JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 Vol.-20, No.-12, December (2025) pp 48-65 ISSN (Print) 0973-8975

ENHANCING IOT SECURITY USING AN INTEGRATED BAGGED-LSTM AND GRADIENT BOOSTING ENSEMBLE TECHNIQUE

Preeti[1], Rajender Nath[2]

[1,2] Department of Computer Science and Applications, Kurukshetra University, Kurukshetra, Haryana, India

Email: [1]preeti.jrf.dcsa@kuk.ac.in, [2]rnath@kuk.ac.in

Corresponding Author: **Preeti**

https://doi.org/10.26782/jmcms.2025.12.00004

(Received: September 18, 2025; Revised: November 17, 2025; December 02, 2025)

Abstract

The Internet of Things (IoT) links billions of devices, boosts innovation, shares information effortlessly, and is reshaping various industries. The most common Distributed Denial of Service (DDoS) attacks target all layers in the IoT architecture. Even though easy to execute, these sorts of attacks may severely harm targeted systems and networks. This Novel hybrid model uses Bagged Long Short-Term Memory (LSTM) and Gradient Boosting (GB) to address large dimensionality, various feature dimensions, low classification accuracy, and high false positive rates in raw traffic data to improve IoT security against DDoS attacks. To reduce input information redundancy, the Boruta-Pearson Feature Selector (BPFS) gathers key features as model inputs. The Bagged-LSTM design minimises variance to detect anomalies, while Gradient Boosting improves prediction accuracy. The CIC-ISD2017 and CIC DDoS2019 datasets were used to test the hybrid model. Experimental results show that the recommended model outperforms current models with an accuracy of 99%. It is impossible to completely protect your server from these threats, but by using the techniques discussed here, these attacks can be prevented, and the server can focus on fulfilling legitimate requests rather than unauthentic ones.

Keywords: DDoS attacks, Gradient Boosting (GB), IoT security, long short-term memory (LSTM),

I. Introduction

The Internet of Things (IoT) has changed life by enabling devices to communicate and exchange data. Applications of IoT include healthcare, smart cities, industrial automation, and transportation, etc. Denial of Service (DDoS) attacks are one of the biggest cybersecurity concerns for IoT, despite its benefits [I]. These attacks use IoT networks' scattered nature to overload devices and interrupt services. This attack happens when the server is overburdened with authorised and unauthorised network traffic. DDoS attacks use several infected devices, frequently from different places, to overload a target system [IV]. As network technology evolves, DDoS attacks

are becoming more frequent, sophisticated, and impactful, making it challenging to detect how they initiate and distinguish between regular and DDoS attack traffic. Thus, academics have researched DDoS attack detection to discriminate between regular and attack traffic. Statistical, machine, and deep learning detection methods exist. All three attacks weaken resources by attacking IoT architectural layers. Some avoidance methods are generic, while others are technological [XXII]. Continuous monitoring, restricting network broadcasting, server redundancy, and network security help reduce DDoS attacks. These general methods are ineffective, hence technology-based prediction procedures like machine learning and IOT were developed. Machine learning model constraints (machine learning cannot efficiently operate with vast datasets) make these approaches less powerful for reliable findings [II]. [XV] suggested a random forest-based DDoS assault detection algorithm using statistical learning. Ye et al. [XVIII] proposed using SVM to classify feature vectors in SDN networks. This method marks DDoS attacks by calculating the information entropy of the source and destination IP addresses and port addresses in the data and using the random forest model to increase the stability of the fitting degree. These feature vectors identify DDoS attacks using source IP, source port, and other network data. Koay et al. [XI] employ information entropy to build features to identify slow DDoS assaults and various classifiers to assess the findings. With the rise of deep learning and big data, more scientists are studying DDoS attack detection using deep learning. Deep learning analyses and learns internal rules from enormous data, builds a network model using multi-layer neurons or perceptual mechanisms, and trains it. Deep learning can handle high-dimensional data and reduce data noise to compensate for machine learning's absence [XX]. A layered spatiotemporal intrusion detection system was proposed by Wang et al. [XXI]. After learning the low-level spatial properties of network traffic with the deep convolutional neural network (CNN), the short-term and long-term memory networks learnt the high-level temporal features. Cheng et al. [VII] suggested a DDoS attack detection approach based on the grey scale matrix feature of network flow of a convolutional neural network. Experimental findings demonstrate that the proposed model outperforms previous in-depth learning methods. Based on IP protocol attack flow and normal flow, a 7-tuple is defined to describe network flow, the binary is converted to grey scale features, and a multi-scale convolutional neural network model is used for feature extraction training. Integrated Bagged-LSTM with Gradient Boosting improves IoT security [V]. Its specific contributions are:

• To address the feature redundancy problem caused by the high dataset dimension, this paper proposes a new feature selection **Boruta-Pearson Feature Selector (BPFS)** algorithm that uses a Boruta algorithm to calculate feature importance and Pearson correlation coefficient analysis to select features.

The Bagged-LSTMs are used to identify DDoS attacks. At the same time, the input data's spatial and temporal features are extracted and learnt thoroughly. Classification is done via a Gradient Boosting classifier. LSTM networks are chosen because of their greater capacity to handle sequential data with temporal dependencies, their robustness against RNN restrictions, and their performance in comparable cybersecurity scenarios. This makes them ideal for DDoS attack detection testing and model dependability and accuracy in real-world circumstances.

This study employs Bagged-LSTM and Gradient Boosting to improve IoT security. Bagging aggregates predictions from several models to strengthen LSTM models, whereas Gradient Boosting refines categorisation using residual patterns. The remaining paper is organised as follows. Section II summarises relevant work. Section III suggests a better hybrid approach. Section IV presents the dataset, evaluation metrics, and experiment outcomes. Finally, Section V closes the article.

