International Journal of Safety and Security Engineering
Vol. 14, No. 5, October, 2024, pp. 1547-1558
Journal homepage: http://iieta.org/journals/ijsse

Deep Learning Based Intrusion Detection System of IoT Technology: Accuracy Versus Computational Complexity

Maryam Lazim Mutleg*, Ali Majeed Mahmood, Muna Mohammed Jawad Al-Nayar

Control and Systems Engineering Department, University of Technology-Iraq, Baghdad 10066, Iraq

Corresponding Author Email: cse.22.21@grad.uotechnology.edu.iq

https://doi.org/10.18280/ijsse.140522

Received: 13 August 2024
Revised: 10 October 2024
Accepted: 17 October 2024
Available online: 31 October 2024

Keywords:
cyber threats, intrusion detection, IoT, LSTM, GS optimization, deep learning, attention mechanism, GRU

ABSTRACT

The Internet of Things' (IoT) rapid growth has resulted in a rise in vulnerabilities, making safeguarding IoT systems against intrusions and illegal access a top priority. Intrusion Detection Systems (IDS) are essential for keeping an eye out for irregularities in network traffic. However, the challenge lies in the IDS's ability to detect attacks within high-speed networks while minimizing computational complexity promptly. To improve detection efficiency in IoT networks, we proposed lightweight detection models in this paper that are based on Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), Gated Recurrent Unit (GRU), and a GRU-based self-attention mechanism. The Grid Search (GS) algorithm optimizes the models by adjusting the hyperparameters, such as the learning rate and the number of hidden units. The proposed models are evaluated using the ToN-IoT dataset. The achieved detection accuracy for all models is as follows: 97% for GRU, 98.1% for LSTM, 98.4% for Bi-LSTM, and 99% for the GRU-based self-attention mechanism. Furthermore, the GRU-based self-attention mechanism has fewer parameters, which leads to a significant saving in classification time of up to 84% compared to GRU. These findings demonstrate that the GRU-based self-attention mechanism is superior in accuracy and computational efficiency, which makes it particularly effective for real-time intrusion detection in IoT networks.

1. INTRODUCTION

One of the most popular terms in information technology is IoT. The IoT will turn physical objects into intelligent virtual objects in the future. Many of the things we see around us will be connected to the internet in one way or another under the IoT paradigm [1]. Thanks to this technology, billions of intelligent devices, also referred to as things can gather a wide range of data about themselves and/or their surroundings via a variety of sensors. After that, they can share this information with authorized parties for a variety of uses, like bettering commercial services or operations or managing and monitoring industrial facilities [2]. Threats to the IoT are growing daily as it becomes more widespread. There are billions of appliances on the Internet right now. Attackers will use the IoT as a target to carry out harmful activities and increase the attack surface of IoT networks through exponential growth [3]. Since many businesses lost physical space as a result of cyberattacks, the damage they caused increased. A sophisticated instrument is needed to identify malicious activity in IoT devices' smart architecture [4]. Multiple devices connected to the internet pose a serious security risk [5]. Because certain data transferred across the IoT system is not secure, harmful attacks can occur [6]. Many studies are being conducted to enhance IoT security to create a reliable and secure network [5]. The three basic principles of secrecy, integrity, honesty, and authenticity should be adhered to when creating solutions. It is crucial to develop various security methods for IoT, such as IDS. Network-based and host-based are the two main categories of intrusion detection systems [7]. The enormous amount of traffic data is one of the main barriers to IDS real-time performance on high-speed networks. Conventional intrusion detection systems find it challenging to monitor traffic data quickly because they often focus on enhancing detection performance rather than paying enough attention to timeliness. Attackers now have a window of opportunity. Access to IDS solutions is also essential for processing traffic data in real time [8]. In time series data processing, a Recurrent Neural Network (RNN) is frequently employed as an advanced deep learning model. An RNN's benefit is its capacity to connect past and current data [9]. Specifically, one kind of RNN that can learn from long dependencies is the Long Short-Term Memory (LSTM) network [10]. It resolves the vanishing gradient issue that arises when training standard RNNs [11]. To improve LSTM-based intrusion detection effectiveness, an optimization algorithm like GS is used to choose the best model parameters. This paper proposed an IDS to detect intrusions in IoT systems as illustrated in Figure 1. The performance metrics are the accuracy and classification time. This paper's primary contributions consist of:

-Three RNN detection models are assessed individually including LSTM, Bi-LSTM, GRU, and GRU-based attention mechanism. The goal is to deploy a low classification time

algorithm at the highest accuracy.

-The ToN-IoT dataset is more impressive and highly accurate which is utilized in this study, rather than most used datasets like UNSW-NB15 and Edge-IIoT, which have been used in previous studies.

-Model hyperparameters are optimized by using the Grid Search optimization technique. Here the number of hidden units for each layer is optimized, which leads to minimizing complexity and classification time.

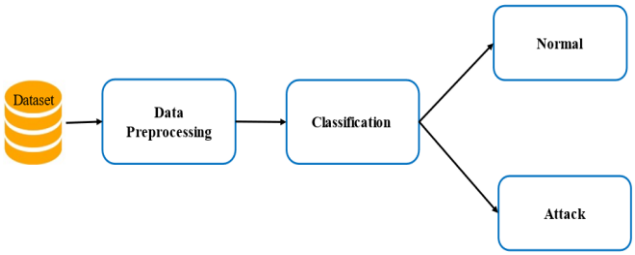

Figure 1. Block diagram for IDS

The rest of the paper is arranged as follows. The second section explores work related to the suggested approach. The third section provides the suggested IDS structure for the IoT system. The fourth section defines the ToN_IoT dataset. Section 5 defines the evaluation metrics for model performance. Section 6 discusses the analysis and outcomes of the proposal. In the last section, we provide future direction as we conclude our study.

2. RELATED WORK

Using DL approaches, this work suggests an IDS that achieves high detection speed and good accuracy. As listed below, many research investigations have been conducted in this area:

An AutoEncoder (AE) architecture for feature learning using a Deep Neural Network (DNN) was provided in reference [12]. The CICIDS2017 and NSL-KDD datasets were used for validation to classify regular and Distributed Denial of Service (DDoS) attack traffic. The accuracy of NSL-KDD and CICIDS2017, according to the experimental results, is 98.43% and 98.92%, respectively. A smaller network that avoids overfitting, minimum reconstruction error, and avoidance of exploding and vanishing gradients are the outcomes of the improvements suggested in this study.

Furthermore, Diaba and Elmusrati [13] used CNN and GRU algorithms to create a hybrid classification method for detecting DDoS assaults on smart grids. The CICIDSS2017 dataset was used to assess the model. Results indicate that the proposed approach has a 99.9% cyberattack detection rate and a 99.7% classification accuracy. Although simulation results show a high accuracy rate, however, the model complexity leads to computationally intensive and this is considered a challenge to the real-time smart grid.

Additionally, Fenani and Semchedine [14] suggested a novel Smart Intrusion Detection (SID) approach based on Federated Learning (FL) in IoT edge computing, where computational load is distributed to network edges for efficient resource utilization. The effectiveness of these models is evaluated using three established IoT field datasets (IoTID20, IoT23, and N-BaIoT) and three well-known deep learning models (DNN, CNN, LSTM). Individual users train their local models using the anomaly-based SID method, sending only parameter updates to the server. The global model is then created by adding up these parameters. IoT users use their local datasets to further train an updated global model that is provided by the central server during each FL training cycle. This methodology optimizes the global model while maintaining the confidentiality of IoT devices.

Saurabh et al. [15] proposed two models of deep learning (Stacked and Bi-Directional LSTM) to identify network and application layer attacks using the UNSW_NB15(they selected 13 features as the most important), and Bot_IoT (with 10 features that were pre-selected) datasets. Several studies have been suggested using Machine Learning (ML) techniques, this paper concluded that these variants are performed the same as the DNN models which are better in terms of accuracy. The experimental results successfully detect several attacks with 99.99% accuracy.

