BiLSTM-CNN Hybrid Intrusion Detection System for IoT Application

Sapna Sadhwani
Birla Institute of Technology and Science, Pilani - Dubai Campus
Mohammed Abdul Hafeez Khan
Birla Institute of Technology and Science, Pilani - Dubai Campus
Raja Muthalagu (raja.m@dubai.bits-pilani.ac.in)
Birla Institute of Technology and Science, Pilani - Dubai Campus
Pranav Mothabhau Pawar
Birla Institute of Technology and Science, Pilani - Dubai Campus

Research Article

Keywords: Internet of Things, Intrusion Detection System, Deep Learning, CNN, BiLSTM

Posted Date: January 3rd, 2024

DOI: https://doi.org/10.21203/rs.3.rs-3820775/v1

Additional Declarations: No competing interests reported.

BiLSTM-CNN Hybrid Intrusion Detection System for IoT Application

Sapna Sadhwani[1], Mohammed Abdul Hafeez Khan[2], Raja Muthalagu[3], and Pranav Mothabhau Pawar[4]

[1,2,3,4] Department of Computer Science Engineering, Birla Institute of Technology and Science Pilani, Dubai Campus, Dubai International Academic City, Dubai, UAE
Email [1]sapna@dubai.bits-pilani.ac.in, [2]f20190091@dubai.bits-pilani.ac.in, [3]raja.m@dubai.bits-pilani.ac.in, [4]pranav@dubai.bits-pilani.ac.in

Abstract. Intrusions in computer networks have increased significantly in recent times and network security mechanisms are not being developed at the same pace at which intrusion attacks are evolving. Therefore, a need has arisen to improve intrusion detection systems (IDS) to make network secure. This research focuses on anomaly-based IDS for security assaults. In this research, deep learning techniques such as Bi-directional Long Short-Term Memory (Bi-LSTM) and Convolutional Neural Networks (CNN) are implemented and subsequently used to design a novel BiLSTM-CNN hybrid IDS for the Internet of Things (IoT). The hybrid intrusion detection system model is created by utilizing the advantages of both the BiLSTM and the CNN's ability to extract temporal and spatial features respectively. The research uses the UNSW-NB 15 dataset for proposed deep learning IDS for IoT networks. The dataset has been split into training and testing data for classifying traffic into normal or attack classes. The models are run on GPU and CPU to illustrate their efficacy and match real-world IoT network communication behavior. The BiLSTM, CNN, and hybrid BiLSTM-CNN models are assessed on various aspects like Precision, Sensitivity, F1-Score, Miscalculation Rate, False Positive Rate, False Negative Rate, and Matthews Correlation Coefficient to evaluate the model's robustness. The findings revealed that the hybrid model surpassed the BiLSTM and CNN models in all aspects. Additionally, the proposed model is compared with the cutting-edge existing approaches in terms of different performance metrics and proved to be better than state-of-the-art models.

Keywords: Internet of Things, Intrusion Detection System, Deep Learning, CNN, BiLSTM

1. Introduction

The Internet of Things (IoT) offers a new application that facilitates smart technology communication within the current networks. Nowadays, everything is being more and more linked to the Internet, and new devices are being developed much more quickly. Because these networked smart things can communicate with one another in unprotected environments, security solutions at different levels are required for the entire communication ecology. IoT technology offers unique features, such as different resource constraints and different network protocol requirements, in contrast to the existing

networks. The attacker uses several IoT system security flaws to perform various attacks. Given the rise in cyberattacks, it is critical to address the implications of the Internet of Things. [6]

1.1 Intrusion Detection System

The term Network security refers to protecting the data, systems, information, and networks from a wide variety of malicious attacks and threats that are luring on the internet. The attacks are aimed at accessing and even destroying information that is important to the organization. An enormous amount of confidential information is inevitably being exchanged over the untrustworthy internet, resulting in major security concerns [39, 26].

The detection of breaches in an IoT network is frequently done using IDS, including internal and external intrusions, as well as anomalies that could signal potential intrusions [21]. An IDS can be a hardware or software application that analyzes and monitors the flow of network traffic or data between users for suspicious activity over a network to prevent intrusion or any network attack. They analyze IoT network traffic or packets in real-time [17]. The underlying principles behind the IDS are 2 basic approaches to analyzing network traffic flow and detecting any intrusion- Anomaly-based IDS and Signature-based IDS.

The Signature-based IDS works much like an antivirus. It has the data of previously known threat patterns, and it detects threats or any intrusion by looking for a particular pattern and then compares it with the data that the IDS already has [12]. This model has a low rate of false alarms and a high detection rate. Attackers are creating new, unidentified attacks as networks and services grow, leaving the model vulnerable to these attacks. An intrusion detection system needs to be smart and efficient in identifying and stopping known and new assaults, including anomaly detection, to secure these network activities [5]. Attacks both known and new can be detected by Anomaly-based IDS, despite a high false alarm rate. This is also effective in detecting zero-day threats.

Monitoring and detection of any kind of IoT network threats are performed by the IDS at both the network and host level. There are 2 types of IDS:

Network Intrusion Detection System (NIDS) provides security to the whole network and all the devices that are connected to the network. It detects any type of malicious activities and IoT threats by scanning the network traffic that is gathered by different networking components like the switch, routers, and much more sophisticated systems like security information and event management (SIEM) [1]. They examine both normal and malicious traffic.

Host-based Intrusion Detection System (HIDS) is installed on the end user's computer to protect them and detect any kind of intrusion. It examines the end user device like the logs file to prevent and detect any malicious activity [20].

Computers and other devices can now learn from a dataset with little assistance from humans thanks to artificial intelligence (AI); IDS have benefited from this ability. AI encompasses machine learning (ML) and deep learning (DL), both of which were applied in the design and development of a successful intrusion detection system. Using manually collected attributes, a machine learning system classifies and detects network traffic. In contrast to machine learning, deep learning can increase and improve the model's detection accuracy. This is because, using its neural network, the deep learning system can extract features from the dataset and then perform classification and detection. [7]

1.2 Learning Techniques for IDS

Numerous machine-learning approaches have been proposed for detecting anomalies in IDS. Technological advancements and the expansion of the internet led to enormous amounts of data being

generated, data volumes are soaring, and information is becoming more complex, resulting in increasingly sophisticated attacks. Thereby making approaches that rely on machine learning not being able to cope with addressing growing security concerns and tackling security threats. Therefore, refrain from relying on the machine learning model to address these major security concerns. Several studies have demonstrated the effectiveness of deep learning techniques at reducing dimensionality and tackling classification problems. In the case of a DL-based IDS, data from previous network traffic records containing both malicious and normal traffic patterns is utilized to train a DL-based IDS [38].

The network traffic complexity can be reduced automatically by DL to discover correlations of data without human intervention. In addition to this, DL-based IDS is considerably more effective in detecting any kind of sophisticated attack pattern or even zero-day exploit, since it uses a large amount of data for training to create an effective detection model [13]. The intrusion detection domain has recently seen a surge in ground-breaking DL techniques.

1.3 Motivation

The intentional construction of the Deep Learning-based IDS makes it ideal for categorization and prediction. IoT networks will see encouraging outcomes from DL. The IoT systems generate a huge amount of data that may be used by learning techniques to make better and more informed decisions. The security system might be strengthened by incorporating intelligence using approaches based on learning. The development of an IDS employing DL techniques for detecting IoT attacks is the main goal of the research because the use of various DL models is becoming increasingly popular. The major goal is to create a deep learning-based IDS for IoT networks that can recognize insider server attacks. The UNSW-NB15 dataset with chosen features has been utilized to train the IDS model for this purpose. Figure 1, illustrates the design of the proposed research.

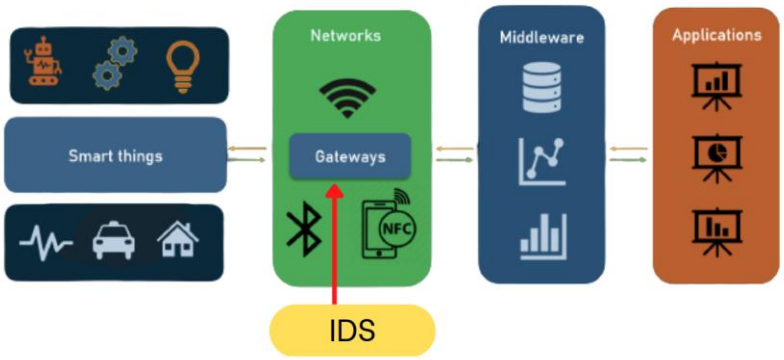

Figure 1. Broad design of the proposed research work

As seen in the figure above, it is equally conceivable that heterogeneous data being transported to a network from many smart applications, like smart airports, smart healthcare, smart energy, smart traffic management, etc., will be compromised by potential security threats. All application data should be protected because they are valuable. The key problem while handling massive sensor data in IoT-based smart applications is to identify any security breach or danger on data being sent to a cloud server using an IDS.

