1-1-2023

# Intrusion detection based on bidirectional long short-term memory with attention mechanism

Yongjie Yang

Shanshan Tu

Raja Hashim Ali

Hisham Alasmary

Muhammad Waqas
*Edith Cowan University*, m.waqas@ecu.edu.au

*See next page for additional authors*



10.32604/cmc.2023.031907
Yang, Y., Tu, S., Ali, R. H., Alasmary, H., Waqas, M., & Amjad, M. N. (2023). Intrusion detection based on bidirectional long short-term memory with attention mechanism. CMC-Computer Material and Continua, 74(1), 801-815.
https://doi.org/10.32604/cmc.2023.031907

## Authors
Yongjie Yang, Shanshan Tu, Raja Hashim Ali, Hisham Alasmary, Muhammad Waqas, and Muhammad Nouman Amjad

*Computers, Materials & Continua*
DOI: 10.32604/cmc.2023.031907
***Article***

Tech Science Press

# Intrusion Detection Based on Bidirectional Long Short-Term Memory with Attention Mechanism

**Yongjie Yang[1], Shanshan Tu[1], Raja Hashim Ali[2], Hisham Alasmary[3,4], Muhammad Waqas[5,6,*] and Muhammad Nouman Amjad[7]**

[1]Engineering Research Center of Intelligent Perception and Autonomous Control, Faculty of Information Technology, Beijing University of Technology, Beijing, 100124, China
[2]Faculty of Computer Science and Engineering, GIK Institute of Engineering Sciences and Technology, Topi, 23460, Pakistan
[3]Department of Computer Science, College of Computer Science, King Khalid University, Abha, Saudi Arabia
[4]Information Security and Cybersecurity Unit, King Khalid University, Abha, Saudi Arabia
[5]Computer Engineering Department, College of Information Technology, University of Bahrain, 32038, Bahrain
[6]School of Engineering, Edith Cowan University, Joondalup Perth, WA, 6027, Australia
[7]School of Engineering, University of Management and Technology, Lahore, Pakistan
*Corresponding Author: Muhammad Waqas. Email: engr.waqas2079@gmail.com
Received: 29 April 2022; Accepted: 24 June 2022

**Abstract:** With the recent developments in the Internet of Things (IoT), the amount of data collected has expanded tremendously, resulting in a higher demand for data storage, computational capacity, and real-time processing capabilities. Cloud computing has traditionally played an important role in establishing IoT. However, fog computing has recently emerged as a new field complementing cloud computing due to its enhanced mobility, location awareness, heterogeneity, scalability, low latency, and geographic distribution. However, IoT networks are vulnerable to unwanted assaults because of their open and shared nature. As a result, various fog computing-based security models that protect IoT networks have been developed. A distributed architecture based on an intrusion detection system (IDS) ensures that a dynamic, scalable IoT environment with the ability to disperse centralized tasks to local fog nodes and which successfully detects advanced malicious threats is available. In this study, we examined the time-related aspects of network traffic data. We presented an intrusion detection model based on a two-layered bidirectional long short-term memory (Bi-LSTM) with an attention mechanism for traffic data classification verified on the UNSW-NB15 benchmark dataset. We showed that the suggested model outperformed numerous leading-edge Network IDS that used machine learning models in terms of accuracy, precision, recall and F1 score.

**Keywords:** Fog computing; intrusion detection; bi-LSTM; attention mechanism

## 1 Introduction

The Internet of Things (IoT) has proliferated in recent years because of the advancements in 5G technology, the maturity of communication technology, and the availability of smart gadgets. Various smart sensors and actuators, including radio frequency identification systems, infrared sensors, laser scanners, positioning systems, and other device technologies, connect smart things in line with established communication protocols. IoT has applications in nearly every field: smart cities, smart transportation, smart grids, smart agriculture, energy management, healthcare, education, and security. In short, the "Internet of Everything" has fundamentally altered human life and work habits. With the growing number of connected devices in the IoT, a considerable amount of necessary data is generated for governments, organizations, and individuals, which is driving the development of advanced information services with a demand for significant storage and computational power, as well as real-time processing capacity [1]. IoT devices continually record and transmit personal data by constantly monitoring our professional and personal activities. Therefore, data security and maintaining the privacy of its customers are critical for IoT applications. It's important to note that IoT devices are subject to several security assaults. Some of these malicious operations may cause a loss of service, while others can inflict catastrophic damage to the system, potentially resulting in tragedy for end users. Since most existing IoT security approaches are centralized and cloud-based [2], these approaches are complicated to deploy and have a considerable transmission delay, with limited mobility, poor scalability, and fewer real-time processing capabilities. Therefore, the security challenges associated with the IoT system cannot be resolved successfully by either the cloud or the isolated attack detection system [3]. On the other hand, a distributed security system allows for interoperability, flexibility, and scalability while securing and managing heterogeneous devices in a unified manner [4].

Fog computing is a popular distributed paradigm that brings processing nodes closer to the physical system and provides processing and storage capabilities at the edge node to detect possible threats quickly and efficiently [5]. Cisco was the pioneer of fog computing, which quickly gained popularity as a viable alternative to cloud computing. Cloud computing is known for issues with high energy consumption and latency. Fog computing extends the cloud to the network's edge, enabling efficient data access, processing, and storage. Fog computing occurs at the fog layer between the cloud and the end-user. Each smart thing is connected to a Fog device in this framework. The fog devices can communicate with one another, and each device is connected to the cloud [6]. The main difference is that the cloud is a centralized cloud network, whereas fog is a decentralized distributed system [7]. In comparison to cloud computing, fog computing provides enhanced mobility, better location awareness, heterogeneity, scalability, low latency, and geographic distribution, enabling a diverse range of IoT systems and applications [8]. In general, the fog computing paradigm aims to reduce data and computing consumption on the cloud server and reduce latency and improve quality of service (QoS) [9]. Moreover, like other services, the IoT system's security mechanism can be designed and delivered at the fog layer, employing fog nodes as agents. In short, the fog node contributes to the IoT system's advantages in deploying distributed and parallel security services [10].

