Received: February 18, 2024. Revised: April 5, 2024.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Network Intrusion Detection System Based on Information Gain with Deep Bidirectional Long Short-Term Memory

Woothukadu Thirumaran Valavan[1]* **Nalini Joseph[1]** **G. Umarani Srikanth[2]**

[1]*Department of Computer Science and Engineering, Bharath Institute of Higher Education and Research (BIHER), Chennai, India*
[2]*Department of Computer Science and Engineering, Panimalar Engineering College, Chennai, India*
* Corresponding author's Email: wtvalavan@gmail.com

Abstract: Network Intrusion Detection System (NIDS) plays a major role in maintaining the integrity and security in computer networks. These systems are created to detect and acknowledge the anomalous activities which specify unauthorized access and malicious internet. Establishing effective NIDS can be difficult, especially identifying network anomalies among the ever-increasing and difficult-to-detect malicious attacks. This study, implemented the Information gain with Deep Bidirectional Long Short-Term Memory (IG-Deep BiLSTM) method utilized to identify the effective intrusions, which will enable an NIDS to gain access to more data. The implemented BiLSTM method can better extract long-term and short-term dependent features and improve classification accuracy. The datasets used to gather data are the ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15 datasets. Next, pre-processing includes data digitization and encoding, as well as data normalization to convert the actual data into a suitable format and remove noise from the data. The IG is used to select the optimal features, and then the Deep BiLSTM is utilized to classify the network intrusion attacks as normal or malicious. Compared with existing methods, the implemented method achieved high accuracy values of 99.95%, 99.95%, 99.50%, and 99.93% using the CIC-IDS-2017, ToN-IoT, BoT-IoT, and UNSW_NB-15 datasets.

Keywords: Deep bidirectional long short-term memory, Information gain, Malicious attack, Network intrusion detection systems, Security.

1. Introduction

A new security technology called Network Intrusion Detection System (NIDS) works together with firewalls, antivirus programs, and access control systems. The goal is to strengthen and improve network security against malicious activity and intrusions [1]. The typical behaviour of anomaly-based NIDS models does not necessitate the explicit identification of attacks in the training set [2]. Any network traffic that deviates from the behaviour modelled by the model is recorded and reported. The model characterises traffic activity in the protected system's normal state [3]. To address unauthorised incidents or attacks in IoT-based environments, a number of academics have already presented IDS. The majority of these IDSs fall under the data-driven or knowledge-based system categories. Any behaviour (or action) departing from the legitimate network profile is referred to as an intrusion by knowledge-based IDS since they require a knowledge repository that offers the genuine network profile [4, 5]. Nevertheless, the security and privacy of IoT devices may be threatened by an increase in botnet attacks, such as Mirai, denial of service (DoS), distributed DoS (DDoS), Gafgayt, and theft [6]. The network's noise and other irrelevant features make it difficult to identify suspicious activity. Due to this difficulty, conducting an investigation takes longer and requires more effort, which lowers the chances of success [7]. The need for interventions that can identify risks and attacks that could harm information networks has increased due to the ongoing rise in internet users and the development

of IoT [8]. In particular, classic IDS find it difficult to provide sufficient performance and efficiency with today's high-bandwidth and high-traffic computer networks. Therefore, typical IDSs completely fail to ensure adequate security needs due to the increasingly sophisticated, automated, and scattered nature of attacks [9].

Transformer-based models are particularly appealing because of their adaptability for use in machine learning (ML)-based NIDs, where data is captured as flows or packet sequences and the performance of NIDS depends on its capability to identify intricate and subtle patterns in this traffic [10]. The volume of transactions in a network increases the risks to this sensitive data; hence, an IoT network needs to have a clever system to identify any unauthorised upgrades and prevent such hazards [11]. The attacks on machines and devices that are network-based have significantly improved due to this deep intrusion. As a result, protecting systems and devices against these types of attacks has become critical [12]. To safeguard the security and privacy of people and organizations, it is important to make sure that data collection and analysis are done in a responsible and secure manner [13]. A number of studies have used both simple and complex DL models, like long short-term memory (LSTM), convolutional neural networks (CNN), deep neural networks, and recent transformer networks, to create an efficient IDS in response to the field's rapid development in ML [14]. Simple ML models included K-means, K nearest neighbour (KNN), and one class support vector machines (OSVM). False positive and false negative results during detection can result from the data's grey regions of assault and normal behavior. These related works cannot address the essential measurement of dataset quality, which is the data tendency of normal and attack behaviours [15]. The main contribution of this research is given below.

- The implemented IG-Deep BiLSTM method is utilized to identify the effective NID based on feature selection and classification models. IG technique is used to select a feature that has the most information found in a specified class. Deep BiLSTM is used to classify the network intrusion accurately.
- Pre-processing stage includes data digitization and encoding, as well as data normalization to convert the actual data into a suitable format and remove noise from data.
- This implemented method is trained and tested using four datasets namely ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15, which provided high evaluation parameters.

This paper is organized as follows: Literature review related to the work is described in Section 2. Implementation description of methodology is in Section 3. Results and discussion are stated in Section 4, and Section 5 is conclusion of this work.

2. Literature survey

Zhang [16] implemented an Ensemble-based Automatic Feature Selection (EAFS) for NID. Hybrid Normalized score of mixed (NSOM) was employed on selected feature subsets dynamically and analyzed the subset performance. This implemented EAFS enhanced the classification of feature subset performance by a union and various subset intersections. However, due to difficulties in characterizing intricately chosen features, this implemented EAFS approach affected the detection model's transparency.