With the inspiration from everything discussed above, this study suggests a novel Bidirectional Long Short-Term Memory (BiLSTM)- convolutional neural network (CNN) hybrid IDS to foresee attacks. The suggested method has a high degree of precision and can reliably identify attacks. A recent, widely used, and well-known UNSW-NB 15 dataset is considered for the research to develop IDS for IoT networks. The model was trained over 175,341 records (training set) and tested over 82,332 records (testing set).

1.4 Contribution

The prime contributions of this research work are as follows:

- The study offers a thorough understanding of the recent work currently being done, concentrating especially on DL methods that have utilized the IoT datasets for IDS creation.
- The research describes the feature selection approach, data normalization, data cleaning, feature scaling, and feature encoding as a part of the pre-processing of data after performing extensive data analysis.
- Performance analysis with binary classification using IoT security datasets, which is, UNSW-NB15. The parameters used to analyze the performance of the proposed model are Precision, Sensitivity, F1-Score, Miscalculation rate, False Positive Rate, False Negative Rate, and Matthews Correlation Coefficient.
- With a great precision of 99.265%, the proposed intelligent BiLTSM-CNN IDS successfully detects attacks for IoT network datasets.
- A few recent state-of-the-art techniques are used to compare the performance of the proposed BiLTSM-CNN IDS over the UNSW-NB 15 dataset such as: 3-layered 1D CNN IDS model [8], 2-layered BiLSTM IDS model [42], MSCNN-LSTM [43], 1D CNN – Multi BiLSTM [32], Deep CNN-WDLSTM [11] and RNN-LSTM [9], RNN-LSTM [4] and CNN-LSTM [10] over precision, accuracy and F1 score and results shows that the hybrid model outperformed all.
- Deep learning-based hybrid BiLSTM- CNN IDS model for detecting attacks on IoT networks is run on CPU and GPU and proved to be a better solution for IoT for being robust and resilient, accurate, precise, lightweight, and efficient in terms of computation time.

The research paper is organized as follows. Section 2 gives the details of the literature survey and states the research gap. The specifics of the UNSW-NB15 dataset are described in Section 3. The proposed BiLSTM- CNN IDS architecture and algorithm, and asymptotic time complexity analysis are presented in Section 4. Additionally, this section also describes the pre-processing of the UNSW-NB15 dataset. Experimental setup, performance measures, and performance outcomes are presented in Section 5 along with a discussion. Also, the proposed intelligent BiLSTM-CNN hybrid model is compared with a few related IDS based on Deep Learning approaches for IoT networks. Lastly, Section 6 provides the conclusions and limitations with suggested future work.

2 Related Works

This section of the manuscript examines prior research that employed various approaches to develop IDS.

For network intrusion detection, a more effective convolutional neural network model has been created by authors [16] where the authors have proposed a 1D-CNN over the KDD'99 dataset. 2

layers of 1-D CNN with max-pooling layers are used to classify the normal and attack categories with the help of softmax classification method. The reported accuracy of the model was 99.83% and was compared consequently with SVM and DBN classifiers, outperforming them both.

Authors [8] proposed a deep-learning method for creating effective and adaptable IDS using 1D-CNN. The UNSW-NB15 dataset is deployed for training and testing the models where the dataset has been configured with training and testing records of 175341 and 82332 respectively. For dealing with the imbalanced dataset, a random oversampling technique has been used to have an equilibrium between the number of normal and attack records. The dataset is used on 1-layered, 2-layered, and 3-layered 1D-CNN models and a 1D-CNN + LSTM model. Subsequently, the 3-layered 1-D CNN along with 2 max pool layers attained the highest accuracy score of 91.17%, a precision score of 87.53%, a recall score of 96.17%, and an f1-score of 91.59%.

By benchmarking simple RNN, GRU, and LSTM models on the KDD'99 Dataset for multi-class classification, evaluated recurrent neural networks and their modifications for intrusion detection. On various combinations of LSTM layers and GRU layers, authors found that these models' accuracy ranged from 94.2% to 96.98% and 94.3% to 95.37%, respectively. The authors [37] reported 64.8% accuracy from a GRU network and 67.5% accuracy from an LSTM network on the multicategory UNSW-NB15 dataset.

Bi-LSTMs and an Attention layer are combined in the model, known as BAT [33], to filter away features that have little bearing on the model's outcome. The model has a BiLSTM, Attention, and a Fully Connected Dense Layer, among other convolutional layers. The NSL-KDD dataset, with its five classes- a normal category and four different sorts of attacks-has been used to test the model. On the multicategory analysis, the accuracy ranged from 82.97% to 84.24%.

Authors [34] worked on the implementation of the Hybrid CNN-LSTM model. They used the CICIDS 2017 dataset to analyze performance using parameters such as recall, accuracy, F-score, and sensitivity. They obtained an overall accuracy of 98.67% and compared their results to other Machine Learning models such as Random Forest and Logistic Regression, but the hybrid CNN-LSTM model outperformed them all.

The research has been done based on Bi-directional LSTM on the network behavior anomaly, where the author proposed a 2-layered Bi-LSTM model using a sigmoid activation function [42]. Binary cross entropy has been used as the loss function with a learning rate of 0.01, batch size of 10, and 10500 epochs. Adam’s optimizer has been utilized for the optimization of the model. Subsequently, an analysis is performed on the UNSW-NB15 dataset resulting in an average precision, recall, and f1-score of 93%, 89%, and 91% respectively.

A Hybrid Convolutional Recurrent Neural Network has been proposed where authors [15] implemented a Deep Learning Hybrid Model by merging a Convolutional Neural Network with a Recurrent Neural Network (HCRNN) for IDS. The CSE-CIC IDS2018 dataset is used, and the performance indicators are Precision, Recall, Accuracy, and F-1 score. When compared against other Machine Learning (ML) models, such as XGBoost (XGB), Logistic Regression (LR), and Decision Tree (DT), their Hybrid HCRNN model outperformed them all, attaining an accuracy of 97.75%.

To include the learning of temporal and spatial features of the data, research [32] suggested an IDS model based on deep learning techniques that combines the distinctive advantage of a Bi-directional LSTM and a Convolutional Neural Network. The model is trained and tested using the UNSW-NB15 and NSL-KDD datasets. The suggested model had a 1-D CNN layer, several Bi-LSTM layers, and layers for batch normalization and reshaping in between. The proposed model's binary classification results for the UNSW-NB15 dataset showed average detection rates, accuracy, and false positive rates

of 94.704%, 93.084%, and 7.70%, respectively, across various Stratified K-Fold Cross Validation ranging from k values between 2 and 10. Subsequently, the model has also been tested for multiclassification of the records, and it achieved the average detection rate, accuracy, and false positive rate for the UNSW-NB15 dataset as 92.506%, 82.084%, and 6.092% respectively.

Authors [43] have used the UNSW-NB15 dataset to develop a hybrid deep learning model comprised of a Long Short-Term Memory (LSTM) and Multiscale Convolutional Network. The model's performance metrics are False Negative Rate (FNR), Accuracy, and False Alarm Rate (FAR). They also evaluated the accuracy of other models such as Lenet-5, Multiscale CNN, and Hast to the hybrid model. With the Adam optimizer and ReLU activation function, the research attains 95.6% accuracy, 1.6% FNR, and 9.8% FAR. The hybrid model outperformed the other four models, which had an accuracy of 63.9% for Lenet-5, 91.4% for MSCNN, and 85.7% for Hast.

The authors developed a hybrid OCNN-HMLSTM based on deep learning [14] that combines an optimized Convolutional Neural Network with Hierarchical Multiscale LSTM. They worked on three separate datasets, UNSW-NB15, NSL-KDD, and ISCX-IDS, and achieved an accuracy of 96.34%, 90.67%, and 95.33%, respectively.

The research proposed by authors [4] is based on deep learning models like ANN, DNN, and RNN for intrusion detection systems for IoT networks. The UNSW-NB15 dataset is created using several distinct files, and binary classification is used to label the data. Instead of testing models individually for each file, the authors of this research have attempted to combine the entire dataset into a single file. After that, the dataset's attack families were employed as a new label to create a multi-classification labeled dataset. The pre-processed UNSW-NB15 dataset is improved for use in this study so that deep-learning models can handle it and find anomalous patterns.

Another research [28] demonstrated a unique deep learning approach, using a Bi-directional Long Short-Term Memory (BiLSTM) Recurrent Neural Network (RNN), for detecting attacks within the IoT network. Using UNSW-NB15, a multi-layer Deep Learning Neural Network is trained. The categorization of normal and attack behaviors on the IoT network into two categories is the main emphasis of this paper. Their proposed BiLSTM model has an attack detection accuracy of over 95%.

Authors proposed a deep learning-based Hybrid CNN-WDLSTM that combines a Convolutional Neural Network with a Weighted-drop LSTM [11]. They trained the model using the UNSW-NB15 dataset and obtained an accuracy of 97.1% for binary classification.