Although fog computing can provide distributed services for the IoT, an intrusion detection system must be installed in the fog node to ensure the Internet of Things security. As an active network security protection system, the Intrusion Detection System (IDS) monitors real-time traffic data generated by the IoT, delivers alerts and actively protects against potential risks whenever a malicious attack or other abnormal event is detected. This is critical to prevent attempts to disrupt the Internet of Things' availability, integrity, and confidentiality. This study provides a distributed IDS based on the Bidirectional Long Short-Term Memory (Bi-LSTM) and Attention method to combat recent IoT attacks. The distributed detection system comprises sensing nodes capable of identifying moving objects in their vicinity. All sensors are identical since they have the same detection radius, and the sensor network is clustered into distinct groups based on the radius. Each member of the cluster is responsible for data collection through environmental monitoring. The acquired data is then routed to nearby fog nodes for processing. IDS are placed at each fog node in the proposed topology to monitor incoming traffic. Thus, fog nodes enable IoT systems to create parallel and distributed collaborative security mechanisms. The log of each network packet is analyzed at the cloud server for the global administration of IoT devices. This research aims to use a distributed integrated design and an intrusion detection system at the fog layer to protect the IoT from attacks. Our primary contributions include the following.

1) We propose a long short-term memory fog computing approach with an attention mechanism-based distributed ensemble architecture to protect the IoT network.
2) We fully utilize traffic slice information and the attention mechanism in the Bi-LSTM model. Then, we set a suitable timestep to verify that the attention mechanism has improved the model's performance.
3) The UNSW-NB15 benchmark dataset is used to conduct experiments. The results indicate that our approach achieves higher accuracy, detection rate, recall and f1-score than other approaches.

The remainder of this work is structured as follows. Section 2 discusses earlier research pertinent to this paper. Section 3 presents the proposed approach for detecting malicious activity in IoT networks based on IDS. Section 4 describes the experimental procedure, findings, and the analysis of the suggested model using the UNSW-NB15 dataset. Finally, this paper is summarized in Section 5.

## 2 Related Work

IDS systems are classified into two categories depending on their signature-matching capabilities: signature-based intrusion detection systems (SIDS) and anomaly-based intrusion detection systems (AIDS) [11]. The term "signature-based intrusion detection" is often referred to as misuse detection or rule-based detection [12]. This approach compares incoming network data to established rules and detects threats based on previously observed characteristics. Note that signature-based approaches can detect known assaults but cannot detect unknown attacks. The second type of intrusion detection is anomaly-based, where the system observes regular network activity and uses it to define a model of normal network traffic. When it finds deviations from the regular traffic pattern, the behavior is classified as a malicious attack activity [13]. However, due to model building and feature engineering complexity, the technique may generate a higher false alarm rate if normal traffic cannot be adequately characterized. However, the advantage of anomaly-based intrusion detection approaches is that they use only samples of normal activity to create models and detect known and unknown attacks.

Existing artificial intelligence techniques can handle privacy protection and fraud anomaly detection in the network [14]. The objective of anomaly detection is to use machine learning algorithms to classify anomalous and normal data. Generally, machine learning-based solutions work by analyzing huge amounts of data generated by network traffic, host processes and users to detect suspicious activities using efficient algorithms [15]. Earlier research has demonstrated success with machine learning-based algorithms for intrusion detection systems. The authors of [16] proposed a system for feature selection that employed five distinct feature selection procedures using filter and wrapper approaches. According to the experimental data, the J48 classifier achieved maximum accuracy. In [17], Lakhan et al. used the CIC-IDS2017 dataset to test and evaluate three machine learning algorithms: Decision Jungle (DJ), Random Forest (RF), and Support Vector Machine (SVM).

Experimental results showed that SVM outperformed the other two machine learning algorithms. In [18], Chand et al. stacked the SVM with nine different classifiers to compare their performance to the solo SVM classifier. The stacked SVM method performed better, particularly when combined with Random Forest. Ling and Wu offer a method for intrusion detection in [19] that integrates various classifiers. The features were selected using Random Forest, and the best features were utilized for training a multi-classifier using SVM, Decision Tree, KNN, and Naive Bayes. Nkiama et al. introduced a recursive feature reduction technique in conjunction with a decision tree classifier [20] to identify significant features. This paper suggested that by lowering the number of characteristics, this strategy produced a high level of accuracy. Ambusaidi et al. introduced a mutual information-based technique for selecting the best feature analytically [21]. This method can handle data features that are linearly and nonlinearly dependent. In [22], Manickam et al. created a comprehensive ICMPv6-DDoS attack dataset to detect ICMPv6-DDoS attacks. They tested the dataset on five machine learning models, and the suggested dataset accurately represented attack traffic, with a high detection accuracy and low false-positive rate.