Mhawi [17] implemented an advanced ensemble Learning (EL) approach, which was utilized for NIDS. This approach employed a hybrid of Correlation Feature Selection coupled with the Forest Panelized Attributes (CFS-FPA) method that had attained efficient feature selection. This approach had high robustness and produced greater reliability in intrusion recognizing and categorizing benign traffic. However, this EL approach was required to enhance its capability to address infrequent traffic problems.

Lazzarini [18] implemented Deep Integrated Stacking for IoT (DIS-IoT) method based on a stacking ensemble to detect intrusions in the IoT environment. This method utilized CICIDS2017, ToN_IoT, and SWaT datasets to detect and classify the attacks in the IoT environment. ToN_IoT dataset attained high performance in both binary and multi-class classifications of attacks. However, this implementation had challenges in real-time processing and also had high computational overhead.

Saikam & Ch [19] implemented the network intrusion detection (NIDS) technique, which combined deep networks and hybrid sampling. This method utilized Elman Spike Neural Network (EESNN) for classified attack categories, and superior classification efficiency was achieved after training. This method used DenseNet 169 and the SAT-Net hierarchical network to effectively extract input data, and EESNN improved the accuracy performance. Due to lower identification rates in minority classes, this method required hybrid DL methods for high identification rates.

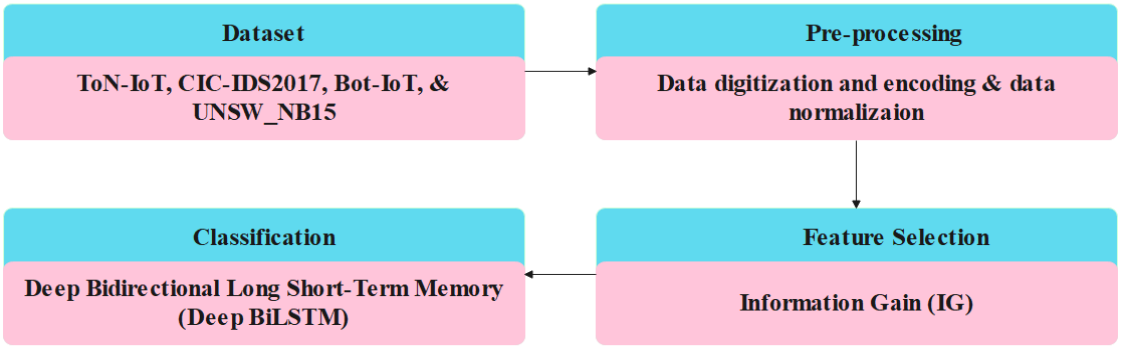

Figure. 1 Block diagram of implemented method

Thakkar & Lohiya [20] implemented a bagging ensemble learning (EL)-based Deep Neural Network (DNN) method for ID and to manage the class imbalance problem. CIC-IDS-2017, NSL-KDD, BoT-IoT, and UNSW_NB-15 datasets were employed to evaluate performance of implemented method. Bagging ensemble learning method was applied along with assigning class weights to assist in the training process and increase the DNN performance. However, the implemented approach was vulnerable to managing the input of malicious data, which caused misclassification.

Anushiya & Lavanya [21] implemented a Genetic-Algorithm and Faster Recurrent Convolutional Neural Network (GA-FR-CNN) model to detect intrusion in IoT. This model utilized Assimilated Artificial Fish Swarm Optimization to increase the recommended system's ability to identify features in feature selection. The implemented GA-FR-CNN model enhanced generalizability of model and also prevented overfitting. Due to low memory requirements and temporal complexity, this method had less intrusion detection.

Bakro [22] implemented a hybrid technique that included information gain, Chi-square, and particle swarm optimization (IG-CS-PSO) utilized for IDS. This IG-CS-PSO technique utilized Random Forest (RF) to identify and categorize different kinds of attacks. Kyoto and UNSW-NB15 datasets were employed to verify and evaluate the implemented technique. With the combined strength of each feature selection, this model achieved improved ID accuracy. However, this method ignored certain essential features, which led to overfitting.

3. Methodology

In this work, a Deep BiLSTM approach is implemented for NIDS. This work involves ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15 datasets for data collection, data digitization and encoding, as well as data normalization, which is used in pre-processing to convert the actual data into a suitable format and remove noise from the data. IG is utilized to select the optimal features from the pre-processed data, and at last, the Deep BiLSTM is employed to classify normal or malicious attacks in NIDS. Fig. 1 denotes the implemented method block diagram.

3.1 Datasets

This work utilizes four datasets, namely ToN_IoT [23], Bot-IoT [24], CIC-IDS 2017 [25], and UNSW-NB15 [26], for intrusion detection. These four datasets for network intrusions can be employed with systems of intrusion detection, and they are explained as follows.

3.1.1. ToN_IoT

An establishment of a large-scale network at Australian Defence Force Academy (ADFA) using University of New South Wales (UNSW) is used to collect the ToN_IoT dataset. A large range of heterogeneous sources is provided by this network, which contains IoT sensors, virtual devices, cloud platforms, and physical systems. The dataset consists of several devices captured from different network perspectives, such as IoT/IIoT, Windows, and Linux.

3.1.2. Bot_IoT

ID in IoT networks is presently utilized by the Bot-IoT dataset. Cyber Range Lab of UNSW Canberra, in a realistic network environment, produced the Bot-IoT dataset. This dataset includes 72 million records in total and combines botnet and regular traffic for the network environment. This dataset is trained on 364,562 records, and during the testing, this was reduced to 243,043 records. Although most of the dataset consists of DDoS-type packets and DDoS, there are a total of four different kinds of attacks.

