Analyzing Deep Learning Model Performance for Intrusion Detection on CIC-IDS2017 Dataset

Md. Sadman Sakib, Noshin Tabassum
Department of Electronics & Communication Engineering
Khulna University of Engineering & Technology, Khulna-9203, Bangladesh

***Abstract*—For real time cyberthreats detection and mitigation, intrusion detection system (IDS) is essential. This research uses the CIC-IDS2017 dataset to assess how well deep learning-based models identify network intrusions. The Preprocessing stages for the data included managing infinite values, handling missing values, eliminating duplicate entries, addressing strongly correlated characteristics and correcting skewness. We used deep learning models including Multi-Layer Perceptron (MLP), 1D Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), CNN+LSTM, Gated Recurrent Unit (GRU), Bidirectional Long Short-Term Memory (BiLSTM), CNN+BiLSTM and Deep Belief Network (DBN). Accuracy, recall, F1 score and precision are among the performance indicators used to evaluate the models. The results are contrasted with previous research that made use of conventional machine learning algorithms. Our study demonstrates the effectiveness of deep learning in improving IDS performance and strengthening cybersecurity defenses.**

Keywords*—Intrusion Detection System, Deep Learning, CIC-IDS2017, Cybersecurity, MLP, CNN, LSTM, BiLSTM, GRU, DBN, Hybrid Models

I. Introduction

In the modern digital age, both individuals and organizations have to deal with an ever-increasing array of cyberthreats. Complex cyberattacks are focusing on weaknesses in organizational processes, personal data and network infrastructure. Firewalls and signature-based detection systems are examples of traditional security measures that are frequently insufficient for identifying new and emerging threats. Researchers have looked into machine learning (ML) and artificial intelligence (AI) to improve the detection capabilities of intrusion detection systems (IDS) in order to overcome these obstacles.

Intrusion detection system keeps an eye on network traffic in order to spot possible breaches of security including malware infections, illegal access and efforts at data exfiltration. IDS can be divided into two main categories: anomaly-based and signature-based. Although signature-based intrusion detection systems rely on pre-established attack patterns, they have trouble identifying new threats. In contrast, anomaly-based intrusion detection systems use statistical and machine learning methods to detect deviations from typical behavior, which increases their ability to react to new threats.

The extensive coverage of real-world attack scenarios in the CIC-IDS2017 dataset makes it a popular choice for assessing intrusion detection programs. Along with regular network traffic, it also covers a variety of attack types, including botnet activities, brute force attacks and Distributed Denial-of-Service (DDoS). Several deep learning models are trained and evaluated in the research using this dataset and their performance is compared against each other as well as with conventional machine learning models from earlier studies.

Deep learning [1] has demonstrated impressive results in recognizing intricate patterns in large datasets. Deep learning architectures are ideal for cybersecurity applications because, in contrast to conventional ML models, they can automatically extract high-level features from raw data. We want to increase the accuracy and effectiveness of IDS by utilizing a variety of deep learning models, including Multi-Layer Perceptron (MLP) [2], 1D Convolutional Neural Networks (CNN) [3], Long Short-Term Memory (LSTM) [4], Gated Recurrent Unit (GRU) [5], Bidirectional Long Short-Term Memory (BiLSTM) [6], Deep Belief Network (DBN) [7] and hybrid architectures. This research contributes to cybersecurity by contrasting deep learning models, pointing out their advantages and disadvantages for intrusion detection and demonstrating the benefits of AI-driven solutions over conventional machine learning techniques.

There are two types of deep learning-based intrusion detection: binary and multiclass. Binary classification, in which cases are divided into two groups (Benign and Attack). The classification of instances into three or more categories, such as Benign, FTP-Patator, SSH-Patator etc. is known as multiclass classification.

This is how the rest of the paper is organized. A survey of the literature on intrusion detection models, covering both deep learning and conventional machine learning techniques is given in Section 2. An overview of the CIC-IDS2017 dataset used in this research is provided in Section 3. The deep learning models used for intrusion detection and the performance evaluation technique are explained in Section 4. The experimental findings, which compare the effectiveness of the chosen deep learning models are shown in Section 5. Our findings are compared to earlier studies in Section 6, with a focus on the main distinctions and advancements. The study is finally concluded and future research directions are outlined in Section 7.

II. Literature Review

The Intrusion Detection System is indispensable to cybersecurity because they can detect suspicious activity on a network. Conventional IDS methods have depended on anomaly-based and signature-based detection strategies. These techniques, however, have a significant false positive rate and have trouble identifying new attacks. As a result, both machine learning and deep learning models have been extensively studied for intrusion detection in order to get over these restrictions.

Numerous studies have looked into intrusion detection using machine learning. For instance, Aksu et al. achieved good accuracy, precision, recall and F-measure by using SVM, KNN, and DT to detect Denial-of-Service (DoS) attacks on the CICIDS-2017 dataset [8]. Similarly to this, Panwar et al. examined several machine learning models on the CICIDS-2017 dataset, such as Naive Bayes, J48 and Decision Tree and showed how effective they performed in detecting intrusions [9]. Cavusoglu et al. evaluated Naive Bayes, J48 and Random Forest's effectiveness on NSL-KDD dataset with a particular goal of detecting DoS attacks [10]. Panwar et al. analyzed eight different supervised machine learning algorithms (GaussianNB (GNB), BernoulliNB (BNB), Decision Tree, KNN, Logistic Regression, SVM, Random Forest and SGD) on the CICIDS-2017 dataset [11]. Negandhi et al. identified different network threats using a Random Forest model and analyzed how well the suggested method worked [12]. Adebiyi et al. used Recurrent Neural Networks (RNN) and K-Nearest Neighbors (KNN), exhibiting remarkable performance in network intrusion detection with high levels of sensitivity, accuracy, and precision. [13] These ML models, however, often perform poorly in dynamic environments and need intensive feature engineering.

The capacity of deep learning models to automatically identify patterns in network traffic data has drawn a lot of interest. Kim et al. designed an intrusion detection system (IDS) using the KDD Cup 1999 dataset, showing the benefits of the Long Short-Term Memory (LSTM) architecture over a Recurrent Neural Network (RNN) [14]. Using deep learning approaches, A. Javaid et al. created a flexible and effective IDS that demonstrated good accuracy, precision and recall on the NSL-KDD dataset [15]. Yin et al. highlighted the benefits of deep learning in capturing sequential dependencies by proposing an RNN-based IDS and contrasting it with conventional ML models [16]. Vinayakumar et al. presented Scale-Hybrid-IDS-AlertNet, a highly scalable and hybrid deep neural network (DNN) architecture that is intended for real-time network traffic and host-level event monitoring in order to proactively generate alerts in an attempt to identify and mitigate potential cyberattacks. [17]. Jiang et al. constructed a reliable anomaly detection technique using the NSL-KDD dataset and neural network technology [18]. Azam et al. explored the constantly evolving field of intrusion detection and cyberattacks, providing a thorough taxonomy of IDS techniques and highlighting developments in machine learning and deep learning powered network intrusion detection systems (NIDS). [19]

Although these researches show how effective machine learning and deep learning is in intrusion detection, most studies focus on a single model or a limited comparison between the models. To determine which deep learning architectures work best, a thorough performance analysis of the various models is still required. In order to close this gap, our work compares and assesses several deep learning models using the CIC-IDS2017 dataset.