II. Related Work

A. DDoS Attack Detection Using LSTM:

LSTM networks are esteemed for their capacity to capture temporal relationships in sequential data, rendering them optimal for identifying harmful patterns in network traffic. Recurrent Neural Networks (RNNs) excel in processing sequential data; however, they are susceptible to gradient vanishing, gradient explosion, and challenges related to long-term dependencies during training [XVIII]. The LSTM layers effectively address the long-term dependency issues inherent in RNNs. The LSTM layers & Units (Cells) incorporate three gates (forgetting gate, input gate, output gate) and one cell state update into the hidden layer of the RNN model, as illustrated in Figure 1.

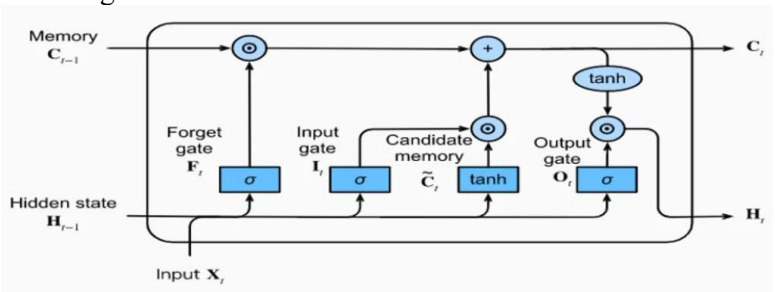


Fig 1. Structure of the LSTM cycle. *(Credits:* https://d2l.ai/chapter_recurrent-modern/lstm.html)

The forgetting gate evaluates the state of the preceding layer of cells, retaining pertinent information while discarding irrelevant data, computed as follows:

$$f_t = \sigma(w_f \cdot [h_t - 1, x_t] + b_f) \quad (1)$$

The bias term of the forgetting gate, and ht−1, represents the output value of the preceding LSTM layer. The σ denotes the sigmoid activation function, while [,] indicates the concatenation of two vectors into a single vector. The input gate assesses the significance of the information, transmitting the pertinent data for the cell state update, thereby completing the update process. This procedure comprises two components: initially, the sigmoid function is employed to ascertain the new information to be incorporated into the cell state; subsequently, the tanh function is utilised to produce a new candidate vector [III]. The calculation process is delineated in Formula (2).

$$\begin{aligned} f_t &= \sigma(w_f \cdot [h_t - 1, x_t] + b_f) \\ \tilde{c}_t &= \tanh(w_c \cdot [h_t - 1, x_t] + b_c) \end{aligned} \quad (2)$$

In this context, wi and bi represent the weights and biases of the inputs, whereas wc and bc denote the weights and biases of the cell state. After the procedure, the preceding cell state ct−1 is revised to the present cell state ct, as delineated in Formula (3).

$$c_t = f_t^* c_t - 1 + i_t^* \tilde{c}_t \tag{3}$$

The input gate regulates the output of the cell state of this layer to determine which cell states are input to the next layer, where ∗ denotes element multiplication, f ∗ t ct−1 denotes deletion information, and it ∗ct denotes new information. Formula (4) provides the calculation formula.

$$o_t = \sigma(w_o \cdot [h_t - 1, x_t] + b_o)$$
$$h_t = o_t^* \tanh(c_t) \tag{4}$$

After the original detection data set is quantified, normalised, and standardised, the LSTM-based DDoS attack detection method inputs the pre-processed data set into the trained LSTM model. The classification results are then input into the Gradient Boosting classifier. Since the one-way LSTM can only retain the previous context information, this paper employs the LSTM network as the time feature extraction model, which can learn the time feature of network traffic better [IV].

Table 1: DDoS Attack Detection Using LSTM-Based Techniques

LSTM-Based Model	Architecture	Performance	Pros	Cons
VanillaLSTM	Single LSTM Layer + Dense	93%	Captures sequential patterns, good for time-series data	High computational cost
BiLSTM	Forward & Backward LSTM Layers	95%	Learns bidirectional dependencies in data	Requires more memory
CNN-LSTM [IX]	CNN for feature extraction + LSTM for sequence learning	98.5%	Extracts spatial + temporal features	Computationally expensive
GRU-LSTM [X]	LSTM + GRU Combination	98.7%	Reduces training time, better for real-time detection	May lose long-term dependencies
Attention-Based LSTM [XIX]	LSTM + Attention Mechanism	95%	Focuses on important time-series features	More complex to implement

B. DDOS Attack Detection Method Based on Ensemble Techniques

Ensemble approaches are methodologies that integrate many algorithms to develop a more resilient and precise prediction model [XVII]. These techniques encompass bagging, boosting, and stacking. Bagging entails training numerous models on distinct subsets of training data, whereas boosting emphasises sequential model training. Stacking entails training many base models and utilising their predictions as input characteristics for a superior model [XVI]. Ensemble approaches can markedly improve DDoS attack detection by minimising misclassification rates, exhibiting reduced susceptibility to overfitting, offering insights into feature significance, and adapting to evolving assault patterns [XII]. Such as bootstrap aggregation, have been applied to reduce variance and enhance model stability. Gradient Boosting algorithms, including XGBoost, LightGBM, and CatBoost, are well-known for their efficiency in handling complex datasets and improving classification accuracy [XIII].