3.1.3. CIC-IDS 2017

CIC-IDS2017 is a network traffic dataset from the Canadian Institute for Cybersecurity (CIC). The collection includes recordings of both malicious and benign network traffic as well as a variety of attack types, including web, botnet, and denial-of-service attacks. Features like packet and flow features that were taken from the network traffic samples are included in the dataset, which has about 10 million instances in total. This dataset was created to provide an accurate depiction of modern network traffic. It is meant to be used in applications of cybersecurity and in development projects involving intrusion detection systems.

3.1.4. UNSW-NB15

Australian Centre for Cyber Security (ACCS) at University of New South Wales (UNSW) Cyber Range Lab produced UNSW-NB15 dataset, which is a dataset of network traffic. Collection contains network traffic recordings for fifteen different types of attacks, such as backdoors, port scans, and SQL injection. To give a realistic representation of modern network traffic, the dataset also contains captures of regular network traffic. Features including packet and flow features that were taken from the network traffic samples are included in the dataset, which has about 2.5 million instances in total.

3.2 Pre-processing

After the collection of data, pre-processing is an important step that is used to clean the data and convert actual data into a suitable format. This pre-processing step includes data digitization, data encoding, and data normalization processing.

3.2.1. Data digitization and encoding

To transform original features into numerical features, binarization method known as the one-hot encoding technique is utilized, which also enables the subsequent feature process.

Furthermore, local features may be extracted in grayscale image form, and to reshape 1D features into 2D features, reshaping a module of spatial feature extraction is important. Based on various datasets, the specific processing techniques vary. To encode data for every feature, an MLP layer is utilized like word embedding later in NLP, where a method of temporal features is extracted. After various subspaces are mapped by the amplified features, the richer features are extracted accordingly, and here the input dimensions are essential for this approach. At the same time, during the training of MLP layer parameters may be adjusted dynamically.

3.2.2. Data normalization process

Following data digitization and encoding, normalization process is executed to remove the data differences due to various dimensions. Reduce the variances dimensions and combine the values by adopting the min-max normalization technique. Eq. (1) represents the specific formula.

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

The normalization processing result is denoted as x_{norm}, original feature is represented as x, maximum and minimum values for feature x are denoted as x_{max} and x_{min}.

3.3 Feature selection

Following pre-processing, the feature selection process is performed. Pre-processed output data is fed as input to this feature selection process. This is the most important stage before the classification phase. This stage seeks to select an optimal subset of existing features and irrelevant features are removed. In this work the IG technique is used for the process of feature selection.

3.3.1. Information gain (IG)

This is a filter-based and one of the most common utilized feature selection technique. This IG technique detects a feature, which has the most information found in a specified class and is utilized by a simple attribute rank due to the irrelevant features and decreased noise. Features' entropy is computed to determine the best feature. Entropy is an uncertainty measure that may be utilized to conclude the feature distribution in cosine form, and Eq. (2) shows the entropy calculation.

$$Entropy\ (S) = \sum_i^c -P_i log_2 P_i \quad (2)$$

Where class i's number of samples is shown as P_i, and number of values in classification class is denoted as c. Value of IG is computed after getting the entropy value, as mentioned below in Eq. (3).

$$Gain\ (S, A) = Entropy(s) - \sum_{values(A)} \frac{|S_v|}{|S|} Entropy\ (S_v) \quad (3)$$

Where sample and attribute are represented as A and S, attribute A's possible value is denoted as v, and A's possible value sets are shown as $values(A)$. Entropy for samples that have v value is denoted as $Entropy\ (S_v)$, number of samples for all data samples is shown as $|S|$, and number of samples for v value is represented as $|S_v|$.

In this research, a filter-based technique of IG is selected as a feature selection technique, which obtains more stable sets of selected features because it prevents overfitting due to its robust nature. Generally, filter-based technique's computational complexity is represented as $O(m.n^2)$, where the number of features or attributes is shown as n, and the number of training data is shown as m. It is low compared to wrapper and embedded- based methods. The wrapper-based methods have an intricate nature, which establishes the greater overfitting issues. Therefore, relevant, significant, and a small number of features are produced utilizing feature selection techniques, and the execution time for classification approaches utilized in the attack or anomaly detection process will decrease with less computational complexity.

From 1 to 77 are the features that are provided for IDS. Weight values of features are ranked using information gain, and error and try method is used to evaluate the minimum weight manually. Minimum weight values are used to rank and group the features in this work. Therefore, groups of features are attained and every feature group are keeping various number of features. Furthermore, the Deep BiLSTM classifier method is used to validate all feature groups, so this can evaluate the effective groups of features, which is sufficient to classify the types of attacks.

3.4 Classification using deep BiLSTM approach

Following feature selection, the classification phase is performed using the selected features. The inherent problem of mitigating exploding gradient issues restricts their extended duration's effectiveness, when Recurrent Neural Networks (RNNs) display proficiency in extracting temporal knowledge. To overcome this drawback, a specialized variant of LSTM adaptation Deep BiLSTM is used for the classification, which is an extension of BiLSTM. This structure excels at searching for sequential spatial features within the local environment, capturing assisted spatial-temporal feature patterns. The detailed LSTM structure is represented in Fig. 2, and its unique ability is emphasized to broadly classify and analyze NIDs based on features selected by IG.

The Deep BiLSTM approach is derived from LSTM, which utilizes three gates, including the input gate, forget gate, and output gate. The unit of LSTM in represented below. The specific information which needs modifications from preceding unit identification by using the input gate is critical. Eq. (4) expresses the input gate.

$$i_t = \sigma(Y_i.x_t + W_i.h_{t-1} + b_i) \tag{4}$$

where input gate at time t denotes i_t, sigmoid function is represented as σ, weight matrix signifies as W and Y, input data at time t indicates x_t, process of matrix multiplication is represented as ".", preceding LSTM unit's output is shown as h_{t-1}, and bias term is denoted as b. Next, the forget gate determines what information could be discarded or retained. The forget gate expression is represented in Eq. (5).