III. Dataset Description

The Canadian Institute for Cybersecurity (CIC) created the popular benchmark dataset CIC-IDS2017 to aid in the research of intrusion detection systems (IDS). Over the course of five days in a controlled setting with 25 users and a variety of devices, it mimics real-world network activity, recording a mix of benign activities and varied cyberattacks. CIC-IDS2017 dataset meets the core evaluation criteria, including complete network configuration, heterogeneity, anonymity, comprehensive interaction, full traffic capture, metadata availability, supported protocols, labeled data, complete data capture and a defined feature set.

In order to provide thorough coverage of contemporary cybersecurity threats, the dataset incorporates a wide range of attack scenarios, including Brute Force (FTP-Patator, SSH-Patator), Denial of Service attacks (Hulk, Slowloris, GoldenEye), Web Attacks (SQL Injection, XSS), Botnet activities and Network Infiltration [20]. It offers 80 flow-based characteristics that were retrieved using CICFlowMeter. These features characterize protocol behavior, packet size, flow time and statistical metrics providing useful information for anomaly identification based on machine learning. Each network flow is classified as malicious, which indicates potentially harmful activities or benign, which represents normal traffic. CICIDS-2017 has established itself as a standard dataset for assessing machine learning-based cybersecurity solutions due to its large feature set and varied depiction of attacks, which aids researchers in creating more efficient and flexible network defense systems.

TABLE I: Attack Types and Their Classification

Day	**Attack Types**	**Classification**
Monday	Normal network activities	-
Tuesday	FTP-Patator, SSH-Patator	Multiclass classification
Wednesday	DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed	Multiclass classification
Thursday Morning	Web Attack - Brute Force, Web Attack - XSS, Web Attack - SQL Injection	Multiclass classification
Thursday Afternoon	Infiltration	Binary classification
Friday Morning	Botnet	Binary classification
Friday Afternoon 1	PortScan	Binary classification
Friday Afternoon 2	DDoS	Binary classification

IV. Methodology

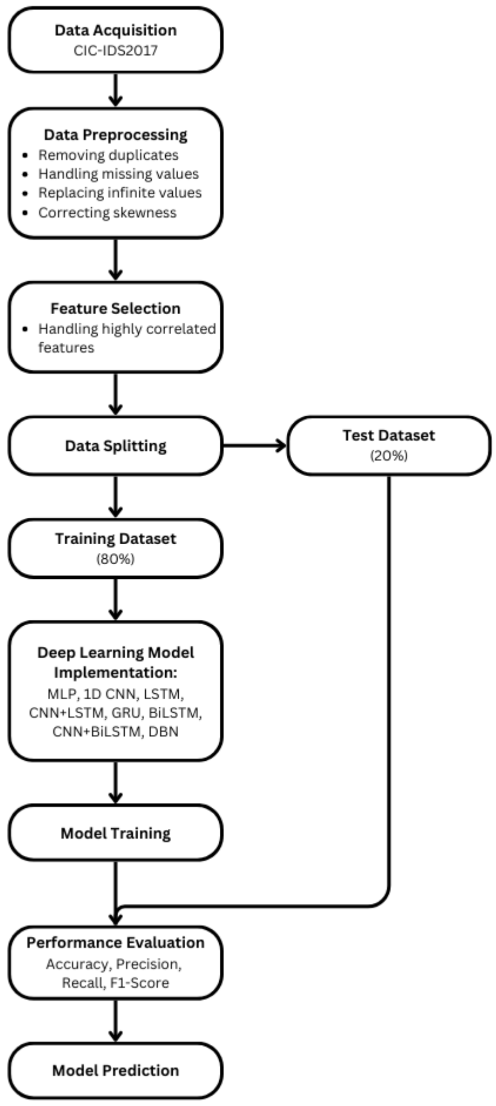

Figure I: Framework of the proposed methodology

Figure 1 shows the framework of our suggested approach. Preprocessing the dataset, Selecting features, Implementing the supervised deep learning models, Training the models, evaluating their performances and finally comparing it to conventional machine learning models are all steps in the method's organised approach.

A. Dataset and Preprocessing

Both normal and attack categories of labelled network traffic data are included in the CIC-IDS2017 dataset. We used the pre-processing steps listed below:

- Removing duplicate rows to eliminate redundancy.
- Handling missing values through imputation or removal.
- Applying correlation analysis to strongly correlated features.
- Skewness correction using transformation methods.
- Maintaining numerical stability by replacing infinite values.

B. Model Implementation

The deep learning models listed below were implemented and trained:

1) **MLP (Multi-Layer Perceptron):** In deep learning, a popular feedforward artificial neural network is the Multi-Layer Perceptron. There are three primary layers in it: an input layer, one or more hidden layers and an output layer. Every neuron in one layer is completely linked to all of the neurons in the layer below it.

   Many deep learning models are based on MLPs, which use non-linear activation functions like Sigmoid, Tanh or ReLU to recognize and understand intricate patterns. Techniques like gradient descent and backpropagation are used in training to reduce the discrepancy between expected and actual results. MLPs perform exceptionally well on structured data, excelling at tasks like regression and classification and are used as benchmark models in deep learning workflows.

2) **1D CNN (1D Convolutional Neural Network):** 1D Convolutional Neural Network is a deep learning model designed to interpret time-series or sequential data by carrying out convolution operations along a single dimension. It finds localized patterns and features such as spikes, trends or repetitions by using learnable filters that move across the input sequence.

   As 1D CNNs are more effective and parameter efficient than fully connected networks, they are ideally suited for jobs involving sequential inputs such as sensor data, audio signals and other time dependent datasets. These networks perform well in tasks like classification, regression and feature extraction because they use numerous layers of convolution and pooling to gradually capture increasingly abstract representations.

3) **LSTM (Long Short-Term Memory):** Long Short-Term Memory networks are a type of recurrent neural network (RNN) that is specifically built to deal with long-term dependencies in sequential input.

   LSTMs have special memory cells with three key gates: the forget gate (which determines which information to discard), the input gate (which decides what new information to retain) and the output gate (which generates output based on the memory). LSTMs can effectively remember and utilize relevant data over long periods. Their effectiveness makes them appropriate for applications such as time series forecasting, speech recognition, language modeling and jobs that need awareness of long-term interdependence.

4) **CNN+LSTM:** Convolutional neural networks and long short-term memory networks are combined in the CNN+LSTM model to process complex sequential data with ease. CNN layers are used initially in this integrated architecture to extract temporal features from input, such as patterns in time series data or sequences. The CNN's features are then fed into LSTM layers, which are intended to capture temporal dynamics and long-term dependencies. This combination works especially well for tasks where both localized feature extraction and sequential dependency modeling are crucial, like time series forecasting, network traffic analysis, speech recognition and video categorization.

