

Intrusion Detection System using Stacking of Deep Learning Models with Bte-Lgbm for IoT Networks

Seshu Bhavani Mallampati*

Computer Science and Engineering (AI&ML), Bhoj Reddy Engineering College for Women, Hyderabad, India

E-mail: seshubhavani.m@gmail.com

ORCID iD: <https://orcid.org/0009-0007-5133-8422>

*Corresponding Author

Hari Seetha

School of Computer Science and Engineering, VIT-AP University, Near Vijayawada, Andhra Pradesh, India

E-mail: seetha.hari@vitap.ac.in

ORCID iD: <https://orcid.org/0000-0001-9822-0009>

Received: August 25, 2025; Revised: June 9, 2026; Accepted: June 23, 2026; Published: August 8, 2026

Abstract: The recent development of the Internet of Things (IoT) has increased the severity of security threats. It is mainly caused by IoT devices' inherent weaknesses, making them vulnerable to attack. Therefore, strengthening the security of such network systems is crucial. This study proposes a novel stacking model to identify attacks in the IoT environment. As a first step, we preprocess the data to make it more reliable. To address the issue of class imbalance, synthetic minority samples are generated by using SMOTE-SVM. Then, a novel stacking model was built by integrating four neural networks: deep neural network (DNN), recurrent neural network (RNN), long short-term memory (LSTM) and gated recurrent unit (GRU) with hyper-parameter tuned Light gradient boosting machine (BTE-LGBM). The performance of the proposed stacking model was evaluated on two recent IoT datasets, namely the ToN_IoT and CIC-IoT23. The efficacy of the proposed stacking model is evaluated and compared with Deep learning, machine learning, and state-of-the-art approaches with respect to metrics such as detection rate, precision, accuracy, and F1 score. The findings of our experiments indicate that the suggested IDS achieves a high accuracy of 99.81% and 99.78% for ToN_IoT and CIC-IoT23 datasets, respectively. It might enhance IoT device security, eventually benefiting consumers who depend on these devices. These findings, however, are based on benchmark datasets and could be impacted by variables including class distribution, attack diversity, and dataset characteristics. More research is needed to determine how well the model performs in real-world IoT contexts with changing attack patterns and heterogeneous devices.

Index Terms: Intrusion Detection Systems, Class Imbalance, Stacking, Deep learning, Internet of Things

1. Introduction

IoT is a powerful tool for connecting people worldwide without human intervention. It links everything to the Internet through data-sensing devices that make intelligent tracking, administration, and control possible. It can be used for intelligent health care, transportation, and smart grids, among other things [1]. However, the exponential growth of devices has also resulted in a significant expansion of the network attack surface, providing hostile hackers with more opportunities for exploitation. Various cyberattacks against the Internet of Things (IoT) have been identified. Some are ransomware, man-in-the-middle, SQL injection, and distributed denial of service (DDoS) attacks [2].

Therefore, detecting intrusions in Internet of Things (IoT) systems is more significant. Intrusion Detection Systems (IDS) play a vital role within a layered defence strategy by effectively identifying and detecting potential breaches before any negative impact occurs on IoT networks. It can determine the difference between regular and unusual network traffic behaviour and stop aberrant behaviours before they cause damage to network devices [3]. It was categorized into distinct categories according to their detecting method. Signature-based IDS and anomaly-based IDS are the primary types of intrusion detection systems. A signature-based model is utilized to detect attacks using signatures (patterns) already present in the IDS database. In contrast, anomaly-based detection effectively identifies previously unidentified threats but is hindered by a high rate of false positives [4].

The other issues encountered in the field of IoT are limited resources, such as low computational power and storage capacity. In recent years, Machine Learning (ML) and Deep Learning (DL) methodologies have been widely adopted by enterprises of all sizes and have been shown to be very effective when applied to enormous amounts of data. ML approaches are used to choose the best features from the dataset, which are subsequently classified using ML algorithms[3]. In contrast, DL models automatically extract the best attributes from a dataset [5]. Moreover, DL has gained significant popularity in the field of cybersecurity. This is primarily due to its ability to provide a more advanced analysis of network data, enabling better identification of abnormalities than conventional ML techniques [4].

Ensemble techniques serve as a popular approach in the field of machine learning. The underlying principle of an ensemble technique is the fusion of various methods to address an identified issue. Rather than depending on a singular model for data analysis, the ensemble approach emphasizes using a collection of models and amalgamates their results. Previous studies have shown evidence that an ensemble model yields higher levels of accuracy in comparison to utilizing a single technique [6]. There are several ensemble techniques. The most popular integrators are stacking, boosting, and bagging.

Breiman [8] proposed the concept of bagging as a means to mitigate the uncertainty of a predictor. The approach employs many iterations of a training set, and each is created using a random sampling process with replacement. The number of instances in each iteration equals the size of the original training set. Each of these datasets is used for training a distinct model. The outputs generated by the models are aggregated using a voting mechanism to provide an overall outcome. Boosting is introduced by Schapire[7]. It aims to improve the accuracy of predictive models. However, boosting works in a fundamentally different way. Instead of combining multiple models in parallel, boosting builds a sequence of models where each model tries to correct the errors made by its predecessor.

In contrast to bagging and boosting, stacking integrates many distinct classifiers, such as a support vector machine (SVM), neural network (NN), logistic regression (LR), etc. Stacking is a technique that involves two levels: the base learner, referred to as level-0, and the meta learner, referred to as level-1. The base learner employs diverse models to acquire knowledge from a given dataset. The results generated by each model are aggregated to form a novel dataset. In the novel dataset, every individual observation is associated with the corresponding actual value that it is intended to forecast. The meta-model then uses the dataset to generate the final outcome. Therefore, stacking can lead to better predictive accuracy when compared to individual model prediction. By combining the strengths of different models, we can reduce the weaknesses of any single model. Additionally, it reduces overfitting too.

In this study, we proposed a novel stacking model by combining various DL and ML models. To assess the effectiveness of our proposed stacking model, we compare outcomes with both level-0 and level-1 models. In addition, we conduct a comparative analysis of our model with state-of-the-art intrusion detection system (IDS) models that use comparable methodologies, as documented in recent literature pertaining to the same datasets. The experimental findings demonstrate that our model performs better than other deep-learning models. Furthermore, our model's outcomes are comparable to those achieved in other research endeavours within the same domain.

The primary contributions are presented as a solution to the above-mentioned issues and limitations.

- In this work, we provided a single-level stacking strategy, whereby four DL models like Deep neural networks (DNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent neural network (GRU) are used in the first step. Finally, an ML-based Light gradient Boosting Machine (LGBM) is used to fit the predictions generated by the individual classification models, resulting in final predictions of IoT intrusions.
- The ML and DL models function more effectively with balanced data; however, network traffic contains fewer attack samples, which makes these models biased towards majority class samples. Therefore, in the proposed method, we increased the minority samples using SVM with SMOTE.
- Bayesian Optimization with Tree-structured Parzen Estimator (BTE) is adopted to estimate the hyperparameters of LGBM. This allows for a more extensive search space to be examined while maintaining the same level of computational cost.
- This study aims to conduct a comprehensive performance assessment and compared the results of our stacked model with individual ML and DL techniques respectively.

1.1. Outline

The subsequent sections of this article are structured in the following manner: Section 2 provides a comprehensive analysis of several ML, DL, and ensemble models currently being used in IDS. Section 3 of the document encompasses the proposed technique. Section 4 provides an overview of the experimental outcomes obtained by benchmarking the proposed stacking model against existing state-of-the-art works. This is followed by a conclusion in Section 5.

2. Literature Review

This section provides a quick summary of recent literary developments. The primary objective focuses on studies that used ML, DL and ensemble learning techniques to detect cyber-attacks in IoT environments.

Elsayed et al. [1] suggested a DL-based Secured Automatic Two-level Intrusion Detection System (SATIDS) to detect attacks in IoT networks. They built their model using the following layers: ReLU, LSTM, dropout, fully connected, and Softmax layer. Moreover, they evaluated their model efficiency by selection optimal features by maximum frequency and minimum relevance (MRMR) and selected 20 optimal features. They have tested their model on InSDN and ToN_IoT datasets. The two datasets contain imbalance; the authors have not addressed issue of imbalance in their work.

Khanday et al. [8] suggest a lightweight intrusion detection system for IoT networks that combines data preprocessing methods with deep learning and machine learning classifiers. Once the data was processed, the optimal features were selected by adopting Extra Tree classifier. They showed that multiple classifier types may be used to build compact intrusion detection systems that are ideally suited for defending against DDoS attacks in IoT networks. For the evaluation, they used linear support vector machine artificial neural networks (ANN), (SVM), a hybrid model in combination with Convolution neural networks and LSTM (CNN-LSTM), LR, and Naive Bayes (NB). Then, for analysis, they used BOT-IoT and TON-IoT datasets. As their model performs better with ToN_IoT but in the case of BoT_IoT the AUC value in the case of DDoS attacks is less compared to all attack class data.

Hossain et al. [9] proposed an innovative ensemble-based machine-learning approach to detect attacks in the network. The suggested method was evaluated on datasets like UNSW-NB15, KDD Cup, NSL-KDD, ToN_IoT, etc., using a variety of ensemble techniques, including Gradient Boosting, RF, Adaboost, Bagging, Gradient XGBoost, and Simple Stacking. The best features are chosen using statistical methods, including principal component analysis, mutual information, and correlation analysis. Their experiments showed that RF outperforms other models. The authors have not mentioned how the hyper parameters were selected and they have not addressed issue of imbalance in their work due to which the model may be biased towards majority samples.

Sharma et al. [5] developed a deep learning-based IDS for identifying attacks in IoT networks. Initially, they removed correlated features, and then synthetic samples were generated using Generative Adversarial Networks. Then, for the classification of attacks, they used deep neural networks. They have evaluated their suggested model on the UNSW-NB15 dataset and obtained an accuracy of 91%. For analysis they used single dataset moreover their findings may have been significantly enhanced with the inclusion of another dataset.

Lazzarini et al. [6] proposed IoT intrusion detection based on a stacking ensemble of deep learning models. The ensemble model was built by combining four separate DL models, Multilayer Perceptron (MLP), CNN, DNN, and LSTM-based models, with a fully connected DL layer. They have conducted experiments on ToN_IoT, CICIDS2017, and SWaT datasets, and obtained accuracies of 99.6%, 98.7%, and 99.6%, respectively. The authors have not addressed the issue of imbalance present in the datasets.