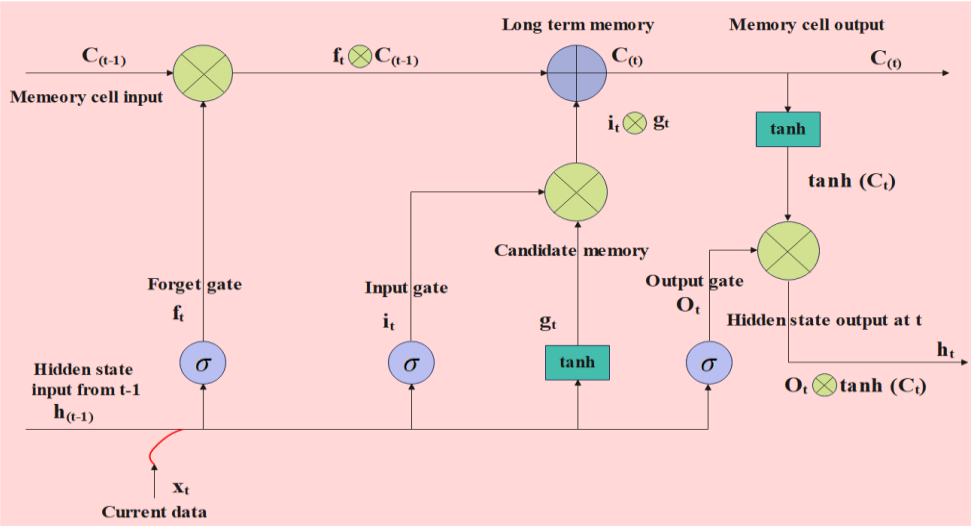

Figure. 2 Architecture of LSTM

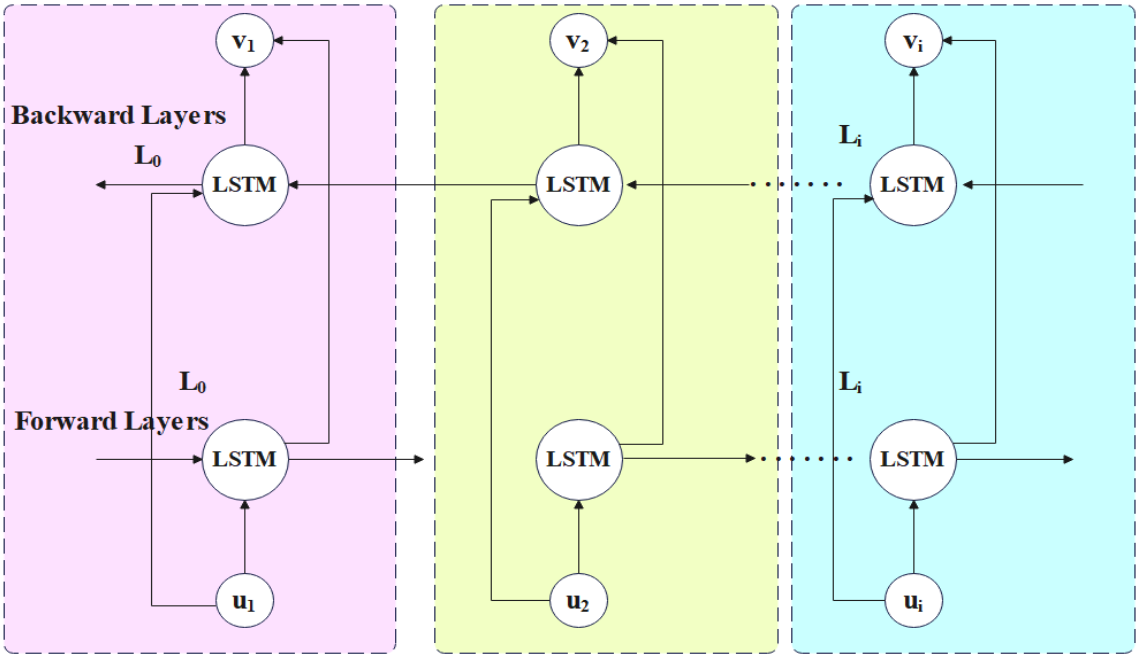

Figure. 3 Deep BiLSTM framework diagram

$$f_t = \sigma(Y_f . x_t + W_f . h_{t-1} + b_c) \quad (5)$$

Where this utilizes a sigmoid function to evaluate both the previously concealed state h_{t-1}, and the current input state x_t . Calculating the information and forgetting old information significance are done by the forget gate which is denoted as f_t.

Then, $tanh$ function produces hidden state h_{t-1}, and current input x_t, for developing a new vector candidate value $\tilde{c}_n$, which can add to the state and is expressed in Eq. (6).

$$\tilde{c}_n = tanh(Y_c . x_t + W_c . h_{t-1} + b_c) \quad (6)$$

The candidate is evaluated through the employment of the tangent activation function denoted by c_t and is expressed in the below Eq. (7).