Traditional machine learning is severely limited since it is incapable of efficiently classifying complex and multi-dimensional intrusion data in the real-world complex network application environment. Deep learning-based NIDS has garnered considerable attention due to their superior performance in dealing with complex, large-scale data and extracting the underlying characteristics of traffic data; as a result, they have emerged as a potential solution for intrusion detection. Vinayakumar et al. [23] suggested a hybrid deep neural network (DNN) model for network and host-level event monitoring. Their research showed that this architecture outperformed classical machine learning classifiers previously implemented. Wu et al. [24] introduced the LuNet deep neural network architecture, which utilized CNN to learn spatial features from traffic data and RNN to learn temporal information. This approach can significantly enhance validation accuracy and decrease the percentage of false positives. Azizjon et al. [25] suggested a CNN-LSTM hybrid algorithm. To address the poor performance caused by imbalanced data, they used random sampling approaches to balance the data. The findings indicate that when trained on balanced data, the 1D-CNN 3-layer model outperformed imbalanced data in precision, recall, and F-score. Xu et al. [26] evaluated the time-related intrusion features and developed a unique DNN model composed of gated recurrent units (GRUs) and multi-layer perceptron (MLP). Kim et al. [27] constructed an intrusion detection model using a variation of the RNN LSTM-RNN. They extracted instances from the KDD Cup99 dataset to discover the super parameters and measure model performance. Roy et al. developed a unique Bi-LSTM network in [28], trained using the UNSW-NB15 dataset as a benchmark, attaining accuracy of above 95%. Sinha et al. introduced an architecture that merged CNN with bidirectional LSTM in [29], and the proposed model demonstrated a high detection rate and a relatively low false-positive rate.

Kathareios et al. [30] presented a two-stage real-time network IDS to reduce manual workload. The initial stage utilizes a shallow auto-encoder to perform adaptive unsupervised anomaly detection. A nearest-neighbour classifier was employed in the second stage to model the manual categorization and filter out false positives. Diro et al. [31] developed a distributed deep learning-based intrusion detection system (IDS) for fog computing IoT systems. The results indicated that a distributed parallel architecture achieves higher precision than a centralized model. Khan et al. [32] proposed a two-stage intrusion detection approach based on stacked auto-encoders. The first stage is classifying normal and pathological network traffic according to the classification probability value. The first stage's output is used as the input for the second stage's procedure of detecting normal and multi-class assaults. Al-Qatf et al. [33] suggested a self-taught learning strategy based on deep learning for acquiring characteristics and lowering the dimension. The suggested framework is constructed by recreating a new feature representation using the sparse auto-encoder mechanism and then feeding the features into the SVM algorithm to increase classification accuracy. Farahnakian et al. [34] proposed a stacked auto-encoder technique. The output of each auto-encoder at the current layer is used as the input to the following layer of auto-encoders. Yang et al. [35] introduced a framework, SAVAER, for learning the latent distribution of the original data using WGAN-GP. The model's decoder generates examples of rare and unknown threats, while the encoder is used to initialize the weights of the DNN's hidden layers and explore high-level feature representations. SAVAER-DNN was shown to be more suitable for data augmentation and to perform better than other state-of-the-art models. Sadaf et al. [36] present a method for detecting unauthorized attacks in the fog environment by utilizing the auto-encoder and isolation forest (IF) concepts. Souza et al. introduced a hybrid binary classification architecture based on DNN and the K-Nearest Neighbor algorithm for use in the fog computing layer [37]. The results indicated that the suggested hybrid strategy outperformed machine learning approaches for IoT systems in terms of precision. However, one significant limitation of the previous research is that they ignored the length of historical information's influence on performance. The network intrusions can be thought of as time-related events.

## 3 System Model

The distributed architecture of an anomaly-based IDS is depicted in Fig. 1. This strategy decentralizes the existing centralized computing architecture and distributes it to local fog nodes in three phases. The first phase involves preprocessing the data from the training dataset. It comprises feature mapping, which converts symbolic features to numeric ones, and feature normalization, which results in an optimal dataset. After preprocessing the data, the dataset is put into a Bi-LSTM and attention algorithm for categorization. The final phase collects data generated by IoT clusters at local fog nodes. The collected data is again preprocessed, and the optimized features are fed into the ideal model. Finally, the suggested model's effectiveness is evaluated using data supplied by IoT devices. If the predicted traffic pattern is normal, IoT devices are permitted to execute typical functions. However, if it detects suspicious activity, the administrator is notified, and the traffic is classified as abnormal. As a result, log details for such devices are forwarded to the cloud server, which maintains the global status of IoT devices.

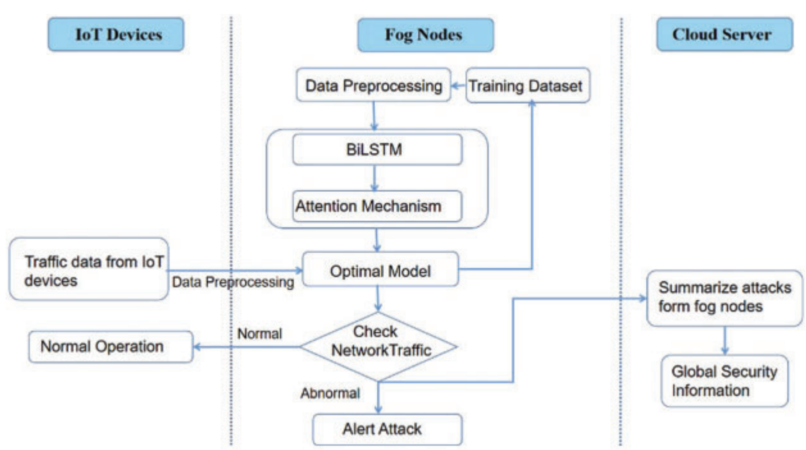


**Figure 1:** Architecture of the proposed IDS model

### *3.1 Bidirectional Long Short-Term Memory*

A recurrent Neural Network (RNN) can extract the temporal features from the input data. The cyclic structure of RNN can preserve historical information and provide sequence modelling capabilities. At timestamp $t$, the network layer accepts the input $x_t$ of the current timestamp and the hidden state of the previous timestamp $h_{t-1}$, so the current state $h_t$ can be defined as follows.