A Knowledge Distillation (KD)--based variant of the Convolutional Neural Network and Gated Recurrent Unit (CNN-GRU) was created by Chen et al. [16]. Using a real-global dataset from an ingenious PiCar testbed and NSL-KDD, the efficacy of their algorithms was verified. It is acknowledged that accuracy and privacy must be traded off. The findings demonstrate the high level of effectiveness and resilience of the DP Federated KD-based IDS for third-party IoT systems of a smart airport.

To extract the most significant characteristics, the study in reference [17] also presented a Self-Attention Convolutional Neural Network (SACNN) structure and an appropriate function extraction technique. Using the X-IIoTID and Edge IIoTset datasets, the performance of the recommended SACNN architecture was evaluated to identify fraudulent activity in IIoT networks. These datasets covered the behaviors of present-day IIoT communication protocols, the features of modern-day devices, distinctive sorts of attacks, and more than a few assault situations.

The hybrid model proposed by Latif et al. [18] is based on seven pre-trained Convolutional Neural Networks (CNN): Xception, VGG16, VGG19, Inception, InceptionResNetV2, EfficientNetB7, and EfficientNetV2L. It is based on Deep Transfer Learning (DTL). The model's hyperparameters have been adjusted using Genetic Algorithms (GA) optimization. The outcomes of the top five models are combined using the bootstrap aggregation ensemble technique. The validation was achieved on a pre-trained Edge_IIoT dataset, and a 100% accuracy rate was achieved to determine the patterns in the data gathered and find the malicious traffic related to the assaults.

Sundaram et al. [19] presented a mixed feature selection method (information gain and recursive feature elimination). A cascaded LSTM classifier that detects binary and multiclass attack categories is employed to enhance attack classifications. Results from experiments using the UNSW-NB15 and NSL-KDD datasets, achieve an accuracy of 99.30% and 98.96% respectively. A workable plan for raising intrusion detection in IoT networks' efficacy and accuracy has been offered.

Strikingly, a Bi-LSTM network coupled with the attention mechanism provides a lower accuracy of 90.73%. By focusing on the most important features of the input data, attention mechanisms are used to highlight the sequences that are most suggestive of intrusions [20]. Bi-LSTMs leverage patterns from both historical and prospective settings to enhance their efficiency further. The decreased accuracy observed might be due to overfitting of the model or subtle details inherent in the dataset employed.

Table 1. An overview of related works

Ref.	Dataset	Classifier	Accuracy
[12]	CICIDS2017 NSL-KDD	AE+DNN	98.43% 98.92%
[13]	CICIDS-2017	CNN+GRU	99.7 %
[14]	N-BaIoT IoTID20, IoT23	LSTM, DNN, CNN	99%
[15]	UNSW_NB15 Bot_IoT	Stacked LSTM, Bi-LSTM	96.60% 96.41% 99.99%
[16]	NSL–KDD, real-world dataset	KD-based CNN-GRU	-
[17]	Edge-IIoTset and X-IIoTID	SACNN	99.62%
[18]	Edge_IIoT	CNN	100%
[19]	NSL-KDD UNSW-NB15	Cascaded LSTM	99.30% 98.96%
[20]	NSL-KDD	Attention mechanism with Bi-LSTM network	90.73%
[21]	KDD99	LSTM-No reduction (all features)	96.51%
[22]	BoT-IoT, ToN-IoT	CNN	99.8%
[23]	KDDCup 99	DNN	95% to 99%
[24]	NSL-KDD	AutoIF	95.4%
[25]	KDD Cup 99, NSL-KDD	SCAE+SVM	99.89%

The model put forward in reference [21] detects threats by using LSTM-based deep learning techniques. Mutual Information (MI) and Principal Component Analysis (PCA) are techniques for feature selection and dimensionality reduction. The studies' findings show that PCA-based models are the most accurate in both training and testing for binary and multiclass categorization. Their accuracy dropped to 96.51% when all features were used for training, though. Similarly, the approach by Gaber et al. [22] blends CNNs and Kernel Principal Component Analysis (KPCA), the first one is used for feature extraction and the second is for attack classification. Although it is remarkable to attain high accuracy and low false negative rates. However, it is computationally complex and leads to latency which is crucial, especially in real-time systems.

Vinayakumar et al. [23] used Deep Neural Networks (DNNs) in IDS to detect and classify evolving cyberattacks, emphasizing the importance of updated malware datasets. They compared DNNs with traditional classifiers across various benchmark datasets, finding that DNNs are more effective at capturing complex IDS patterns. The study introduced scale-hybrid-IDS-AlertNet, as a scalable framework for real-time network and host-level monitoring using DNNs. Using hyperparameter selection techniques on the KDDCup 99 dataset, the ideal network topologies and parameters were determined. Experiments ranging from 0.01 to 0.5 learning rates over 1,000 epochs proved the efficacy of the DNN model, which was subsequently applied to additional datasets to confirm its scalability and robustness.

Sadaf and Sultana [24], addressed the vulnerabilities of fog devices in Fog Computing, which act as intermediaries for local processing and reduce communication delays between end-users and the cloud. Emphasizing the critical importance of authenticating network traffic, given the sensitive nature of the information passing through, a novel IDS named AutoIF is introduced. This DL-based technique combines Isolation Forest (IF) and AE specifically for fog environments and aims for binary classification of packets. The approach achieves a remarkable accuracy of 95.4% when tested on the NSL-KDD dataset.

In the study of Yu et al. [25], the focus was on utilizing deep learning techniques to automatically derive crucial feature representations and achieve efficient high detection rates. A novel method employing a stacked contractive autoencoder (SCAE) is introduced for unsupervised feature extraction. Through this method, enhanced and robust low-dimensional features are autonomously extracted from raw network traffic. The SCAE and a Support Vector Machine (SVM) classification algorithm have been combined to create a novel cloud intrusion detection system. By combining deep and shallow learning methodologies, our combined SCAE+SVM methodology greatly reduces analytical overhead. Experiments on two well-known intrusion detection assessment datasets, KDD Cup 99 and NSL-KDD, show that this suggested approach performs better in terms of detection effectiveness than three other state-of-the-art approaches. As shown in Table 1, the use of DL approaches in IDS has been thoroughly investigated in several studies.

3. THE PROPOSED APPROACH'S DESIGN

The goal of this effort is to develop an effective deep learning method to address the issue of cyberattacks in Internet of Things networks. The selection of LSTM, BI-LSTM, GRU, and GRU-based self-attention mechanisms for the IDS was based on their ability to handle sequential data, lightweight models, less complex compared to CNN, more accurate, and it was used in other applications and was of high accuracy. The choice of LSTM was made because of its capacity to recognize long-term dependencies in network traffic, which is an essential component for spotting temporal patterns in intrusion detection. While Bi-LSTM model utilizes data from both past and future sequences to enhance detection accuracy by considering the entire context of network behaviors. Switching to GRU provides a computationally cost-effective alternative to LSTM, which makes it suitable for resource-constrained IoT devices while maintaining effective pattern recognition. It was simpler for the model to concentrate on the most crucial elements of the input sequence when self-attention mechanisms based on GRU were added. By including self-attention, the model can decide to focus on important segments of the network flow, enhancing detection performance. The whole framework of the suggested method is shown in Figure 2. The following subsections contain the design's specifics.