To solve the issue of class imbalance, Thakkar et al. [10] describe an IDS for IoT networks using an ensemble-based deep learning model. They used the Bagging strategy with class weights to get a more even distribution of classes in the training set that is then fed into the DNN for training and classification. As a result, the efficacy of the intended IDS is enhanced by using an ensemble bagging learning strategy and balancing the distribution of classes with class weights. Experiments are conducted using NSL-KDD, UNSW_NB-15, CIC-IDS-2017, and BoT-IoT datasets and attained accuracies of 98.90%, 96.70%, 98.74%, and 98.99%, respectively. Though their model has good accuracy with all the datasets but their model has less detection rate with respect BoT-IoT dataset.

To support the development of security analytics applications in actual IoT operations, Neto et al. [11] provided a novel IoT attack dataset named CIC-IoT23 with recent attacks. Then, they tested their proposed dataset on various ML and DL models like perceptron, LR, deep neural networks, AdaBoost and RF. Their results proved that the data set can be effectively worked with both ML and DL models. Though their model has a notable accuracy but the score in terms of precision, recall, and F1-score are less because they have not handled the imbalance in the datasets where the models are biased towards majority samples.

To identify attacks, Zhou et al. [12] suggested an intrusion detection method based on improvised LSTM. They incorporating the notions of state change, incremental LSTM can capture the dynamic information of traffic. On UNSW-NB15 and CICIDS2017 datasets, they evaluated their model. Their finding proved that their model works effectively in identifying attacks with an accuracy of 98.01% and 99.74%. Though the model has attained good accuracy, but it may exhibit sensitivity to stealth assaults, since the incremental LSTM model is specifically built to capture the dynamic characteristics present in the network data.

Bhavani et al. [13] proposed an IDS to detect distributed denial of service attacks in an IoT environment. They built their model by selecting optimal features using Boruta feature selection. Then, the optimal features are trained using the LGBM model. To balance the dataset, they applied K-Means SMOTE. Moreover, to enhance the performance, the optimal features of LGBM are tuned by the SHAP-hyper-tune method. They evaluated their model on the CIC-IDS 2017 dataset and obtained an accuracy of 99.9%. They have used very less data for training their model which may lead to overfitting and model generalization and using more datasets might have greatly enhanced their results.

Kasongo et al. [14] described an IDS framework for IoT, cloud computing, etc., by employing several recurrent neural network approaches in conjunction with the XGBoost feature selection method. The feature selection method was applied to NSL-KDD and the UNSW-NB15 datasets and selected 22 and 17 optimal features, respectively. Then, the optimal features are trained by using various Recurrent Neural Networks (RNNs), namely, Long-Short Term Memory (LSTM), Gated Recurrent Unit (GRU) and Simple RNN. Form the analysis they have suggested that Simple-RNN would

be the best option in a situation when computer resources are less and also, they have not addressed the issue of class imbalance.

Latif et al. [15] developed a dense random neural network (Dnrrann) for identifying attacks in IoT networks. To improve the effectiveness of their model, they tuned the parameters of Dnrrann. Then, they tested their method on ToN_IoT dataset. Their experimental results showed their models work effectively in identifying attacks with an accuracy of 99.15%. Although they provide an outstanding examination of that technique, the study might have benefitted from additional validation using more IoT datasets.

De Souza et al. [16] suggested a two-phase ensemble model for detecting attacks in the IoT Fog environment. In the initial phase of the process, the traffic is analysed using an Extra Tree binary classifier. Then, an ensemble method is built by combining Extra Tree, Deep Neural Network, and RF in the second step of the process to investigate events identified as attacks. They have tested their model on IoTID20, Bot-IoT, CICIDS2018, and NSL-KDD datasets. The authors provide only theoretical view of countermeasure module in their framework.

Thockchom et al. [17] proposed an ensemble-based IDS using machine learning models like LR, Gaussian NB, and decision tree as the base models and stochastic gradient descent as the meta-classifier. The suggested ensemble model was tested on KDD Cup, UNSW-NB15, and CIC-IDS2017 datasets. Their experimental results showed that ensemble-based IDS performs better when compared to individual models by an accuracy of 99.84%, 93.88% and 99.80%. While they have good performance, the outcomes of UNSW-NB can be further improved by tuning hyperparameters their model.

Abbas et al. [18] suggested an ensemble framework for identifying attacks in the IoT environment. They built their model by integrating NB, LR, and decision tree with hard voting. They tested the approach with the CICIDS2017 dataset and obtained an accuracy of 88.92% on binary classification. They have used only one metric to analyse the results moreover they have not addressed the issue of imbalance in CICIDS2017 dataset and the performance can be further improved by tuning the models.

Kumar et al. [19] put forward a methodology for detecting cyber-attacks in IoMT (internet of medical things) networks, which combines ensemble learning and a fog-cloud architecture. The ensemble architecture incorporates Naive Bayes, Decision Tree, and Random Forest as primary individual learners at the first level. In the subsequent stage, the classification outcomes are used by XGBoost in order to discern between normal and attack instances. Furthermore, in the context of dynamic and heterogeneous networks such as the IoMT, fog and cloud computing, they suggested a deployment architecture for their framework. Specifically, they suggest the use of Software as a Service on the fog side and Infrastructure as a Service on the cloud side.

Awajan [20] offers a cutting-edge intrusion detection solution for Internet of Things devices that is based on DL. The author designed a four-layer deep Fully linked (FC) network architecture to identify malicious traffic. The experimental outcomes of FC show dependable performance for both simulated and actual attacks. Although the authors study presented the benefit of using authentic data, it relied on a comparatively limited dataset that may lack the necessary diversity to serve as an extensive training platform for their model.

Hongsheng et al. [21] provides a hierarchical intrusion detection model that integrates various deep learning models with attention mechanisms to address the issues of poor detection accuracy, a structure that has to be tuned, and a high false positive rate. The optimal attributes are initially extracted, and noise is reduced using the stacked convolutional denoising autoencoders model. Then, to extract spatial features, CNN is used to extract the temporal aspects of network data by using the BiLSTM. Then, a Self-Attention mechanism is included to weight each time step's output in order to gather and remember the pertinent data. Finally, SoftMax is used to classify the data, and they have tested their model on NSL-KDD and obtained an accuracy of 99.26%. While the model has a notable accuracy but the model has higher false positive rate of 7.53%.

To identify anomalies, Aktar et al. [22] suggested a deep learning-based model that uses a contractive autoencoder. They used the compressed form of the input data to train their model, and then they used a stochastic threshold technique to identify attacks. Their proposed model was tested on NSL-KDD, CIC-IDS2017, and CIC-DDoS2019 datasets and the accuracy is recorded as 96.08%, 94.25% and 97.58%, respectively. Ding et al. [23] to identify attacks against Internet of Things devices introduces a deep learning framework named DeepAK. They built their model in three blocks: the temporal representation block, the detection block, and the residual-based-spatial (RBS) representation block. To extract a feature representation from the previous layer's output, the RBS block uses five residual blocks. The four convolutional layers are linked in parallel via a skip connection inside each block to prevent disappearing or expanding gradients. Subsequently, the second block learns a temporal representation to identify cyber threats using the retrieved spatial model. They have evaluated their model on Edge-IIoTset, UNSW-NB15, and TON-IoT datasets, and obtained an accuracy of 94.96%, 98.41%, and 90.57% respectively.

Popoola et al. [24] proposed a stacked recurrent neural networks bot identifying Botnet attacks in smart homes. They combine various recurrent neural networks to build a stacked model. They evaluated their model on the BoT-IoT dataset. Their experimental outcomes proved that stacked RNNs perform better when compared to RNNs. The model generalizability can be analysed by evaluating their model on various datasets.

The literature review shows that some models use individual ML and DL approaches. At the same time, some ensemble strategies are described in the literature. But still, there is a need for effective attack detection methods despite the volumes and forms of network traffic constantly changing. Most IDS datasets contain class imbalances, which might

lead to detection biased towards majority classes. Therefore, there is a critical requirement to include assessment since specific approaches fail to take the data imbalance issue into account. For validation of the model, some of the authors use a single dataset. Moreover, the IoT context was not considered in some works.

3. Methodology

This study presents a novel approach to detect IoT network attacks using stacked ensemble IDS. The overall architecture is exhibited in Fig. 1. Further, a comprehensive overview of datasets, data preparation, handling of class imbalance, and proposed stacking model are described in the following subsections.

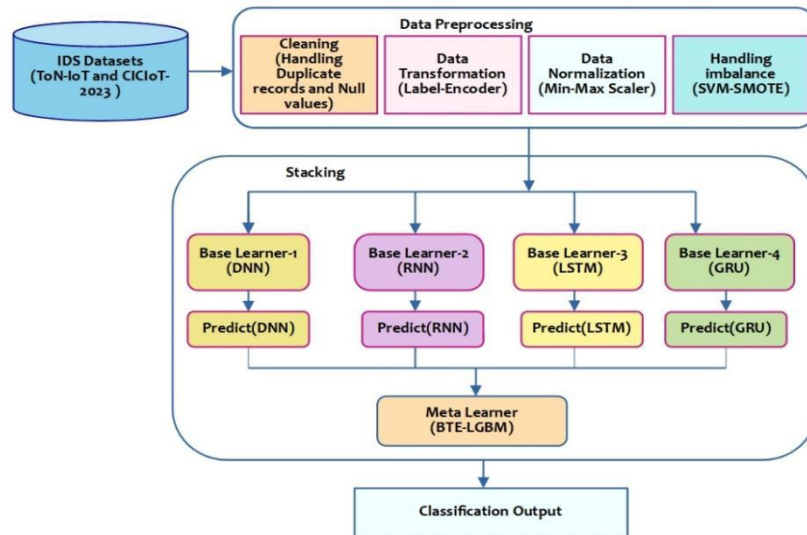


Fig. 1. Proposed IDS Architecture

3.1. Datasets

3.1.1. ToN_IoT dataset

This dataset was gathered by the UNSW Canberra Cyber IoT Lab, School of Engineering and Information Technology, UNSW Canberra @ The Australian Defense Force Academy (ADFA). The dataset was acquired by parallel processing to collect a diverse range of cyber-attack events from IoT networks. The network consists of various components, such as virtual devices, IoT sensors, cloud platforms, and physical systems, which provide diverse data sources. It contains nine attack classes and one normal class, and the frequency distribution of normal and attack classes is shown in Table 1. Finally, it contains 43 attributes, with a total of 461,043 records. In the proposed study, we have considered all attack instances and converted them into a single class for analysing binary classification, and the data distribution is shown in Table 3

Table 1. Class distribution of the ToN_IoT dataset

Class	Name of class	No.of instances	Total
Attack classes	Backdoor	20000	1,61,043
	DDoS	20000	
	DoS	20000	
	Injection	20000	
	MITM	1043	
	Password	20000	
	Ransomware	20000	
	Scanning	20000	
	XSS	20000	
Normal class	Normal	300000	3,00,000

3.1.2. CIC IoT Dataset 2023

Neto et al. [11] developed a novel IoT data set in the Canadian Institute for Cybersecurity IoT lab. A total of 33 attacks were carried out on an IoT topology of 105 devices. The attacks are categorized into seven distinct types: DoS, spoofing, brute force, Recon, DDoS, Web-based, and Mirai. The dataset is available in PCAP and 169 CSV files. The original data distribution of 169 csv files is shown in Table 2. In the current work, we have considered 1 CSV file for training and testing our proposed model, which contains 238687 instances, as shown in Table 3.