Table 2: DDoS Attack Detection Using Ensemble Machine Learning Techniques

Ensemble Technique [VI]	Base Models Used	Pros	Cons
Bagging	Decision Tree, Random Forest	Reduces variance, prevents overfitting	Requires high computational resources
Boosting (AdaBoost, Gradient Boosting)	Decision Tree, SVM, Logistic Regression	Improves weak learners, high accuracy	Sensitive to noisy data, prone to overfitting
Stacking	Random Forest, SVM, Neural Networks	Combines multiple classifiers with higher predictive power	Requires more training time
Voting Classifier (Hard & Soft Voting)	KNN, Decision Tree, Naïve Bayes	Simple to implement, balances multiple models	Not always better than a single strong classifier

While LSTM and Gradient Boosting have been studied individually for DDoS detection, combining these techniques into a unified framework for IoT security remains underexplored. This study aims to bridge this gap by proposing a Bagged-LSTM with Gradient Boosting hybrid model.

III. Research Methodology

In this paper, we construct a hybrid model of Bagged LSTM with Gradient Boosting using a novel feature selection Boruta-Pearson Feature Selector (BPFS) algorithm. The research methodology is depicted in Figure 2.

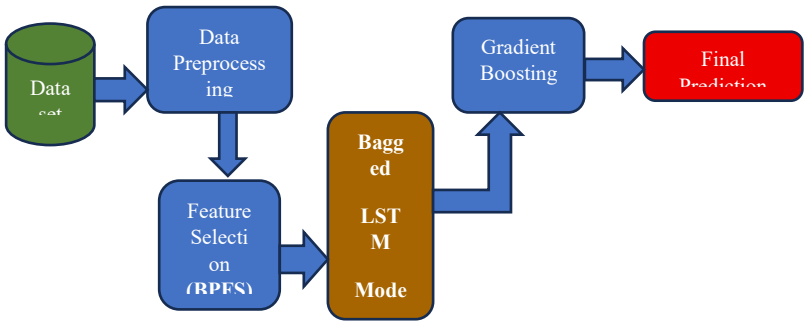


Figure 2. The Bagged-LSTM Framework

A. Data Collection:

In the domain of DDoS attack detection, there exists a limited number of datasets suitable for training deep learning systems. This paper utilised two publicly accessible datasets obtained from the Canadian Institute of Cybersecurity (CIC) and Wireshark in a simulated environment: CIC-IS2017 and CICDDoS2019 [VIII] for evaluating the performance of the Hybrid model. The two datasets were first created with two distinct usage patterns and multi-level attacks, including different DoS and DDoS attacks. The gathered traffic was pre-processed using the CICFlowMeter program to provide a CSV file encompassing diverse DoS and DDoS traffic statistics. The CIC-IDS2017 dataset, developed in 2017 by the Canadian Institute for Cybersecurity, employs 80 characteristics to monitor both benign and malicious attacks. The CIC-DDoS2019 dataset comprises 86 attributes of network traffic packets created by an open-source application that creates network packets and gathers their information. Feature selection techniques were employed to extract a total of 52-dimensional features, including flow features, basic features, connection features, time features, general characteristics, many more created features, and label features. After preprocessing, the CIC DDoS2019 dataset retains 67-dimensional characteristics. Normal traffic is denoted by 0, whereas DDoS attack traffic is denoted by 1. To verify the effectiveness of the proposed model in detecting multi-class attacks, multi-class experiments were conducted based on the CIC-DDoS2019 dataset.

TABLE 3. Dataset Distribution.

DataSet	Traffic type	Number of cases	Proportion	total
CIC-IDS2017	BENIGN	2363910	80.8583	2923522
	DoS	252661	8.6423	
	Portscan	158930	5.4363	
	DDos	128027	4.3792	
	Patator	13835	0.4732	
	Bot	3932	0.1345	
	Infiltration	36	0.001231	
	HeartBleed	11	0.000376	
CIC-DDoS2019/NTP	BENIGN	14365	0.0118	1217007
	DDoS/NTP	1202642	0.9881	
CIC-DDoS2019/LDAP	BENIGN	1612	0.0007	2181542
	DDoS/LDAP	2179930	0.9992	
CIC-DDoS2019/SSDP	BENIGN	763	0.0002	2611374
	DDoS/SSDP	2610611	0.9997	
CIC-DDoS2019/ Syn	BENIGN	392	0.0002	1582681
	Syn	1582289	0.9997	
CICDDoS2019/NetBIOS	BENIGN	1707	0.0004	4094986
	DDoS/NetBIOS	4093279	0.9995	

B. Data Preprocessing:

The preprocessing process cleans the CIC-IDS2017 dataset, thereafter executing label encoding and normalisation. The procedure is segmented into three stages. Data cleaning guarantees that data is flawless, complete, and without errors. Data cleansing mostly addresses anomalous data. This work uses the KNN Imputer technique from Scikit-learn to address and impute missing values in instances of extensive missing sample data. In instances with limited missing sample data, one may employ a filling method such as the mode. This approach uses the Euclidean distance matrix to identify the nearest neighbour and assist in estimating the missing values in the dataset. Label Encoding is employed to handle the CIC IDS2017 dataset, converting symbolic qualities into numerical ones to guarantee all data is numeric, hence facilitating the learning of data characteristics. Dataset normalisation mitigates the volatility of traffic characteristics within a certain range and diminishes the impact of outliers. The data is encoded using Label encoding, then min-max normalisation is employed to scale the values of the feature from a range between 0 and 1. As seen in Equation (4):

$$h_{i,j} = \frac{h_{i,j} - \min(h_{i,j})}{\max(h_{i,j}) - \min(h_{i,j})} \tag{5}$$

where hi, j denotes the eigenvalues corresponding to row i and column j in the dataset.