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (7)$$

Eventually, the current state is determined and provided in the equation, where point-to-point multiplication is represented by $\odot$. At last, the output gate determines the sigmoid layer which is first activated to select cell state components as output Eq. (8). Here, LSTM defines the cell states and chooses the output portion with Eq. (9).

$$g_t = \sigma(Y_g . x_t + W_g . h_{t-1} + b_g) \quad (8)$$

$$h_t = g_t \odot \tanh(c_t) \quad (9)$$

Here the output gate's weight matrix denotes W and Y, and the output gate's bias denotes b. g_t is represented as the output gate, and $tanh$ function computes the final output function which is denoted as h_t. The baseline LSTM approach detects current NIDs based on only prior data. It is apparent that if the data are analyzed unidirectionally, there is a possibility that some information will be lost. Fig. 3 represents the two layers of LSTM, which operate in both forward and backward directions and consist of Deep BiLSTM. The output layer v_t of Deep BiLSTM is expressed in Eq. (10).

$$v_t = [\overrightarrow{h_t}\ \overleftarrow{h_t}] \quad (10)$$

Where v_t is the output of two LSTM units' combination, $\overleftarrow{h_t}$ and $\overrightarrow{h_t}$ denoted the forward and backward outputs of LSTM units. The idea behind the RNN is that this Deep BiLSTM approach introduces the data one by one sequentially to the neural network, adding the temporal variables as well, instead of sending the complete set of input data in a singular instance. This method passes the initial value into the network and receives a similar output if it has an array of input values. Then, this method passed the next input along with the previous output for the subsequent output. This Deep BiLSTM approach enhances the detection of intrusion behavior by evaluating network traffic patterns, and providing accurate classification of network intrusion.

4. Results and discussion

This implemented method is trained using four datasets, namely the ToN-IoT, CIC-IDS-2017, NSL-KDD, BoT-IoT, and UNSW_NB-15 datasets. The following are the system requirements for the proposed research: processor: intel core i7, RAM: 16 GB, and Windows 10 (64-bit) operating system. The effectiveness of implemented method is computed in terms of recall, precision, accuracy, and f1-score, which are explained below.

4.1 Evaluation parameters

The parameters like accuracy, f-measure, recall, precision, and false alarm rate (FAR) are employed to calculate method's performance. Parameters are denoted mathematically in Eqs (11), (12), (13), (14), AND (15).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (11)$$

$$F-measure = \frac{Precision*Recall}{Precision+Recall} \quad (12)$$

$$Recall = \frac{TP}{TP+FN} \quad (13)$$

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

$$FPR = \frac{FP}{FP+TN} \quad (15)$$

where True Positive, False Negative, True Negative, and False Positive are represented by TP, $FN, TN, and\ FP$.

4.2 Quantitative and qualitative analysis

In this section, performance of IG-Deep BiLSTM method in terms of precision, F1-measure, accuracy, and recall is analysed based on the ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15 datasets. Different tables are presented below to show the implemented method's effectiveness.

Table 1 represents performance of implemented feature selection method. Implemented IG method is compared with state-of-the-art feature selection methods including Chi-square, Principal component analysis (PCA), Variance Threshold, and recursive feature elimination (RFE). The implemented IG method achieved the highest values of 91.89% precision, 92.98% accuracy, 92.60% f1-measure, 91.99% recall, and 2.2 FAR when compared with other feature selection methods.

Table 1. Performance of Feature selection method

Feature selection methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
Chi-square	63.12	82.23	79.82	83.76	3.8
PCA	72.45	79.56	84.71	85.19	3.5
Variance Threshold	69.78	86.89	71.93	78.43	3.2
RFE	85.57	80.31	78.94	89.67	2.5
IG	**92.98**	**91.89**	**91.99**	**92.60**	**2.2**

Table 2. Performance of the classification models

Classification methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
RNN	85.82	90.40	81.87	89.29	4.0
GRU	79.71	82.67	90.54	83.40	3.8
CNN	88.93	78.15	86.50	92.98	4.9
LSTM	91.30	84.39	88.31	80.99	2.9
Deep BiLSTM	95.35	94.76	94.98	95.87	1.9

Table 3. Performance of the implemented method utilizing the ToN-IoT dataset

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
IG-RNN	85.60	89.35	79.98	80.64	2.0
IG-GRU	81.12	90.78	84.45	86.89	5.5
IG-CNN	94.67	87.49	93.56	90.23	4.0
IG-LSTM	90.82	93.71	88.93	95.91	3.2
IG-Deep BiLSTM	**99.95**	**99.93**	**99.85**	**99.80**	**2.8**

Table 2 represents performance of classification models. Implemented Deep BiLSTM method is compared with state-of-the-art classifier models such as RNN, Gated Recurrent Unit (GRU), Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM). Implemented Deep BiLSTM method achieved the highest values of 94.76% precision, 95.35% accuracy, 95.87% f1-measure, 94.98% recall, and 1.9 FAR while compared with other classifier models.

Table 3 denotes the performance of the implemented method using the ToN-IoT dataset. The implemented IG-Deep BiLSTM method is compared with state-of-the-art methods like IG-RNN, IG-GRU, IG-CNN, and IG-LSTM. This implemented IG-Deep BiLSTM method attained better values of 99.95% accuracy, 99.58% recall, 99.93% precision, 99.80% f1-measure, and 2.8 FAR when compared to other methods.

Table 4 shows the performance of the implemented method using CIC-IDS-2017 dataset. Implemented IG-Deep BiLSTM method is compared with state-of-the-art methods like IG-LSTM, IG-RNN, IG-CNN, and IG-GRU. This implemented IG-Deep BiLSTM method attained better values of 99.98% recall, 99.95% f1-measure, 99.95% accuracy, 99.86% precision, and 3.3 FAR when compared to other methods

Table 5 represents the performance of the implemented method using the BoT-IoT dataset. The implemented IG-Deep BiLSTM method is compared with state-of-the-art methods like IG-GRU, IG-CNN, IG-LSTM, and IG-RNN. This implemented IG-Deep BiLSTM method attained better values of 99.90% recall, 99.91% f1-measure, 99.50% accuracy, 99.93% precision, and 2.5 FAR when compared to other methods.