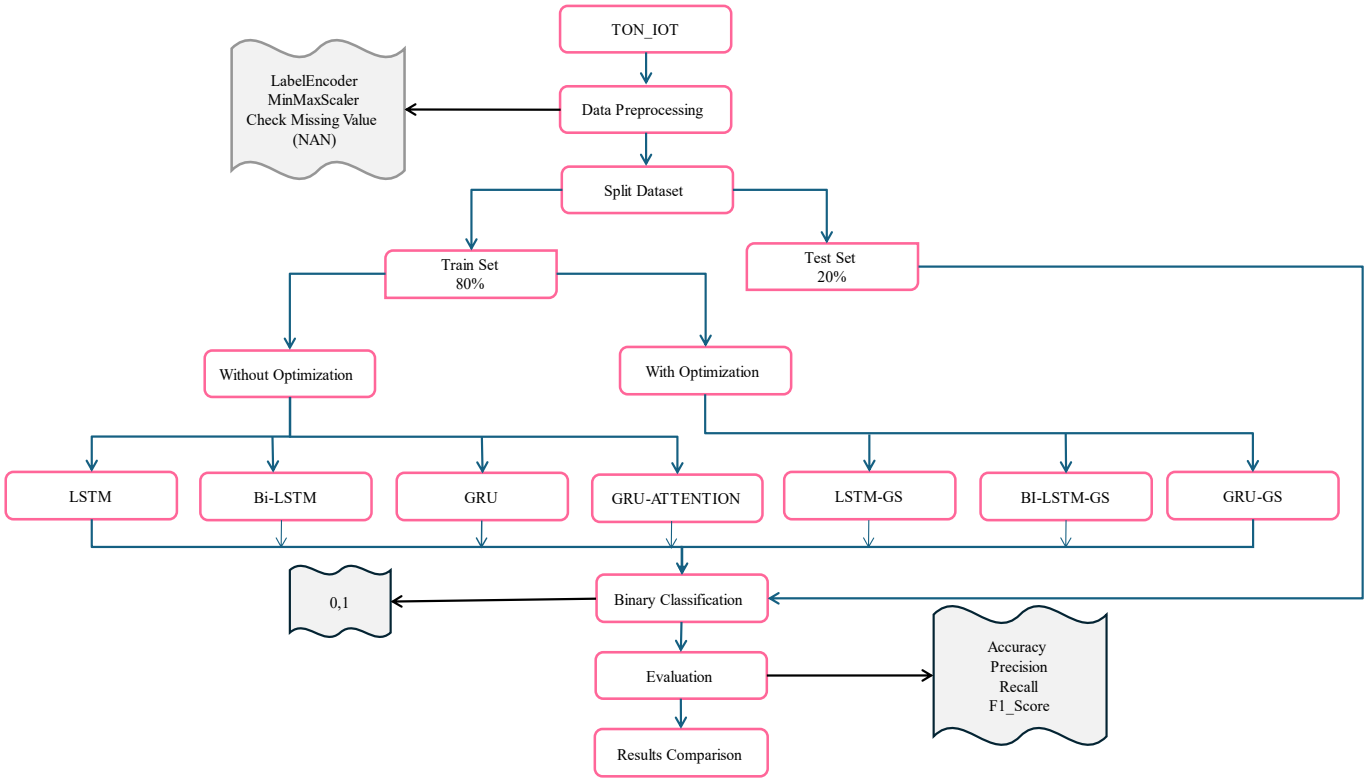

Figure 2. The entire flowchart of the proposed approach

3.1 The concepts of LSTM

One of the most significant varieties of RNN that can learn from lengthy dependencies is Long Short-Term Memory (LSTM). A typical recurrent cell cannot learn as well as an LSTM cell. In particular, LSTMs are used to preserve long-term data sequences using a memory cell structure [26]. The memory cell, also known as the cell state, is the basic building block of the LSTM. It uses gate activation functions to control the flow of data into, through, and out of the network [27]. As shown in Figure 3, the memory cell has three gates: input, forget, and output gates.

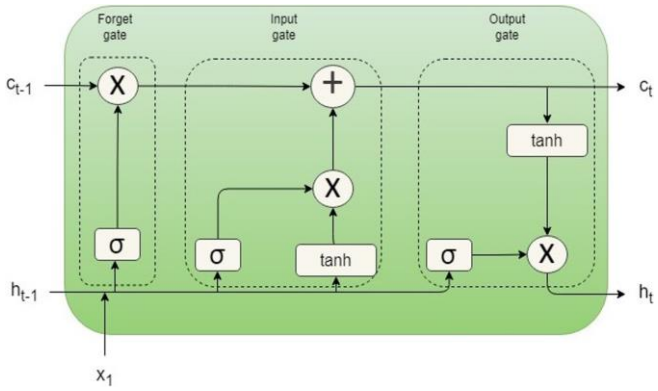

Figure 3. The architecture of standard LSTM [27]

The forget gate decides which data in the cell state can be removed. When the value is 0, this information is destroyed; however, when the forget gate, ft, equals 1, it is retained. The input gate controls which fresh information can be added to the cell state. Based on the state of the cell, the output gate controls the data that can be output [28]. The mathematical expressions for an LSTM cell are as follows.

3.1.1 Forget gate

It outlines which data from the prior cell state should be removed and which should be retained. Information flow from the preceding cell state is controlled by it. The following equations define it:

$$ft = \sigma(Wf[ht-1, Xt] + bf) \quad (1)$$

where, ft represents the output of the Foregate gate, σ is the sigmoid function, Wf and $ht{-}1$ denote the related weight matrices for the foregate gate, bf is the bias and Xt refers the input to the current timestamp.

3.1.2 Input gate

It controls how much new data enters the LSTM cell.

$$It = \sigma(Wi[ht-1, Xt] + bi) \quad (2)$$

$$C = tanh(Wc[ht-1, Xt] + bc) \quad (3)$$

where, Wi and Wi are wights of the input layer, bi, and bc are the bias terms for the input gate and the $tanh$ represents the activation function.

$$Ct = ft * Ct - 1 + it * C \quad (4)$$

where, it is the output of the input gate, C is the cell state, $Ct{-}1$ is the previous cell state and Ct represents the current cell state.

3.1.3 Output gate

It controls the flow of data from the modified cell state to the model's output. It is defined by the following equations.

$$Ot = \sigma(Wo[ht-1, Xt] + bo) \quad (5)$$

$$Ht = Ot * tanh(Ct) \quad (6)$$

where, Ot is the output vector, Wo is the weight of the output gate, bo is the bias, and σ is the sigmoid activation function. The proposed structure is outlined in Figure 4.

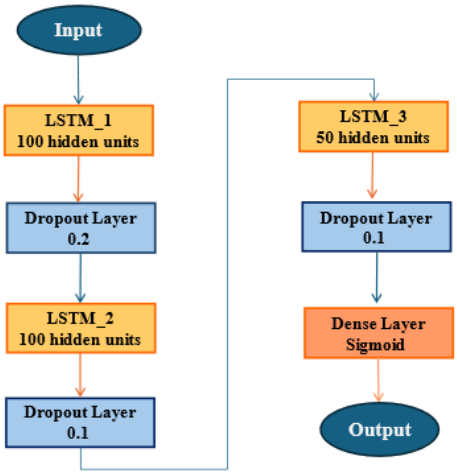

Figure 4. The LSTM's proposed structure

3.2 Bi-Directional LSTM

The LSTM model is the basis for the BiLSTM sequence processing model. One of its two LSTMs processes information in a forward direction, while the other does the opposite. The input layer, forward transmission layer, reverse transmission layer, and output layer are the four layers that comprise the BiLSTM. One of the most significant advantages of BiLSTM, which has been designed for speech and handwriting recognition, is that its input sequence is based on previous and upcoming sequences. BiLSTM effectively improves the context provided for the algorithm by increasing the quantity of network information available. It was discovered that, despite its high cost, training the data with BiLSTM models yields more accurate predictions than LSTM models in the normal mode [10]. The performance of the network is harmed by the absence of crucial information since the gates of the LSTM cells mentioned above are not directly connected to the cell state. Gers and Schmidhuber [29] enhanced the LSTM cell by adding a peephole connection to address this issue. The following are the equations for the mathematical expressions:

$$ft = \sigma(Wfhht-1 + Wfxxt + Pf \cdot ct-1 + bf) \quad (7)$$

$$it = \sigma(Wihht-1 + Wixxt + Pi \cdot ct-1 + bi) \quad (8)$$

$$ct = ft \cdot ct-1 + it \cdot tanh(Wchht-1 + Wcxxt + bc) \quad (9)$$

$$ot = \sigma(Wohht-1 + Woxxt + Po \cdot ct + bo) \quad (10)$$

$$ht = ot \cdot tanh(ct) \quad (11)$$

When an LSTM has a peephole connection, it can examine its internal states and acquire precise and reliable timing algorithms without the need for external guidance from its trainer [30]. The BiLSTM represents an upgraded form of the LSTM that operates in two directions to enhance feature extraction abilities as demonstrated in Figure 5.