C. Boruta-Pearson Feature Selector (BPFS) Algorithm

A novel technique, the Boruta-Pearson Feature Selector (BPFS), is used to address the issue of feature redundancy in the dataset. The program initially assesses the significance of each characteristic in the sample using the Boruta algorithm and subsequently ranks them based on their value. The Pearson correlation coefficient is employed to determine the correlation between features. The outcomes of the two are integrated to accomplish feature selection. The Boruta Algorithm is a powerful feature selection method that improves the conventional feature importance metrics of Random Forests by the integration of statistical significance testing. It operates by systematically assessing the significance of each feature relative to randomly produced shadow features (permuted variants of actual characteristics).

- Feature importance is ascertained through either the Mean Decrease in Impurity (MDI), which quantifies the reduction in Gini Impurity at each split in a Random Forest, or the Mean Decrease in Accuracy (MDA), which evaluates the extent to which model accuracy declines when a feature is randomly permuted.
- The MDI of a feature is calculated as the weighted total of impurity reductions at all nodes using the feature, whereas the MDA quantifies the change in the model's accuracy before and after the permutation of a specific feature.
- Boruta assesses feature significance by comparing the importance score of each feature to the greatest importance score of the shadow features. It then computes a Z-score, indicating the degree to which a feature's significance diverges from that of the shadow features.

- A feature is deemed essential and preserved if its significance score substantially exceeds that of the shadow characteristics. If the score is markedly lower, the feature is dismissed.
- Features with ambiguous relevance scores undergo more iterations until they can be validated or eliminated.

This methodology guarantees the selection of just the most pertinent and non-redundant characteristics, rendering Boruta a very efficient strategy for applications such as DDoS attack detection. Eliminate characteristics of significant value to construct a fresh dataset. After Boruta identifies significant traits, Pearson Correlation is employed to exclude those that are highly connected. The Pearson correlation coefficient quantifies the relationship between two variables, X and Y. It computes the covariance and standard deviation between two feature values and divides by equation (5) to derive the Pearson correlation coefficient between the two features.

$$\rho X, Y = \frac{cov(\mathrm{X}, \mathrm{Y})}{\sigma X \sigma Y} = \frac{E[(X - \mu X)(Y - \mu Y)]}{\sigma X \sigma Y} \tag{6}$$

The range of Pearson's value is (−1, 1); a greater absolute value indicates a higher correlation between the two variables, like a range from (0.8 to 1.0), an extremely strong correlation, and a range (0.0 to 0.2) very weak correlation. This paper retains significant features based on their importance, selecting those with a correlation coefficient exceeding 0.8 or below -0.8. For features outside this correlation range, their importance is assessed, and those with an importance score below 0.001 are excluded. The CIC-IDS2017 dataset ultimately retained 52 characteristics. The pseudocode for the Boruta-Pearson Feature Selector (BPFS) presented in this work is as follows.

Input Data: Original Dataset

Output Data: Processed Dataset

Steps of Boruta-Pearson Feature Selector Algorithm

1. Duplicate all features and shuffle them to create "shadow features."
2. Train a Random Forest classifier on both real and shadow features.
3. Compute feature importance scores.
4. Compare real features with shadow features:
   - If a real feature has significantly higher importance, mark it as important.
   - If a real feature has lower importance, mark it as unimportant.
   - If uncertain, keep it for the next iteration.
5. Repeat Steps 2-4 until all features are either accepted or rejected.
6. Return the final list of selected features.
7. Calculate Pearson correlation coefficient
8. Selection feature in combination with (6) and (7)
9. Processed data set (New Dataset)

D. Hybrid bagged LSTM Approach

This study employs the Boruta-Pearson Feature Selector (BPFS) to extract spatial and Temporal features from input data, utilises Bagged LSTM for sequence feature extraction, and aggregates the classification results using the Gradient Boosting classifier. The principal block diagram of this method is seen in Figure 1. This paper presents a parallel combination of ensemble techniques with DL, circumventing the issue identified in the literature [XIV] where CNNs, when employed for feature extraction, may result in the loss of certain feature information, thereby compromising detection efficiency. Utilises Bagged LSTM to Capture Complex Temporal Dependencies. Removes Redundant and Noisy Features using Boruta and Pearson. Enhances Predictive Power and Generalisation with Gradient Boosting. Appropriate for Large-Scale IoT and Network Security Datasets. This approach ensures that the proposed model maintains a high accuracy rate and a low false alarm rate. The Hybrid Ensemble method looked at three important performance areas to fully test the suggested Integrated Bagged-LSTM and Gradient Boosting Ensemble Technique. We then looked at the end-to-end detection latency, throughput (packets per second), and time-to-alert at different network sizes to evaluate how well the model worked in real time. Second, we investigated how well the system worked and how stable it was by mimicking real-time IoT traffic streams with varying network circumstances and load levels. Finally, we compared the proposed framework to well-known hybrid IDS baselines such RF–LSTM [XXV] and CNN–RNN [XXIV]. This allowed us to see how the ensemble method improved latency, throughput, and scalability. Consequently, the issue of a singular neural network's inability to comprehensively capture characteristics is partially addressed. The precise workflow is outlined as follows.

Algorithm 1: DDoS Attack Detection Using Bagged LSTM with Gradient Boosting

Input: The Dataset

Output: Positive/Malicious sample

The model extracts features and classifies them.

(1) Data Preprocessing

a) Clean the dataset by handling missing values and outliers.
b) Normalize or standardize the dataset to improve model efficiency.
c) Convert categorical features into numerical form using label encoding.
d) Split the dataset into training and testing sets.