An intrusion detection approach based on Kullback Gaussian deep belief network (KG-DBN) has been introduced to lower the high false positive rates [39]. To make the pre-processing of the data simpler, the dataset is treated using probabilistic mass function (PMF) encoding and the Min-Max normalization approach. The UNSW-NB15 and NSL-KDD open datasets are used in the simulation studies. The accuracy of the suggested approach is 96.17% and 86.49%, for the UNSW-NB15 and NSL-KDD datasets respectively.

Layers of CNN and LSTM are stacked and a CNN-LSTM hybrid model is proposed by researchers [10]. The UNSW-NB15, CIC-IDS2017, and WSN-DS datasets (all of which included benign and attack records) are used to assess the model. While the detection rate and FAR results are positive, the model is not able to give a high detection rate for some types of threats, such as web attacks in CIC-IDS2017 and worms, backdoors, and analysis in UNSW-NB15.

Table 1. Summary of existing Intrusion Detection System using Deep Learning techniques

Research Work	Proposed Methods	Dataset	Performance Metrics
[16]	CNN model	KDD99	Accuracy (99.83%)
[42]	2 layered BiLSTM	UNSW-NB15	Precision (93%), recall (89%), and f1-score (91%)
[8]	1D CNN 3 layers	UNSW-NB15	Accuracy (91.17%), Precision (87.53%), Recall (96.17%), F1 score (91.59%)
[33]	BAT	NSL-KDD	Accuracy (84.24%)
[34]	DL-IDS	CICIDS2017	Accuracy (98.67%), F1 score (93.32%)
[32]	1D CNN and multiple layers of BiLTSM	UNSW-NB15	Average detection rate (94.704%), Accuracy (93.084%), and False positive rate (7.70%)
[43]	MSCNN-LSTM	UNSW-NB15	Accuracy (89.8%), False Alarm rate (47.4%), False Negative Rate (8.6%)
[11]	Deep CNN-WDLSTM	UNSW-NB15	Precision (97%), Recall (97%), F1-score (97%) Accuracy (97.17%)
[35]	IDS based on improved DBN	NSL-KDD and UNSW-NB15	96.17% and 86.49% accuracy for NSL-KDD and UNSW-NB15 respectively
[19]	Multi CNN Fusion	NSL-KDD	Accuracy (86.95%)
[9]	RNN-LSTM	UNSW-NB15	Accuracy (87%)
[4]	RNN-LSTM	UNSW-NB15	Accuracy (85.42%)
[10]	CNN-LSTM	UNSW-NB15	Precision (94.69%), F1 Score (94.77%), Accuracy (93.95%)

For Internet of Things-based applications, such as healthcare, machine learning, and metaheuristic algorithms based on a hybrid intelligent intrusion detection system (HIIDS) over the NSL-KDD dataset is presented [29]. To reduce computation costs, also to attain precise classification of the normal and attack classes based on chosen features, supervised learning techniques like Decision Tree (DT) and Known Nearest Neighbor (KNN) are used. The optimum feature selection is done using metaheuristic algorithms like Particle Swarm Optimization (PSO), Differential Evaluation (DE), and Genetic Algorithm (GA). Additionally, a hybrid approach to feature selection and classification has been provided..

It can be seen in Table 1 that several works have been done on Intrusion detection in IoT networks over the past few years. Most current work emphasizes binary classification over different data sets with good results. However, the recent state-of-the-art work fails to achieve higher precision while maintaining a high accuracy and F1 score. Hence, the research focuses on binary classification using basic Deep learning models like CNN and BiLSTM and eventually designing a hybrid BiLSTM-CNN model to detect an attack over the UNSW-NB15 dataset with remarkable precision while giving the best accuracy and F1 score.

3 Dataset

For this study, the UNSW-NB15 data set is used because it only contains the two classes "attack" and "normal," which cover current attack patterns, and contain current normal traffic patterns. The initial network packet for the UNSW-NB15 dataset has been created by the network-wide laboratory of the Australian Centre for Cyber Security (ACCS) using the IXIA PerfectStorm tool [22]. This technology is mostly used to produce current synthetic attack actions as well as genuine, everyday modern behavior. 100GB of unprocessed traffic is captured using the tcpdump utility. This dataset [23] has nine different types of attacks. Table 2 displays the record distribution for both the training set and the test set.

Table 2. Traffic Distribution of UNSW-NB15 dataset.

Traffic	No of records in Training set	No of records in Testing set
Normal	56,000	37,000
Generic	40,000	18,871
Exploits	33,393	11,132
Fuzzers	18,184	6,062
Dos	12,264	4,089
Reconnaissance	10,491	3,496
Analysis	2,000	677
Backdoor	1,746	583
Shellcode	1,133	378
Worms	130	44
Total	**175,341**	**82,332**

The training set file (UNSW-NB15 training-set.csv) has a total of 175,341 records, whereas the testing set file (UNSW-NB15 testing-set.csv) has a total of 82,332 records. The training set's distribution of normal and attack records is shown in Figure 2, and the distribution of attacks in the training set is shown in Figure 3.

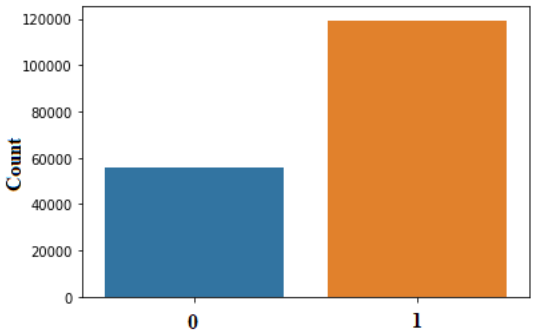

Figure 2. The training set's distribution of normal and attack records

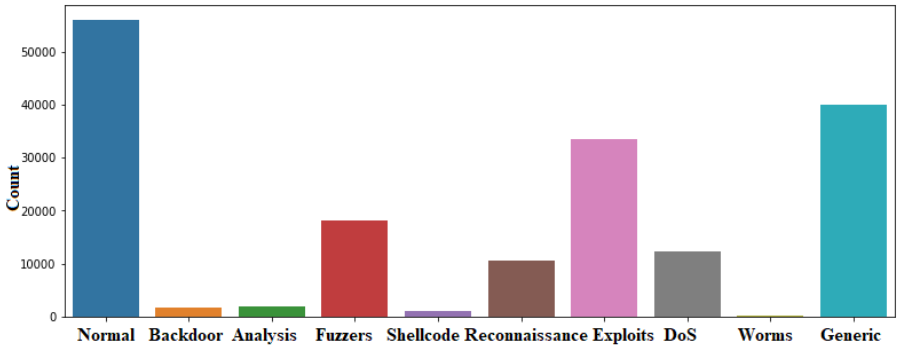

Figure 3. Distribution of nine types of attacks in the training set

4 Proposed Methodology

To improve the optimal results for the classification of the attacks, the usage of multiple DL models has been facilitated in this study. The DL models are described in more detail in the part that follows. It also shows how the BiLSTM-CNN architecture helped attain the best results.

4.1Proposed DL Models

The study in the field of deep learning multi-layered hierarchical networks concentrated on deep networks with classification training. Depending on how architecture and technologies are implemented, the network's deep IDS can be classified. Initially, the BiLSTM model is used, which is followed by the CNN model and ultimately resulted in the development of a hybrid BiLSTM-CNN model architecture.

4.2BiLSTM Model

The Long Short-Term Memory (LSTM) model describes how information spreads throughout LSTM cells. To store information in each cell, incoming data is passed through several gates that remove unnecessary information before being retrieved and added to the cell state [24]. An LSTM

modification known as the Bidirectional LSTM (BiLSTM) employs a duplicate LSTM that receives an input that is a reversed copy of the original input. In some situations, the BiLSTM can yield a noticeable performance improvement, but at the expense of twice the necessary trainable parameters for every given hidden dimension size [27]. To produce an output, the Bidirectional Recurrent Neural Network (BRNN) combines two hidden layers of the recurrent architecture in the opposite direction. [30]. The versatility of the recurrent architecture's input data is increased by its bidirectional characteristic. Additionally, the recurrent bidirectional network does not need the input data to be fixed before training and increases the reachability of future state inputs to the current state. [28].

Figure 4 illustrates the BiLSTM model's architecture employed in this study. It consists of two BiLSTM layers, with a max pooling layer following the first layer and a global max pooling layer following the second. The vector is flattened as a result of this layer's output, which is then sent to a fully connected layer and finally to an output layer.