$$\overrightarrow{ht} = LSTM(xt, \overrightarrow{ht}-1) \quad (12)$$

$$\overleftarrow{ht} = LSTM(xt, \overleftarrow{ht}-1) \quad (13)$$

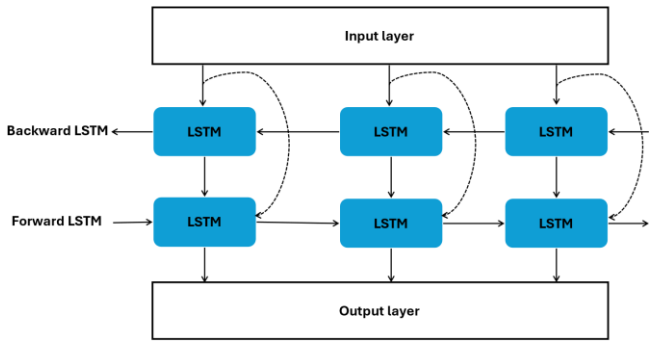

Figure 5. The proposed Bi-LSTM configuration

3.3 GRU

GRU has a forget gate and is an LSTM variant. To cut down on the number of parameters and training time, the GRU cell integrates the LSTM's forget and input gates as an update gate. Consequently, by reducing the additional parameters. It reduces the computational load. The GRU cell only has two gates: a reset gate and an update gate. The GRU cell's mathematical formulas are as follows: Eq. (14) and Eq. (15) compute the reset gate (*rt*) and update gate (*zt*) [31]:

$$rt = \sigma(Wrhht-1 + Wrxxt + br) \quad (14)$$

In Eq. (14) the reset gate establishes the appropriate weight to be assigned to previously collected data.

$$zt = \sigma(Wzhht-1 + Wzxxt + bz) \quad (15)$$

The update gate in Eq. (15) regulates how much the unit updates its activation or data from the previous step (*ht*-1) and is sent to the next cell. Where *xt* is the input vector, *Wr*, *Wz*, *Wh*, *Ur*, and *U* refer to weight matrices, *br* and *bz* stand for the bias.

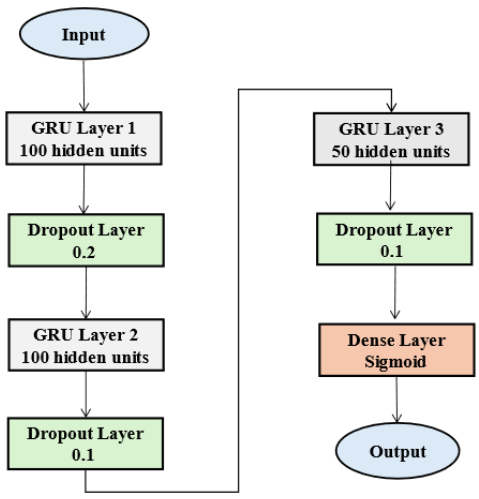

Figure 6. The proposed GRU structure

$$ht = \tanh(Whh(rt \cdot ht - 1) + Whxxt + bz) \quad (16)$$

$$htt = (1 - zt) \cdot ht - 1 + zt \cdot ht \quad (17)$$

In Eq. (16) and Eq. (17), ht is the output vector and htt is the candidate output. The other symbols are the same as before. The proposed structure is shown in Figure 6.

3.4 GRU based-attention mechanism

We propose a three-layer GRU architecture with a self-attention mechanism as shown in Figure 7 that is a type of attention mechanism applied to the preprocessing dataset to focus on the most impact feature. Human behavior serves as an inspiration for the creation of the attention mechanism. To some degree, human attention is demonstrated when people concentrate primarily on specific local areas of a picture or words in a sentence. The attention mechanism helps to make the most of a few resources. The interconnected elements of the content are then extracted over long distances using the self-attention technique. By applying the self-attention technique, the suggested model may focus more on the crucial elements of network intrusion in the ToN-IoT dataset. This process may have an impact on the internal components of the source or the target. Therefore, during the training phase, the self-attention mechanism computation can improve the effectiveness of the learning characteristics [32].

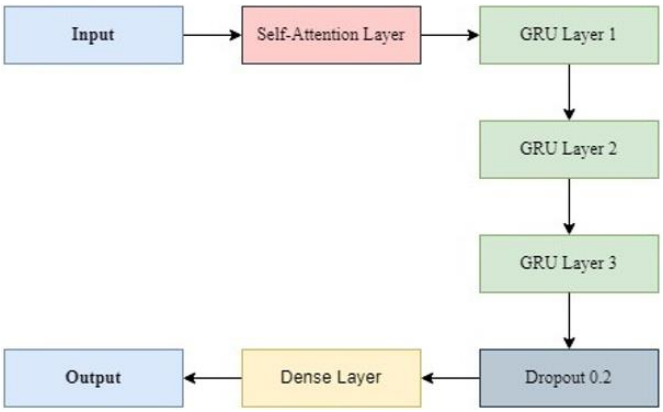

Figure 7. The proposed block diagram of the self-attention-GRU hybrid model

Furthermore, any two pieces of the relevant text can be instantly connected by self-attention through a calculating phase in the calculation process. The significantly shorter distance between them makes it easier to use the long-distance dependent characteristics effectively. The formula for calculating the output result of self-attention is as follows:

$$Pt = tanh(W \cdot ht) \quad (18)$$

where, Eq. (5), *ht* acquires the output *Pt*. Comparing *Pt* in Eq. (18) yields the allocation coefficient with a trainable parameter matrix *Q*.

$$\alpha t, i = \frac{exp\big(score(Pt, Q)\big)}{\sum_{t=1}^{T} exp\ (score(Pt, Q))} \quad (19)$$

Ultimately, the concentrated vector *S* is achieved, as demonstrated by Eq. (20).

$$S = \sum_{t=1}^{T} at.ht \quad (20)$$

3.5 GS optimization algorithm

GS is the traditional method for optimizing hyperparameters. It consists of doing an exhaustive search over a specific area of the training algorithm's hyperparameter space. Before we can conduct a grid search, we must define a boundary because the parameter space of the machine-learning approach includes spaces with actual or infinite values for some parameters [33]. GS algorithm has been used to optimize the number of hidden units. The algorithm steps are outlined in Figure 8.