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{1}$$

where $W_{xh}$ and $W_{hh}$ are the weight matrices and $b$ is the bias variable, $\sigma$ represents the activation function. However, recurrent neural networks are prone to gradient vanish and gradient explosion. Compared with the basic RNN, LSTM is better at processing more extended sequence signal data and is widely used in sequence prediction and natural language processing tasks. Fig. 2 shows the structure of an LSTM cell.

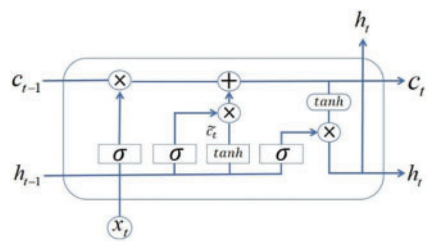


**Figure 2:** Structure of an LSTM cell

Compared with the basic RNN network, which has only one state vector $h_t$, LSTM adds a new state vector $c_t$, and at the same time introduces a gate mechanism to control the forgetting and refreshing of information through the gate control unit. Three gates in each LSTM unit control the internal information flow: input gate, forget gate, and output gate. $c_t$ can be used as the internal state vector memory, $h$ can be regarded as the output vector. The forget gate determines how much information the previous memory $c_{t-1}$ retained, which can be defined as:

$$g_f = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \tag{2}$$

The input gate determines how much information the current memory can hold. First, a new input vector $\tilde{c}_t$ is obtained by performing a nonlinear transformation on the current input $x_t$ and the output of the previous timestamp, which can be defined as:

$$\tilde{c}_t = tanh(W_c[h_{t-1}, x_t] + b_c) \tag{3}$$

The input gate $g_i$ determines the acceptance of the new input $\tilde{c}_t$, which can be defined as:

$$g_i = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{4}$$

Under the control of the forget gate and the input gate, LSTM selectively reads the memory of the previous timestamp and the new input of the current timestamp. The memory of can be defined as:

$$c_t = g_f c_{t-1} + g_i\tilde{c}_t \tag{5}$$

In LSTM, the output of the memory unit and the output information is under the control of the output gate. The output gate can be defined as:

$$g_o = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{6}$$

Therefore, the output of LSTM can be defined as:

$$h_t = g_o * tanh(c_t) \tag{7}$$

Bi-LSTM combines the forward and backward information, which makes up for the lack of contextual semantic information in LSTM. The bidirectional structure provides complete past and future context information for each moment in the input sequence of the output layer. As shown in Fig. 3, the Bi-LSTM network can better extract long-term and short-term dependent features and improve classification accuracy.

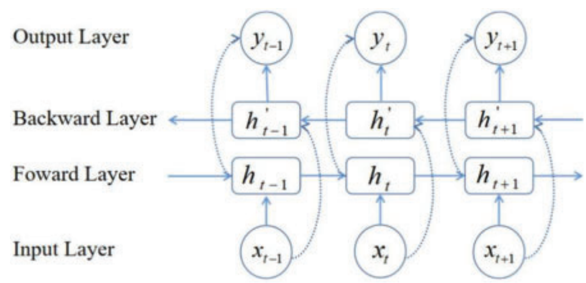


**Figure 3:** Structure of Bi-LSTM

### *3.2 Slice-Based Attention*

Based on our previous research, we know the traffic data is time-related. Therefore, traffic information of multiple adjacent moments is beneficial to learn the current traffic type. Therefore, we combined a few pieces of traffic data as slice traffic. Furthermore, dot-product attention is utilized to reduce calculation consumption during the optimized matrix multiplication, as shown in Fig. 4.

$$u_i = tanh(W_w h_i + b_w) \tag{8}$$

For each time step, hidden representation $u_i$ of hidden state $h_i$ can be obtained through a single layer perception.

$$\alpha_i = \frac{exp(u_i^T u_s)}{\sum_i exp(u_i^T u_s)} \tag{9}$$

Then, we use the similarity of $u_i$ and $u_w$ to evaluate the importance of traffic pieces at different moments $i$. Attention weight $\alpha$ can be calculated through a SoftMax function.

$$v = \sum_i \alpha_i h_i \tag{10}$$

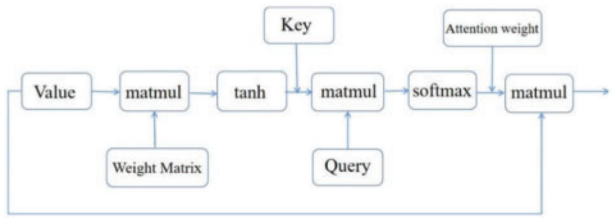


**Figure 4:** Illustration of dot-product attention

## 4 Experiment

### *4.1 Data Description*

To ensure the evaluation's efficiency, we used the UNSW-NB15 dataset, which is extensively used in intrusion detection investigations. The UNSW-NB15 intrusion dataset, created in 2015 by the Australian Centre for Cyber Security (ACCS) to generate a hybrid of real normal activities and synthetic contemporary attack behaviors in network traffic, is widely used as a benchmark dataset in the field of intrusion detection and prevention. The entire UNSW-NB15 dataset has been partitioned into a training and testing set. The training set has 175,341 records, whereas the testing set contains 82,332 records. Each record in the UNSW-NB15 dataset has 44 features: flow features, fundamental features, content features, time features, additional produced features, and labels, as shown in Tab. 1. The records are grouped into two broad categories: normal and attack. Attack records are further classified into nine categories: fuzzers, analysis, backdoors, denial of service, exploits, generic, reconnaissance, shellcode, and worms.