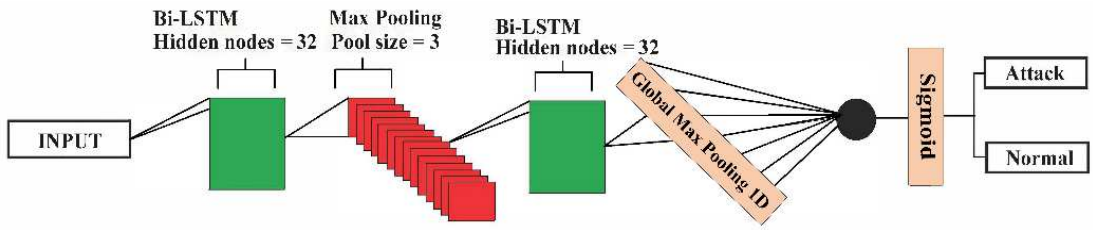

Figure 4. The architecture of the BiLSTM Model

The BiLSTM model's first LSTM layer, which uses 32 hidden nodes, is followed by a layer with a 1-dimensional maximum pooling. The prior layer's (max pool layer) output is used as an input in the following layer of BiLSTM. To the global max pooling layer the output of the BiLSTM layer is then passed to flatten the vector size before being passed to a fully connected layer with a sigmoid as an activation function to determine the input class required to start an action to suppress the network traffic flow attack, whether it be a regular network flow or an anomaly.

4.3CNN Model

Convolutional layers are used in CNN, a deep learning architecture, to map the features of the input data. By using various filter sizes, the layers are organized into various feature mappings. Based on the feature mapping results, CNN can learn more about the input data [2]. The convolutional layer is typically followed by a pooling layer to provide accurate output dimensions even with various filters. By lowering the output dimensions while keeping the crucial information, the pooling layer can minimize the computational strain [40].

Figure 5 illustrates the CNN model's architecture employed in this study. It comprises two convolutional layers, the first of which is followed by a max pooling layer and the second by a global max pooling layer. This layer's output flattens the tensor and is then supplied to a fully connected layer, which is followed by an output layer. Both convolutional layers' filters are of size 7.

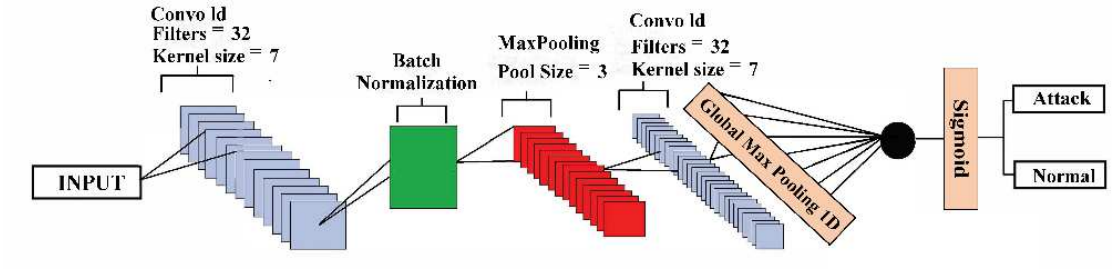

Figure 5. The architecture of the CNN Model

The CNN model's initial convolutional layer has 32 convolution filters and is put through batch normalization, max pooling layer, and three as stride value. To minimize the amount of training epochs, the batch normalization layer is seen to be a crucial component of CNN. As a result, the CNN's learning process becomes stable [3]. At this point, the output is 11 x 32 in size. The output of the preceding max pool layer is fed into the subsequent convolution layer. The tensor size produced in this convolutional layer is 5 x 32. The output is thereafter passed to a layer that is fully connected with the sigmoid activation function to find out which input class is necessary to start an operation that suppresses network traffic, whether it be a typical network flow or an anomaly. The output is thereafter passed to a global max pooling layer that flattens the tensor size to 32.

4.4Proposed Hybrid BiLSTM-CNN Model

As seen in Figure 6, a hybrid deep neural network model is created by combining homogenous neural networks. By altering the initialization of the network's weights and the input characteristics, an ensemble of different classifiers is created.

This study developed a hybrid deep learning model for effective network intrusion detection. CNN plays a significant role in the IDS by extracting pertinent features from large amounts of data while the BiLSTM preserves the long-term relationships among the obtained features to prevent overfitting on recurrent connections [32]. The implemented CNN and BiLSTM architectures explained and demonstrated above are consequently compared with the proposed hybrid model, confirming its satisfactory performance.

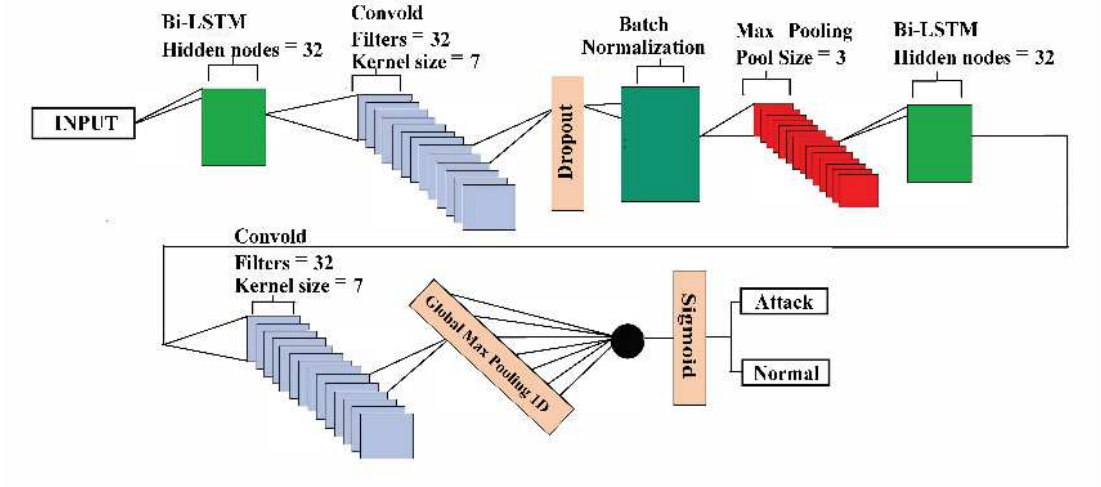

Figure 6. The architecture of the hybrid BiLSTM-CNN Model

The proposed model has multiple BiLSTM and CNN layers. The input is first reshaped in this architecture before being supplied to the BiLSTM layer. This layer then generates the output and feeds it as input to a 1-dimensional convolutional layer. This convolutional layer outputs a tensor which is followed by a dropout layer and then batch normalization and then after and max pooling layer. The dropout parameter in the proposed model is set to 0.1.

The output tensor of the preceding max pool layer is then input to the second BiLSTM layer. Similar to how the convolutional layer output is delivered, the output of this BiLSTM layer is then passed to a global max pooling layer, which flattens the vector size. When a network traffic flow intrusion occurs, whether it be a regular network flow or an anomaly, the output of it is then supplied to a layer that is fully connected with sigmoid as the activation function to identify the input class required to start an action to suppress it.

Due to the BiLSTM recurrent bidirectional architecture, the BiLSTM and CNN layers in this architecture are accountable for learning the temporal and spatial relationship respectively between the network flow and adjusting the temporal and spatial dynamics [18]. These layers also provide the framework for network flow feature extraction.

4.5Algorithm and Time Complexity of the BiLSTM-CNN Model

Algorithm: Binary classification Pseudocode for IDS

Require: Epoch = 30, Batch size=1024, Input shape($I_{0,d}$) = (39,1)
Ensure: Precision, Sensitivity, F1 score, Accuracy, Miscalculation rate, False Negative Rate, False positive Rate, Matthew's Correlation Coefficient (for 2 class)
1. Sequential model definition
2. **for** Epochs = 1 to 30 **do**
3. Evaluate BiLSTM (BiLSTM1) operations with 32 hidden nodes using the following equations
   a. Forget gate

$$f_t = \sigma\left(w_{f_h} * [h_{t-1}], w_{f_x} * [x_t], b_f\right) \quad (1)$$

b. Input gate

$$i_t = \sigma(w_{i_h} * [h_{t-1}], w_{i_x} * [x_t], b_i) \quad (2)$$

$$\tilde{c}_t = tanh\left(w_{c_h} * [h_{t-1}], w_{c_x} * [x_t], b_c\right) \quad (3)$$

$$c_t = f_t * \tilde{c}_t + i_t * \tilde{c}_t \quad (4)$$

c. output gate

$$o_t = \sigma\left(w_{o_h} * [h_{t-1}], w_{o_x} * [x_t], b_o\right) \quad (5)$$

$$h_t = o_t * \tanh(c_t) \quad (6)$$

4. Taking filters f and kernel size k compute 1-D Convolution (CL1) operation using below equation

$$CL1_{i,j}^{f} = \sigma\left(\sum_{p=0}^{k}\sum_{q=0}^{k}((\mathrm{I}_{0,d})_{i+m,j+n} * W_{p,q}^{f} + B^{f}\right) \quad (7)$$

Where (i,j) are indices of output, (p,q) indices of f^{th} filters, and weight Wand bias B

5. Set the drop out value to 0.1 to enable the model to learn more useful characteristics also avoiding overfitting problem
6. Perform batch normalization to prevent and regulate overfitting, making the model much for efficient.
7. With pool size =3, Use below equation to extract the most activated features with 1-D maxpooling (MPL1).