Table 2. Original data distribution of CIC IoT Dataset 2023

Class	Name of class	No.of instances	Total
Attack classes	Web	24,829	4,55,88,384
	Brute force	13,064	
	Spoofing	4,86,504	
	Mirai	26,34,124	
	DoS	80,90,738	
	DDoS	33984560	
	Recon	3,54,565	
Normal class	Normal	10,98,195	10,98,195

Table 3. Data considered for Proposed analysis

Dataset	Class	Records
CIC IoT Dataset 2023	Normal	5600
	Attack	233087
ToN-IoT	Normal	300000
	Attack	161043

3.2. Data Preprocessing

Data preparation is an essential step in the whole process since it is necessary to clean up and convert the data before machine learning models can use it productively. The final model's accuracy and efficacy significantly depend on the input data quality. Firstly, it allows in the identification and elimination of inconsistencies in the dataset. Secondly, it facilitates the data normalization, enhancing its analysability. Lastly, it contributes to making the data more manageable and suitable for ML and DL models. The Scikit-learn package was used to do preliminary processing on the data before training the proposed model. The procedure consists of different stages.

Initially "*isnull()*" method is applied to check the Null values in both datasets. By the analysis, it is observed that the datasets have no missing values or Null values. The duplicate records are removed. Then, it is identified that the ToN_IoT data set contains 26 categorical features such as 'ssl_version', 'ssl_cipher', 'weird_name', and 'dns_query', etc. Python offers diverse libraries to transform the data, such as label encoding and one-hot encoding. We adopted label encoder to convert categorical data to numerical. Because it does not affect the dimensionality of the dataset and does not result in additional time expenses for the training of the classifier. The subsequent step is data normalization, which entails scaling the values within the range of 0 to 1. This phase is crucial in order to mitigate the presence of outliers within the data, which may bias the model while training. Hence, we adopted the Min-Max Scaler of the Scikit-learn package to normalize the data. The mathematical form is as follows

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

3.3. Data Resampling

In this work, we used ToN_IoT and CIC-IoT23 datasets. By observing Table 3, it was identified that there is an imbalance in the datasets. Under these circumstances, traditional classifiers tend to focus on the majority class and overlook the minority class. Thus, to prevent problems with data imbalance, it is essential to distribute data uniformly according to class labels. The best method of handle this is data resampling. Hence, the proposed work uses SMOTE with SVM to balance the data. It is an improved form of SMOTE variations in which an SVM classifier is included to address the issue of incorrectly identified observations in borderline. After training SVM on the entire dataset, the boundary line

was generated using support vectors. Then, artificial instances are generated randomly by connecting each minority class support vector with a certain number of its closest neighbours. The approach used for this generation will be either interpolation or extrapolation, as shown in Fig. 2., depending on the density of majority class samples. While conventional SMOTE may introduce noisy synthetic samples, our study employs SVM-SMOTE, which generates minority-class samples near the SVM decision boundary rather than randomly interpolating all minority instances. This approach reduces the likelihood of creating noisy or redundant samples and improves class discrimination.

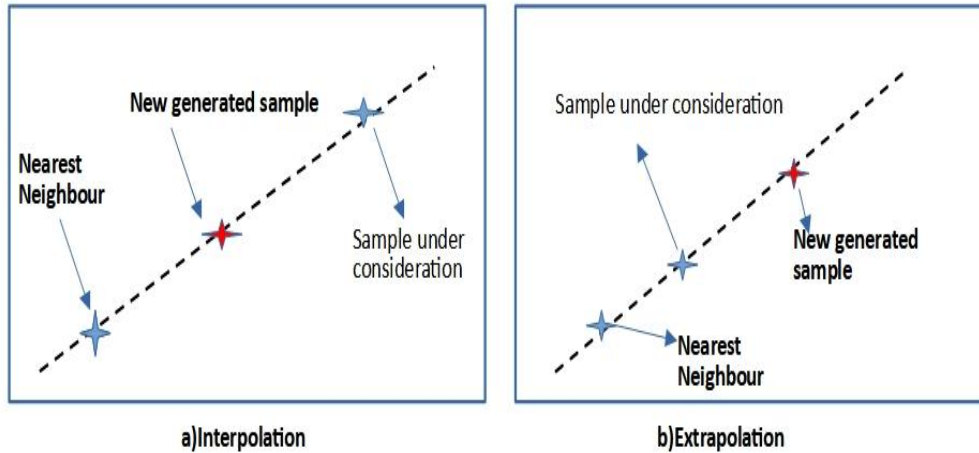


Fig. 2. Synthetic samples generated by a) interpolation and b) extrapolation

3.4. Proposed stacking model

The stacking approach fuses base learners to form an ensemble model, much like the bagging and boosting methods. Despite this, it is essential to note that two significant distinctions exist among boosting/bagging and stacking strategies. In the ensemble learning framework, stacking often involves diverse base learners, which refer to classification algorithms that vary from one another. However, in the case of bagging and boosting techniques, which typically employ homogeneous learners or classification algorithms that are identical to one another. Furthermore, in bagging and boosting, the basic learners are merged in a deterministic manner. Typically, this task is accomplished via either a weighted summation or a voting approach. In the stacking method, the base learners are aggregated by means of a non-deterministic process to a meta-learner. The benefit of this proposed model is that it is a combination of different classification approaches that mitigates the limitation of each of the classifiers to improve overall performance. Therefore, it improves the power of interpreting extremely diverse data. Stacking is the only strategy that reduced the false positive rate by a significant amount [25]. The current study used four DL models as base learners: DNN, RNN, LSTM, and GRU. Additionally, LGBM is employed as a meta-learner to construct the ensemble. The basic framework is shown in Fig. 1, while the subsequent subsections include descriptions of individual base learners and the meta learner.

3.4.1. Base Learners

The stacking ensemble methodology used in our study employs DNN, LSTM, RNN, and GRU as base learners.

a) Deep Neural networks (DNN)

The multi-layered deep neural network (DNN) architecture, as shown in Fig. 3, is built for the proposed stacking model. This approach provides a more comprehensive examination of the data, necessitating a higher computational capacity [6]. It comprises an input layer with dimension equivalent to the number of attributes in the datasets adopted for training, four densely connected hidden layers, and an output layer containing a single neuron for binary categorization. The ReLU activation function is used in each hidden layer to enhance the efficacy of the learning process. In addition, a dropout layer is included after each dense layer in order to promote generalization and mitigate the problem of overfitting. Moreover, the Sigmoid activation function is used in the output layer to provide predictions. In addition, the DNN architecture's performance is assessed by using the binary cross-entropy loss function. The configuration details of DNN are provided in Table 4.

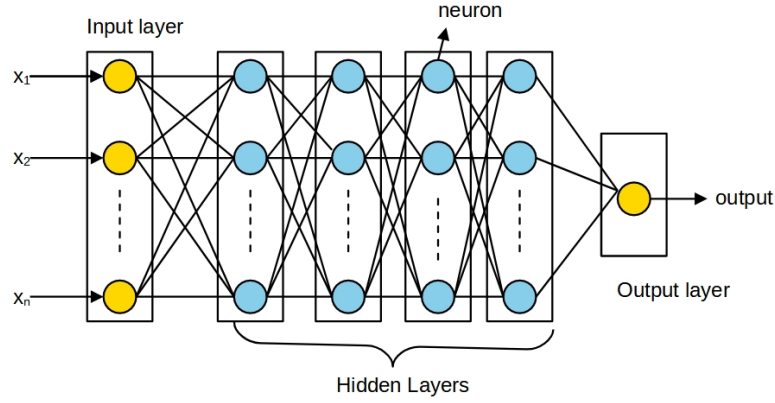


Fig. 3. DNN architecture

b) Recurrent neural networks (RNN)

RNN is adopted as a part of stacking. In contrast to conventional neural networks, RNNs have the ability to process information bidirectionally, hence avoiding the limitation of unidirectional information processing. It can iterate over several layers and exhibit the capacity to temporarily retain information for further use [26]. The structure of the RNN is depicted in Fig. 4. RNN is visually represented by unrolling the network to demonstrate the hierarchical nature. A forward-backward propagation approach is used to develop the RNN network through time steps [14]. The mathematical form of RNN is as follows from (2)-(4).