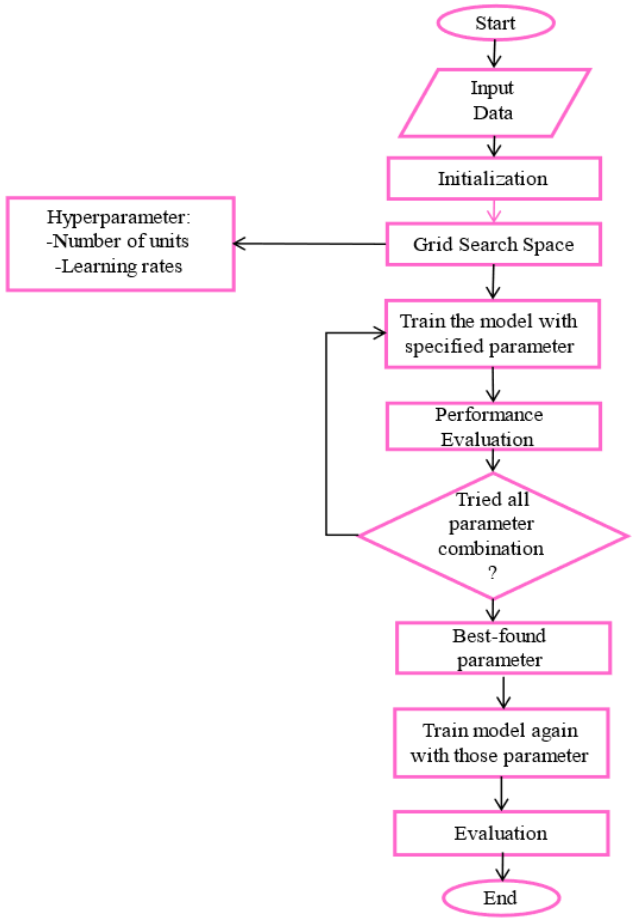

Figure 8. Flowchart of GS for proposed mode

4. DATASET DESCRIPTION AND COLLECTION

Telemetry data from IoT services, network traffic data from IoT networks, and operating system logs are all included in a recently released diverse dataset in 2020 [34]. In particular, the elements of network traffic flows are the subject of this study. At ACCS's Cyber Range Lab, a variety of attack simulations were run in a realistic medium-scale network environment to create the dataset. The collection consists of 461,043 records from various sorts of network assaults spread across 45 distinct columns. Of the forty-five features in the ToN-IoT dataset, twenty-seven are strings, and eighteen are numbers, of which seventeen are integers and one is decimal as illustrated in Table 2.

Table 2. Dataset statistics for ToN-IoT

Statistic	Count
Number of Instances	461,043
Number of Columns	45
Number of Categorical Features	27
Number of Numerical Features	18

5. EVALUATION METRICS FOR MODEL PERFORMANCE

The effectiveness of different models is assessed using a range of indicators, including the classification time. The following definitions apply to the evaluation criteria used.

5.1 Confusion matrix

The performance of a classification model when applied to a set of data whose actual values are known is displayed in a table known as the confusion matrix. Commonly used terms include False Positives (FP), False Negatives (FN), True Positives (TP), and True Negatives (TN). In Table 3, the matrix can be found.

Table 3. Confusion matrix

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

5.2 Accuracy

The proportion of accurately anticipated cases to all instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{21}$$

5.3 Precision

The percentage of accurately predicted positive observations compared to all anticipated positives.

$$Precision = \frac{TP}{TP + FP} \tag{22}$$

5.4 Recall

The percentage of accurately predicted positive instances to all observations conducted during the actual class.

$$Recall = \frac{TP}{TP + FN} \tag{23}$$

5.5 F1-score

The precision and recall weighted average are called the F1-score.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision \times Recall} \tag{24}$$

6. RESULT AND DISCUSSION

In order to distinguish between normal and abnormal attacks, this study examined the effectiveness of LSTM, Bi-LSTM, GRU, and a novel hybrid model called the GRU-based self-attention mechanism. The ToN-IoT dataset was used to train and assess each model. It is important to note that Intel(R), Core (TM) i7-8550U CPU @ 1.80GHz (8 CPUs) with 8192MB RAM is used to run the simulation. Various scenarios have been tested (LSTM, Bi-LSTM, GRU, and GRU-based Attention mechanism) to ensure excellent accuracy and efficiency and increase classification speed. To select the best model hyperparameters for the job, A varied number of layers and different values for the cells used for learning in each layer were used in the development of the model.

To select the model with the maximum efficiency and the fewest false alarms, all scenarios with varying numbers of layers were compared. It has been shown that increasing the number of layers results in longer learning times and worse accuracy, which is not helpful for this purpose. The proposed models involve three layers the two first layers with 100 cells, the third with 50 cells, and one output layer with one cell that uses a sigmoid function, the best model hyperparameters are shown in Table 4. It eliminates overfitting between training and testing and provides the maximum accuracy for training. Additionally, dropout layers were added to prevent overfitting, every layer applies with tan activation. Except for the last layer, which uses sigmoid and binary_crossentropy loss.

Table 4. Model hyper-parameters

No. of Layers	No. of Hidden Units	No. of Epochs	Batch Size	Dropout Rate	Learning Rate
3	100 100 50	5	80	0.2 0.1 0.1	0.0015

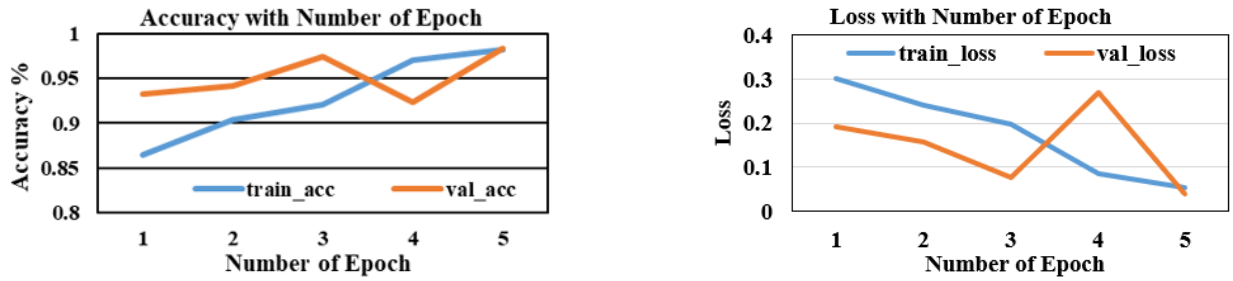

Figure 9. Measurement of accuracy and loss in training compared to validation for 5 epochs utilizing the LSTM classifier

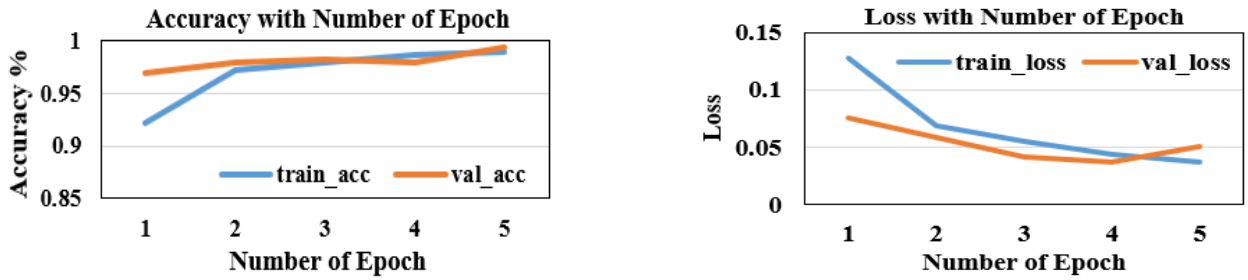

Figure 10. Bi-LSTM classifier accuracy and loss measure in training versus validation over five epochs

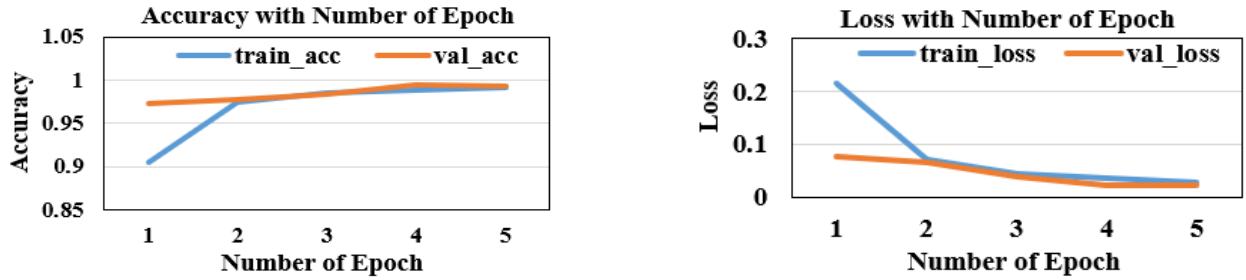

Figure 11. GRU classifier accuracy and loss measure in training versus validation over five epochs