$$MPL1_{i,j}^{f} = MaxPool(CL1_{i,j}^{f}) \quad (8)$$

8. Compute BiLSTM (BiLSTM2) operation as step 3.
9. Compute 1-D Convolution (CL2) as step 4
10. Perform Global Max Pooling
11. Use fully connected dense layer with one neuron and sigmoid activation function to detect and categorize attacks.
12. Using Adam optimizer and binary cross-entropy as loss function, compile the model
13. Fit the model on training set (175341 records), where validation set is 0.2
14. Replicate steps 3 to 14 for each Epochs (1 to 30)
15. **end for**
16. For testing set evaluate and predict the model
17. Generate the confusion matrix and classification result
18. Plot graphs for training and validation accuracy and loss on CPU and GPU over 30 epochs
19. Print performance matrix

The proposed BilSTM-CNN model IDS's asymptotic time complexity is examined in this section. The time complexity of LSTM model is O(4*Ih*+ $4h^2$+ 3*h* + *hk),* where I is the number of inputs, h is the node count in the hidden layer and k is the number of outputs. For LSTM model time complexity is 2 * O(4*Ih*+ $4h^2$+ 3*h* + *hk).* Therefore, the time complexity of two BiLSTM layers is 4 * O(4*Ih*+ $4h^2$+ 3*h* + *hk).*

The addition of row-wise dot product of two matrices, one of which is the kernel with size k and the other which is the input with size Ic, results in the 1D convolution. Since d is the number of channels, for a single convolution layer the computational complexity is O($I_c kd$) [36]. The complexity of

convolution is O($I_c kdf$) with f being the number of filters. Therefore, 2*O($I_c kdf$) is the temporal complexity when using two 1D-convolution layers.

There are no trainable parameters in the max-pooling layer. Therefore, it will facilitate a decrease in computational overhead. With the help of the kernel size, k, from the input data given, I_m the maximum weight is selected. It is fundamentally an unsorted, one-dimensional array. The temporal complexity is therefore 2*O(I_m) with two 1D-max-pooling layers.

The data from the max-pooling layer is flattened before being input into the output layer which is a fully connected dense layer, where it is processed to detect various attacks. Each neuron, I_{fc}, is connected to the Sigmoid activation neuron, I_s, by this layer. Therefore, the temporal complexity of one dense layer that is fully connected is O($I_{fc} I_s$).

The proposed BiLSTM-CNN IDS's asymptotic time complexity is:

$$\{4 * O(4Ih + 4h^2 + 3h + hk) + 2 * O(I_c kdf) + 2 * O(I_m) + O(I_{fc} I_s)\} \quad (9)$$

4.6 Proposed Model Flowchart with Step-by-Step Phases

The five stages of the proposed model's flow are as follows: data preprocessing and feature extraction; normalization; splitting the dataset into training and validation sets; building DL models; testing them on the UNSW-NB15 testing set and subsequently evaluating the IDS based on the accuracy and various performance metrics. Figure 7 illustrates the flow of the proposed model.

Data Pre-processing and Feature Extraction. There were some unnecessary and redundant values in the dataset that needed to be deleted. As a result, 4 attributes—id, proto, service, and state—are removed from the dataset. There are a total 41 columns in the improved dataset. The dataset contains values of the object datatype (labeled data), which are not directly usable in the raw format. The data is subsequently labelled using Label Encoding, a successful method for transforming labels into a numeric form that can be ingested into deep learning models [31]. The dataset is likewise verified for any pre-existing null values; as none were present, the attributes are normalized.

Normalization. The famous min-max normalization technique is deployed in order to build consistent neural networks. This kind of normalization helps to lessen any bias brought on by the preference of some dominant qualities in the dataset over others and is a reliable and robust strategy for keeping all data linkages [25].

Dataset Splitting. The dataset, as discussed above was split by the author of the data into training-set and testing-set. The training-set had 175,341 values, while the testing-set contained 82,332 values. However, to avoid the model's overfitting, the training-set is further separated into training and validation sets with an 80:20 ratio. The model's performance could be assessed on the fresh data due to the validation set that is provided with the initial test against unexplored data. Additionally, insightful information from the validation is helpful when optimizing the hyperparameters. [41]

Deep-Intrusion Detection System. The proposed IDS for IoT network is based on DL techniques with 3 types of DL models that are: CNN, BiLSTM and hybrid BiLSTM-CNN.

Testing the Model and Comparing Results. The models are subsequently tested using the 175,341 testing-set data, after which they are compared on the basis of precision, accuracy, sensitivity, F1-Score, Miscalculation Rate, False Positive Rate, False Negative Rate, and Matthew's Correlation Coefficient.

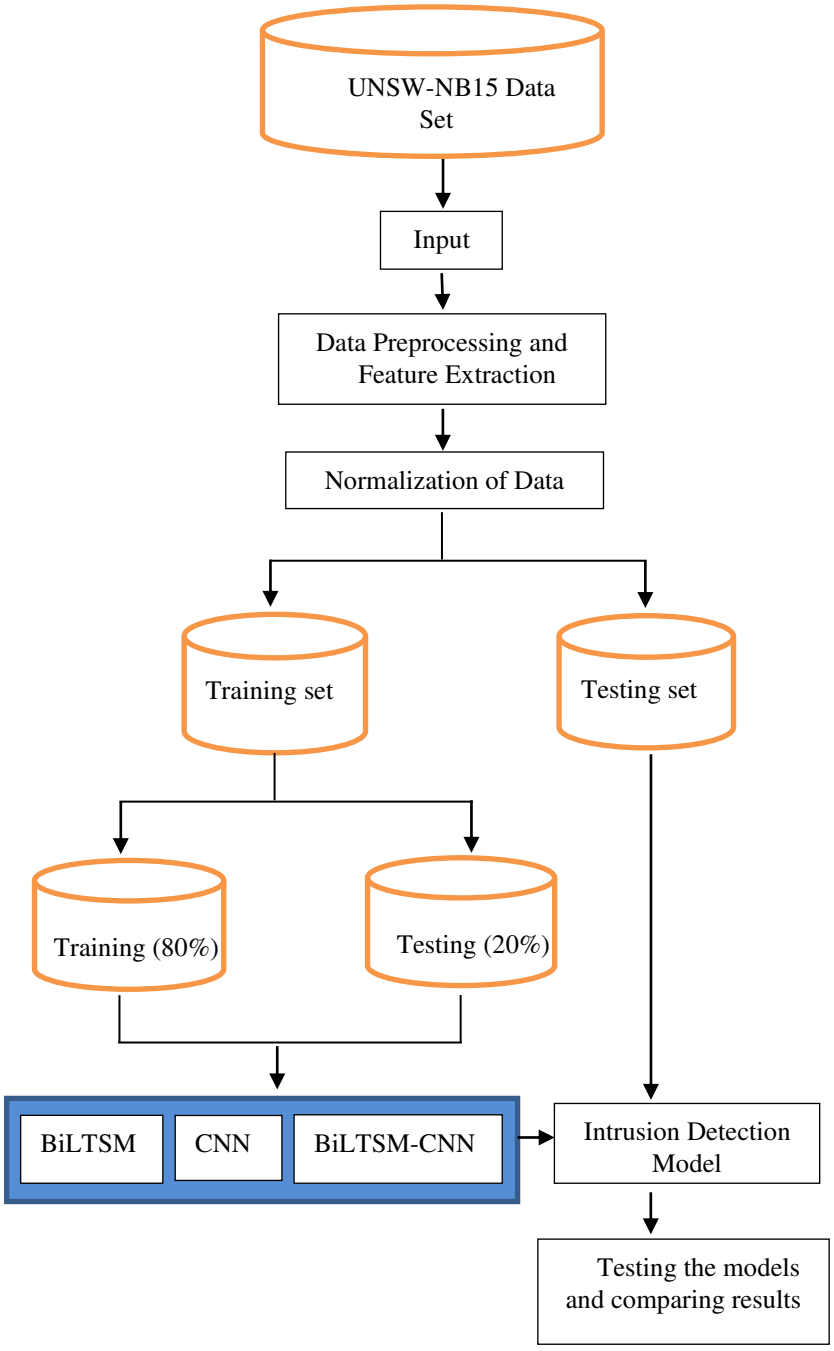

Figure 7. Flowchart of the proposed model

5 Experimental Results

The Jupyter Notebook environment and Python 3.10.9 are used to create the DL models. The proposed BiLSTM-CNN model is trained using Adam's propagation as an optimizer and binary cross entropy as the empirical loss function for 30 epochs with a batch size of 1024. After 30 epochs, it has been found that the validation loss grew, which decreased the validation accuracy. The epochs are fixed to 30 as a result to avoid the model from overfitting. RMSprop and Adam's propagation, two distinct optimizers, were used to evaluate the model. For optimization Adam's propagation is employed for the proposed model since it generated better outcomes than RMSprop in terms of loss and accuracy.

The CNN, BiLSTM, and the hybrid BiLSTM-CNN models are run on two different processors: GPU and CPU. The model is trained and tested using an NVIDIA Quadro P1000 GPU running on CUDA version 11.3 and CUDNN version 8.2, and the CPU used is an Intel(R) Core (TM) i7-1065G7 CPU running at 1.30GHz.