$$h_c = \bar{\omega}(h_{c-1}, x_c) \quad (2)$$

$$h_c = \bar{\omega}([Wh_c + Uh_{c-1}] + b) \quad (3)$$

$$y_t = \bar{\omega}([Wh_c + Uh_{c-1}] + b) \quad (4)$$

Let X_c the input to RNN h_c be the current hidden output at the time c , $\bar{\omega}$ is a nonlinear activation function, h_{c-1} is the output of the previous state at the time c , and the weights associated with the connections between the input layer and the first hidden layer are denoted as W . The weight associated with the connection between the present hidden layer and the subsequent hidden layer is represented by the symbol U and b is the bias. Further, at each iteration i , the error between the actual and the predicted output is calculated as follows

$$L(\bar{y}_i, y) = \sum_{i=1}^N (\bar{y}_i - y_i) \quad (5)$$

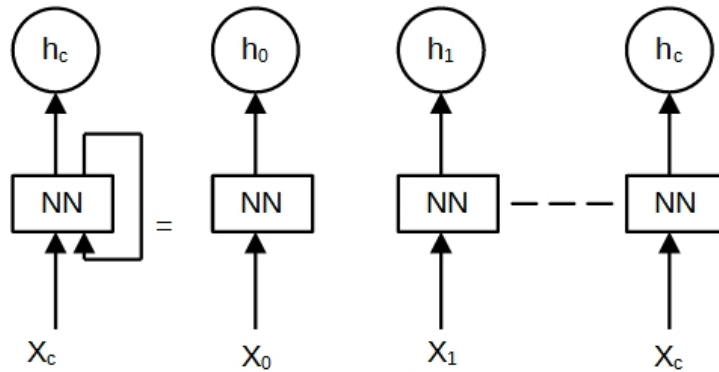


Fig. 4. Unrolled RNN

c) Long Short-Term Memory (LSTM)

The training of RNNs may be hampered by the issues of vanishing or exploding gradients, which limits their ability to learn long-term dependencies to a certain degree. In order to address this constraint, long short-term memory (LSTM), a variant of the Recurrent Neural Network (RNN), was specifically developed to capture and understand the complex relationship between the output and input variables, particularly in environments where there are long-range dependencies

[27]. LSTM exhibits outstanding efficiency in handling sequential data. The LSTM cell within an LSTM network is responsible for retaining and preserving the network's state over time. The use of input gates, output gates, and forget gates enables the provision of communication with cells as shown in Fig. 5. The mathematical representation of the information flow inside an LSTM network is as follows:

$$d_c = \sigma(W_g \cdot [k_{c-1}, x_c] + b_d) \quad (6)$$

$$f_c = \sigma(W_c \cdot [k_{c-1}, x_c] + b_c) \quad (7)$$

$$S'_c = \tanh(W_s \cdot [k_{c-1}, x_c]_{b_c}) \quad (8)$$

$$S_c = d_c * S_{c-1} + P_t * S'_c \quad (9)$$

$$g_c = \sigma(W_g \cdot [k_{c-1}, x_c] + b_g) \quad (10)$$

$$k_c = g_c * \tanh(S_c) \quad (11)$$

where LSTM adopts sigmoid and tanh activations, W is the weight, b is the bias, S is the cell state, and K_c is the output.

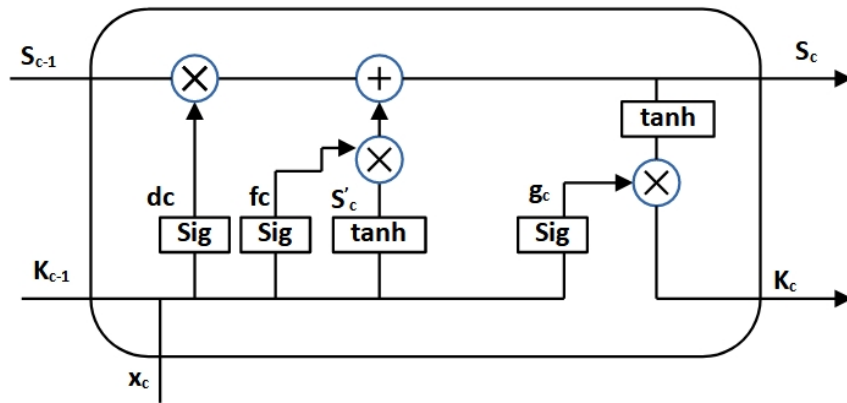


Fig. 5. LSTM architecture

d) Gated Recurrent Unit (GRU)

GRU networks belong to the class of RNN, which was proposed in 1997. The technique addresses the issue of vanishing and exploding gradients in conventional RNNs during the process of learning long-term dependencies. Equations 12-14 delineate the process through which memory cells in each hidden layer of the GRU networks are updated at each time step, as shown in Fig. 6. Like the LSTM model, the input value connects in the interaction process with the information derived from the previous state. This connection facilitates the computation of various values associated with intermediate gates [28]. These computed values are later used to determine the output value. The mathematical formulation of the GRU is represented as follows:

$$a_c = \sigma(W_a \cdot [S_{c-1}, x_c] + x_c) \quad (12)$$

$$b_c = \sigma(W_b \cdot [S_{c-1}, x_c]) \quad (13)$$

$$\tilde{S}_c = \tanh(W \cdot [a_c * S_{c-1}, x_c]) \quad (14)$$

$$S_c = (1 - b_c) * S_{c-1} + b_c * \tilde{S}_c \quad (15)$$

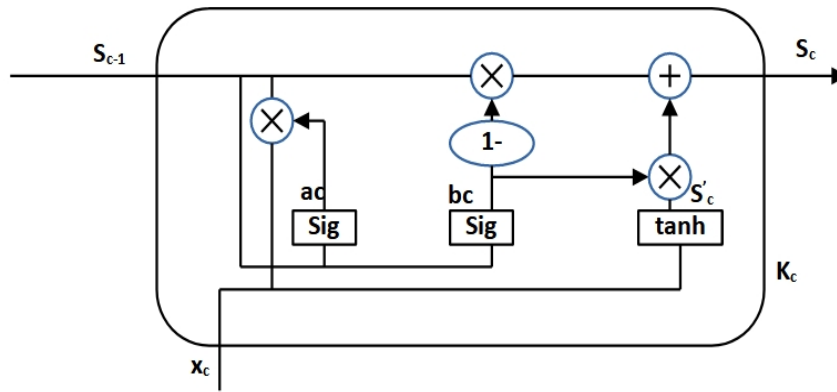


Fig. 6. GRU architecture

where W is the weight, a_c is the reset gate vector, b_c is the update gate vector, S'_c is the candidate vector, and S_c is the output of GRU. The GRU employs both the Sigmoid (σ) and hyperbolic tangent ($\tanh$) activation functions to narrow down the range of information.

In the proposed work, RNN, LSTM, and GRU models contain four layers with 64 units, respectively, and output dense layer with 1 unit for binary classification. In addition, a dropout layer is included after each layer to mitigate the problem of overfitting. Further binary cross-entropy is used to calculate loss generated by the model, and to promote generalization, the Adam optimizer is applied. Moreover, the models are trained for 100 epochs with a batch size of 512. The configuration details are shown in Table 4.

Table 4. Configuration details for CICIOT23 and ToN IoT datasets

DNN configuration		RNN/LSTM/GRU
Parameter	Value	Value
Model	Sequential	Sequential
Input Size	ToN IoT-43, CICIOT23-46	ToN IoT-(1,43), CICIOT23-(1,46)
Number of layers	4	4
Number of neurons in hidden layers	512,128,64,32	64
Activation at the hidden layer	ReLU	sigmoid
Activation at the output layer	Sigmoid	tanh
Epochs	100	100
Batch size	512	512
Dropout	0.01	0.01
Loss	Binary cross entropy	Binary cross entropy
Optimizer	Adam	Adam

3.4.2. Meta learner

Instead of using a deep learning model as the meta-learner, we used a machine learning-based Bayesian Optimization with Tree-structured Parzen Estimator with LightGBM. Because LightGBM considerably minimizes both computational time and memory usage.

a) BTE-LGBM

Microsoft's research team devised a novel, effective, lightweight gradient-boosting decision tree (GBDT) in 2017 called LightGBM [29]. LightGBM utilizes cutting-edge methods, namely the histogram method based on the traditional gradient boosting tree, an independent feature merge algorithm, and a depth-constrained leaf-wise strategy. These techniques aim to effectively handle datasets with a high volume of instances and a large number of attributes, respectively. The underlying concept is the linear combination of N weak regression trees to form robust regression trees, and it is represented as follows

$$Y(x) = \sum_{n=1}^N y_n(x) \quad (16)$$

The (BTE) Tree-structured Parzen Estimator is a method for optimizing hyperparameters sequentially based on the Bayesian optimization principles described by the Bayes theorem [30]. In the current work, while training the meta learner (LGBM) the BTE optimizer is used to identify the optimal hyperparameters in order to enhance the performance of the model and mitigate the occurrence of overfitting. This approach can provide extensive variables inside the search space, including log-uniform, uniform, and normal distribution values. The main objective of Bayesian optimization is to minimize the objective function $f(x)$ as follows

$$x_{op} \in \operatorname{argmin} f(x) \quad (17)$$

The optimal hyperparameters of search space X is represented as x_{op} . Where Bayesian optimization employs an iterative approach for searching the optimal hyperparameter, using the acquisition function by which it balances the trade-off between exploration and exploitation. The objective of the Tree-structured Parzen Estimator for Bayesian optimization is to identify the best hyperparameters of the model by simultaneously modelling the conditional probability distribution $p(x|y)$ and the prior probability distribution $p(y)$. The distribution $p(y)$ denotes the set of solutions, whereas $p(x|y)$ reflects the distribution of the parameter x given the knowledge of the solution. Furthermore, the use of the optimization criteria known as Expected Improvement is employed to direct the exploration of the configuration space. It is determined as follows:

$$p\left(\frac{x}{y}\right) = \begin{cases} a(x) & \& y \leq y^* \\ b(x) & \& y > y^* \end{cases} \quad (18)$$

where y^* is the predefined threshold, the function $a(x)$ represents the density estimate that yields a loss function value less than y^* , whereas $b(x)$ represents the density composition that yields a loss function value bigger than $x(i)$. Table 5 shows the optimal hyperparameters of LGBM tuned by BTE.

Table 5. Optimal parameters of LGBM

Ton_IoT	CIC-IoT23
Num leaves =56	Num leaves= 56
learning rate= 0.1	learning rate=0.3
n_estimators =72	n_estimators= 99
max_depth=44	max_depth=30
bagging fraction =0.5	bagging fraction= 0.5

3.4.3. Model Architecture

The proposed deep stacking model employs base models such as DNN, LSTM, RNN, and GRU. The base models are integrated with a meta-learner. Furthermore, it undergoes an extra training phase to get the final output. As a result, the stacking model becomes a stand-alone model that combines a meta-learner with four separate sub-models. Each sub-model has a distinct categorization methodology. Combining them makes it possible for the proposed model to simulate the properties found in the data effectively.