Table 6 signifies the performance of the implemented method using the UNSW_NB-15 dataset. The implemented IG-Deep BiLSTM method is compared with state-of-the-art methods like IG-GRU, IG-CNN, IG-LSTM, and IG-RNN. This implemented IG-Deep BiLSTM method attained better values of 99.80% f1-measure, 99.93% accuracy, 99.85% precision, 99.90% recall, and 1.4 FAR when compared to other methods.

4.2.1. Comfusion matrix

Figure 4 represents the confusion matrix utilizing the different datasets. The confusion matrix for four datasets including CIC-IDS-2017, ToN-IoT, BoT-IoT, and UNSW-NB15 datasets are illustrates in Figure 4 (a), (b), (c), and (d).

Table 4. Performance of the implemented method utilizing the CIC-IDS-2017 dataset

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
IG-RNN	87.58	80.41	90.69	87.63	5.0
IG-GRU	92.90	84.35	85.58	90.81	3.5
IG-CNN	89.61	95.94	92.10	83.75	3.8
IG-LSTM	94.75	89.55	93.97	94.92	4.0
IG-Deep BiLSTM	**99.95**	**99.86**	**99.98**	**99.95**	**3.3**

Table 5. Performance of the implemented method utiliozing the BoT-IoT dataset

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
IG-GRU	80.76	90.98	91.75	85.64	4.6
IG-CNN	79.84	84.65	93.86	84.31	4.8
IG-LSTM	95.91	92.32	93.48	92.90	3.4
IG-Deep BiLSTM	**99.50**	**99.93**	**99.90**	**99.91**	**2.5**

Table 6. Performance of the implemented method using the UNSW_NB-15 dataset

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Measure (%)	FAR
IG-RNN	93.78	88.70	90.17	91.43	1.9
IG-GRU	89.45	93.32	87.93	85.59	2.3
IG-CNN	85.12	94.65	92.28	90.67	1,8
IG-LSTM	92.90	89.98	89.53	94.82	1.5
IG-Deep BiLSTM	**99.93**	**99.85**	**99.90**	**99.80**	**1.4**

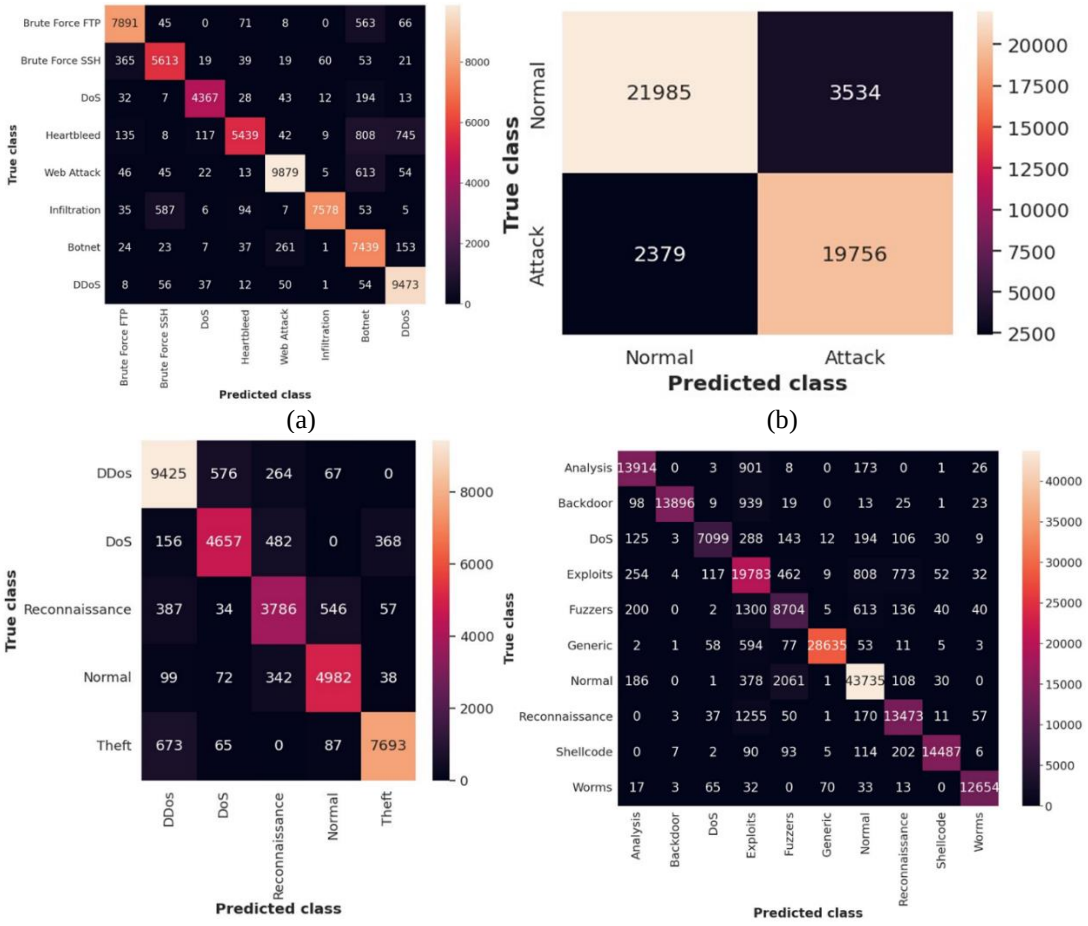

Figure. 4 Confusion matrix for four datasets: (a) CIC-IDS-2017, (b) ToN-IoT, (c) BoT-IoT, and (d) UNSW-NB15 datasets

Table 7. Comparative analysis of the implemented method with existing methods using four datasets.