**Table 1:** Features of the UNSW-NB15 dataset

| No | Feature | Type | No | Feature | Type |
|---|---|---|---|---|---|
| 1 | id | Nominal | 23 | dtcpb | Integer |
| 2 | dur | Float | 24 | dwin | Integer |
| 3 | proto | Nominal | 25 | tcprtt | Float |
| 4 | service | Nominal | 26 | synack | Float |
| 5 | state | Nominal | 27 | ackdat | Float |
| 6 | spkts | Integer | 28 | smean | Integer |
| 7 | dpkts | Integer | 29 | dmean | Integer |
| 8 | sbytes | Integer | 30 | trans_depth | Integer |
| 9 | dbytes | Integer | 31 | response_body_len | Integer |
| 10 | rate | Integer | 32 | ct_srv_src | Integer |
| 11 | sttl | Integer | 33 | ct_state_ttl | Integer |
| 12 | dttl | Integer | 34 | ct_dst_ltm | Integer |
| 13 | sload | Float | 35 | ct_src_dport_ltm | Integer |
| 14 | dload | Float | 36 | ct_dst_sport_ltm | Integer |
| 15 | sloss | Integer | 37 | ct_dst_src_ltm | Integer |
| 16 | dloss | Integer | 38 | is_ftp_login | Binary |
| 17 | sinpkt | Integer | 39 | ct_ftp_cmd | Integer |
| 18 | dinpkt | Integer | 40 | ct_flw_http_mthd | Integer |

(Continued)

**Table 1:** Continued

| No | Feature | Type | No | Feature | Type |
|---|---|---|---|---|---|
| 19 | sjit | Float | 41 | ct_src_ltm | Integer |
| 20 | djit | Float | 42 | ct_srv_dst | Integer |
| 21 | swin | Integer | 43 | is_sm_ips_ports | Binary |
| 22 | stcpb | Integer | 44 | attack_cat | Nominal |

### *4.2 Data Preprocessing*

Data preprocessing is required to meet the input requirements for deep learning methods. This includes but is not limited to numerical processing and feature normalization.

#### *4.2.1 Numerical Processing*

The process of converting symbolic features to numerical data is called "feature transformation". This stage is important since the neural network's input is a digital matrix, and numerical operations are the sole option for neural networks in deep learning. As a result, we digitize the symbolic characteristics using the one-hot encoding method.

#### *4.2.2 Normalization*

Although the data set has been numerically processed, the range of minimum and maximum values for distinct feature data is highly diverse, significantly reducing the reliability of training results. The numerical data must be normalized to eliminate the big difference. As shown in Eq. (11), min-max standardization is used to translate the various numerical values of the features to the range [0, 1] without disturbing the linear relationship between the original data.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{11}$$

### *4.3 Evaluation Matrix*

Tab. 2 defines the confusion matrix to evaluate the model's performance. The dataset's samples can be classified into four types: True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). TP shows the number of anomalous attacks; FP denotes the number of normal samples incorrectly classified as anomalous; TN denotes the number of normal records correctly classified as normal; FN denotes the number of anomalous records incorrectly classified as normal. Various evaluation measures such as accuracy, precision, recall (detection rate), and f1 score are used to validate the proposed model.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{12}$$

$$Precision = \frac{TP}{TP + FP} \tag{13}$$

$$Recall = \frac{TP}{TP + FN} \tag{14}$$

$$F1\_score = \frac{2 \times Precision \times Recall}{Precision \times Recall} \tag{15}$$

**Table 2:** Confusion matrix

| Predicted Class | Actual Class | |
|---|---|---|
| | Anomaly | Normal |
| Anomaly | TP | FP |
| Normal | FN | TN |

### *4.4 Experimental Process*

All experiments are run on a computer equipped with an Intel(R) Core (TM) i7–10875H CPU running at 2.30 GHz, 16.0 GB RAM, and one NVIDIA GeForce RTX 2070 GPU. The programming environment is Python 3.7.4 and Tensorflow 2.1.0. Tab. 3 shows the parameters needed to build the model. To meet the criterion of the input dimension for Bi-LSTM, the dataset must first be reshaped into a three-dimensional shape. Following data preprocessing, the 44-dimensional features are converted to 196-dimensional features. After that, all 196 features are concatenated into a single piece of data called a vector. Thus, the model's final input has the shape (batch-size, timestep, 196), where batch-size is a hyper-parameter indicating the number of samples supplied to the model at a given time and timestep is the duration of historical events. Next, a dense layer is coupled to the input layer to construct the attention mechanism. This dense layer contains the same number of hidden units as the input layer. After that, two Bi-LSTM layers are layered together for processing time-series data. Each timestep generates an output, and all steps receive dot-product attention. Finally, dense layers are connected to the output of the attention layer, with only one categorization unit in the output layer.