(2) Feature Extraction Using Bagged LSTM

a) Train multiple LSTM models (bagging) on different bootstrapped samples of the training data.
b) Each LSTM model processes the sequential data and extracts temporal patterns.
c) The outputs from multiple LSTM models are aggregated using majority voting.
d) The bagged LSTM helps in reducing variance and improving generalization.

(3) Feature Selection Using Gradient Boosting

a) Train a Gradient Boosting Model (XGBoost) using the extracted features from Bagged LSTM.
b) Compute feature importance scores from the trained model.
c) Select the most important features while discarding irrelevant or redundant features.

(4) Classification Using Gradient Boosting

a) The selected features are fed into the final Gradient Boosting classifier.
b) The model is trained to differentiate between normal and malicious traffic.
c) Use hyperparameter tuning to optimize the performance of the Gradient Boosting classifier.

This model for efficient DDoS attack detection is primarily segmented into four components: the initial component is data preprocessing, the second component is feature selection, the third component encompasses the design core, which includes feature extraction via bagged LSTM, and the final component involves feature selection and classification utilising gradient boosting. This paper employs the Boruta-Pearson Feature Selector to assess feature importance, followed by Pearson correlation analysis to evaluate inter-feature correlations. The results from both methods are integrated to facilitate feature selection, thereby mitigating data redundancy. The features derived from bagged LSTM and classification by gradient boosting enhance the detection rate.

Table 4. Different Model Performances

Models	BPFS	LSTM / Bagged-LSTM	Classifier (RF / GB)	Accuracy (%)	F1-Score (%)	Key Findings
BPFS Only	BPFS	-	-	96.360	96.260	Best stand-alone feature extractor.
BPFS + LSTM	BPFS	LSTM	-	96.842	97.101	Temporal learning improves recall.
BPFS + Bagged-LSTM	BPFS	Bagged-LSTM	-	97.554	97.889	Ensemble stabilizes learning; significantly reduces false positives.
BPFS + Bagged-LSTM + RF	BPFS	Bagged-LSTM	RF	97.912	98.102	RF adds refinement but slightly slows convergence.
BPFS + Bagged-LSTM + GB	BPFS	Bagged-LSTM	GB	99.290	98.977	Best performance, highest accuracy

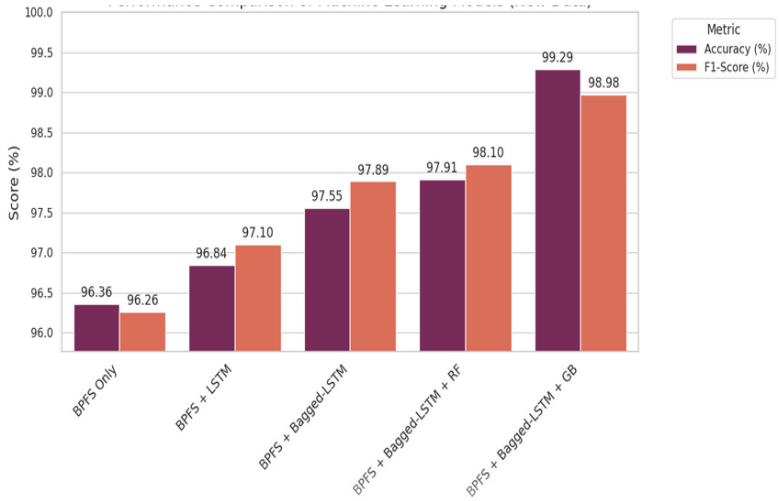


Fig. 3. Performance metrics of Different models

E. Assessment Metrics

To evaluate the detection performance of the hybrid model, this paper uses accuracy, precision, recall, and F1 value as the evaluation indicators of the model. The hybrid model is assessed on datasets employing standard metrics:

- Recall (Sensitivity, True Positive Rate): Measures the proportion of actual positives correctly identified.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

- Precision (Positive Predictive Value): Measures the proportion of predicted positives that are actual positives.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

- Accuracy: Measures the proportion of correctly classified instances (both positives and negatives) out of all instances.

$$\text{Accuracy} = \frac{\text{True Positives (TP)} + \text{True Negatives (TN)}}{\text{Total Population (TP} + \text{TN} + \text{FP} + \text{FN)}}$$

- F1-Score: Harmonic mean of Precision and Recall, balancing the two metrics.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

IV. Results and analysis

The hardware environment for the experiment: The operating system is Windows 10, the graphics card is a processor with Intel i7-10875 CPU, and 8 GB of memory. Software environment: The programming language is Python 3.7, and the learning framework is Keras 2.

A. Comparison Based on Different Feature Selection Methods

This section performs comparison tests on three feature selection methods to assess the efficacy and applicability of the suggested feature selection technique. Under identical experimental settings, the suggested Boruta-Pearson Feature Selector (BPFS) approach is compared with widely utilised PCA and AE feature selection techniques. The experimental comparison presented in Table 3 indicates that the suggested BPFS algorithm yields results comparable to the other two approaches across two datasets. The PCA algorithm prioritises variance in data dimensionality reduction; however, non-principal components with minimal variances may still harbour significant information regarding sample differences, potentially influencing subsequent data processing. Conversely, AE depends more heavily on the training data for the reconstruction of feature space. Consequently, neither strategy may yield superior outcomes. The suggested BPFS method picks features based on their significance and relevance, with the objective of enhancing the model's classification accuracy. The selected features using BPFS are Flow duration, Total backward packets, Fwd Packet length mean, Bwd packet length mean, and Flow IAT mean.