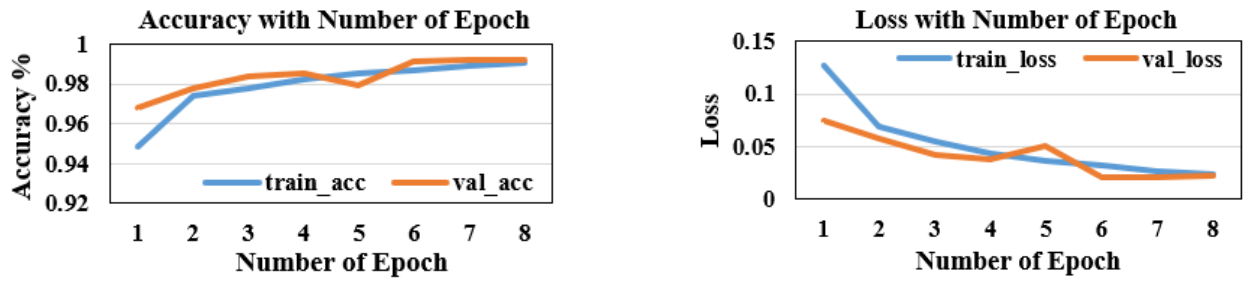

Figure 12. Accuracy and loss measure in training vs. validation for 8 epochs using attention-GRU hybrid classifier

	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	59920
1	0.98	0.98	0.98	32289
Accuracy			0.99	92209
Macro avg.	0.98	0.98	0.98	92209
Weighted avg.	0.99	0.99	0.99	92209

LSTM

	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	59920
1	0.98	0.99	0.98	32289
Accuracy			0.99	92209
Macro avg.	0.99	0.99	0.99	92209
Weighted avg.	0.99	0.99	0.99	92209

Bi-LSTM

	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	59920
1	0.99	0.98	0.99	32289
Accuracy			0.99	92209
Macro avg.	0.99	0.99	0.99	92209
Weighted avg.	0.99	0.99	0.99	92209

GRU

	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	59920
1	0.99	0.99	0.99	32289
Accuracy			0.99	92209
Macro avg.	0.99	0.99	0.99	92209
Weighted avg.	0.99	0.99	0.99	92209

GRU-based Self-Attention Mechanism

Figure 13. The classification report for the proposed models

Figures 9-12 show the computed accuracy and loss curves for our models in binary classification in both training and testing. It shows how the more epochs there are, the higher the test accuracy. As mentioned earlier, 20% of the ToN-IoT dataset is used to evaluate the trained model.

Moreover, the thorough classification report offers a more profound understanding of the model's class discrimination capabilities. This highlights the model's balanced performance since the precision, recall, and F1-score are all close to 99% for both classes as shown in Figure 13. In particular, class 0 precision and recall are 99%, for LSTM which also yields a 99% F1-score. Class 1 exhibits a comparable pattern, with

precision and recall values approaching 98%, with an F1-score of 98% and an accuracy of 99%. For (GRU, Bi-LSTM, and GRU-based attention) the large test dataset of 92,209 entries yielded an overall accuracy of 99%, proving the ability of the model to extrapolate outside of the training set.

In addition, Figure 14 summarizes the confusion matrix which is particularly useful for investigating the instances of false positives and false negatives. It is an essential tool for comprehending the particular kinds of mistakes that our suggested models make. It can be noticed that GRU obtained the lowest false negative rate. The matrix shows that our model properly recognized 59,627 out of the 59,920 samples of class 0 while incorrectly identifying 293 instances as class 1. These 293 instances are false positives, as seen in Figure 14. On the contrary, out of the 32,289 class 1 instances that the model accurately identified, 31,819 were mistakenly categorized 470 as class 0. These 470 instances serve as instances of false negatives.

Furthermore, a model's sensitivity, sometimes referred to as its True Positive Rate, indicates how well it can identify positive cases. The sensitivity of the GRU model is approximately 99.69% for class 0 and 99.26% for class 1. This suggests that the model performs exceptionally well in accurately detecting cases for both classes, with class 0 showing a marginally higher sensitivity. On the other hand, Specificity, often known as the True Negative Rate, shows how well the model classifies negative examples. Class 0 has a specificity of about 99.26%, whereas class 1 has a specificity of 99.69%, according to the specificity values, which nearly match the sensitivity scores.

For the results shown in Table 5, the testing phase for the LSTM took 0.743 ms, because the cell state and output layer are separated leading to a complex architecture. In addition, a large number of trainable parameters increased memory usage. On the other hand, the architecture using Bidirectional LSTM took the longest classification time 1.202 ms to test, which doubles the number of computations done in comparison to LSTM, due to its bidirectional processing of the input sequence this increases memory usage, combining two layers of LSTM increase complexity of the model all these can lead to slower processing time. Furthermore, GRU took 0.55 ms because of its quicker training speed, less memory utilization, less complicated processes, smaller parameters, and simpler structure. Moreover, the GRU-based attention mechanism took 0.075 ms by focusing on important features, the model can make faster training without processing unrelated features thus reducing computations and leading to high classification time compared to other IDS approaches.

Table 5. A comparison of various algorithms in the classification of binary targets (normal/attack) based on their accuracy and classification time

Algorithm	Testing Accuracy	Classification Time [ms]
GRU	97.2%	0.55
LSTM	98.1%	0.743
Bi-LSTM	98.4%	1.202
GRU-based Self Attention Mechanism	99%	0.3