Datasets	Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-measure (%)
CIC-IDS-2017	EAFS [16]	99.92	N/A	N/A	99.92
	EL [17]	99.7	N/A	N/A	N/A
	Bagging EL-based DNN [20]	98.74	99.77	99.96	99.86
	IG-Deep BiLSTM	**99.95**	**99.86**	**99.98**	**99.95**
ToN-IoT	DIS-IoT [18]	99.50	N/A	N/A	N/A
	EESNN [19]	99.89	99.87	99.42	N/A
	IG-Deep BiLSTM	**99.95**	**99.93**	**99.85**	**99.80**
BoT-IoT	Bagging EL-based DNN [20]	98.99	98.90	91.30	94.95
	GA-FR-CNN [21]	93.77	86.66	95.87	91.03
	IG-Deep BiLSTM	**99.50**	**99.93**	**99.90**	**99.91**
UNSW_NB-15	EAFS [16]	98.36	N/A	N/A	98.31
	IG-CS-PSO-RF [22]	98.39	98.39	98.54	98.46
	Bagging EL-based DNN [20]	96.70	98.90	98.67	98.78
	IG-Deep BiLSTM	**99.93**	**99.85**	**99.90**	**99.80**

4.3 Comparative analysis

The implemented model's performance is evaluated using parameters like recall, accuracy, f1-measure, and precision, as shown in this section. The comparative analysis of existing and implemented methods is compared using the datasets ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15, which are represented in Table 7. The implemented method outperformed all other approaches such as EAFS [16], EL [17], DIS-IoT [18], EESNN [19], Bagging EL-based DNN [20], GA-FR-CNN [21], and IG-CS-PSO-RF [22], and

the implemented method achieved the highest performance values.

4.4 Discussion

The existing methods limitations and the implemented approach's benefits are discussed in this section. Some limitations of the existing methods are as follows: the EAFS [16] method had difficulties in characterizing intricately chosen features, this implemented EAFS approach affected the detection model's transparency. The EL [17] approach requires enhancement of its capability to address infrequent traffic problems. The GA-FR-CNN [21] method had less intrusion detection due to low memory requirements and temporal complexity. The IG-CS-PSO [22] method ignored certain essential features; hence, this produced overfitting. To overcome these issues, the IG-Deep BiLSTM approach is implemented in this work. This implemented method is trained and tested utilizing four datasets namely ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15. IG is utilized in the feature selection stage for effectively identifying the features that mostly provide the difference between malicious and normal network behavior, which increases the efficiency of accuracy in NIDS. The Deep BiLSTM technique is employed in the classification phase to essentially evaluate network traffic patterns and sequences, which can improve the detection of complex intrusion behaviors. At the same time, this technique's bidirectional nature extracts rich hierarchical features, and provides a complete network data representation for NID's accurate classification.

5. Conclusion

Demand for NIDS is growing as network intrusion continues to change. ID is a significant link in the IoT for the information security protection field, and application systems are contributed to intelligent devices to protect user information. A method named IG-Deep BiLSTM is implemented effectively for NIDS, and this method consists of four phases including datasets, pre-processing, feature selection, and classification. At first, data are collected using four datasets: ToN-IoT, CIC-IDS-2017, BoT-IoT, and UNSW_NB-15. The second phase is pre-processing, where collected data are pre-processed using data digitization and encoding, and a data normalization process is used to clean the data, convert the actual data into a suitable format, and remove the data differences due to various dimensions. Feature selection is the third phase. The IG technique is used to select a feature that has information mostly found in a specified class. It is employed by simple attribute rank because of irrelevant features and reduced noise. Finally the selected features are passed through the classification phase. Here the Deep BiLSTM approach is used to overcome the inherent problem of mitigating exploding gradient issues. This structure excels at searching for sequential spatial features within the local environment, capturing assisted spatial-temporal feature patterns, and increasing the classification accuracy of NIDS. This method is used to improve the capability to generalize the new data and reduce the impact of overfitting. Compared with existing methods like EAFS [16], EL [17], DIS-IoT [18], EESNN [19], Bagging EL-based DNN [20], GA-FR-CNN [21], and IG-CS-PSO-RF [22], the implemented method achieved the highest values of accuracy 99.95%, 99.95%, 99.50%, and 99.93% using CIC-IDS-2017, ToN-IoT, BoT-IoT, and UNSW_NB-15 datasets. In the future, the classification of class-wise predictions will be improved in the implemented technique.

Notation Description

symbol	Description
x_{norm}	normalization processing result
x	original feature
x_{max} and x_{min}.	maximum and minimum values for feature x
P_i	class i's number of samples
c	number of values in classification class
A	attribute
S	sample
v	attribute A's possible value
$values(A)$	A's possible value sets
$Entropy\ (S_v)$	Entropy for samples that have v value
$\|S\|$	number of samples for all data samples
$\|S_v\|$	number of samples for v value
i_t	input gate at time t
σ	sigmoid function
W and Y	weight matrix
x_t	input data at time t
.	process of matrix multiplication
h_{t-1}	preceding LSTM unit's output
b	bias term
f_t	forget gate
h_{t-1}	hidden state
c_t	tangent activation function
$\odot$	point-to-point multiplication
v_t	output layer
h_t	final output function

TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, have been done by 1st author. The supervision and project administration, have been done by 2nd and 3rd author.

References

[1] S. Mohamed, and R. Ejbali, "Deep SARSA-based reinforcement learning approach for anomaly network intrusion detection system", *International Journal of Information Security*, Vol. 22, pp. 235-247, 2023.