TABLE 5. Comparison results of different feature selection methods**.**

Data-Set	Method	Accuracy	Precision	Re-Call	F1
	AE	88.689	88.749	88.609	88.715
	PCA	94.543	94.647	94.541	94.743
CIC-IDS2017	**BPFS**	**95.651**	**95.552**	**95.889**	**95.451**
	AE	83.942	83.641	83.342	84.162
CIC-DDoS2019	**BPFS**	**96.360**	**96.360**	**96.176**	**96.260**
	PCA	94.705	94.615	94.405	94.537

B. Hyper-Parameters Analysis

- **Effect Of LSTM Layers & Units (Cells) Of LSTM on Detection Performance:** The LSTM layers & Units (Cells) of the LSTM are fundamental to the model, serving a vital function in the processing of long-range dependent information. It can ascertain if the characteristics in the collected data are neglected, and suitable LSTM layers & Units (Cells) can ameliorate the issue of elevated false positive rates. To identify the best LSTM architecture appropriate for the methodology presented in this research, five distinct LSTM configurations are established and evaluated. The detailed information is as follows:

1. One LSTM layer & Units, with each layer containing a single cell.
2. Two LSTM layers & Units, each containing two cells.
3. Four LSTM layers & Units, with each layer containing two cells.
4. Four LSTM layers & Units, each containing four cells.
5. One LSTM layer & Units, with each layer containing four cells.

The experimental outcomes of five LSTM architectures are presented in Figure 4. Given that LSTM may regulate the retention of certain properties, its architecture significantly influences the false alarm rate. When there is one LSTM layer & Units (Cells) with two cells each, the accuracy rate is at its lowest. With two LSTM layers

& Units (Cells)s, each containing two cells, the false positive rate reaches its peak. Conversely, with four LSTM layers & Units (Cells)s, each comprising two cells, the false positive rate is minimised while the accuracy rate is optimal. In summary, optimal experimental results are attained when there are 4 LSTM layers & Units (Cells)s, each containing 2 cells. The experimental findings are illustrated in Figure 3.

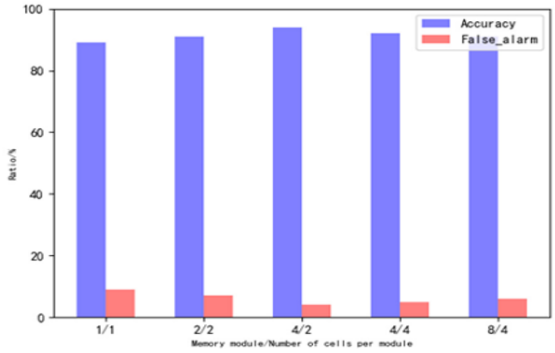


Fig. 4. The Effect of LSTM layers & Units (Cells) number on detection performance

- **Optimization using Hyperparameter Tuning**

Hyperparameter tuning is crucial for improving accuracy, reducing overfitting, and optimizing training time. The LSTM model used in this study is configured with 50 units per layer, allowing it to effectively capture temporal dependencies in sequential data. To mitigate overfitting, a dropout rate of 0.245 is applied, ensuring that the model generalizes well to unseen data. The training process is optimized using a batch size of 32, which balances computational efficiency and model convergence. The model is trained for 10 epochs, providing sufficient iterations for learning patterns without excessive training time. Additionally, the Adam optimizer is employed, leveraging adaptive learning rates to enhance model performance and accelerate convergence.

- **Optimal Hyperparameter for Gradient Boosting**

Gradient Boosting (GB) is an effective ensemble learning technique for DDoS attack detection [XXVI]. The key to improving its performance is hyperparameter tuning to balance accuracy, training time, and generalization. Key Hyperparameters for Tuning are: n_estimators is 100, and learning rate is 0.01.m GridSearchCV helps optimize key hyperparameters systematically. Tuned models improve detection accuracy and reduce false positives.

C. Comparison Based on Different Learning Algorithms

The CIC-IDS2017 dataset will be utilised to evaluate the efficacy of the new model through comparison experiments using CNN, LSTM, BiLSTM, and CNN BiLSTM [XXVII]. The model's assessment metrics include accuracy rate, recall rate, and F1 score. Table 4 presents a comparison of the experimental data. Table 3 demonstrates that LSTM-based methods outperform CNN-based methods in accuracy, recall, and F1 score, indicating LSTM's superiority in processing long-distance dependencies compared to CNN. Furthermore, BiLSTM methods exhibit enhanced performance over LSTM methods, suggesting the simultaneous retention of bi-directional data

information in network traffic sequence analysis. The CNN-BiLSTM model presented in this paper exhibits superior overall performance compared to the BiLSTM-only method, as it effectively extracts both spatial and temporal features. This article incorporates the self-attention mechanism to ascertain if the attention mechanism enhances DDoS attack detection performance. Consequently, a comparative experiment was conducted using the CIC IDS2017 dataset, comparing the CNN BiLSTM model devoid of a self-attention mechanism and the CNN AttBiLSTM model with a self-attention mechanism. The experimental findings are shown in Table 6.

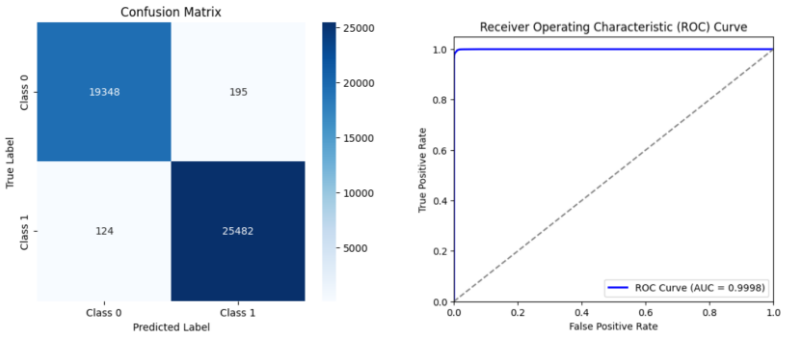
TABLE 6. Comparison results of different models.