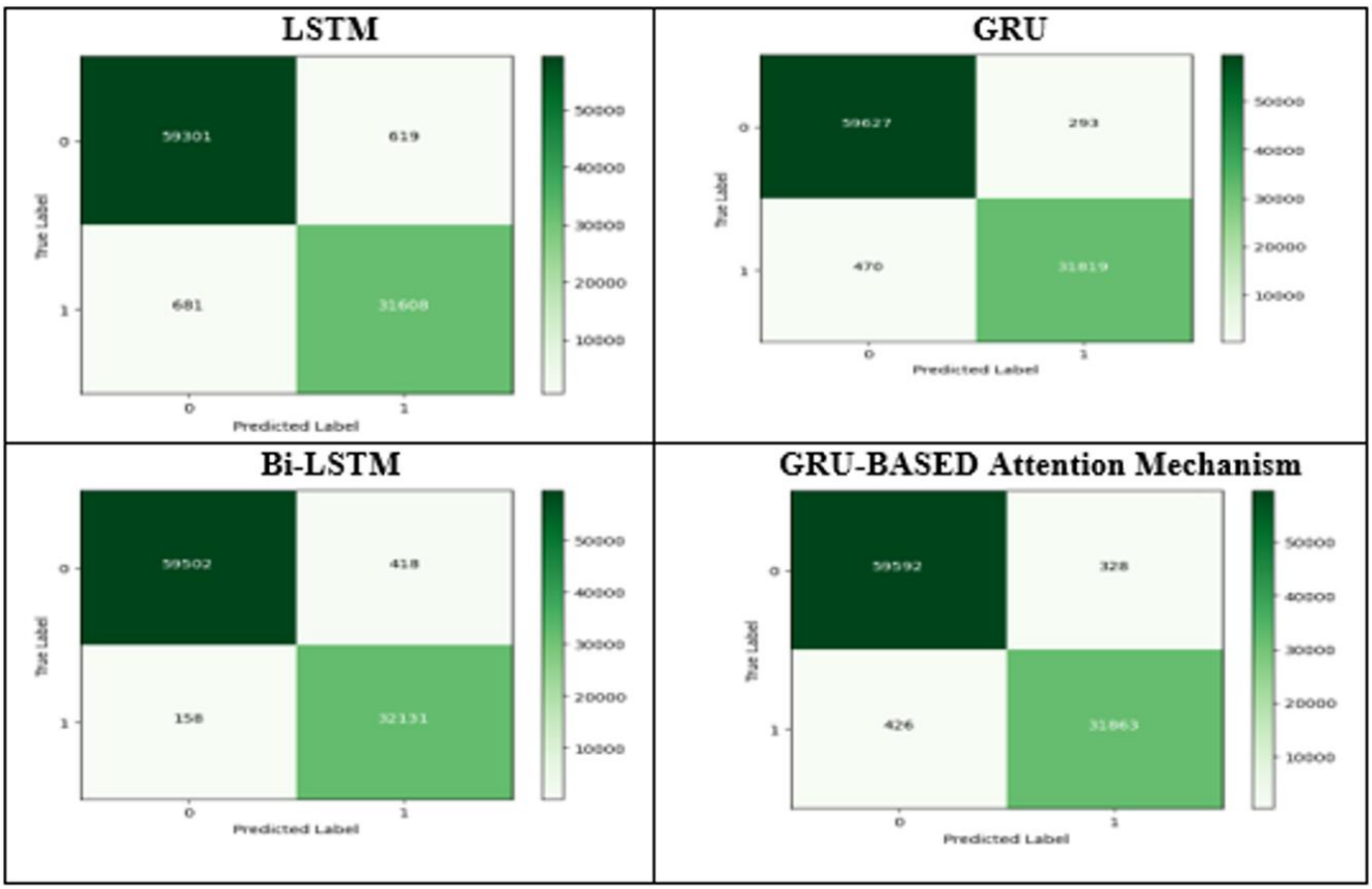

Figure 14. Confusion matrices for the proposed models

6.1 Simulation results with grid search optimization algorithm

Many strategies minimize complexity like feature dimensionality reduction (using PCA algorithm), lightweight models (VGG, Inception v2, ResNet), feature selection algorithm, GPU with TensorFlow reduce processing time for training phase, and optimization algorithm here A key parameter, the number of hidden units and learning rate, have been optimized using the Grid Search algorithm. Table 6 below shows the optimized parameters. The fine-tuning of hyper-parameters for the proposed models through the utilization of the GS algorithm revealed significant fluctuations in performance throughout various iterations. The fine-tuning process was evaluated by examining precision, recall, and F1-score for two classes, which indicated the model's proficiency in accurate classification within a dataset consisting of 92,209 data points.

Table 6. Model hyperparameters

No. of Layers	No.s of Hidden Units	Learning Rate
3	64 64 64	0.001

Table 7. The comparison of various algorithms with GS in the classification of binary targets (normal/attack) based on their accuracy and classification time

Algorithm	Testing Accuracy	Classification Time [ms]
GRU	97.2%	0.489
LSTM	98.1%	0.521
Bi-LSTM	98.4%	0.846
GRU-based Self Attention Mechanism	99%	0.075

The accuracy, precision, recall, and F1-score for every model were all 99%. Higher accuracy, precision, recall, and f1-score levels are obtained by successfully applying the GS algorithm to refine the LSTM, GRU, Bi-LSTM, and GRU-based attention models. It is evident that the model, which GS has improved, has competent and fair classification abilities for all classes in the dataset. However, like any optimization process, it is important to understand and closely monitor the effect of variation in results over several rounds and the possibility of either over-fitting or under-fitting in specific cases. The effect of GS is seen in Table 7.

Figure 15 illustrates the accomplishments of each architectural design with decreasing time taken for classification.

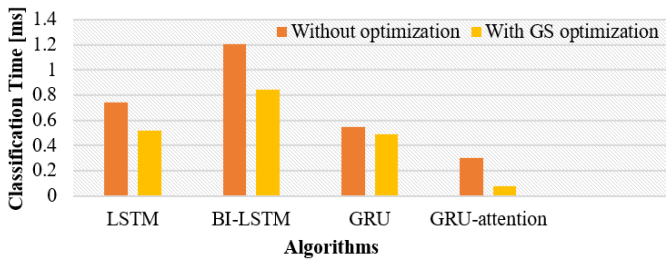

Figure 15. The classification time comparison

We compared the proposed approach with the application of the GS optimization algorithm against alternative intrusion detection techniques. The evaluation of the proposed method shows a remarkable accuracy rate of 99% as illustrated in Table 8. Then, the approach combines the GS algorithm with the strength of GRU deep-learning architectures, which are renowned for their skill in sequential-data modeling and attention mechanism that focus on the most important feature in the dataset. To prevent over-fitting and help the model learn the best possible representation of the data, the Grid Search method optimizes and minimizes the hidden units of GRU layers. Such high accuracy confirms our method's reliability and highlights the need to integrate deep learning with conventional optimization techniques for intrusion detection.

Now, to provide a brief quantitative comparison between the proposed deep learning models, we compared the models based on different key factors including the accuracy, the training time, and the corresponding complexity of each model as shown in Table 9.

In fact, based on the number of parameters of each network, the GRU has less complexity, followed by the proposed GRU-based self-attention mechanism. However, including the self-attention layers improves the classification accuracy.

Table 8. Comparison with the most related research studies

Ref.	Dataset	Method	Accuracy
[20]	NSL-KDD	Attention mechanism with Bi-LSTM network	90.73%
[21]	KDD99	LSTM-No reduction (all features)	96.51%
[22]	BoT-IoT, ToN-IoT	CNN	99.8%
The proposed approach	ToN-IoT	GRU-based Self Attention Mechanism with GS Algorithm	99%

Table 9. Quantitative comparisons between the applied DL models

Key Factor	LSTM	Bi- LSTM	GRU	Hybrid
Accuracy	Good, but not better than the hybrid model	Good, but not better than the hybrid model	Good, but not better than the LSTM, BI-LSTM, and hybrid model	The most superior of all models, making it the best for applications requiring precision
Training time	Fast compared to Bi-LSTM	The model that is the longest takes longer to train than the others	Faster than LSTM and Bi-LSTM but slower than hybrid model	It can be considered the fastest model out of all models due to the lower training time
Complexity	Exhibits a moderate complexity level with 151,451 parameters	Highly complex with 422,901 parameters	Least complex with 114,351 parameters	Simpler with 126,995 parameters

7. CONCLUSION

The Internet of Things is currently one of the fastest-growing technologies. This information will then be shared across authorized entities to accomplish a number of goals, like overseeing and controlling manufacturing facilities or improving corporate operations and procedures. Since there are more gadgets connected to the Internet, there are more targeted entities being attacked. Protecting these devices from hackers, illegal access, and alteration is essential. Based on lightweight models that can identify intrusions and achieve high classification speed and accuracy, this study suggested an IDS for critical-time IoT systems. Only one target binary class—normal or attack—is used in the ToN-IoT dataset. The suggested models show improvement when the GS optimization process is used. Examined and contrasted with previous systems, the suggested system performs better.

The achieved detection accuracy for all models is as follows: 97% for GRU, 98.1% for LSTM, 98.4% for Bi-LSTM, and 99% for the GRU-based self-attention mechanism. GRU-based self-attention mechanism has fewer parameters, which leads to a significant saving in classification time by up to 84% compared to GRU, making it well-suited for real-time detection.

These findings demonstrate that the GRU-based self-attention mechanism is superior in terms of accuracy and computational efficiency, making it ideal for real-time intrusion detection in IoT networks. In future directions, it can focus on using multi-class targets to detect specific attack types (e.g., DoS and DDoS) and using advanced DL techniques like TensorFlow that facilitate system adaptation and detection of zero-day attacks.

REFERENCES

[1] Alghuried, A. (2017). A model for anomalies detection in internet of things (IoT) using inverse weight clustering and decision tree. Masters dissertation, Technological University Dublin.