   The CNN+LSTM model efficiently captures both spatial and temporal correlations by combining CNNs for feature recognition and LSTMs for sequence learning, giving predictions that are more accurate and dependable.

5) **GRU (Gated Recurrent Unit):** Gated Recurrent Unit is a form of recurrent neural network that, like the Long Short-Term Memory, is intended to detect long-term dependencies in sequential input.

   GRUs try to solve the vanishing gradient problem that standard RNNs suffer by employing a simpler architecture with fewer gates. GRUs employ two gates: the update gate, which governs how much of the prior memory is maintained and the reset gate, which decides how much of the previous information to discard while processing new input. GRUs are especially helpful for tasks like time series prediction, machine translation and speech recognition where comprehending temporal connections is crucial because of their simpler structure, which enables them to learn effectively without sacrificing speed.

6) **BiLSTM (Bidirectional Long Short-Term Memory):** Bidirectional Long Short-Term Memory is an extension of regular LSTM that processes sequential data both forwards and backwards. Typical LSTM processes information from past to future. But BiLSTM is able to capture information from both past and future contexts by employing two LSTM layers, one that processes the input sequence from left to right and another that processes it from right to left. This bidirectional approach enables the network to learn from both the previous and succeeding elements in a sequence, making it especially useful for tasks such as speech recognition, sentiment analysis and named entity recognition, where context from both directions can significantly improve performance.

   BiLSTMs offer a deeper comprehension of sequential data by utilizing both forward and backward temporal dependencies.

7) **CNN+BiLSTM:** Convolutional neural networks and bidirectional long short-term memory networks are combined in a CNN+BiLSTM model to handle complex data with temporal and spatial relationships.

   Usually, the CNN layers in this architecture are employed first to automatically extract local characteristics from the input data, including temporal features in time series or spatial patterns in pictures. The BiLSTM layers receive the extracted features after which they analyze the data both forward and backward, capturing context from the past and future. This combination works especially well for tasks that need a comprehension of both spatial patterns and bidirectional temporal connections, such as speech recognition, video classification, emotion detection and sequence labeling. The CNN+BiLSTM model enables the extraction of precise characteristics and a thorough grasp of sequential relationships, resulting in more accurate and robust predictions.

8) **DBN (Deep Belief Network):** A Deep Belief Network is a form of generative neural network made up of several layers of stochastic and latent variables. Each Restricted Boltzmann Machine (RBM) learns a probabilistic distribution over its input and it is made up of multiple RBMs stacked on top of one another.

   DBN's layers are trained layer-by-layer, with the first layer being taught like an RBM and the others being trained in an unsupervised greedy method. Once the layers have been trained, the network is fine-tuned with supervised learning techniques such as backpropagation. DBNs are particularly effective for unsupervised learning tasks and can be used for feature extraction, dimensionality reduction and generative modeling. DBNs have been applied to image recognition, audio processing and collaborative filtering because of their capacity to learn hierarchical representations of data.

C. Hardware and Software used

For training and evaluating deep learning models, we utilized Google Colab, a cloud-based Jupyter notebook environment that provides access to high-performance computing resources. Its support for GPU and TPU acceleration makes it popular for deep learning and machine learning tasks.

Hardware Configuration (Google Colab):

- **CPU:** Intel Xeon processor (allocated dynamically based on session availability).
- **GPU:** NVIDIA Tesla T4, K80, P100, or A100.
- **TPU:** Google Tensor Processing Units (optional, for deep learning tasks).
- **RAM:** Up to 12–25 GB (allocated dynamically based on session).

D. Software Stack

- **Operating System:** Ubuntu (Google Colab operates on a Linux-based cloud environment).
- **Programming Language:** Python 3.x
- **Deep Learning Frameworks:** TensorFlow, Keras, PyTorch.
- **Machine Learning Libraries:** Scikit-learn, NumPy, Pandas, Matplotlib & Seaborn (for visualisation).

E. Performance Metrics

We have chosen five important measures to evaluate the models' performance: Accuracy, Recall, Precision, F1 Score and Training time.

True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN) are the four basic indications that are used to construct these measures.

True Positive (TP): The number of harmful instances that are accurately identified as malicious.

False Positive (FP): The number of harmless cases that are mistakenly labeled as harmful.

False Negative (FN): The number of malicious cases that are mistakenly categorized as benign.

True Negative (TN): The number of benign cases that are accurately identified as benign.

(i) Accuracy: The percentage of true predictions (including true positives and true negatives) to all predictions is how accuracy evaluates a model's overall correctness.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

(ii) Precision: Precision is concerned with the degree of accurate projections. This represents the ratio of real positive predictions to all positive predictions (true positives and false positives).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

(iii) Recall: Recall is the proportion of true positive predictions to all actual positives (true positives and false negatives).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

(iv) F1 Score: The F1 score is calculated as the harmonic mean of precision and recall. It is especially helpful when there is a trade-off between precision and recall or when datasets are unbalanced.

$$\text{F1 Score} = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{4}$$

(v) Training time: The amount of time needed for a machine learning or deep learning model to finish its training process is referred to as training time. Processing the dataset, performing forward and backward propagation, updating model's parameters and finishing all iterations are included in this time frame.

V. Result Analysis

Our model was implemented in Python and we used the CIC-IDS2017 dataset to test a number of deep learning algorithms such as Multi-Layer Perceptron (MLP), 1D Convolutional Neural Network (1D CNN), Long Short-Term Memory (LSTM), CNN+LSTM, Gated Recurrent Unit (GRU), Bidirectional LSTM (BiLSTM), CNN+BiLSTM and Deep Belief Network (DBN).
Tables II, III, IV, V and VI represents the numerical data while Figures II, III, IV, V and VI illustrate the data through bar graph representation. Similarly, Tables VII, VIII, IX, X and XI provide a summary of the multiclass classification results, while Figures VII, VIII, IX, X and XI correlate to the equivalent visual representations.