**Table 3:** Parameters of the proposed model

| Parameter | Values |
|---|---|
| Optimizer | Adam |
| Learning rate | 0.01 |
| Loss function | Binary cross-entropy |
| Batch size | 1024 |
| Monitor | Val-accuracy |
| Activation unit | Sigmod |
| Dropout | 0.5 |
| Epochs | 160 |
| Input layer size | 196 |

## 5 Experimental Results and Analysis

Fig. 5 illustrates the model's optimal training and validation accuracy. We set the ratio of training set and validation set to 7:3. The accuracy rate increases significantly when neural network parameters are optimized at the start of training. This also demonstrates that the model can quickly determine the gradient decline direction at this point. After around 15 epochs, the model determines the ideal set of parameters for the current conditions, resulting in the maximum accuracy rate. We attempt to train the model again and see that the accuracy rate does not continue to increase. Within around 40 epochs, the accuracy rate swings within a narrow range without seeing any major decrease. After training, validation accuracy did not improve from 0.991. The model's accuracy is 99.05 percent, the precision is 98.9 percent, the detection rate is 99.36 percent, and the f1 score is 99.15 percent on the testing dataset, indicating that our model has a high detection rate.

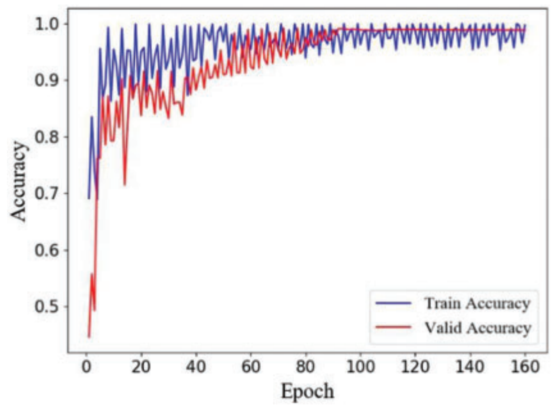


**Figure 5:** Train and test accuracy score of the model

As illustrated in Fig. 6, as model training proceeds, our proposed method's loss on the training and validation datasets converges. As the accuracy of our suggested model increased, the loss of our proposed model decreased quite quickly in the first 20 epochs. This condition exists because of model parameter optimization. As can be observed, loss tends to converge around the 18th epoch. While there were some minor ups and downs during the follow-up training process, the diversification is not immediately apparent. The training loss should converge to around 0.11 after about 40 epochs. Meanwhile, the validation loss should drop to around 0.08.

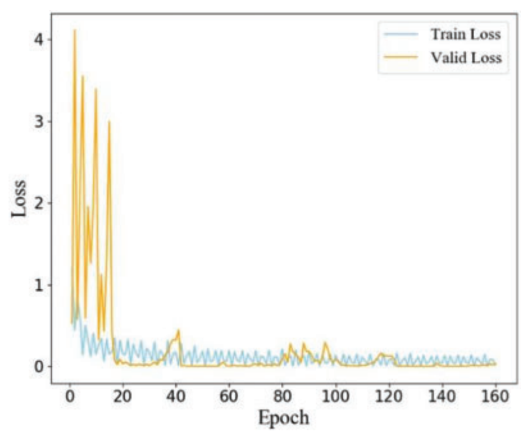


**Figure 6:** Loss score of the model

To evaluate the proposed model's performance, we execute network intrusion detection on the UNSW-NB15 dataset using seven classic machine learning techniques (Logistic Regression, SVM, Naive Bayes, K-Nearest Neighbor, Decision Tree, Random Forest, and Adaboost). These techniques have been widely utilized to detect intrusions. Comparative experimental findings are provided in

Fig. 7. Although the improvement in precision is not evident compared to other conventional machine learning models, our suggested framework outperformed well-known classifiers in terms of overall accuracy, recall, and f1 score.

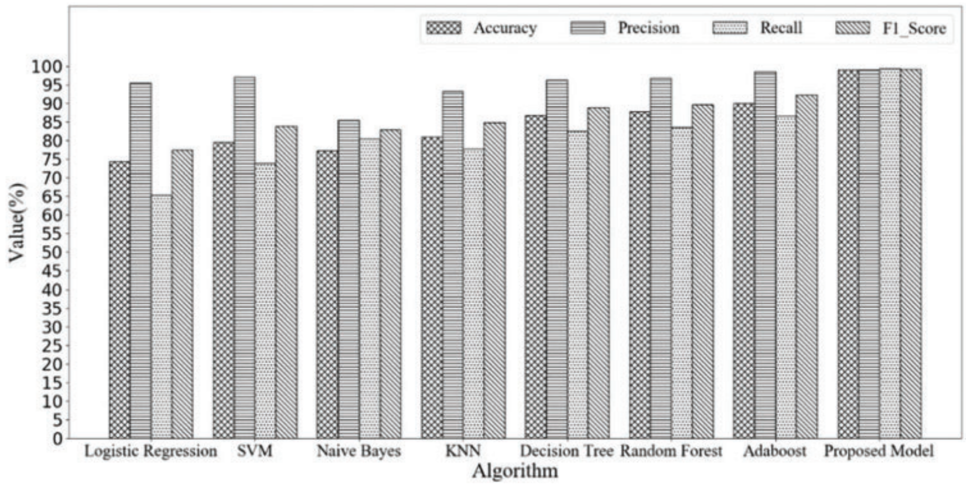


**Figure 7:** Performance comparison between the machine learning models and proposed model

Additionally, the suggested strategy is compared to previous work using the UNSW-NB15 dataset, as illustrated in Tab. 4. The suggested framework achieves the highest overall accuracy, detection rate, and f1 score on the UNSW-NB15 dataset compared to the other seven classification models. However, its precision is slightly lower than the Bi-LSTM proposed in [28]. The preceding comparative experimental results demonstrate unequivocally that the Bi-LSTM with attention mechanism model is superior at detecting network intrusions.