For the proposed BiLSTM- CNN model, the training and validation accuracy and loss values are noted. The accuracy and loss values for the 1, 5, 10, 15, 25, and 30 epochs on the GPU and CPU, respectively, are shown in Tables 3 and 4.

Table 3. Accuracy and loss attained by the BiLSTM-CNN Model at various epochs run on the GPU

No of Epochs	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
1	0.8266	0.3590	0.3194	0.8080
5	0.9423	0.1414	0.3436	2.2380
10	0.9507	0.1200	0.8814	0.3013
15	0.9541	0.1106	0.8954	0.2620
20	0.9568	0.1047	0.8935	0.1090
25	0.9601	0.0009	0.8954	0.1071
30	**0.9751**	**0.0955**	**0.9134**	**0.1054**

After 30 epochs of training on the GPU, as mentioned in the Table 3, the model achieved accuracy of 97.51% and 91.04% on the training and validation sets, respectively. The loss produced after 30 epochs when the proposed model is run of GPU on the training and validation sets, are 0.0955 and 0.1054, respectively.

Table 4. Accuracy and loss attained by the BiLSTM-CNN Model at various epochs run on the CPU

No of Epochs	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
1	0.8330	0.3487	0.5364	0.6709
5	0.9425	0.1408	0.9056	0.2365
10	0.9504	0.1210	0.8874	0.2916
15	0.9547	0.1105	0.8655	0.3505

20	0.9573	0.1024	0.8913	0.2146
25	0.9589	0.0994	0.8890	0.2011
30	**0.9738**	**0.0948**	**0.9089**	**0.1544**

Similarly, after 30 epochs of training on the CPU, the model achieves an accuracy of 97.38% and 90.89% on the training and validation sets, respectively, as shown in Table 4. The loss produced after 30 epochs when the proposed model is run on the CPU on the training and validation sets are 0.0948 and 0.1544, respectively.

The graphs shown in Figure 8 and Figure 10 depict the training and validation accuracy, respectively, while the graphs shown in Figure 9 and Figure 11 depict the training and validation loss for 30 epochs. The graphs shown in Figures 8 and 9 depict the proposed model running on the GPU, while the graphs shown in Figures 10 and 11 predict it running on the CPU. Figures 8 and 10 illustrate the correlation between epochs and accuracy, while Figures 9 and 11 illustrate the correlation between epochs and loss for the proposed work over 30 epochs for the training and validation datasets.

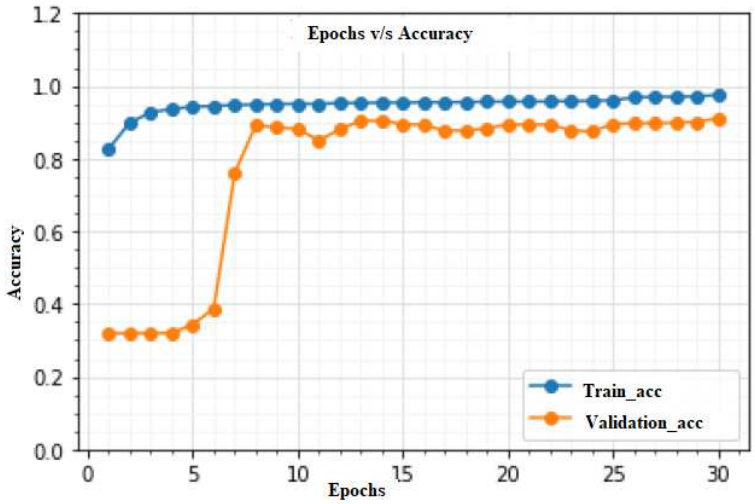

Figure 8. Train and validation accuracy of BiLSTM-CNN model run on GPU

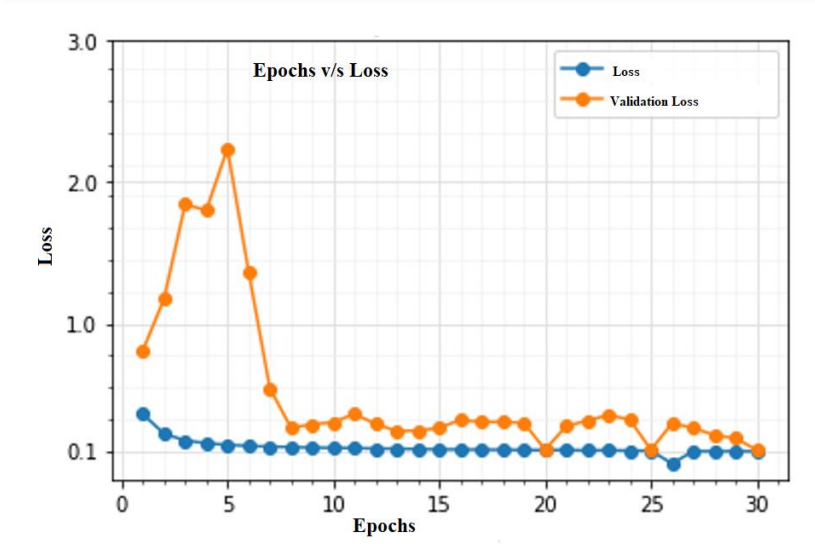

Figure 9. Train and validation loss of BiLSTM-CNN model run on GPU

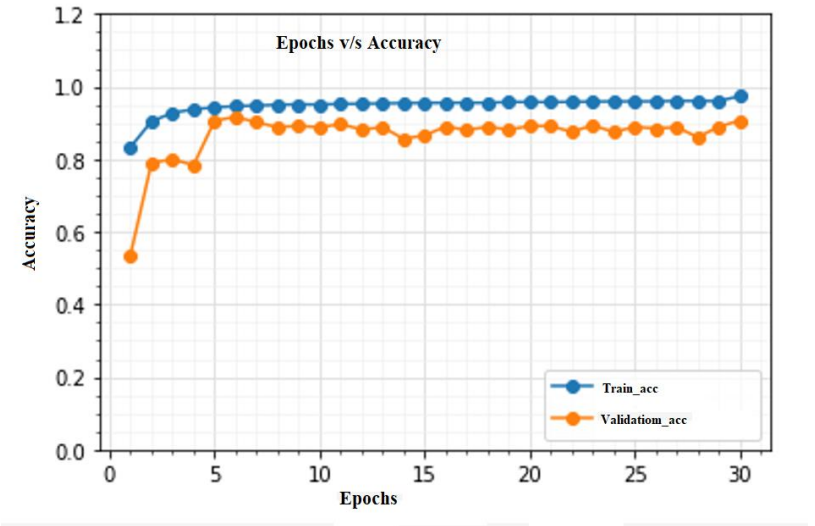

Figure 10. Train and validation accuracy of BiLSTM-CNN model run on CPU

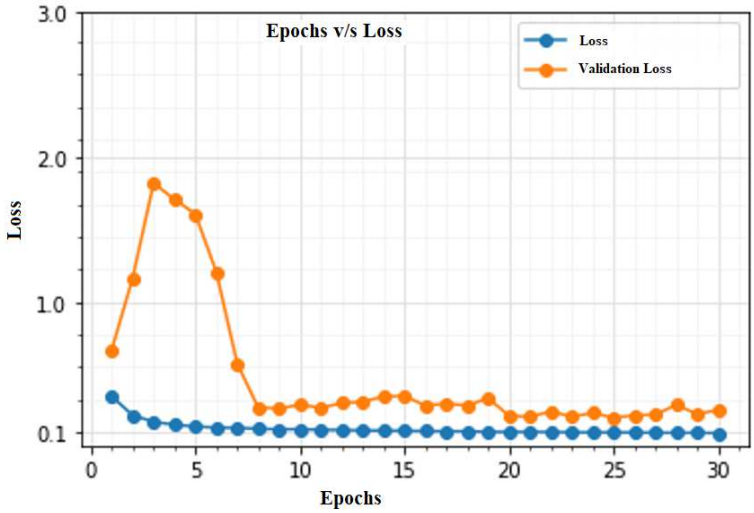

Figure 11. Train and validation loss of BiLSTM-CNN model run on CPU

The accuracy and loss convergence of the models on training set, validation set, and testing set are recorded on the GPU and CPU, as seen in Table 5 and 6, respectively. When the models are run on GPU, the BiLSTM model on the training, validation, and testing set attains the accuracy of 92.51%, 88.16%, and 87.15% respectively, while keeping a loss convergence of 0.1795, 0.2527, and 0.057. Similarly, the CNN model attains 95.82%, 89.12%, and 88.22% respectively, keeping a loss convergence of 0.1040, 0.2869, and 0.057. The hybrid BiLSTM-CNN model obtains the highest accuracy on all the datasets, with 97.51% accuracy and 0.0948 loss convergence on training set, 90.89% of accuracy and 0.1054 loss convergence on the validation set, and 91.14% accuracy and 0.296 loss convergence on the testing set.