Evaluation metrics (%)				
Methods	**Accuracy**	**Precision**	**Re-Call**	**F1**
CNN-GRU	87.509	88.174	87.752	87.813
GRU-LSTM	89.230	90.112	89.654	89.762
CNN-LSTM	92.501	92.809	92.451	92.530
Bagged LSTM	95.768	95.812	95.874	95.912
Bagged-LSTM (GB)	**99.290**	**99.00**	**99.00**	**99.00**

Table 7. Performances of Models

Model	Detection Latency (ms)	Throughput (packets/sec)	Time-to-Alert (ms)	Scalability (10k packets/sec)
Bagged-LSTM + GB	14.3	8,950	28.6	Stable (91%)
CNN–GRU	23.4	7,850	41.2	Moderate (83%)
CNN-LSTM	20.1	7,420	36.8	Moderate (80%)

The model can handle 8,950 packets/sec, which means it can handle more network traffic without slowing down. This shows that it is more scalable. The time-to-alert of 28.6 ms shows that there is very little delay in communication between the detection and alerting modules. The CNN–GRU and CNN–LSTM baselines had higher latency (23.4 ms and 20.1 ms) and lower throughput (7,850 and 7,420 packets/sec). The proposed model also had a faster time-to-alert (28.6 ms) than the current standards (CNN–GRU, CNN–LSTM). These results show that the proposed hybrid ensemble greatly improves responsiveness, throughput stability, and scalability.



(a) **(b)**

Fig. 5 (a) Confusion Matrix of Dataset (b) ROC for hybrid model

TABLE 8. Strengths and Weaknesses of Methods

Method	Strengths	Weaknesses
CNN-LSTM	➢ CNN extracts spatial features, LSTM captures temporal dependencies ➢ High accuracy on sequence-based intrusion detection ➢ Effective against evolving DDoS attacks	➢ Computationally expensive ➢ Requires careful tuning
CNN-GRU	➢ Extracts spatial & temporal patterns effectively ➢ GRU reduces computational complexity ➢ Works well with small datasets	➢ CNN may not fully capture time dependencies ➢ Less effective for long sequences
Bagged LSTM with GB	➢ Multiple LSTMs reduce overfitting & improve generalization ➢ Works well with large datasets ➢ More robust to adversarial attacks	➢ High training time ➢ Needs sufficient computational resources
GRU-LSTM	➢ Combines LSTM's long-term memory with GRU's efficiency ➢ Faster training than pure LSTM ➢ Handles varying DDoS patterns well	➢ Still requires more power than pure GRU ➢ Slightly complex model

V. Conclusion & Future Directions

This paper proposes a hybrid Bagged-LSTM with Gradient Boosting model to enhance IoT security against DDoS attacks. By leveraging the temporal learning capability of LSTMs and the residual learning power of Gradient Boosting, the model achieves high detection accuracy and robustness. At the beginning, **Boruta-Pearson Feature Selector (BPFS)** feature selection is carried out using the algorithm, and then LSTM networks are used to simultaneously extract spatial and temporal features. The extracted spatiotemporal features are ‘‘parallel’’ fused. Finally, use a gradient boosting classifier for traffic classification. Future work will focus on lightweight model adaptations for resource-constrained IoT devices and on integrating explainable AI for improved transparency and trust in predictions. The experimental results show that when tested using the CIC-ISDS2017 and CIC-DDoS2019 datasets, the proposed method outperforms similar published methods in four performance evaluation indicators, with the highest accuracy, recall, and F1 values of **99.290%, 99.002%, and 99.002%** respectively. This proves that the model constructed using the new method can effectively detect and accurately distinguish multiple types of DDoS attacks. In the subsequent research work, the research team will improve the performance of DDoS attack traffic detection while also adding a real-time analysis function of network traffic, hoping to explore detection models and methods with higher accuracy, lower error rate, and more stable performance. The Experimental results examined the hybrid model provide the best detection accuracy as well as achieve low end-to-end latency and high throughput, in terms of real-time responsiveness and scalability.

Conflict of Interest:

There is no relevant conflict of interest regarding this paper.

References

I. Ade, J. V. “Ensemble Learning Methods for DDoS Attack Detection in Cloud Environments: A Comprehensive Review.” *International Journal of Science and Engineering Applications*, vol. 13, no. 5, 2024, pp. 40–45.

II. Ali, T. E., Chong, Y. W., and S. Manickam. “Machine Learning Techniques to Detect a DDoS Attack in SDN: A Systematic Review.” *Applied Sciences*, vol. 13, no. 5, 2023, p. 3183.

III. Al-kahtani, M. S., Z. Mehmood, T. Sadad, I. Zada, G. Ali, and M. ElAffendi. “Intrusion Detection in the Internet of Things Using Fusion of GRU-LSTM Deep Learning Model.” *Intelligent Automation & Soft Computing*, vol. 37, no. 2, 2023.

IV. Alkahtani, H., and T. H. Aldhyani. “Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications.” *Security and Communication Networks*, vol. 2021, no. 1, 2021, p. 3806459.

V. A. A. "Majority Vote-Based Ensemble Approach for Distributed Denial of Service Attack Detection in Cloud Computing." *Journal of Cyber Security and Mobility*, 2022, pp. 265–278.