**Table 4:** Performance comparison of different classification models

| Algorithm | Accuracy | Precision | Recall | F1_Score |
|---|---|---|---|---|
| DNN [20] | 76.5 | 94.6 | 69.5 | 80.1 |
| CNN [21] | 91.20 | 87.53 | 96.17 | 91.59 |
| Auto-Encoder [29] | 89.71 | 89.74 | 89.85 | 89.79 |
| LuNet [21] | 97.40 | N/A | 98.18 | N/A |
| CNN + LSTM [22] | 89.93 | 86.15 | 95.15 | 90.43 |
| SAVAER-DNN [32] | 93.01 | 95.21 | 91.94 | 93.54 |
| BiLSTM [25] | 95.71 | 100 | 96.00 | 98.00 |
| **Proposed Model** | **99.05** | **98.96** | **99.36** | **99.15** |

## 6 Conclusion

This paper conducts intrusion detection on fog nodes for IoT applications. The intrusion detection is accomplished by imbuing local fog nodes with intelligence via the Bi-LSTM and attention algorithms. Local fog nodes identify attacks based on traffic generated by IoT devices and send them to cloud servers to summarize the global security condition of IoT applications. We develop a two-layer bidirectional long short-term memory network with an attention mechanism to distinguish traffic data, considering that traffic data is time-related. It achieves the highest accuracy of 99.05 percent, the highest detection rate of 99.36 percent, and the highest f1 score of 99.154 percent on the UNSW-NB15 dataset. In every case, the proposed technique outperforms traditional machine learning classifiers. Furthermore, when compared to other approaches, our proposed design exceeds state-of-the-art models. However, due to the high computational cost of complicated DNN structures, they were not trained using the other benchmark IDS datasets in this research.

**Funding Statement:** The authors are thankful for the Beijing Natural Science Foundation (No. 4212015), Natural Science Foundation of China (No. 61801008), China Ministry of Education-China Mobile Scientific Research Foundation (No. MCM20200102), China Postdoctoral Science Foundation (No. 2020M670074), Beijing Municipal Commission of Education Foundation (No. KM201910005025). The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through large groups Project under Grant Number RGP.2/201/43.

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] J. Pacheco, V. H. Benitez, L. C. Félix-Herrán and P. Satam, "Artificial neural networks-based intrusion detection system for internet of things fog nodes," *IEEE Access*, vol. 8, pp. 73907–73918, 2020.

[2] S. Prabavathy, K. Sundarakantham and S. M. Shalinie, "Design of cognitive fog computing for intrusion detection in internet of things," *Journal of Comm. and Networks*, vol. 20, no. 3, pp. 291–298, 2018.

[3] M. Haus, M. Waqas, A. Y. Ding, Y. Li, S. Tarkoma *et al.,* "Security and privacy in device-to-device (D2D) communication: A review," *IEEE Comm. Surveys & Tut*, vol. 19, no. 2, pp. 1054–1079, 2017.

[4] M. Waqas, M. Ahmed, J. Zhang and Y. Li, "Confidential information ensurance through physical layer security in device-to-device communication," in *IEEE Global Communications Conf.*, Abu Dhabi, UAE, pp. 1–7, 2019.

[5] J. Wan, M. Waqas, S. Tu, S. M. Hussain, A. Shah *et al.,* "An efficient impersonation attack detection method in fog computing," *Computers, Materials & Continua*, vol. 68, no. 1, pp. 267–281, 2021.

[6] M. Waqas, Y. Niu, M. Ahmed, Y. Li, D. Jin *et al.,* "Mobility-aware fog computing in dynamic environments: Understandings and implementation," *IEEE Access*, vol. 7, pp. 38867–38879, 2019.

[7] S. Tu, M. Waqas, S. U. Rehman, T. Mir, Z. Halim *et al.,* "Social phenomena and fog computing networks: A novel perspective for future networks," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 32–44, 2022.

[8] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[9] A. Lakhan, M. A. Mohammed, O. I. Obaid, C. Chakraborty, K. H. Abdulkareen *et al.,* "Efficient deep-reinforcement learning aware resource allocation in SDN-enabled fog paradigm," *Automated Software Engineering*, vol. 29, no. 1, pp. 1–25, 2022.

[10] S. Tu, M. Waqas, S. Rehman, T. Mir, G. Abbas *et al.,* "Reinforcement learning assisted impersonation attack detection in device-to-device communications," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1474–1479, 2021.

[11] B. Zhang, M. Waqas, S. Tu, S. M. Hussain and S. U. Rehman, "Power allocation strategy for secret key generation method in wireless communications," *Computers, Materials & Continua*, vol. 68, no. 2, pp. 2179–2188, 2021.