Table 5. Accuracy and loss convergence of models on datasets run on the GPU

Model	Accuracy (Training Set) **± 0.015%**	Loss Convergence (Training Set)	Accuracy (Validation Set) **± 0.015%**	Loss Convergence (Validation Set)	Accuracy (Testing Set) **± 0.015%**	Loss Convergence (Testing Set)
BiLSTM	0.9251	0.1795	0.8816	0.2527	87.15	0.057
CNN	0.9582	0.1040	0.8912	0.2869	88.22	0.112
Proposed BiLSTM -CNN	**0.9751**	**0.0948**	**0.9089**	**0.1054**	**91.14**	**0.296**

Table 6. Accuracy and loss convergence of models on datasets run on the CPU

Model	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss

	(Training Set) **± 0.015%**	Convergence (Training Set)	(Validation Set) **± 0.015%**	Convergence (Validation Set)	(Testing Set) **± 0.015%**	Convergence (Testing Set)
BiLSTM	0.9279	0.1731	0.8711	0.2705	86.88	0.104
CNN	0.9584	0.1033	0.8847	0.3191	87.46	0.260
Proposed BiLSTM -CNN	**0.9738**	**0.0955**	**0.9089**	**0.1544**	**90.63**	**0.208**

Figures 12 and 13 show the graphical representations of the accuracies reached by the three models on training, validation, and testing datasets performed on the GPU and CPU, respectively.

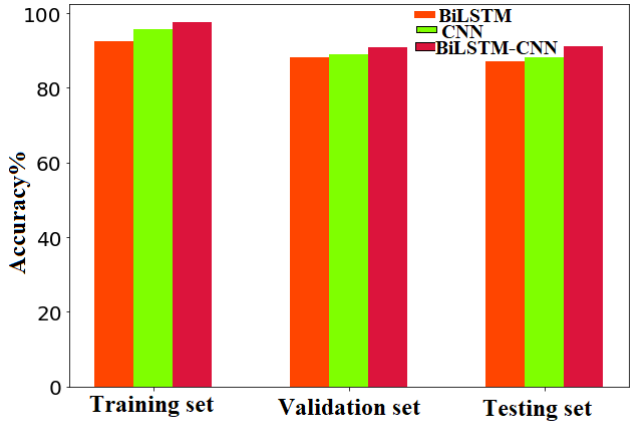

Figure 12. Accuracies of the models on datasets run on the GPU

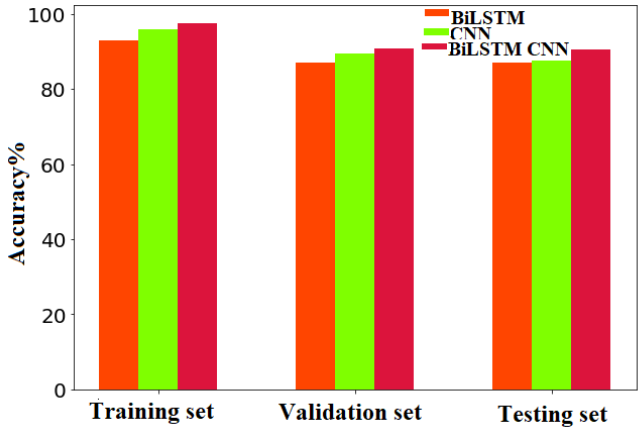

Figure 13. Accuracies of the models on datasets run on the CPU

The BiLSTM model on the training, validation, and testing set achieves an accuracy of 92.79%, 87.11%, and 86.88% correspondingly when the models are run on a CPU, while maintaining a loss

convergence of 0.1731, 0.2705, and 0.104. Similarly, the CNN model attains 95.84%, 88.47%, and 87.46% respectively, keeping a loss convergence of 0.1033, 0.3191, and 0.260. With 97.38% accuracy and 0.0955 loss convergence on the training set, 90.89% accuracy and 0.1554 loss convergence on the validation set, and 90.63% accuracy and 0.208 loss convergence on the testing set, the hybrid BiLSTM-CNN model demonstrates its great performance and resilience.

Now for computing and comparing the results obtained from the confusion matrices, 7 criteria are used as shown in Table 7. These are Precision, Sensitivity, F1-Score, Miscalculation rate, False Positive Rate, False Negative Rate, and Matthews Correlation Coefficient.

Table 7. Metrics for evaluating the performance of the models

Measure	Derivations
Precision (PPV)	TP / (TP + FP)
Sensitivity (TPR)	TP / (TP + FN)
F1-Score (F1)	2TP / (2TP + FP + FN)
Miscalculation Rate (MR)	(FP + FN) / (TP + TN + FP + FN)
False Positive Rate (FPR)	FP / (FP + TN)
False Negative Rate (FNR)	FN / (FN + TP)
Matthews Correlation Coefficient (MCC)	TP*TN – FP*FN / sqrt ((TP + FP) *(TP + FN) *(TN++ FP)*(TN + FN))

Where,
TP = True Positive,
FP = False Positive
TN= True Negative
FN= False Negative

Table 8 represents the comparison for the binary classification on the testing-set with regards to the three DL classification models based on the discoursed performance metrics.

Table 8. Performance Matrix

Model	PPV (%)	TPR (%)	F1 (%)	MR (%)	FPR (%)	FNR (%)	MCC (%)
BiLSTM	98.123	82.294	89.514	13.124	3.356	17.707	74.358
CNN	98.167	83.697	90.356	12.161	3.333	16.304	75.923
Proposed BiLSTM-CNN	**99.265**	**87.641**	**93.091**	**8.857**	**1.384**	**12.360**	**82.120**

The proposed intelligent hybrid BiLSTM-CNN model outperforms the other deep learning BiLSTM and CNN model for IDS in all aspects, as seen in Table 8. Since the BiLSTM model predicts more false positives, its PPV and FPR suffer. Furthermore, because it predicts large false negative values, the achieved TPR is poor but FNR is substantially greater. As a result of the large number of false positives and false negatives, the F1 and MCC are both low, while the MR is much higher. The CNN model performs slightly better than the preceding BiLSTM model as it predicts a fewer number of

false positive and false negatives, due to which it achieved a higher PPV, TPR, F1, and MCC of 98.167%, 83.697%, 90.956% respectively, while maintain a low FPR, FNR and MR of 3.333%, 16.304%, and 12.161% respectively. Consequently, when the hybrid BiLSTM-CNN model is tested, the proposed model performs significantly better than the previous two models, as it predicts a high number of true positive and true negatives, while keeping the false positive and negatives substantially low. Therefore, the proposed hybrid model achieves the best results obtaining a PPV, TPR, F1, MR, FPR, FNR, and MCC of 99.625%, 87.641%, 93.091%, 8.857%, 1.384%, 12.360%, and 82.120% respectively.

As exhibited in Table 9, the proposed model's precision, F1 score and accuracy on the UNSW-NB15 dataset is also compared to earlier research based on deep learning IDS. Due to the proposed model's substantial performance in keeping false positives significantly low while identifying a high number of true positives, the set of precision, accuracy and F1 score are the greatest when compared to previous research works.

Table 9. Comparison of the Proposed Model with Previous Work

Research	Model	PPV (%)	F1 Score	Accuracy %
[8]	3-layered 1D CNN	80.33	88.46	85.86
[42]	2-layered BiLSTM	93.00	91.00	-
[43]	MSCNN-LSTM	-	-	89.8
[32]	1D CNN – Multi BiLSTM	-	-	93.084
[11]	Deep CNN-WDLSTM	97.00	97	97.17
[9]	RNN-LSTM	-	-	87.00
[4]	RNN-LSTM	**-**	**-**	85.42
[10]	CNN-LSTM	94.69	94.77	93.95
Proposed Research	**Proposed BiLSTM-CNN**	**99.265**	**93.091**	**97.51**

6 Conclusion and Future Work

This study presents deep learning models based on BiLSTM, CNN, and their fusion to construct a hybrid BiLSTM-CNN architecture, a new collaborative IDS for detecting intrusive behaviors in IoT network. A hybrid intrusion detection system model is developed in the study by utilizing the advantages of both the Bi-LSTM and CNN ability to extract temporal and spatial features, respectively. To improve the model's performance, dropout layers, batch normalization and standardization are also implemented. This study involves pre-processing and normalizing UNSW-NB15 so that it could be used with deep learning models to identify anomalous patterns. Initially, the behavior of the CNN, BiLSTM, and BiLSTM-CNN models are evaluated based on several performance metrics. The BiLSTM-CNN hybrid model offers the highest detection rate and accuracy, according to the results. Hence, the proposed hybrid model is assessed further. The precision, F1 score and accuracy are used to evaluate the proposed model's robustness. The findings reveal that the proposed hybrid model surpasses the existing hybrid DL models in all aspects. It attains a PPV, TPR, F1, MR, FPR, FNR, and MCC of 99.625%, 87.641%, 93.091%, 8.857%, 1.384%, 12.360%, and 82.120% respectively. The proposed hybrid model not just has a high detection rate but also a low rate

of false alarms. Further, the models are run on GPU and CPU to demonstrate their effectiveness and to compare real-world IoT network communication behavior. Consequently, the efficiency of the proposed technique is validated.