4. Performance Evaluation

4.1. Experimental setup

The experimental configuration comprises a workstation equipped with an Intel® Core™ i7-5960X CPU, 64 GB of RAM, and an NVIDIA GeForce RTX 2060 GPU. The workstation operates on the Windows 10 Operating System (OS). The implementation was done with Python programming language, using Scikit-Learn, TensorFlow, and Panda's libraries.

4.2. Evaluation Measures

The effectiveness of the proposed stacking model is analyzed by employing various performance metrics like Accuracy (AC), Precision (Pe), F1-score, Detection Rate/Recall (DR) and False Positive Rate (FPR). These metrics are constructed using the following terms: true positive (TP), false positive (FP), true negative (TN), and false negative (FN)

$$AC = \frac{TP + TN}{TP + FP + TN + FN} \quad (19)$$

$$DR = \frac{TP}{TP + FN} \quad (20)$$

$$Pe = \frac{TP}{TP + FP} \quad (21)$$

$$F1 - Score = \frac{2 * Pe + DR}{Pe + DR} \quad (22)$$

$$FPR = \frac{FP}{TN + FP} \quad (23)$$

IoT intrusion detection datasets frequently show class imbalance, meaning that attack instances and regular traffic are not equally represented. In these situations, Accuracy by itself might give an unduly positive evaluation of model performance. As a result, more measures as shown in equation 20,21,22 and 23 are needed to fully assess the IDS.

5. Results and discussion

This section deals with the performance analysis of the proposed stacking IDS in two cases, and it is compared with the latest state-of-the-art works. The cases are as follows

- Analysis of the proposed stacking model with imbalanced data.
- Analysis of the proposed stacking model with balanced data.

5.1. Analysis of the proposed stacking model with imbalanced data

In this part, the tests were conducted utilizing the imbalanced data of the ToN_IoT and CIC_IoT23 datasets, as shown in Table 3. To begin with, Tables 6 and 7 show the outcomes of the two datasets. To check the effectiveness of the proposed stacking model, the outcomes were tested with the base models RNN, LSTM, GRU and meta-model -LGBM separately. The outcomes indicate that all algorithms exhibit better performance in identifying abnormalities. Even though the proposed framework presents a distinct enhancement compared to the other approaches. However, in the case of imbalanced data, accuracy is not the metric to be considered. Batchu et al. [31] state that F1-score is an essential metric to consider when the data contains dissimilarity. Therefore, we have analysed F1-score of two datasets. In the case of ToN_IoT dataset, F1-score is 88.96%, 88.5%, 93.48%, 90.75%, 83.59%, and 94.25% for DNN, RNN, LSTM, GRU, LGBM, and proposed stacking model. Similarly, for CIC-IoT23 dataset F1-Score is 93.37%, 89.35%, 90.69%, 83.59%, and 94.25% for DNN, RNN, LSTM, GRU, LGBM, and proposed stacking model, respectively. The outcomes showed that the F1-score of all models on the two datasets is not up to the mark. Fig. 7-8 show the ROC plot of ToN_IoT and CIC-IoT23 datasets. Fig. 7 shows that the AUC values of the LSTM and RNN are less when compared to other models, whereas the proposed stacking model has a high AUC value of 96.3%. Similarly, Fig. 8 shows that DNN has less AUC value of 75.1%, and the proposed model has attained an AUC of 88.3%. Hence, it can be concluded that with the imbalanced data, the proposed stacking model does not have a reasonable detection rate. Therefore, to improve the model performance, the class dissimilarity is handled by using SVM-SMOTE, as discussed in Section 3.3. As observed Table 6 and 7 the training time of proposed model is 3.2 and 3.9 seconds for two datasets respectively.

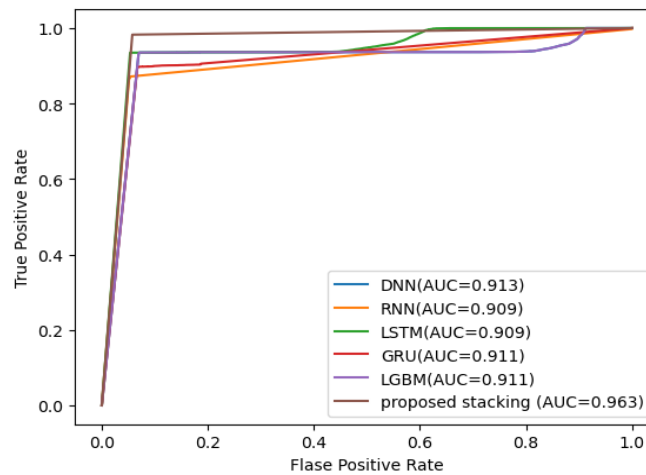


Fig. 7. ROC Plot for ToN_IoT imbalance dataset

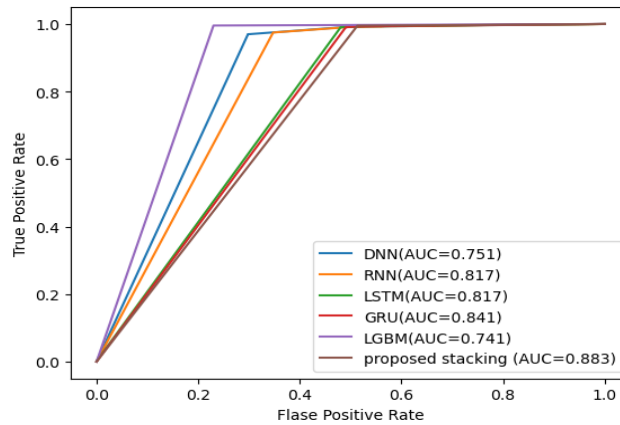


Fig. 8. ROC plot of CIC-IoT23 imbalance dataset.

Table 6. Outcomes of ToN_IoT imbalanced dataset

Model	Accuracy	Precision	Recall	F1-Score	FPR	Time (s)
DNN	91.98	88.27	98.66	88.96	0.11	2.4
RNN	91.89	90.44	86.64	88.5	0.09	2.2
LSTM	94.21	90.73	93.48	92.09	0.09	1.8
GRU	93.14	88.23	93.42	90.75	0.11	2.2
LGBM	88.16	83.48	83.69	83.59	0.41	1.6
Proposed Stacking Model	95.68	90.56	98.26	94.25	0.05	3.2

Table 7. Outcomes of CIC-IoT imbalanced dataset

Accuracy	Precision	Recall	F1-Score	FPR	Time (s)
95.52	94.08	93.38	93.73	0.59	2.2
92.44	85.56	90.15	89.35	0.09	2.1
93.13	88.01	93.54	90.69	0.12	3.0
93.2	88.37	93.28	93.76	0.16	2.9
97.05	94.8	99.55	97.11	0.15	0.9
97.58	99.75	96.41	98.06	0.11	3.9

5.2. Analysis of the proposed stacking model with balanced data.

To enhance the performance of the proposed stacking model, minority synthetic samples are generated by using SVM-SMOTE. The outcomes of the proposed stacking model and the individual models of the two balanced datasets are shown in Tables 8-9. The analysis shows that the performance of all models was improved in the case of two datasets when compared with imbalanced outcomes.

Table 8. Outcomes of ToN-IoT balanced dataset

Model	Accuracy	precision	Recall	F1-Score	FPR	Time (s)
DNN	97.26	94.91	99.36	97.27	0.05	2.7
RNN	98.25	96.81	99.58	98.28	0.31	2.4
LSTM	98.26	97.00	99.47	98.22	0.02	3.1
GRU	97.26	95.02	99.75	97.33	0.04	2.9
LGBM	98.88	98.49	99.28	98.88	0.32	2.6
Proposed Stacking Model	99.81	99.65	99.61	99.8	0.02	3.8

Table 9. Outcomes of CIC-IoT23 balanced dataset

Model	Accuracy	precision	Recall	F1-Score	FPR	Time/Sec
DNN	98.79	99.25	98.35	98.79	0.05	2.8
RNN	98.48	97.59	99.07	98.34	0.20	2.5
LSTM	98.74	98.99	99.02	99.35	0.08	3.6
GRU	99.06	98.94	99.09	98.96	0.09	3.1
LGBM	98.93	98.27	98.64	98.74	0.05	1.1
Proposed Stacking Model	99.78	99.94	99.83	99.89	0.04	4.8

From Table 8, we can observe that the base models, RNN, LSTM and GRU, have accuracies of 97.26%, 98.25%, 98.26% and 97.26%, whereas meta learner BTE-LGBM has obtained an accuracy of 98.88%. When the four DL models are stacked with meta learner BTE-LGBM, the performance is enhanced, and the accuracy of the proposed stacking model is 99.81%. This enhancement is due to an increase in the minority class samples. The recall of GRU is more when compared to the proposed stacking model by 0.14%. The FPR of LGBM and RNN are higher when compared to other models, and it is recorded as 0.32 and 0.31, respectively. The proposed stacking model has an FPR of 0.004%, indicating the model has a reasonable detection rate. Similar to Table 8, Table 9 shows the performance of the proposed stacking model on the CIC-IoT23 dataset. It can be observed that the proposed model performs better when compared to base models (DNN, RNN, LSTM and GRU) and meta-model (BTE-LGBM) with an accuracy of 99.78%. Moreover, the proposed stacking model has less FPR of 0.04, whereas RNN is 0.24%, which is more when compared to other models. Therefore, when the base models are stacked with the meta-model, the performance is improved over the two datasets.