[12] Y. N. Nguimbous, R. Ksantini and A. Bouhoula, “Anomaly-based intrusion detection using auto-encoder,” in *Int. Conf. on Software, Telecommunications and Computer Networks*, Split, Croatia, pp. 1–5, 2019.
[13] M. Azizjon, A. Jumabek and W. Kim, “1D CNN based network intrusion detection with normalization on imbalanced data,” in *Int. Conf. on Artificial Intelligence in Information and Communication (ICAIIC)*, Fukuoka, Japan, pp. 218–224, 2020.
[14] S. Tu, M. Waqas, Z. Halim, S. U. Rehman, G. Abbas *et al.,* “The role of artificial intelligence and machine learning in wireless networks security: Principle, practice and challenges,” *Artificial Intelligence Review*, pp. 1–47, 2022.
[15] A. H. Azizan, S. A. Mostafa, A. Mustapha, C. F. M. Foozy, M. H. A. Wahab *et al.,* “A machine learning approach for improving the performance of network intrusion detection systems,” *Annals of Emerging Technologies in Computing*, vol. 5, no. 5, pp. 201–208, 2021.
[16] H. M. Anwer, M. Farouk and A. Abdel-Hamid, “A framework for efficient network anomaly intrusion detection with features selection,” in *9th Int. Conf. on Information and Communication Systems (ICICS)*, Irbid, Jordan, pp. 157–162, 2018.
[17] A. Lakhan, M. A. Mohammed, J. Nedoma, R. Martinek, P. Tiwari *et al.,* “Federated-learning based privacy preservation and fraud-enabled blockchain IoMT system for healthcare,” *IEEE Journal of Biomedical and Health Informatics*, 2022.
[18] N. Chand, P. Mishra, C. R. Krishna, E. S. Pilli and M. C. Govil, “A comparative analysis of SVM and its stacking with other classification algorithm for intrusion detection,” in *Int. Conf. on Advances in Computing, Communication, & Automation (ICACCA)*, Dehradun, India, pp. 1–6, 2016.
[19] J. Ling and C. Wu, “Feature selection and deep learning-based approach for network intrusion detection,” in *3rd Int. Conf. on Mechatronics Engineering and Info. Technology*, Dalian, China, Atlantis Press, 2019.
[20] H. Nkiama, S. Zainudeen and M. Saidu, “A subset feature elimination mechanism for intrusion detection system,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 4, pp. 148–157, 2016.
[21] M. A. Ambusaidi, X. He, P. Nanda and Z. Tan, “Building an intrusion detection system using a filter-based feature selection algorithm,” *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986–2998, 2016.
[22] S. Manickam, A. H. B. AIghuraibawi, R. Abdullah, Z. A. A. Alyasseri, K. H. Abdulkareem *et al.,* “Labelled dataset on distributed denial-of-service (DDoS) attacks based on internet control message protocol version 6 (ICMPv6),” *Wireless Communications and Mobile Computing*, vol. 2022, pp. 8060333, 2022.
[23] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat *et al.,* “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, vol. 7, pp. 41525–41550, 2019.
[24] P. Wu and H. Guo, “LuNet: A deep neural network for network intrusion detection,” in *IEEE Symp. Series on Computational Intelligence (SSCI)*, Xiamen, China, pp. 617–624, 2019.
[25] M. Azizjon, A. Jumabek and W. Kim, “1D CNN based network intrusion detection with normalization on imbalanced data,” in *Int. Conf. on Artificial Intelligence in Information and Communication (ICAIIC)*, Fukuoka, Japan, pp. 218–224, 2020.
[26] C. Xu, J. Shen, X. Du and F. Zhang, “An intrusion detection system using a deep neural network with gated recurrent units,” *IEEE Access*, vol. 6, pp. 48697–48707, 2018.
[27] J. Kim, J. Kim, H. L. Thi Thu and H. Kim, “Long short-term memory recurrent neural network classifier for intrusion detection,” in *Int. Conf. on Platform Technology and Service*, Jeju, Korea, pp. 1–5, 2016.
[28] B. Roy and H. Cheung, “A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network,” in *28th Int. Telecommunication Networks and Applications Conf. (ITNAC)*, Sydeny, NSW, Australia, pp. 1–6, 2018.
[29] J. Sinha and M. Manollas, “Efficient deep CNN-BiLSTM model for network intrusion detection,” in *3rd Int. Conf. on Artificial Intelligence and Pattern Recognition*, New York, USA, pp. 223–231, 2020.
[30] G. Kathareios, A. Anghel, A. Mate, R. Clauberg and M. Gusat, “Catch it if you can: Real-time network anomaly detection with low false alarm rates,” in *16th IEEE Int. Conf. on Machine Learning and Applications (ICMLA)*, Cancun, Mexico, pp. 924–929, 2017.

[31] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for internet of things," *Future Generation Computer Systems*, vol. 82, no. 5, pp. 761–768, 2017.

[32] F. A. Khan, A. Gumaei, A. Derhab and A. Hussain, "A novel two-stage deep learning model for efficient network intrusion detection," *IEEE Access*, vol. 7, pp. 30373–30385, 2019.

[33] M. Al-Qatf, Y. Lasheng, M. Al-Habib and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection," *IEEE Access*, vol. 6, pp. 52843–52856, 2018.

[34] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in *20th Int. Conf. on Advanced Communication Technology (ICACT)*, Chuncheon, Korea, 2018.

[35] Y. Yang, K. Zheng, B. Wu, Y. Yang and X. Wang, "Network intrusion detection based on supervised adversarial variational auto-encoder with regularization," *IEEE Access*, vol. 8, pp. 42169–42184, 2020.

[36] K. Sadaf and J. Sultana, "Intrusion detection based on autoencoder and isolation forest in fog computing," *IEEE Access*, vol. 8, pp. 167059–167068, 2020.

[37] C. Souza, C. B. Westphall and R. B. Machado, "Hybrid approach to intrusion detection in fog-based IoT environments," *Computer Networks*, vol. 180, pp. 107417, 2020.