[2] O.A. Alghanam, W. Almobaideen, M. Saadeh, and O. Adwan, "An improved PIO feature selection algorithm for IoT network intrusion detection system based on ensemble learning", *Expert Systems with Applications*, Vol. 213, pp. 118745, 2023.

[3] H. Yang, J. Xu, Y. Xiao, and L. Hu, "SPE-ACGAN: A Resampling Approach for Class Imbalance Problem in Network Intrusion Detection Systems", *Electronics*, Vol. 12, No. 15, pp. 3323, 2023.

[4] A. Singh, P.K. Chouhan, and G.S. Aujla, "SecureFlow: Knowledge and data-driven ensemble for intrusion detection and dynamic rule configuration in software-defined IoT environment", *Ad Hoc Networks*, Vol. 156, pp. 103404, 2024.

[5] N. Jeffrey, Q. Tan, and J.R. Villar, "A hybrid methodology for anomaly detection in Cyber–Physical Systems", *Neurocomputing*, Vol. 568, pp. 127068, 2024.

[6] J. Azimjonov, and T. Kim, "Stochastic gradient descent classifier-based lightweight intrusion detection systems using the efficient feature subsets of datasets", *Expert Systems with Applications*, Vol. 237, pp. 121493, 2024.

[7] H.M. Saleh, H. Marouane, and A. Fakhfakh, "Stochastic Gradient Descent Intrusions Detection for Wireless Sensor Network Attack Detection System Using Machine Learning", *IEEE Access*, Vol. 12, pp. 3825-3836, 2024.

[8] E. Osa, P.E. Orukpe, and U. Iruansi, "Design and implementation of a deep neural network approach for intrusion detection systems", *e-Prime-Advances in Electrical Engineering*, Electronics and Energy, Vol. 7, pp. 100434, 2024.

[9] K. Cengiz, S. Lipsa, R.K. Dash, N. Ivković, and M. Konecki, "A novel intrusion detection system based on artificial neural network and genetic algorithm with a new dimensionality reduction technique for UAV communication", *IEEE Access*, Vol. 12, pp. 4925-4937, 2024.

[10] L.D. Manocchio, S. Layeghy, W.W. Lo, G.K. Kulatilleke, M. Sarhan, and M. Portmann, "Flowtransformer: A transformer framework for flow-based network intrusion detection systems", *Expert Systems with Applications*, Vol. 241, p. 122564, 2024.

[11] A. Biju, and S.W. Franklin, "Evaluated bird swarm optimization based on deep belief network (EBSO-DBN) classification technique for IOT network intrusion detection", *Automatika*, Vol. 65, No. 1, pp. 108-116, 2024.

[12] S. Patthi, and S. Singh, "2-layer classification model with correlated common feature selection for intrusion detection system in networks", *Multimedia Tools and Applications*, 2024.

[13] D. Srivastav, and P. Srivastava, "A two-tier hybrid ensemble learning pipeline for intrusion detection systems in IoT networks", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 14, No. 4, pp. 3913-3927, 2023.

[14] T.N. Hoang, and D. Kim, "Supervised contrastive ResNet and transfer learning for the in-vehicle intrusion detection system", *Expert Systems with Applications*, Vol. 238, pp. 122181, 2024.

[15] C.H. Tseng, W.J. Tsaur, and Y.M. Shen, "Classification Tendency Difference Index Model for Feature Selection and Extraction in Wireless Intrusion Detection", *Future Internet*, Vol. 16, No. 1, pp. 25, 2024.

[16] Y. Zhang, H. Zhang, and B. Zhang, "An effective ensemble automatic feature selection method for network intrusion detection", *Information*, Vol. 13, No. 7, pp. 314, 2022.

[17] D.N. Mhawi, A. Aldallal, and S. Hassan, "Advanced feature-selection-based hybrid ensemble learning algorithms for network intrusion detection systems", *Symmetry*, Vol. 14, No. 7, pp. 1461, 2022.

[18] R. Lazzarini, H. Tianfield, and V. Charissis, "A stacking ensemble of deep learning models for IoT intrusion detection", *Knowledge-Based Systems*, Vol. 279, pp. 110941, 2023.
[19] J. Saikam, and K. Ch, "EESNN: Hybrid Deep Learning Empowered Spatial-Temporal Features for Network Intrusion Detection System", *IEEE Access*, Vol. 12, pp. 15930-15945, 2024.
[20] A. Thakkar, and R. Lohiya, "Attack classification of imbalanced intrusion data for IoT network using ensemble learning-based deep neural network", *IEEE Internet of Things Journal*, Vol. 10, No. 13, pp. 11888-11895, 2023.
[21] R. Anushiya, and V.S. Lavanya, "A new deep-learning with swarm based feature selection for intelligent intrusion detection for the Internet of things", *Measurement: Sensors*, Vol. 26, pp. 100700, 2023.
[22] M. Bakro, R.R. Kumar, A. Alabrah, Z. Ashraf, M.N. Ahmed, M. Shameem, and A. Abdelsalam, "An Improved Design for a Cloud Intrusion Detection System Using Hybrid Features Selection Approach with ML Classifier", *IEEE Access*, Vol. 11, pp. 64228-64247, 2023.
[23] ToN-IoT dataset link: https://www.kaggle.com/datasets/amaniabourida/ton-iot
[24] BoT-IoT dataset link: https://www.kaggle.com/datasets/vigneshvenkateswaran/bot-iot
[25] CIC-IDS-2017 dataset link: https://www.kaggle.com/datasets/cicdataset/cicids2017/data
[26] UNSW_NB-15 dataset link: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15/code