TABLE II: Accuracy for Binary Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DDoS	0.999756	0.999779	0.999690	0.999690	0.999601	0.999646	0.999601	0.660480
PortScan	0.999878	0.999703	0.999808	0.999651	0.999878	0.999895	0.999878	0.991779
Infiltration	0.999913	0.999931	0.999913	0.999931	0.999913	0.999931	0.999948	0.999879
Botnet	0.996231	0.996179	0.996310	0.996283	0.996310	0.996388	0.996336	0.990211

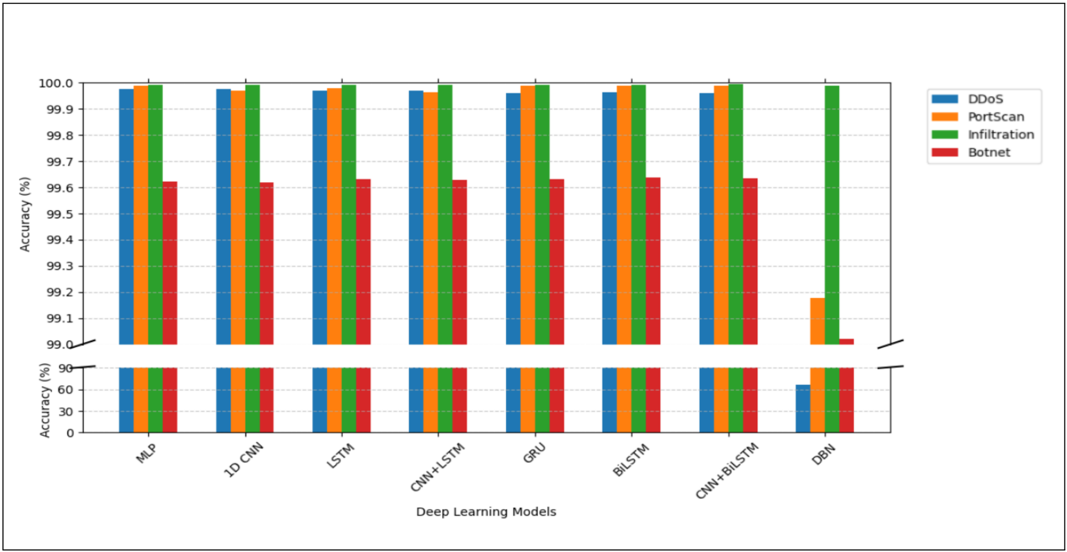

Figure II: Accuracy for Binary Classification

TABLE III: Precision for Binary Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DDoS	0.999756	0.999779	0.999690	0.999690	0.999601	0.999646	0.999601	0.659933
PortScan	0.999878	0.999703	0.999808	0.999651	0.999878	0.999895	0.999878	0.991782
Infiltration	0.999908	0.999924	0.999908	0.999931	0.999913	0.999924	0.999948	0.999757
Botnet	0.996213	0.996161	0.996308	0.996238	0.996308	0.996386	0.996278	0.980518

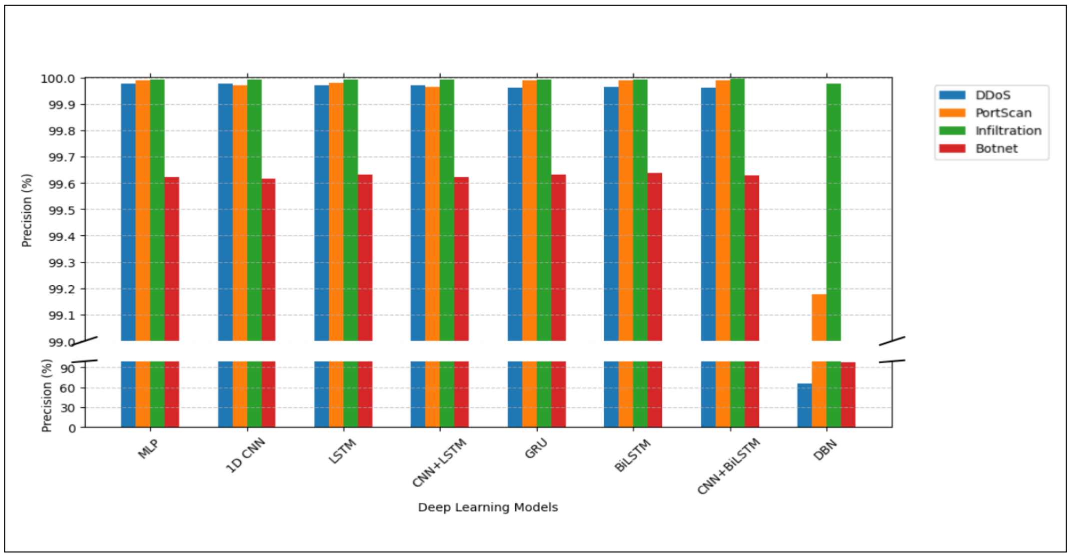

Figure. III: Precision for Binary Classification

TABLE IV: Recall for Binary Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DDoS	0.999756	0.999779	0.999690	0.999690	0.999601	0.999646	0.999601	0.660480
PortScan	0.999878	0.999703	0.999808	0.999651	0.999878	0.999895	0.999878	0.991779
Infiltration	0.999913	0.999931	0.999913	0.999931	0.999913	0.999931	0.999948	0.999879
Botnet	0.996231	0.996179	0.996310	0.996283	0.996310	0.996388	0.996336	0.990211

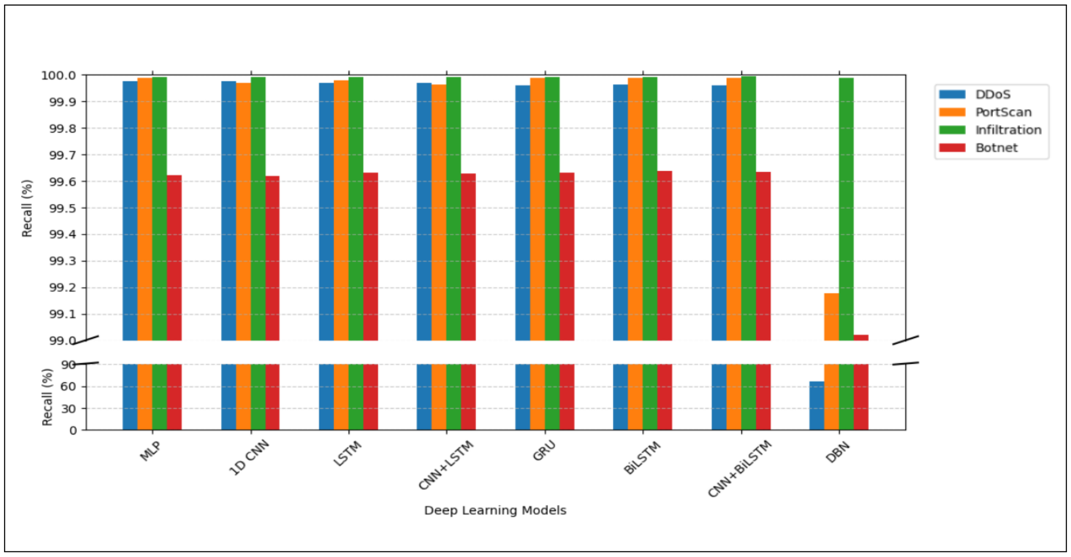

Figure IV: Recall for Binary Classification

TABLE V: F1 Score for Binary Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DDoS	0.999756	0.999779	0.999690	0.999690	0.999601	0.999646	0.999601	0.645405
PortScan	0.999878	0.999703	0.999808	0.999651	0.999878	0.999895	0.999878	0.991778
Infiltration	0.999910	0.999925	0.999910	0.999917	0.999889	0.999925	0.999941	0.999818
Botnet	0.995801	0.995735	0.995892	0.995881	0.995892	0.995990	0.995953	0.985341