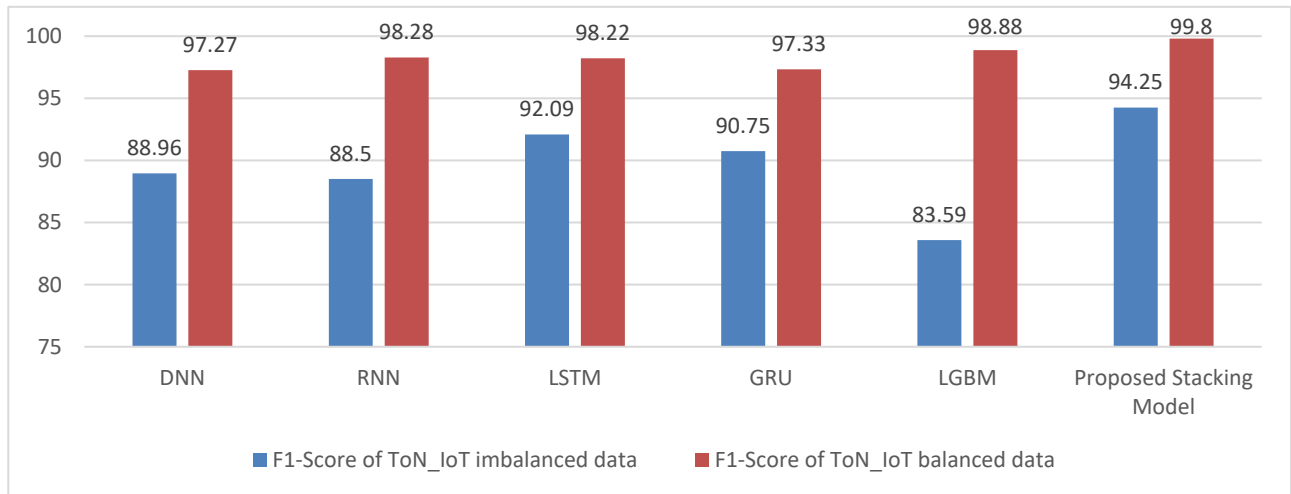


Fig. 9. Comparison of F1-Score of ToN_IoT dataset

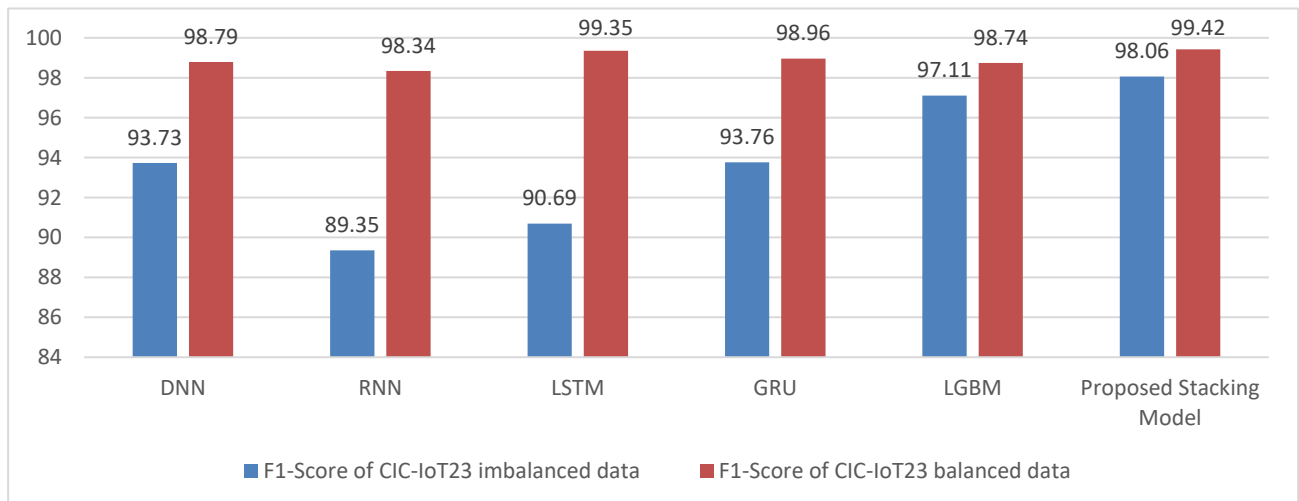


Fig. 10. Comparison of F1-Score of CIC-IoT23 dataset

Further, the F1-score metric is analysed. Fig. 9-10. show the comparison of the F1-score on imbalance and balanced datasets. It is observed that the F1-score was improved when the synthetic samples were generated for the minority class.

In the case of ToN_IoT imbalance data, F1-score is 88.96%, 88.5%, 92.09%, 90.75%, 83.59% and 92.25 for DNN, RNN, LSTM, GRU, BTE-LGBM, and proposed stacking model respectively. Whereas when the data is balanced, the F1-score was improved by 8.31% for DNN, 9.78% for RNN, 6.13% for LSTM, 6.58% for GRU and 15.29% for LGBM and 5.63% with the proposed stacking model. Similarly, for the CIC-IoT23 dataset, F1-score was improved by 5% on DNN, 8.99% for RNN, 8.66% for LSTM, 4.93% for GRU, 1.63% for LGBM and the proposed stacking model F1-score was improved by 1.36%. Fig. 11-12. show the ROC plots of two balanced datasets, and it is observed that the detection rate of all models was improved when the data was balanced. The proposed stacking model has a better AUC value of 99.9% on the ToN-IoT and 99.7% on CIC-IoT23 dataset, respectively. Finally, the computation time of two datasets are 3.2 and 4.8 seconds respectively. The stacking framework contains more parameters than any individual model so the training time is slightly more when compared to individual models.

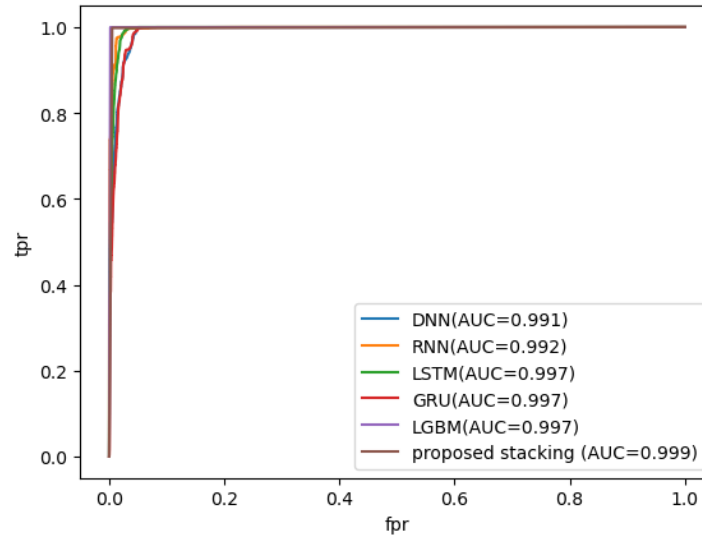


Fig. 11. ROC plot of balanced ToN_IoT dataset.

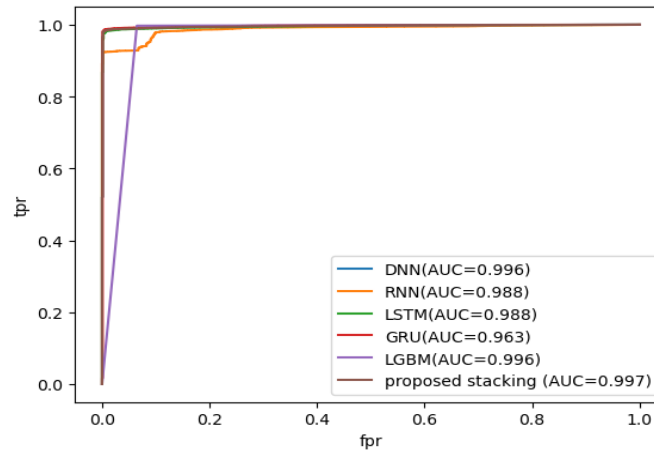


Fig. 12. ROC plot of balanced CIC-IoT23 dataset.

5.3. Comparative analysis

The performance comparison between the proposed stacking IDS system and existing methods using the ToN_IoT dataset is shown in Fig. 13-16. The proposed stacking model is compared with existing IDS models of ToN-IoT as discussed in literature such as SATIDS[1], MRMR-SATIDS[1], DIS-IoT[6], Ensemble learning[19], deepAK_IoT [23], and Dnnrann [15]. The analysis shows that the proposed stacking model has outperformed other models by an accuracy of 99.85%, precision of 99.65%, and F1-score of 99.8%. Regarding the detection rate, DIS-IoT [6] recorded a higher detection rate than the proposed approach, but our model performs better in the case of all other metrics. Moreover, our proposed stacking model is compared with existing stacking models of ToN_IoT datasets like DIS-IoT[6] and Ensemble learning [19]. Fig. 13 shows our proposed stacking model performs better when compared to DIS-IoT[6] and Ensemble learning [19] by an accuracy of 99.81%.

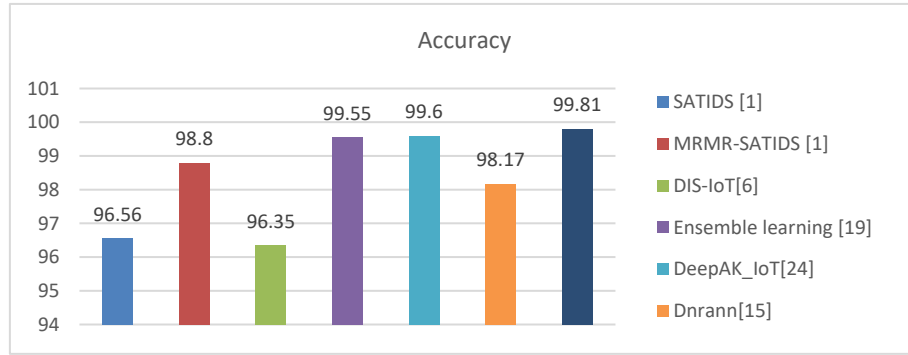


Fig. 13. Comparison of proposed model Accuracy with respect to existing approaches using the ToN_IoT dataset.

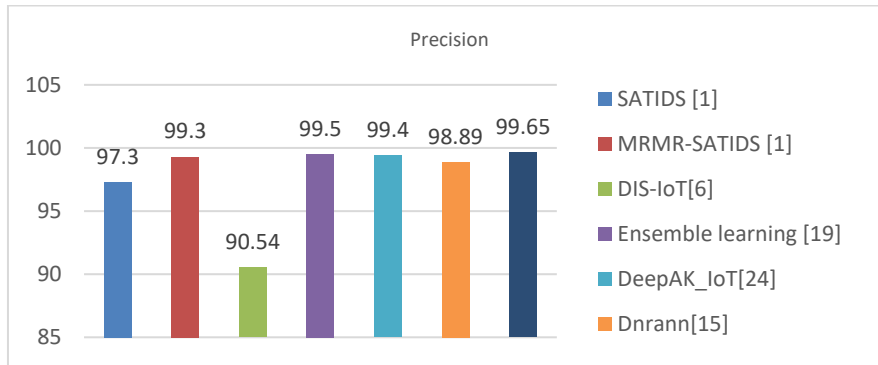


Fig. 14. Comparison of proposed model precision with respect to existing approaches using the ToN_IoT dataset.