[2] Mhawi, D.N., Hashem, S.H. (2022). Proposed hybrid ensemble learning algorithms for an efficient intrusion detection system. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 22(2): 73-84. https://doi.org/10.33103/uot.ijccce.22.2.7

[3] Ahmad, I.A., Al-Nayar, M.M.J., Mahmood, A.M. (2023). A comparative study of Gaussian mixture algorithm and K-means algorithm for efficient energy clustering in MWSN. Bulletin of Electrical Engineering and Informatics, 12(6): 3727-3735. https://doi.org/10.11591/eei.v12i6.5707

[4] Al Hasan, R.A., Hamza, E.K. (2023). An improved intrusion detection system using machine learning with singular value decomposition and principal component analysis. International Journal of Intelligent Engineering & Systems, 16(4): 25-38. https://doi.org/10.22266/ijies2023.0831.03

[5] Jassim, S.A., Farhan, A.K. (2022). Designing a new lightweight AES algorithm to improve the security of the IoT environment. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 22(2): 96-108. https://doi.org/10.33103/uot.ijccce.22.2.9

[6] Hussein, A.Y., Sadiq, A.T. (2022). Meerkat clan-based feature selection in random forest algorithm for IoT intrusion detection. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 22(3): 15-24. https://doi.org/10.33103/uot.ijccce.22.3.2

[7] Hussein, A.Y., Falcarin, P., Sadiq, A.T. (2021). Enhancement performance of random forest algorithm via one hot encoding for IoT IDS. Periodicals of Engineering and Natural Sciences, 9(3): 579-591. https://doi.org/10.21533/pen.v9i3.2204

[8] Jin, D., Lu, Y., Qin, J., Cheng, Z., Mao, Z. (2020). SwiftIDS: Real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism. Computers & Security, 97: 101984. https://doi.org/10.1016/j.cose.2020.101984

[9] Lin, Z., Sun, X., Ji, Y. (2022). Landslide displacement prediction model using time series analysis method and modified LSTM model. Electronics, 11(10): 1519. https://doi.org/10.3390/electronics11101519

[10] Selim, I.M., Sadek, R.A. (2023). DAE-BILSTM: A fog-based intrusion detection model using deep learning for IoT. Journal of Theoretical and Applied Information Technology, 101(5): 2027-2042.

[11] Imrana, Y., Xiang, Y., Ali, L., Abdul-Rauf, Z. (2021). A bidirectional LSTM deep learning approach for intrusion detection. Expert Systems with Applications, 185: 115524. https://doi.org/10.1016/j.eswa.2021.115524

[12] Bhardwaj, A., Mangat, V., Vig, R. (2020). Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of DDoS attacks in cloud. IEEE Access, 8: 181916-181929. https://doi.org/10.1109/ACCESS.2020.3028690

[13] Diaba, S.Y., Elmusrati, M. (2023). Proposed algorithm for smart grid DDoS detection based on deep learning. Neural Networks, 159: 175-184. https://doi.org/10.1016/j.neunet.2022.12.011

[14] Fenanir, S., Semchedine, F. (2023). Smart intrusion detection in IoT edge computing using federated learning. Revue d'Intelligence Artificielle, 37(5): 1133-1145. https://doi.org/10.18280/ria.370505

[15] Saurabh, K., Sood, S., Kumar, P.A., Singh, U., Vyas, R., Vyas, O.P., Khondoker, R. (2022). LBDMIDS: LSTM based deep learning model for intrusion detection systems for IoT networks. In 2022 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, pp. 753-759. https://doi.org/10.1109/AIIoT54504.2022.9817245

[16] Chen, Y., Al-Rubaye, S., Tsourdos, A., Baker, L., Gillingham, C. (2023). Differentially-private federated intrusion detection via knowledge distillation in third-party iot systems of smart airports. In ICC 2023 - IEEE International Conference on Communications, Rome, Italy, pp. 603-608. https://doi.org/10.1109/ICC45041.2023.10279722

[17] Qathrady, M.A., Ullah, S., Alshehri, M.S., Ahmad, J., Almakdi, S., Alqhtani, S.M., Khan, M.A., Ghaleb, B. (2024). SACNN-IDS: A self-attention convolutional neural network for intrusion detection in industrial internet of things. CAAI Transactions on Intelligence Technology. https://doi.org/10.1049/cit2.12352

[18] Latif, S., Boulila, W., Koubaa, A., Zou, Z., Ahmad, J. (2024). Dtl-ids: An optimized intrusion detection framework using deep transfer learning and genetic algorithm. Journal of Network and Computer Applications, 221: 103784. https://doi.org/10.1016/j.jnca.2023.103784

[19] Sundaram, K., Natarajan, Y., Perumalsamy, A., Yusuf

Ali, A.A. (2024). A novel hybrid feature selection with cascaded LSTM: Enhancing security in IoT networks. Wireless Communications and Mobile Computing, 2024(1): 5522431. https://doi.org/10.1155/2024/5522431

[20] Fu, Y., Du, Y., Cao, Z., Li, Q., Xiang, W. (2022). A deep learning model for network intrusion detection with imbalanced data. Electronics, 11(6): 898. https://doi.org/10.3390/electronics11060898

[21] Laghrissi, F., Douzi, S., Douzi, K., Hssina, B. (2021). Intrusion detection systems using long short-term memory (LSTM). Journal of Big Data, 8(1): 65. https://doi.org/10.1186/s40537-021-00448-4

[22] Gaber, T., Awotunde, J.B., Torky, M., Ajagbe, S.A., Hammoudeh, M., Li, W. (2023). Metaverse-IDS: Deep learning-based intrusion detection system for Metaverse-IoT networks. Internet of Things, 24: 100977. https://doi.org/10.1016/j.iot.2023.100977

[23] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7: 41525-41550. https://doi.org/10.1109/ACCESS.2019.2895334

[24] Sadaf, K., Sultana, J. (2020). Intrusion detection based on autoencoder and isolation forest in fog computing. IEEE Access, 8: 167059-167068. https://doi.org/10.1109/ACCESS.2020.3022855

[25] Yu, L., Xu, L., Jiang, X. (2023). An effective method for detecting unknown types of attacks based on LOG-COSH variational autoencoder. Applied Sciences, 13(22): 12492. https://doi.org/10.3390/app132212492

[26] Jabber, S.A., Jafer, S.H. (2023). A novel approach to intrusion-detectionsystem: Combining LSTM and the snakealgorithm. Jordanian Journal of Computers and Information Technology, 9(4): 360-376.

[27] Albaqami, H., Hassan, G.M., Datta, A. (2023). MP-SeizNet: A multi-path CNN Bi-LSTM Network for seizure-type classification using EEG. Biomedical Signal Processing and Control, 84: 104780. https://doi.org/10.1016/j.bspc.2023.104780

[28] Nasser, A.R., Mahmood, A.M. (2021). Cloud-based Parkinson's disease diagnosis using machine learning. Mathematical Modelling of Engineering Problems, 8(6): 915-922. https://doi.org/10.18280/mmep.080610

[29] Yu, Y., Si, X., Hu, C., Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. Neural Computation, 31(7): 1235-1270. https://doi.org/10.1162/neco_a_01199

[30] Salman, H., Singhal, P., Shankar, T., Yin, P., et al. (2018). Learning to sequence robot behaviors for visual navigation. arXiv preprint arXiv:1803.01446. https://doi.org/10.48550/arXiv.1803.01446

[31] Al-Khazraji, H., Nasser, A.R., Khlil, S. (2022). An intelligent demand forecasting model using a hybrid of metaheuristic optimization and deep learning algorithm for predicting concrete block production. IAES International Journal of Artificial Intelligence, 11(2): 649-657. https://doi.org/10.11591/ijai.v11.i2.pp649-657

[32] Lu, C., Dai, H.N., Zhou, J., Wang, H. (2021). Exploring self-attention mechanism of deep learning in cloud intrusion detection. In Cloud Computing: 10th EAI International Conference, CloudComp 2020, Qufu, China, pp. 57-73. https://doi.org/10.1007/978-3-030-69992-5_5

[33] Liashchynskyi, P., Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: A big comparison for NAS. arXiv preprint arXiv:1912.06059. https://doi.org/10.48550/arXiv.1912.06059

[34] Sarhan, M., Layeghy, S., Moustafa, N., Portmann, M. (2021). Netflow datasets for machine learning-based network intrusion detection systems. In Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, pp. 117-135. https://doi.org/10.1007/978-3-030-72802-1_9