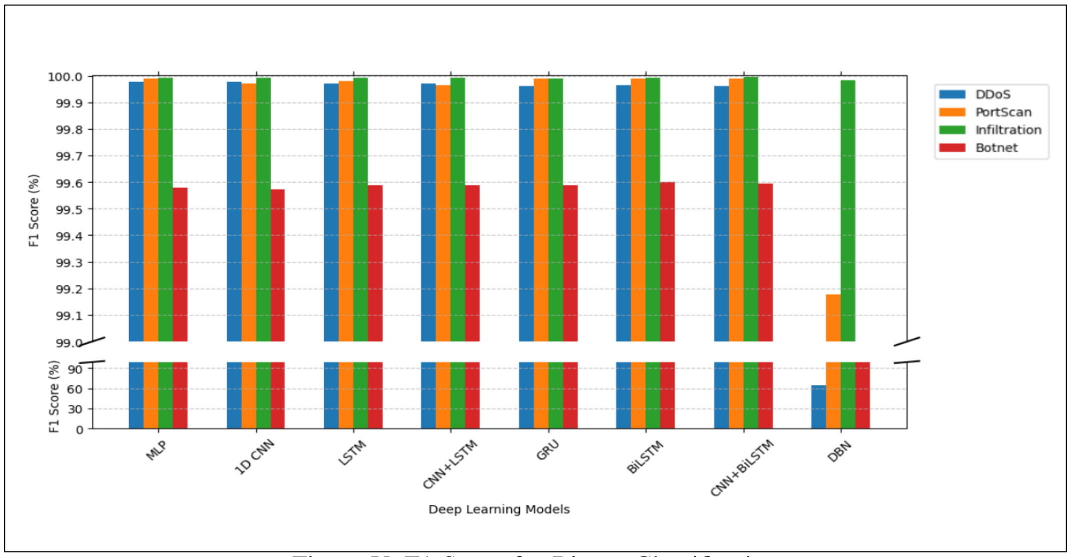

Figure V: F1 Score for Binary Classification

TABLE VI: Training Time for Binary Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DDoS	573.64	1469.57	1553.06	2827.95	1445.01	1821.25	2992.93	59.36
PortScan	817.45	1681.39	1527.93	3821.07	1498.10	2605.87	3931.47	78.62
Infiltration	685.58	1716.84	1494.21	3889.12	1494.31	2586.94	3941.73	75.29
Botnet	549.74	1410.61	802.83	2684.26	796.93	1547.99	2706.94	51.45

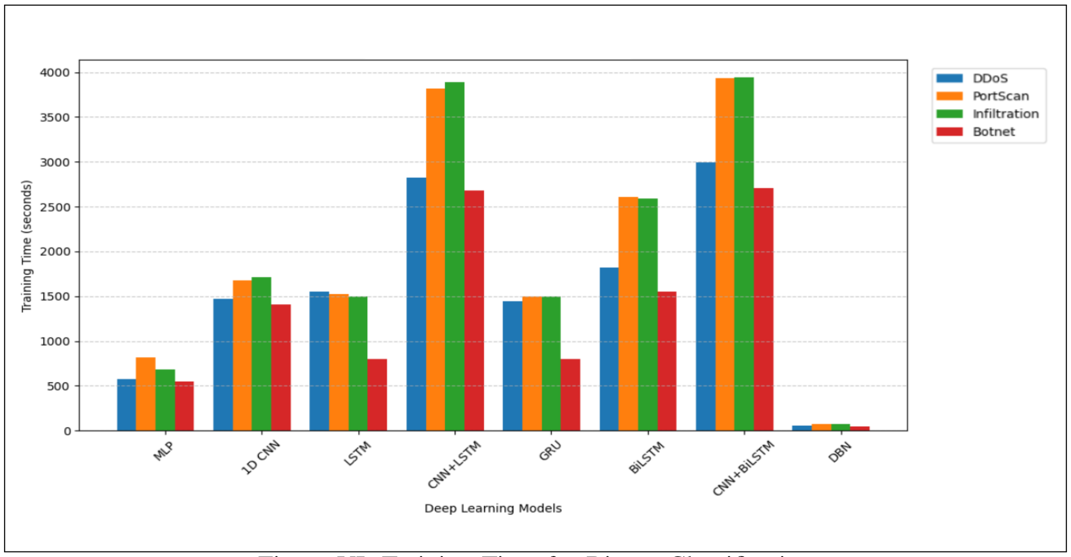

Figure VI: Training Time for Binary Classification

Among the models tested for detecting binary-class attacks, such as DDoS, Portscan, Infiltration and Botnet attacks, **1D CNN model** exhibited remarkable performance in identifying DDoS attacks, excelling in accuracy, precision, recall and F1 score. Similarly, the **BiLSTM model** proved highly effective for detecting Portscan attacks, demonstrating impressive results across these same metrics. For Infiltration attack detection, the hybrid **CNN+BiLSTM architecture** delivered outstanding performance in terms of accuracy, precision, recall and F1 score. Lastly, the **BiLSTM model** also proved to be a reliable choice to identify botnet attacks, showing commendable results in all evaluation parameters.

TABLE VII: Accuracy for Multiclass Classification

Attacks	**MLP**	**1D CNN**	**LSTM**	**CNN + LSTM**	**GRU**	**BiLSTM**	**CNN + BiLSTM**	**DBN**
DoS	0.992861	0.996925	0.996665	0.996016	0.996976	0.997214	0.997062	0.753163
Web Attack	0.992722	0.992957	0.993045	0.992692	0.992898	0.993045	0.993338	0.986383
Brute Force	0.999249	0.999596	0.999114	0.999473	0.999439	0.998385	0.999619	0.968929