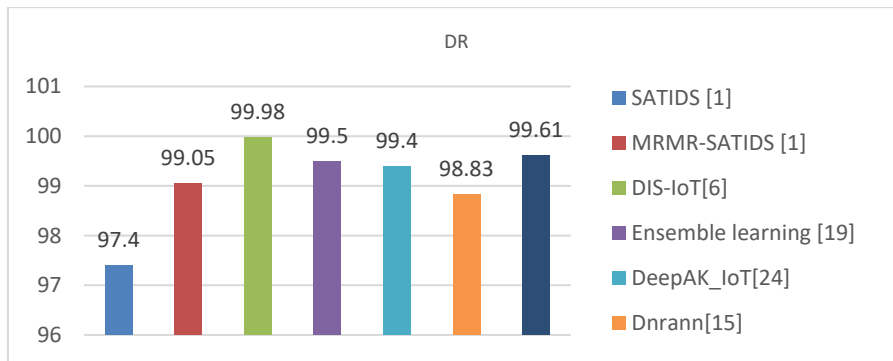


Fig. 15. Comparison of proposed model detection rate with respect to exiting approaches using the ToN_IoT dataset.

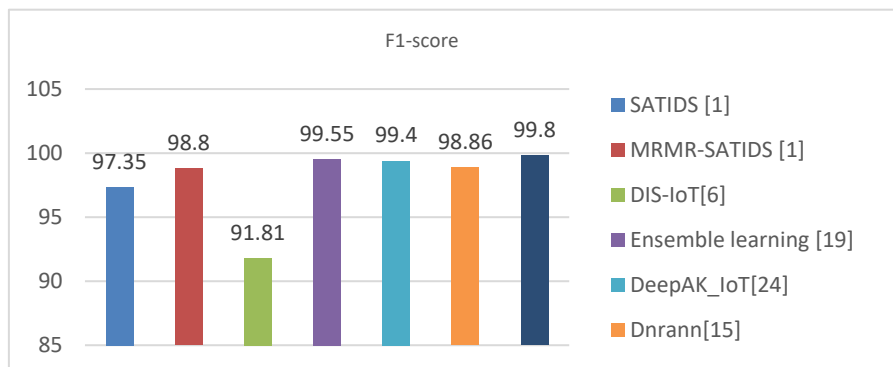


Fig. 16. Comparison of proposed model F1-score with respect to exiting approaches using the ToN_IoT dataset.

Table 10. shows the performance of our proposed approach over the existing methods of the CIC-IoT23 dataset. It is observed that the proposed stacking model outperforms the existing models. Carlos et al. [11] has developed the CIC-IoT23 dataset and tested various ML and DL models like LR, perceptron, RF, Adaboost, and DNN. Their results are shown in Table 10. To our knowledge, there is no stacking model on the CIC-IoT23 dataset as of now to compare with

the proposed stacking model. Further, to assess the performance of our proposed model, we have also evaluated SVM and K-nearest neighbours (KNN) models on the CIC-IoT23 dataset. Out of all the models, the perceptron model has attained a lesser accuracy of 98.17% compared to other models. The models like Adaboost[11], RF[11], and DNN[11] have accuracy above 99%, but other metrics like precision, detection rate, and F1-score are less when compared to our model. The precision rate of Adaboost[11], RF[11], and DNN[11] are lesser by 3.38%, 3.43%, and 5.19% whereas DR is lesser by 5.1%, 3.3%, 6.51%. Finally, F1-score is lesser by 4.2%, 3.37%, and 5.86%, respectively. This happened because the authors have not handled the imbalance issue in the data set, so the models are biased towards the majority class. As we tested KNN and SVM on balanced data, the bias issue was not reflected in outcomes. The accuracy of KNN and SVM are recorded as 99.16% and 98.42%. But our stacking model has performed better when compared to all other models by an accuracy of 99.78%, precision of 99.94, detection rate of 99.83 and F1-score of 99.89.

Table 10. Comparison of the proposed stacking model with existing approaches on CIC-IoT23 dataset

Model	Accuracy	Pe	DR	F1-score	Stacking model
LR[11]	98.9	86.31	89.04	87.26	NO
perceptron[11]	98.17	82.54	79.7	81.05	NO
Adaboost[11]	99.58	96.56	94.73	95.62	NO
RF[11]	99.68	96.51	96.53	96.52	NO
DNN[11]	99.44	94.75	93.32	94.03	NO
KNN	99.16	99.28	99.05	99.16	NO
SVM	98.42	97.75	98.74	98.24	NO
Proposed stacking model	99.78	99.94	99.83	99.89	Yes

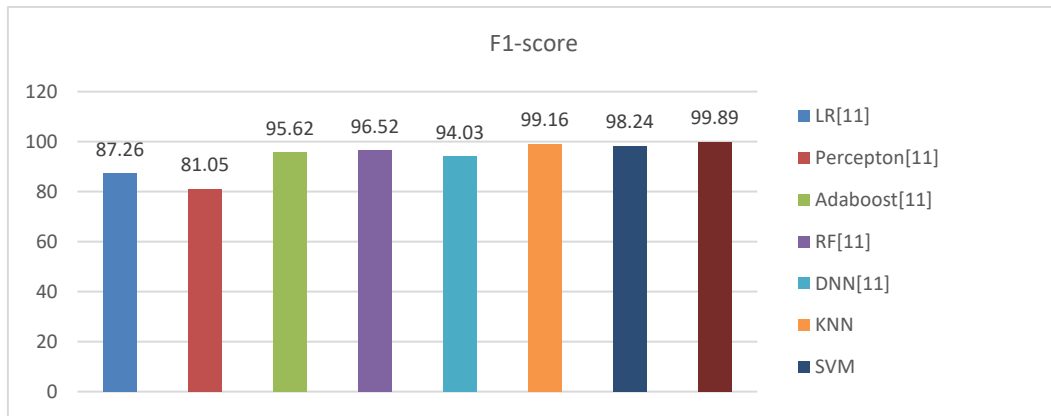


Fig. 17. Comparison of proposed model F1-score over the existing models of CIC-IoT23 dataset

In the case of imbalanced datasets, the F1 score is an important metric to be considered. Fig. 16-17. show the F1-score comparison of the proposed approach with respect to existing methods on the two datasets. In the case of the ToN_IoT and CIC-IoT23 datasets, the F1-score is more, and it is noted as 99.8% and 99.89%, respectively. A classification model that provides an optimal balance of precision and recall is more advantageous, which will try to maximize the F1-score, and hence, our proposed stacked model outperformed all other existing models in terms of F1-score on two datasets. Therefore, it can be concluded that the proposed stacking model exhibits superior performance compared to the conventional IDS models and is effective in detecting attacks in the IoT environment.

6. Conclusion

IoT security requires the detection and monitoring of network attacks in order to effectively manage and mitigate unauthorized traffic flows. The rising frequency of attacks emphasizes the need for developing efficient methods of identifying abnormalities in data. This study proposes a novel stacking model to detect attacks in IoT networks. Prior to being used with the proposed stacking model, the dataset is pre-processed to eliminate noise. Then, class dissimilarity is handled to reduce the bias issue in the datasets. Further, the stacking model is built by using four DL models (DNN, RNN, LSTM, and GRU) as base learners and a hyperparameter-tuned BTE-LGBM as a meta-model to benefit from the various classification standards that each sub-model provides. The proposed stacking model was tested on two recent realistic IoT data sets, namely ToN_IoT and CIC-IoT23 datasets. The experiment outcomes demonstrate that the proposed model has

outstanding results in terms of accuracy, detection rate, precision, and F1-Score with the less false positive rate on two datasets. Further research will explore the proposed stacking model with real IoT devices. The suggested stacking model has four DL models and one ML model, which requires more resources. Hence, conducting experiments with the proposed model in a topology structured around edge networking would be more suitable than deploying it directly on low-power Internet of Things (IoT) sensors.

All the Declarations and Statements

Author Contributions Statement

Seshu Bhavani Mallampati – Handled data acquisition, dataset preprocessing, and implementing the research model. Model Training, Validation, and Performance Evaluation, Formal Analysis, Visualization, and Statistical Analysis: Performed in-depth analysis of experimental, results, prepared performance charts, and ensured the statistical robustness of the evaluation, Writing.

Hari Seetha – Conceptualization, Methodology, and Supervision: Proposed research ideas, Constructed the overall, framework, and supervised project execution. Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Funding Declaration

Not applicable

Data Availability Statement

This study analyzed publicly available datasets.

Ethical Declarations

This study does not involve human participants or animals.

Acknowledgments

We sincerely thank the experts for their professional evaluation and valuable recommendations, which have Contributed to improving the quality of the experiment and the reliability of its results.

Declaration of Generative AI in Scholarly Writing

During the preparation of this manuscript, the authors reviewed the content carefully and take full responsibility for the accuracy, originality, and integrity of the final manuscript.