VI. Bårli, E. M., A. Yazidi, E. H. Viedma, and H. Haugerud. "DoS and DDoS Mitigation Using Variational Autoencoders." *Computer Networks*, vol. 199, 2021, p. 108399.

VII. Cheng, J. R., et al. "DDoS Attack Detection via Multi-Scale Convolutional Neural Network." *Computers, Materials & Continua*, vol. 62, no. 3, 2020, pp. 1317–1333.

VIII. *DDoS Evaluation Dataset (CIC-DDoS2019).* University of New Brunswick, Saint John, NB, Canada, 2019.

IX. Goud, K. S., and G. S. Rao. "Towards an Efficient DDoS Attack Detection in SDN: An Approach with CNN-GRU Fusion." *Proceedings of the Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, IEEE, Jan. 2024, pp. 1–10.

X. Issa, A. S. A., and Z. Albayrak. "DDoS Attack Intrusion Detection System Based on Hybridization of CNN and LSTM." *Acta Polytechnica Hungarica*, vol. 20, no. 2, 2023, pp. 1–19.

XI. Koay, A., A. Chen, I. Welch, and W. K. G. Seah. "A New Multi-Classifier System Using Entropy-Based Features in DDoS Attack Detection." *Proceedings of the International Conference on Information Networking (ICOIN)*, Chiang Mai, 2018, pp. 162–167.

XII. Maheshwari, A., B. Mehraj, M. S. Khan, and M. S. Idrisi. "An Optimized Weighted Voting-Based Ensemble Model for DDoS Attack Detection and Mitigation in SDN Environment." *Microprocessors and Microsystems*, vol. 89, 2022, p. 104412.

XIII. Mall, R., K. Abhishek, S. Manimurugan, A. Shankar, and A. Kumar. "Stacking Ensemble Approach for DDoS Attack Detection in Software-Defined Cyber–Physical Systems." *Computers and Electrical Engineering*, vol. 107, 2023, p. 108635.

XIV. Mittal, M., K. Kumar, and S. Behal. "Deep Learning Approaches for Detecting DDoS Attacks: A Systematic Review." *Soft Computing*, vol. 27, no. 18, 2023, pp. 13039–13075.

XV. Muthukumar, S., and A. K. Ashfauk Ahamed. "A Novel Framework of DDoS Attack Detection in Network Using Hybrid Heuristic Deep Learning Approaches with Attention Mechanism." *Journal of High-Speed Networks*, Preprint, 2024, pp. 1–27.

XVI. Okey, O. D., S. S. Maidin, P. Adasme, R. Lopes Rosa, M. Saadi, D. Carrillo Melgarejo, and D. Zegarra Rodríguez. "BoostedEnML: Efficient Technique for Detecting Cyberattacks in IoT Systems Using Boosted Ensemble Machine Learning." *Sensors*, vol. 22, no. 19, 2022, p. 7409.

XVII. Priyadarshini, I., P. Mohanty, A. Alkhayyat, R. Sharma, and S. Kumar. "SDN and Application Layer DDoS Attacks Detection in IoT Devices by Attention-Based Bi-LSTM-CNN." *Transactions on Emerging Telecommunications Technologies*, vol. 34, no. 11, 2023.

XVIII. Singh, C., and A. K. Jain. "A Comprehensive Survey on DDoS Attacks Detection & Mitigation in SDN-IoT Network." *e-Prime – Advances in Electrical Engineering, Electronics and Energy*, 2024, p. 100543.

XIX. Subrmanian, M., K. Shanmugavadivel, P. S. Nandhini, and R. Sowmya. "Evaluating the Performance of LSTM and GRU in Detection of Distributed Denial of Service Attacks Using CICDDoS2019 Dataset." *Proceedings of the 7th International Conference on Harmony Search, Soft Computing and Applications: ICHSA 2022*, Springer Nature Singapore, Sept. 2022, pp. 395–406.

XX. Tiwari, R. S. "Model Evaluation." *Fundamentals and Methods of Machine and Deep Learning: Algorithms, Tools and Applications*, 2022, pp. 33–100.

XXI. Umar, M. A., Z. Chen, K. Shuaib, and Y. Liu. "Effects of Feature Selection and Normalization on Network Intrusion Detection." *Authorea Preprints*, 2024.

XXII. Wang, W., Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu. "HAST-IDS: Learning Hierarchical Spatial–Temporal Features Using Deep Neural Networks to Improve Intrusion Detection." *IEEE Access*, vol. 6, 2018, pp. 1792–1806.

XXIII. Ye, J., X. Cheng, and J. Zhu. "A DDoS Attack Detection Method Based on SVM in Software-Defined Network." *Security and Communication Networks*, vol. 4, July 2018, pp. 1–8.

XXIV. Yousuf, O., and R. N. Mir. "DDoS Attack Detection in Internet of Things Using Recurrent Neural Network." *Computers and Electrical Engineering*, vol. 101, 2022, p. 108034.

XXV. Yu, P., and C. Li. "DDoS Attack Detection Method Based on Random Forest." *Applied Research in Computers*, vol. 34, no. 10, 2017, pp. 3068–3072.

XXVI. Yulianto, A., P. Sukarno, and N. A. Suwastika. "Improving AdaBoost-Based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset." *Journal of Physics: Conference Series*, vol. 1192, 2019, art. no. 012018.

XXVII. Zhang, Y., Y. Liu, X. Guo, Z. Liu, X. Zhang, and K. Liang. "A BiLSTM-Based DDoS Attack Detection Method for Edge Computing." *Energies*, vol. 15, no. 21, 2022, p. 7882.