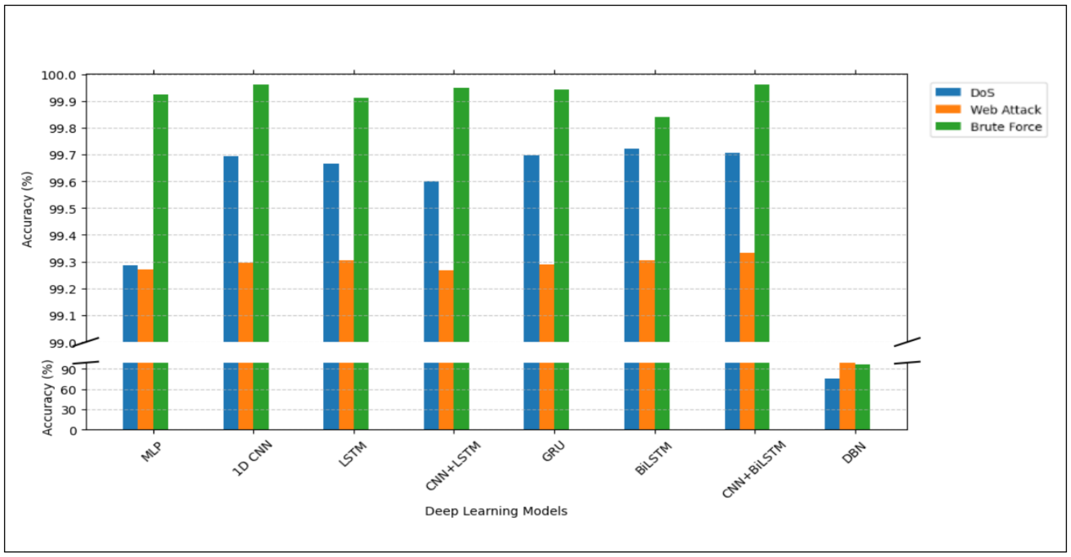

Figure VII: Accuracy for Multiclass Classification

TABLE VIII: Precision for Multiclass Classification

Attacks	**MLP**	**1D CNN**	**LSTM**	**CNN + LSTM**	**GRU**	**BiLSTM**	**CNN + BiLSTM**	**DBN**
DoS	0.992965	0.996944	0.996687	0.996048	0.996992	0.997228	0.997077	0.727405
Web Attack	0.994345	0.995408	0.995473	0.991101	0.995569	0.995473	0.994639	0.972951
Brute Force	0.999247	0.999596	0.999130	0.999474	0.999438	0.998370	0.999618	0.938823

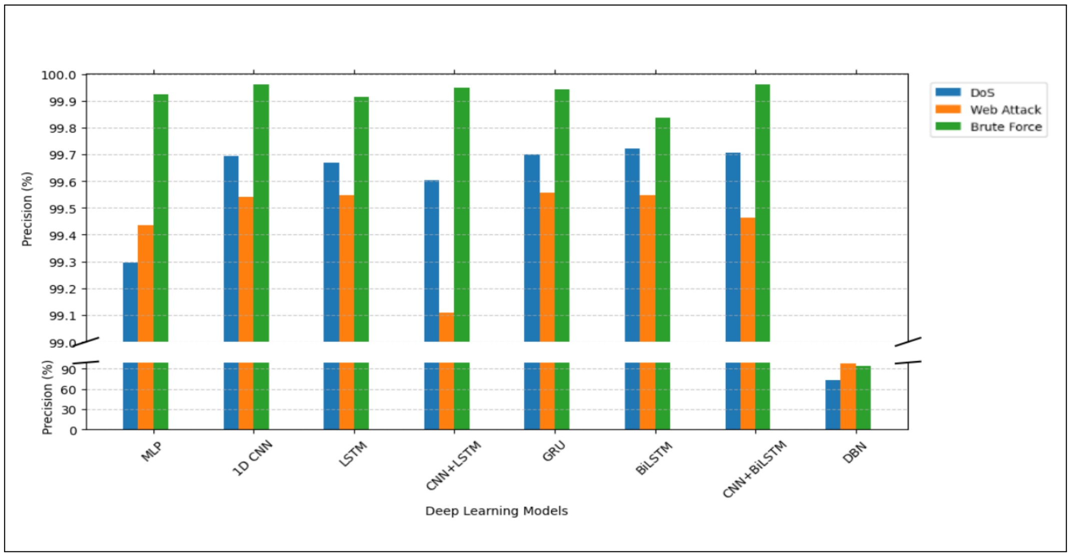

Figure VII: Precision for Multiclass Classification

TABLE IX: Recall for Multiclass Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DoS	0.992861	0.996925	0.996665	0.996016	0.996976	0.997214	0.997062	0.753163
Web Attack	0.992722	0.992957	0.993045	0.992692	0.992898	0.993045	0.993338	0.986383
Brute Force	0.999249	0.999596	0.999114	0.999473	0.999439	0.998385	0.999619	0.968929

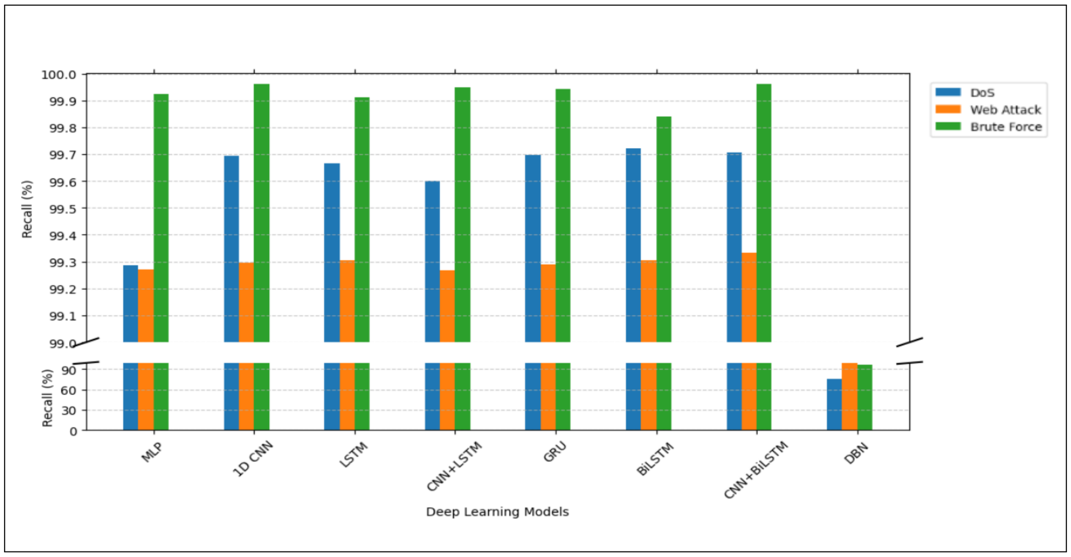

Figure IX: Recall for Multiclass Classification

TABLE X: F1 Score for Multiclass Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DoS	0.992874	0.996927	0.996667	0.996016	0.996979	0.997216	0.997064	0.739822
Web Attack	0.991591	0.991792	0.991866	0.991412	0.991842	0.991866	0.992079	0.979621
Brute Force	0.999247	0.999596	0.999119	0.999473	0.999438	0.998374	0.999618	0.953638

