy, X. Yuan, and S. Khorsandroo, "Anomaly detection on iot network intrusion using machine learning," in *2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, 2020, pp. 1–5.

[22] D. Andrešič, P. Šaloun, and B. Pečíková, *Large Astronomical Time Series Pre-processing for Classification Using Artificial Neural Networks*. Cham: Springer International Publishing, 2021, pp. 265–293. [Online]. Available: https://doi.org/10.1007/978-3-030-65867-0_12

[23] S. Mao and E. Sejdić, "A review of recurrent neural network-based methods in computational physiology," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2022.

[24] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, 2020.

[25] B. Roy and H. Cheung, "A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network," in *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, 2018, pp. 1–6.

[26] P. Agrawal and A. Duvey, "Detecting infiltration and intrusive behaviours in wireless networks using deep learning and long short-term memory."

[27] M. Waqas, K. Kumar, A. A. Laghari, U. Saeed, M. M. Rind, A. A. Shaikh, F. Hussain, A. Rai, and A. Q. Qazi, "Botnet attack detection in internet of things devices over cloud environment via machine learning," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 4, p. e6662, 2022.

[28] D. C. Asogwa, S. O. Anigbogu, I. E. Onyenwe, and F. A. Sani, "Text classification using hybrid machine learning algorithms on big data," *CoRR*, vol. abs/2103.16624, 2021. [Online]. Available: https://arxiv.org/abs/2103.16624

[29] M. A. Khan, "Hcrnnids: Hybrid convolutional recurrent neural network-based network intrusion detection system," *Processes*, vol. 9, no. 5, 2021. [Online]. Available: https://www.mdpi.com/2227-9717/9/5/834

[30] D. Alghazzawi, O. Bamasag, H. Ullah, and M. Z. Asghar, "Efficient detection of ddos attacks using a hybrid deep learning model with improved feature selection," *Applied Sciences*, vol. 11, no. 24, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/24/11634

[31] P. Sun, P. Liu, Q. Li, C. Liu, X. Lu, R. Hao, and J. Chen, "Dl-ids: Extracting features using cnn-lstm hybrid network for intrusion detection system," *Security and Communication Networks*, vol. 2020, p. 8890306, 2020.

[32] M. Al-Omari, M. Rawashdeh, F. Qutaishat, and K. Al-Faour, "An intelligent tree-based intrusion detection model for cyber security," *Journal of Network and Systems Management*, vol. 29, no. 1, pp. 20–38, 2021.

[33] M. H. Alwan, Y. I. Hammadi, O. A. Mahmood, A. Muthanna, and A. Koucheryavy, "High density sensor networks intrusion detection system for anomaly intruders using the slime mould algorithm," *Electronics*, vol. 11, no. 20, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/20/3332

[34] D. Yu, X. Hou, C. Li, Q. Lv, Y. Wang, and N. Li, "Anomaly detection in unstructured logs using attention-based bi-lstm network," in *2021 7th IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC)*, 2021, pp. 403–407.

[35] M. Abdallah, N. A. Le Khac, H. Jahromi, and A. D. Jurcut, "A hybrid cnn-lstm based approach for anomaly detection systems in sdns," in *The 16th International Conference on Availability, Reliability and Security (ARES 2021)*. New York, NY, USA: ACM, August 2021, p. 7. [Online]. Available: https://doi.org/10.1145/3465481.3469190

[36] M. A. Abdou, "Literature review: efficient deep neural networks techniques for medical image analysis," *Neural Computing and Applications*, vol. 34, no. 8, pp. 5791–5812, 2022.

[37] P. Devan and N. Khare, "An efficient xgboost–dnn-based classification model for network intrusion detection system," *Neural Computing and Applications*, vol. 32, pp. 12 499–12 514, 2020.

[38] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, 2019.

[39] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 2016, pp. 258–263.

[40] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," 09 2016.

[41] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLOS ONE*, vol. 11, no. 6, pp. 1–17, 06 2016. [Online]. Available: https://doi.org/10.1371/journal.pone.0155781

[42] B. Mohammed and E. K. Gbashi, "Intrusion detection system for nsl-kdd dataset based on deep learning and recursive feature elimination," *Engineering and Technology Journal*, pp. 1069–1079, 2021.

[43] S. Wunderlich, M. Ring, D. Landes, and A. Hotho, "Comparison of system call representations for intrusion detection," in *Advances in Intelligent Systems and Computing*. Springer International Publishing, apr 2019, pp. 14–24. [Online]. Available: https://doi.org/10.1007%2F978-3-030-20005-3_2

[44] W.-F. Zheng, "Intrusion detection based on convolutional neural network," in *2020 International Conference on Computer Engineering and Application (ICCEA)*, 2020, pp. 273–277.

[45] M. H. Kotb and R. Ming, "Comparing smote family techniques in predicting insurance premium defaulting using machine learning models," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 9, 2021. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2021.0120970