In the future, this research can be extended by deploying the framework in a real-world setting and providing further findings and explanations. Future work will refine the suggested method even further so that it can be applied to additional publicly available intrusion detection datasets and achieve higher accuracy and can perform multiclass classification of attacks. The model presented here can be employed in applications that speed up the study and development of system architectures. It can reduce the time required to validate new types of DL models and IDSs. Other deep learning networks and their hybrids can be investigated in future studies to improve attack detection performance and be deployed in real Internet of Things networks.

The positive results of this work will be leveraged and improved upon by data scientists and researchers in order to spur the advancement of highly accurate yet practical DL solutions for detecting malicious attacks from reliable and resilient system designs and to formulate immense innovations in the domain of IoT network security.

References

1. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J. and Ahmad, F. (2021): Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1), p.e 4150.
2. Albawi, S., Mohammed, T.A. and Al-Zawi, S. (2017): Understanding of a convolutional neural network. International conference on engineering and technology (ICET) (pp. 1-6). Ieee.
3. Ali, Abbas M., Kayhan Ghafoor, Aos Mulahuwaish, and Halgurd Maghdid. (2022): COVID-19 pneumonia level detection using deep learning algorithm and transfer learning." Evolutionary Intelligence. 1-12.
4. Aleesa, Ahmed, M. O. H. A. M. M. E. D. Younis, Ahmed A. Mohammed, and N. Sahar. (2021): Deep-intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques. Journal of Engineering Science and Technology 16, no. 711-727
5. Alhakami, W., ALharbi, A., Bourouis, S., Alroobaea, R. and Bouguila, N. (2019): Network anomaly intrusion detection using a nonparametric Bayesian approach and feature selection. *IEEE Access*, *7*, pp.52181-52190.
6. Aljanabi, Mohammad, Mohd Arfian Ismail, and Ahmed Hussein Ali. (2021): Intrusion detection systems, issues, challenges, and needs. International Journal of Computational Intelligence Systems 14, no. 560-571.
7. Alkhawaldeh, Rami S., Bilal Al-Ahmad, Amel Ksibi, Nazeeh Ghatasheh, Evon M. Abu-Taieh, Ghadah Aldehim, Manel Ayadi, and Samar M. Alkhawaldeh (2023): Convolution Neural Network Bidirectional Long Short-Term Memory for Heartbeat

Arrhythmia Classification. International Journal of Computational Intelligence Systems 16, no. 197.

8. Azizjon, M., Jumabek, A. and Kim, W. (2020): 1D CNN based network intrusion detection with normalization on imbalanced data. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* (pp. 218-224). IEEE.
9. Guizani, N. and Ghafoor, A. (2020): A network function virtualization system for detecting malware in large IoT based networks. IEEE Journal on Selected Areas in Communications, 38(6), pp.1218-1228.
10. Halbouni, Asmaa, Teddy Surya Gunawan, Mohamed Hadi Habaebi, Murad Halbouni, Mira Kartiwi, and Robiah Ahmad. (2022): CNN-LSTM: hybrid deep neural network for network intrusion detection system. IEEE Access 10.
11. Hassan, M.M., Gumaei, A., Alsanad, A., Alrubaian, M. and Fortino, G. (2020): A hybrid deep learning model for efficient intrusion detection in big data environment. Information Sciences, 513, pp.386-396.
12. Ioulianou, P., Vasilakis, V., Moscholios, I. and Logothetis, M. (2018): A signature-based intrusion detection system for the internet of things. *Information and Communication Technology Form*.
13. Javaid, A., Niyaz, Q., Sun, W. and Alam, M. (2016): 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)* (pp. 21-26).
14. Kanna, P.R. and Santhi, P. (2022): Hybrid Intrusion Detection using MapReduce based Black Widow Optimized Convolutional Long Short-Term Memory Neural Networks. Expert Systems with Applications, 194, p.116545.
15. Khan, M.A. (2021): HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system. Processes, 9(5), p.834.
16. Khan, R.U., Zhang, X., Alazab, M. and Kumar, R. (2019): May. An improved convolutional neural network model for intrusion detection in networks. In 2019 Cybersecurity and cyberforensics conference (CCC) (pp. 74-77). IEEE.
17. Khraisat, A., Gondal, I., Vamplew, P. and Kamruzzaman, J. (2019): Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, *2*(1), pp.1-22.
18. Li, A. and Yi, S. (2022): Intelligent Intrusion Detection Method of Industrial Internet of Things Based on CNN-BiLSTM. Security and Communication Networks.
19. Li, Y., Xu, Y., Liu, Z., Hou, H., Zheng, Y., Xin, Y., Zhao, Y. and Cui, L. (2020): Robust detection for network intrusion of industrial IoT based on multi-CNN fusion. Measurement, 154, p.107450.

20. Liu, M., Xue, Z., Xu, X., Zhong, C. and Chen, J. (2018): Host-based intrusion detection system with system calls: Review and future trends. *ACM Computing Surveys (CSUR)*, *51*(5), pp.1-36.
21. Mishra, P., Pilli, E.S., Varadharajan, V. and Tupakula, U. (2017): Intrusion detection techniques in cloud environment: A survey. *Journal of Network and Computer Applications*, *77*, pp.18-47.
22. Moustafa, N. and Slay, J. (2015): UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In 2015 military communications and information systems conference (MilCIS) (pp. 1-6). IEEE.
23. Moustafa, N. and Slay, J. (2016): The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1-3), pp.18-31.
24. Nguyen, Ngoc Anh, Tien Dat Dang, Elena Verdú, and Vijender Kumar Solanki. (2023): Short-term forecasting electricity load by long short-term memory and reinforcement learning for optimization of hyper-parameters. Evolutionary Intelligence 16, no. 5. 1729-1746
25. Patro, S. and Sahu, K.K. (2015): Normalization: A preprocessing stage. arXiv preprint arXiv:1503.06462.
26. Rekha, H., and M. Siddappa. (2022): Hybrid deep learning model for attack detection in internet of things." Service Oriented Computing and Applications 16, no. 4. 293-312
27. Rogers, Brendan, Nasimul Noman, Stephan Chalup, and Pablo Moscato. (2023): A comparative analysis of deep neural network architectures for sentence classification using genetic algorithm. Evolutionary Intelligence 1-20.
28. Roy, B. and Cheung, H. (2018): 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) (pp. 1-6). IEEE.
29. Saif, S., Das, P., Biswas, S., Khari, M. and Shanmuganathan, V. (2022): HIIDS: Hybrid intelligent intrusion detection system empowered with machine learning and metaheuristic algorithms for application in IoT based healthcare. Microprocessors and Microsystems, p.104622.
30. Schuster, M. and Paliwal, K.K. (1997): Bidirectional recurrent neural networks. In IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, Nov. 1997, doi: 10.1109/78.650093
31. Shah, D., Xue, Z.Y. and Aamodt, T.M (2022): Label encoding for regression networks. International Conference on Learning Representations. arXiv preprint arXiv:2212.01927.

32. Sinha, J. and Manollas, M. (2020): Efficient deep CNN-BILSTM model for network intrusion detection. In *Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Pattern Recognition* (pp. 223-231).
33. Su, T., Sun, H., Zhu, J., Wang, S. and Li, Y. (2020): BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. IEEE Access, 8, pp.29575-29585.
34. Sun, P., Liu, P., Li, Q., Liu, C., Lu, X., Hao, R. and Chen, J. (2020): DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system. Security and communication networks.
35. Tian, Q., Han, D., Li, K.C., Liu, X., Duan, L. and Castiglione, A. (2020): An intrusion detection approach based on improved deep belief network. Applied Intelligence, 50(10), pp.3162-3178.
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. (2017): Attention is all you need. Advances in neural information processing systems, 30.
37. Vinayakumar, R., Soman, K.P. and Poornachandran, P. (2017): Evaluation of recurrent neural network and its variants for intrusion detection system (IDS). International Journal of Information System Modeling and Design (IJISMD), 8(3), pp.43-63.
38. Vinayakumar, R., Alazab, M., Soman, K.P., Poornachandran, P., Al-Nemrat, A. and Venkatraman, S. (2019): Deep learning approach for intelligent intrusion detection system. *Ieee Access*, *7*, pp.41525-41550.
39. Von Solms, B. and Von Solms, R. (2018): Cybersecurity and information security–what goes where? *Information & Computer Security*.
40. Wu, H. and Gu, X. (2015): November. Max-pooling dropout for regularization of convolutional neural networks. In International Conference on Neural Information Processing (pp. 46-54). Springer, Cham.
41. Xu, Y. and Goodacre, R. (2018): On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. Journal of analysis and testing, 2(3), pp.249-262.
42. Yang, S.U. (2019): Research on network behavior anomaly analysis based on bidirectional LSTM. In *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)* (pp. 798-802). IEEE.
43. Zhang, J., Ling, Y., Fu, X., Yang, X., Xiong, G. and Zhang, R. (2020): Model of the intrusion detection system based on the integration of spatial-temporal features. *Computers & Security*, *89*, p.101681.