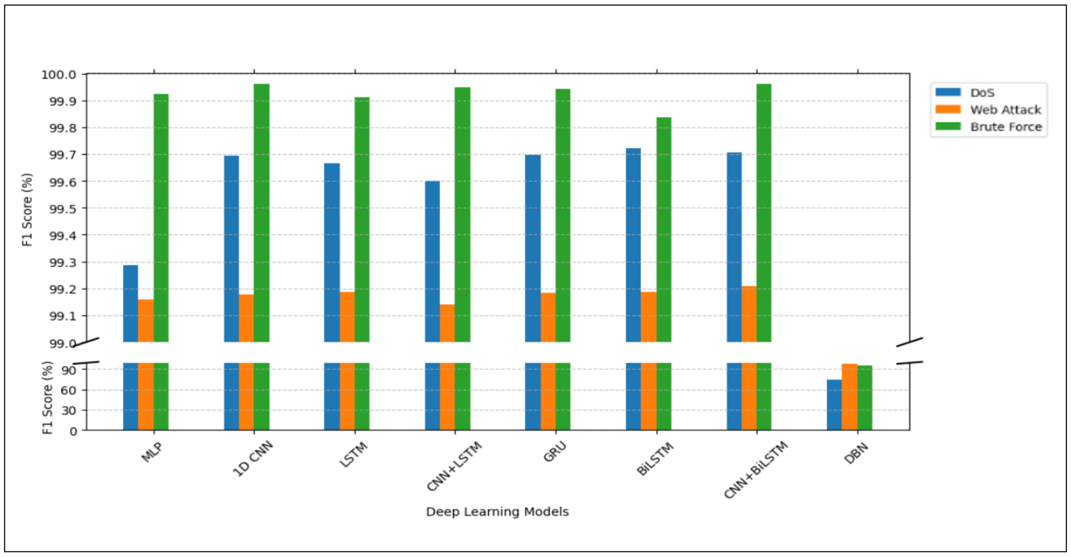

Figure X: F1 Score for Multiclass Classification

TABLE XI: Training Time for Multiclass Classification

Attacks	MLP	1D CNN	LSTM	CNN + LSTM	GRU	BiLSTM	CNN + BiLSTM	DBN
DoS	1363.47	3943.31	3126.01	8793.69	2806.15	5092.25	9118.58	196.93
Web Attack	473.32	1378.99	746.83	2538.64	797.81	1488.18	2573.67	46.15
Brute Force	793.46	2625.29	2413.37	5221.72	1569.33	3824.58	5180.22	109.94

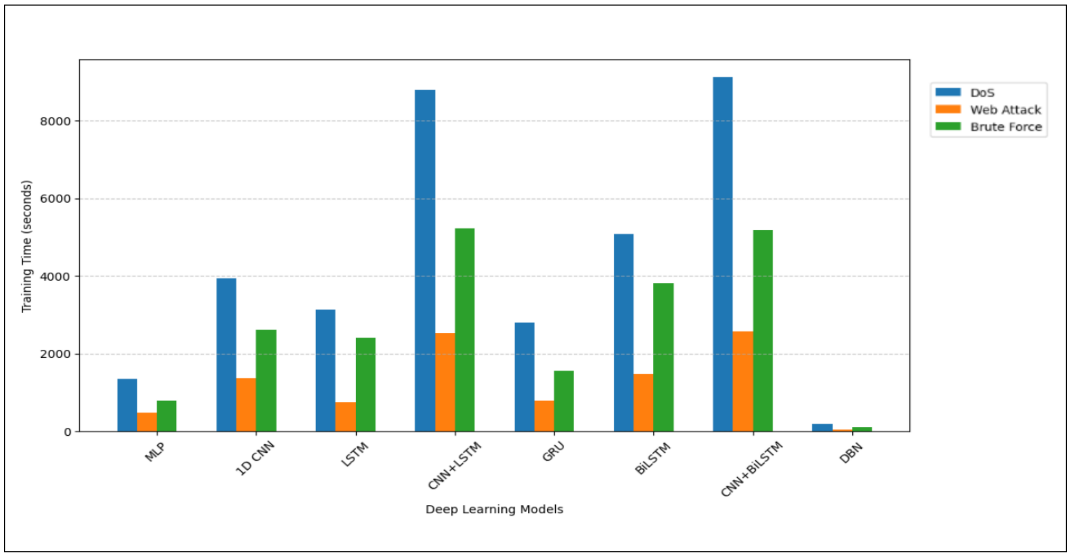

Figure XI: Training Time for Multiclass Classification

When detecting multi-class attacks in the CIC-IDS2017 dataset, such as DoS, Web attacks and Brute Force attacks, the models demonstrated distinct strengths. The **BiLSTM model** performed exceptionally well in identifying DoS attacks, achieving strong results in accuracy, precision, recall and F1 score. In case of Web attack detection, the hybrid **CNN + BiLSTM architecture** delivered notable performance, consistently excelling in all evaluation metrics. Similarly, for detection of brute force attacks, **CNN + BiLSTM model** exhibited excellent accuracy, precision, recall and F1 score, highlighting its effectiveness in handling this type of intrusion.

TABLE XII: Accuracy Comparison Between Our Results and Previous Studies [11]

Attacks	GNB	BNB	DT	KNN	LR	SVM	RF	SGD
DDoS	0.881810	0.970194	0.999873	0.999725	0.997447	0.998776	0.999888	0.997806
PortScan	0.994697	0.993733	0.999850	0.999800	0.996924	0.999235	0.999867	0.999035
Infiltration	0.908904	0.995462	0.999810	0.999825	0.999818	0.999818	0.999837	0.999817
Botnet	0.995787	0.791659	0.999810	0.999077	0.994400	0.993793	0.999845	0.991375
DoS	0.859371	0.847384	0.999660	0.998996	0.984594	0.984737	0.999787	0.989636
Web Attack	0.973865	0.984235	0.995527	0.999524	0.990776	0.989769	0.999840	0.987394
Brute Force	0.978153	0.992855	0.999867	0.999881	0.992471	0.992492	0.999901	0.986352

TABLE XIII: Precision Comparison Between Our Results and Previous Studies [11]

Attacks	GNB	BNB	DT	KNN	LR	SVM	RF	SGD
DDoS	0.972054	0.997430	0.999814	0.999667	0.996381	0.999116	0.999862	0.997430
PortScan	0.997471	0.994420	0.999819	0.999849	0.995224	0.999037	0.999879	0.994239
Infiltration	0.001989	0.967462	0.818181	0.998896	0.999675	0.999734	0.999800	0.998867
Botnet	0.976833	0.979840	0.983091	0.935632	0.796296	0.996002	0.999806	0.999276
DoS	0.906173	0.997722	0.999245	0.997984	0.942425	0.986583	0.999853	0.987634
Web Attack	0.304523	0.787878	0.986842	0.969696	0.193548	0.977725	0.999745	0.968694
Brute Force	0.697058	0.998589	0.995783	0.994574	0.262370	0.969712	0.998756	0.969713

TABLE XIV: Recall Comparison Between Our Results and Previous Studies [11]

Attacks	GNB	BNB	DT	KNN	LR	SVM	RF	SGD
DDoS	0.816112	0.999481	0.999762	0.999762	0.999370	0.999111	0.999825	0.998741
PortScan	0.996562	0.994210	0.999835	0.999788	0.999215	0.999259	0.999851	0.999185
Infiltration	0.846153	0.996742	0.692307	0.230769	0.153846	0.995632	0.998767	0.979783
Botnet	0.843705	0.959192	0.978365	0.978365	0.620192	0.997733	0.998481	0.815835
DoS	0.764854	0.926485	0.998763	0.999245	0.989945	0.965743	0.999811	0.986537
Web Attack	0.835920	0.973392	0.997782	0.992348	0.336021	0.999771	0.999843	0.957373
Brute Force	0.506410	0.998589	0.999084	0.999389	0.747738	0.998544	0.999896	0.987473

TABLE XV: F1 Score Comparison Between Our Results and Previous Studies [11]