Abbreviations

The following abbreviations are used in this manuscript:

IoT- Internet of Things
DNN- Deep neural network
RNN- Recurrent Neural Network
LSTM- Long Short-Term Memory
GRU-Gated Recurrent Unit
IDS-Intrusion Detection System
DDoS-Distributed Denial of Service
ML- Machine Learning
DL- Deep Learning
SVM-Support Vector Machine
NN- Neural Network
LR-Logistic Regression
TON_IoT Telemetry Optimization and Network Internet of Things
BTE- Bayesian Optimization with Tree-structured Parzen Estimator
ANN-Artificial Neural Networks
GBDT-Gradient-Boosting Decision Tree

LR-Logistic Regression
 NB- Naive Bayes
 MLP- Multilayer Perceptron
 CNN-Convolution Neural Networks
 ADFA -Australian Defense Force Academy
 AC-Accuracy
 Pe- Precision (Pe)
 DR- Detection Rate/Recall
 FPR -False Positive Rate
 TP-True Positive
 FP- False Positive
 TN- True Negative
 FN- False Negative

APPENDIX A/B/C..., with appendix title
 Not Applicable

References

- [1] R. A. Elsayed, R. A. Hamada, M. I. Abdalla, and S. A. Elsaid, "Securing IoT and SDN systems using deep-learning based automatic intrusion detection," *Ain Shams Eng. J.*, vol. 14, no. 10, p. 102211, 2023, doi: 10.1016/j.asej.2023.102211.
- [2] Y. Cao, Z. Wang, H. Ding, J. Zhang, and B. Li, "An intrusion detection system based on stacked ensemble learning for IoT network," *Comput. Electr. Eng.*, vol. 110, no. July, p. 108836, 2023, doi: 10.1016/j.compeleceng.2023.108836.
- [3] S. Bhavani Mallampati and S. Hari, "Fusion of Feature Ranking Methods for an Effective Intrusion Detection System," *Comput. Mater. Contin.*, vol. 76, no. 2, pp. 1721–1744, 2023, doi: 10.32604/cmc.2023.040567.
- [4] S. B. Mallampati and H. Seetha, "A Review on Recent Approaches of Machine Learning , Deep Learning , and Explainable Artificial Intelligence in Intrusion Detection Systems," vol. 17, no. 1, pp. 29–54, 2023, doi: 10.30486/mjee.2023.1976657.1046.
- [5] B. Sharma, L. Sharma, C. Lal, and S. Roy, "Anomaly based network intrusion detection for IoT attacks using deep learning technique," *Comput. Electr. Eng.*, vol. 107, no. February, p. 108626, 2023, doi: 10.1016/j.compeleceng.2023.108626.
- [6] R. Lazzarini, H. Tianfield, and V. Charissis, "A stacking ensemble of deep learning models for IoT intrusion detection," *Knowledge-Based Syst.*, vol. 279, p. 110941, 2023, doi: 10.1016/j.knosys.2023.110941.
- [7] R. E. Schapire, "A brief introduction to boosting," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2, pp. 1401–1406, 1999.
- [8] S. A. Khanday, H. Fatima, and N. Rakesh, "Implementation of intrusion detection model for DDoS attacks in Lightweight IoT Networks," *Expert Syst. Appl.*, vol. 215, no. November 2022, 2023, doi: 10.1016/j.eswa.2022.119330.
- [9] A. Hossain and S. Islam, "Ensuring network security with a robust intrusion detection system using," *Array*, vol. 19, no. June, p. 100306, 2023, doi: 10.1016/j.array.2023.100306.
- [10] A. Thakkar and R. Lohiya, "Attack Classification of Imbalanced Intrusion Data for IoT network Using Ensemble Learning-based Deep Neural Network," *IEEE Internet Things J.*, pp. 1–8, 2023, doi: 10.1109/JIOT.2023.3244810.
- [11] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," *Sensors*, vol. 23, no. 13, 2023, doi: 10.3390/s23135941.
- [12] H. Zhou, L. Kang, H. Pan, G. Wei, and Y. Feng, "An intrusion detection approach based on incremental long short-term memory," *Int. J. Inf. Secur.*, vol. 22, no. 2, pp. 433–446, 2023, doi: 10.1007/s10207-022-00632-4.
- [13] H. S. & R. K. B. Seshu Bhavani Mallampati, "PCB-LGBM: A Hybrid Feature Selection by Pearson Correlation and Boruta-LGBM for Intrusion Detection Systems," in *International Conference on Computational Intelligence and Data Engineering*, 2022, pp. 523–533, [Online]. Available: https://doi.org/10.1007/978-981-99-0609-3_37.
- [14] S. M. Kasongo, "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework," *Comput. Commun.*, vol. 199, pp. 113–125, Feb. 2023, doi: 10.1016/j.comcom.2022.12.010.
- [15] S. Latif et al., "Intrusion Detection Framework for the Internet of Things Using a Dense Random Neural Network," *IEEE Trans. Ind. Informatics*, vol. 18, no. 9, pp. 6435–6444, 2022, doi: 10.1109/TII.2021.3130248.
- [16] C. A. de Souza, C. B. Westphall, and R. B. Machado, "Two-step ensemble approach for intrusion detection and identification in IoT and fog computing environments," *Comput. Electr. Eng.*, vol. 98, no. December 2021, p. 107694, 2022, doi: 10.1016/j.compeleceng.2022.107694.
- [17] N. Thockchom and M. Marjit, "A novel ensemble learning-based model for network intrusion detection," *Complex Intell. Syst.*, 2023, doi: 10.1007/s40747-023-01013-7.
- [18] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, "A New Ensemble-Based Intrusion Detection System for Internet of Things," *Arab. J. Sci. Eng.*, 2021, doi: 10.1007/s13369-021-06086-5.
- [19] P. Kumar, G. P. Gupta, and R. Tripathi, "An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks," *Comput. Commun.*, vol. 166, no. April 2020, pp. 110–124, 2021, doi: 10.1016/j.comcom.2020.12.003.
- [20] A. Awajan, "A Novel Deep Learning-Based Intrusion Detection System for IoT Networks," *Computers*, vol. 12, no. 2, 2023, doi: 10.3390/computers12020034.

- [21] H. Xu, L. Sun, G. Fan, W. Li, and G. Kuang, "A Hierarchical Intrusion Detection Model Combining Multiple Deep Learning Models With Attention Mechanism," *IEEE Access*, vol. 11, no. July, pp. 1–1, 2023, doi: 10.1109/access.2023.3290613.
- [22] S. Aktar and A. Yasin Nur, "Towards DDoS attack detection using deep learning approach," *Comput. Secur.*, vol. 129, p. 103251, 2023, doi: 10.1016/j.cose.2023.103251.
- [23] W. Ding, M. Abdel-Basset, and R. Mohamed, "DeepAK-IoT: An effective deep learning model for cyberattack detection in IoT networks," *Inf. Sci. (Ny)*, vol. 634, no. January, pp. 157–171, 2023, doi: 10.1016/j.ins.2023.03.052.
- [24] S. I. Popoola, B. Adebisi, M. Hammoudeh, H. Gacanin, and G. Gui, "Stacked recurrent neural network for botnet detection in smart homes," *Comput. Electr. Eng.*, vol. 92, no. June, 2021, doi: 10.1016/j.compeleceng.2021.107039.
- [25] I. Syarif, E. Zaluska, A. Prugel-Bennett, and G. Wills, "Application of bagging, boosting and stacking to intrusion detection," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7376 LNAI, pp. 593–602, 2012, doi: 10.1007/978-3-642-31537-4_46.
- [26] M. Ibrahim and R. Elhafiz, "Modeling an intrusion detection using recurrent neural networks," *J. Eng. Res.*, vol. 11, no. 1, p. 100013, 2023, doi: 10.1016/j.jer.2023.100013.
- [27] P. Sun et al., "DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system," *Secur. Commun. Networks*, vol. 2020, 2020, doi: 10.1155/2020/8890306.
- [28] T. A. Tang, D. McLernon, L. Mhamdi, S. A. R. Zaidi, and M. Ghogho, "Intrusion detection in sdn-based networks: Deep recurrent neural network approach," *Adv. Sci. Technol. Secur. Appl.*, no. NetSoft, pp. 175–195, 2019, doi: 10.1007/978-3-030-13057-2_8.
- [29] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 3147–3155, 2017.
- [30] S. Watanabe, "Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance," pp. 1–74, 2023, [Online]. Available: <http://arxiv.org/abs/2304.11127>.
- [31] R. K. Batchu and H. Seetha, "A Hybrid Detection System for DDoS Attacks Based on Deep Sparse Autoencoder and Light Gradient Boost Machine," *J. Inf. Knowl. Manag.*, vol. 22, no. 1, 2022, doi: 10.1142/S021964922250071X.

Authors' Profiles



Dr. Seshu Bhavani Mallampati obtained her Master's degree from JNTUH and her Ph.D. from the School of Computer Science and Engineering, VIT-AP University, Amaravathi, India. She worked on Cyber Security during her Ph.D. She is a dedicated academician and accomplished researcher with extensive experience in teaching, research, and academic administration. With over 16 years of experience in the field of Computer Science and Engineering, she has made significant contributions to both academic excellence and institutional development. She has guided several undergraduate and postgraduate students in their academic projects and research work, helping them achieve academic and professional success. She has also organized and participated in various workshops, seminars, and faculty development programs, promoting continuous learning and innovation. She has published several papers in international journals and has presented at various international conferences. She has served as a reviewer for international journals and conferences. She is currently working as Associate Professor at Bhoj Reddy Engineering College for Women, Hyderabad, India.



Dr. Hari Seetha obtained her Master's degree from National Institute of Technology (formerly R. E. C.) Warangal and her Ph.D. from the School of Computer Science and Engineering, VIT University, Vellore, India. She worked on Large Data Classification during her Ph.D. She has research interests in the fields of pattern recognition, data mining, text mining, soft computing and machine learning. She received Best paper award for the paper entitled "On improving the generalization of SVM Classifier" in Fifth International Conference on Information Processing held at Bangalore. She has published several research papers in national and international journals of repute. She has been one of the editors for the edited volume on "Modern Technologies for Bigdata Classification and Clustering" published by IGI Global in 2017. She is a member of editorial board for various International Journals. She guided four Ph.D. students and several scholars are working under her guidance. She is listed in the 2014 edition of Who's who in the world published by Marquis Who's Who, as the biographical reference representing the world's most accomplished individuals. She was a co-investigator to a major research project sponsored by the Department of Science and Technology, Government of India. She served as a Division Chair for Software Systems division and Program Chair for B.Tech (CSE) programme, in the School of Computer Science and Engineering at VIT University, Vellore, India. She was Assistant Director (Ranking and Accreditation) at VIT University, Vellore, Tamil Nadu, India. She also served as Dean School of Computer Science and Engineering at VIT-AP University, Near Vijayawada, Andhra Pradesh, India. She is currently working as a Professor at VIT-AP University.

How to cite this paper

Seshu Bhavani Mallampati, Hari Seetha, "Intrusion Detection System using Stacking of Deep Learning Models with Bte-Lgbm for IoT Networks", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.16, No.4, pp. 44-64, 2026.
DOI:10.5815/ijwmt.2026.04.04