Attacks	GNB	BNB	DT	KNN	LR	SVM	RF	SGD
DDoS	0.887283	0.908362	0.999843	0.999814	0.997773	0.999114	0.999897	0.998085
PortScan	0.997016	0.994314	0.999824	0.999818	0.999215	0.999147	0.999864	0.996707
Infiltration	0.003968	0.981883	0.749999	0.374946	0.266854	0.997678	0.999283	0.989232
Botnet	0.905401	0.969406	0.980722	0.956531	0.797842	0.997066	0.999143	0.898285
DoS	0.829537	0.960784	0.999307	0.998610	0.965600	0.976051	0.999527	0.987088
Web Attack	0.449220	0.870864	0.992281	0.980891	0.245619	0.988625	0.999793	0.963000
Brute Force	0.586633	0.998589	0.997430	0.996975	0.383441	0.983916	0.999325	0.978512

TABLE XVI: Comparison of Our Results with Past Studies [17] [20] [21]

Machine Learning Algorithms	**Sharafaldin et al. 2018 [20]**			
	Accuracy	Precision	Recall	F1 Score
Random Forest	-	0.98	0.97	0.97
Naïve Bayes	-	0.88	0.84	0.84
KNN	-	0.96	0.96	0.97
	Kahraman Kostas 2018 [21]			
Random Forest	0.94	0.94	0.94	0.94
Naïve Bayes	0.86	0.86	0.87	0.86
KNN	0.97	0.97	0.97	0.97
	Vinayakumar et al. 2019 : Binary Classification [17]			
Random Forest	0.940	0.849	0.969	0.905
Naïve Bayes	0.313	0.300	0.979	0.459
KNN	0.910	0.781	0.968	0.865
Logistic Regression	0.839	0.685	0.850	0.758
Decision Tree	0.935	0.839	0.965	0.898
SVM	0.799	0.992	0.328	0.493
	Vinayakumar et al. 2019 : Multiclass Classification [17]			
Random Forest	0.944	0.970	0.944	0.953
Naïve Bayes	0.250	0.767	0.250	0.188
KNN	0.909	0.949	0.909	0.922
Logistic Regression	0.870	0.889	0.870	0.868
Decision Tree	0.940	0.965	0.940	0.949
SVM	0.915	0.757	0.915	0.723

In terms of intrusion detection, our analysis shows that deep learning models perform noticeably better than conventional machine learning models. The models demonstrated their effectiveness in identifying intricate patterns in network traffic data by achieving high accuracy and F1 scores.

In contrast to earlier studies that employed machine learning models, our deep learning methodology resulted in a significant improvement in detection rates, lowering false positives while preserving high recall. These findings highlight the potential for improving the intrusion detection system through deep learning techniques.

VI. Conclusion

This paper offers a thorough analysis of deep learning methods for intrusion detection with the CICIDS-2017 dataset. Through the application of several deep learning models, including MLP, 1D CNN, GRU, DBN, LSTM, BiLSTM, CNN+LSTM and CNN+BiLSTM, we were able to demonstrate the ability of these architectures to accurately detect network intrusions. The experimental findings demonstrate that deep learning models perform noticeably better than conventional machine learning algorithms in terms of accuracy, precision, recall and F1-score by utilizing sophisticated neural network topologies.

Future research can investigate ensemble learning strategies, which mix several models to increase resilience against adversarial attacks and generalization. Furthermore, integrating lightweight deep learning architectures optimized for low-latency inference into real-time intrusion detection systems may improve their practical applicability. Transfer learning strategies should be further explored in order to adapt previously trained IDS models to new datasets without necessitating a significant amount of retraining. Future IDS can attain higher accuracy, scalability and real-time threat mitigation capabilities by improving hybrid architectures and incorporating adaptive learning mechanisms, enhancing cybersecurity defenses against new attacks.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] S. Amari, "A theory of adaptive pattern classifier," *IEEE Transactions*, vol. EC, no. 16, pp. 279–307, 1967.

[3] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, 1989.

[4] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," in *Proceedings of the 9th International Conference on Neural Information Processing Systems (NIPS'96)*. MIT Press, 1996, pp. 473–479.

[5] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734.

[6] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005, iJCNN 2005.

[7] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 07 2006.

[8] D. Aksu, S. Üstebay, M. A. Aydin, and T. Atmaca, "Intrusion detection with comparative analysis of supervised learning techniques and fisher score feature selection algorithm," in *Computer and Information Sciences*, T. Czachórski, E. Gelenbe, K. Grochla, and R. Lent, Eds. Cham: Springer International Publishing, 2018, pp. 141–149.

[9] S. S. Panwar, Y. P. Raiwani, and L. S. Panwar, "Evaluation of network intrusion detection with features selection and machine learning algorithms on cicids-2017 dataset," in *International Conference on Advances in Engineering Science Management & Technology (ICAESMT) - 2019, Uttaranchal University, Dehradun, India*, 2019.

[10] Ünal Çavuşoğlu, "A new hybrid approach for intrusion detection using machine learning methods," *Applied Intelligence*, vol. 49, no. 8, pp. 2735–2761, February 2019.

[11] S. S. Panwar, Y. P. Raiwani, and L. S. Panwar, "An intrusion detection model for cicids-2017 dataset using machine learning algorithms," in *2022 International Conference on Advances in Computing, Communication and Materials (ICACCM)*, 2022, pp. 1–10.

[12] P. Negandhi, Y. Trivedi, and R. Mangrulkar, "Intrusion detection system using random forest on the nsl-kdd dataset," in *Emerging Research in Computing, Information, Communication and Applications: ERCICA 2018, Volume 2*. Springer, 2019, pp. 519–531.

[13] M. O. Adebiyi, O. G. Atanda, C. Okeke, A. A. Adebiyi, and A. A. Adebiyi, "Network intrusion detection using k-nearest neighbors (knn) and recurrent neural networks (rnn)," in *2024 International Conference on Science, Engineering and Business for Driving Sustainable Development Goals (SEB4SDG)*, 2024, pp. 1–8.

[14] J. Kim, J. Kim, H. L. Thi Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *2016 International Conference on Platform Technology and Service (PlatCon)*, 2016, pp. 1–5.

[15] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 21–26.

[16] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[17] 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.

[18] F. Jiang, Y. Fu, B. B. Gupta, Y. Liang, S. Rho, F. Lou, F. Meng, and Z. Tian, "Deep learning based multi-channel intelligent attack detection for data security," *IEEE Transactions on Sustainable Computing*, vol. 5, no. 2, pp. 204–212, 2020.

[19] Z. Azam, M. M. Islam, and M. N. Huda, "Comparative analysis of intrusion detection systems and machine learning-based model analysis through decision tree," *IEEE Access*, vol. 11, pp. 80 348–80 391, 2023.

[20] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani *et al.*, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." 2018.

[21] K. Kostas, "Anomaly detection in networks using machine learning," School of Computer Science and Electronic Engineering, University of Essex, Aug. 2018.