Journal of Intelligent & Fuzzy Systems 45 (2023) 1775–1790
DOI:10.3233/JIFS-223246
IOS Press

HSDL-based intelligent threat detection framework for IoT network

D. Santhadevi* and B. Janet
National Institute of Technology, Tiruchirappalli, India

Abstract. Many Internet of Things (IoT) devices are susceptible to cyber-attacks. Attackers can exploit these flaws using the internet and remote access. An efficient Intelligent threat detection framework is proposed for IoT networks. This paper considers four key layout ideas while building a deep learning-based intelligent threat detection system at the edge of the IoT. Based on these concepts, the Hybrid Stacked Deep Learning (HSDL) model is presented. Raw IoT traffic data is pre-processed with spark. Deep Vectorized Convolution Neural Network (VCNN) and Stacked Long Short Term Memory Network build the classification model (SLSTM). VCNN is used for extracting meaningful features of network traffic data, and SLSTM is used for classification and prevents the DL model from overfitting. Three benchmark datasets (NBaIoT-balanced, UNSW-NB15 & UNSW_BOT_IoT- imbalanced) are used to test the proposed hybrid technique. The results are compared with state-of-the-art models.

Keywords: Hybrid stacked deep learning, stacked LSTM, Vectorized Convolutional Neural Network, IoT-network security, edge computing

1. Introduction

The Internet of Things (IoT) network is a collection of intelligent items connected by the worldwide internet, such as sensors, digital appliances, smartphones, embedded electronic gadgets, integrated automobiles, healthcare equipment, computers; it is rapidly expanding and becoming an integral part of our daily lives. There are a variety of applications that use this technology: smart health care systems, smart parking, intelligent traffic control, smart agricultural science, and intelligent transportation.

IoT makes human life easier, but there is a concern about privacy and security [1]. IoT networks are turn into a popular target for hackers. According to research from Palo Alto Networks' Unit 42, 98 percent of IoT device communication is unencrypted, and 40 percent of attacks target IoT device flaws [2]. Adversaries might then utilize the susceptible devices to connect with IoT botnet and launch advanced, large-scale assaults. For example, Mirai [3], the original IoT botnet, was competent to infect unprotected surveillance (CCTV) cameras utilizing default credentials to execute a DDoS assault at DNS servers in October 2016. In some regions of the United States, internet connectivity was disrupted because of this assault. Mozi, an IoT botnet identified in April 2020, was proven capable of performing multiple DDoS assaults [4, 5]. Most preventative approaches fail somewhere at the level of edge. Edge security is required; intrusion detection systems are frequently employed to identify malicious network traffic [6, 7].

As the number of assaults on IoT networks and systems grows, they become increasingly sophisticated and undetected because IoT network creates a huge volume of heterogeneous and time-series data. When such data is subjected to big data analytics, it is feasible to uncover previously unknown patterns, uncover hidden relationships, and obtain new insights [8]. The

*Corresponding author. D. Santhadevi, Research Scholar, National Institute of Technology, Tiruchirappalli, India. E-mail: 405116002@nitt.edu.

Intelligent threat detection system should be evolved to deal with security issues.

Many classic machine learning (ML) approaches utilized for intrusion detection [9] have shallow architectures and are unsuitable for detecting malicious activity in huge data context. They have trouble recognizing unexpected assaults, providing real-time solutions, and managing the usual intrusion in huge data sets.

Deep learning (DL) techniques have been used to manage massive data in many applications in recent years [10, 11]. Deep Belief Networks are primarily applied for pattern recognition, and it trains quicker than other DL approaches. CNN is commonly employed in image-processing applications that has more discriminative capacity.

DL algorithms perform well when handling highly heterogeneous data [12–14]. They may indeed assess complicated and big data to gain knowledge, identify relationships within data. DL Algorithms use prior attack patterns to discover new and previously unknown attack patterns [15–17]. IoT devices have limited storage and compute capabilities due to that heavy activity such as big-data analysis, and the development of learning models must be offloaded to edge and cloud servers [18–20]. As a result, computational offloading [21] could assist in minimizing task execution delays and improve energy efficiency in battery-powered and portable IoT devices, but that may create specific security issues [22].

Further efforts are required to enhance the practice of malicious activity detection in the IoT network. The deep learning based intelligent threat detection framework is proposed in this paper. The key contributions are as following:

- Established with four basic design concepts for constructing HSDL-based ITDS for IoT network, including distributed processing, input vector selection, over fitting management, model optimization, and testing on real-time IoT datasets.
- Examined state-of-the-art approaches, discovered gaps, and analyzed the key conflicts for our work based on the stated core design principles.
- An efficient pre-processing with the spark gives distributed environment to process data.
- Combination of VCNN and Stacked LSTM is used for intelligent threat detection systems on the Internet of Things.
- The deepVCNN collected significant characteristics from network data flow.
- The SLSTM is utilized here for maintaining long-term interdependence with retrieved variables.
- The drop-connect regularisation approach is used here in the hidden layer of the SLSTM to avoid overfitting.
- RMSProp is used for model optimization, and the hyper-parameters of the model are optimized by observation and experimentation. For malware classification, the suggested model is tested using the publicly available three IoT network traffic datasets.

The article's structure proceeds as follows: Section II reviewed the related work. In Section III, the proposed model is described. The experimental setup, dataset selection, and experimental results are discussed in Section IV. Section V presents the comparison of Results with the state-of-the-art model. Section VI covers the conclusion and future work of this paper.

2. Related works

In the digital world, humans are increasingly reliant on technologies. Every public and private sector uses smart technology, consisting of sensors, sensors integrated with products, and actuators. These sensor-based devices are networked and communicate massive amounts of data every second. These devices provide low-hanging fruit for attackers to access any network and steal data remotely. As a result, there is a need for intelligent systems smart enough to make judgments to secure information from intrusion. Many applications make use of ML and DL artificial intelligence algorithms.

This section includes recent research on IoT security threats and mitigation strategies. Deep Learning methodologies have surpassed standard machine learning methods in every underlying technology. McDermott CD [23] et al. developed an RNN-based bidirectional LSTM for deep packet inspection threat detection. Their investigation was generated on their dataset that was primarily for detecting Mirai attacks. The bidirectional model lengthens processing time yet provides a superior progressive model at points of time. Based on network flow, they created a labelled dataset for botnet identification. Jiyeon Kim [24] et al. developed a device-level botnet assault detection system using ML and DL algorithms. Sajad Homayoun [25] et al. discovered a correlation between autoencoder characteristics and the CNN model for network botnet traffic. Hammoudeh M [26] et al. suggested a signature-based fraudulent traffic detection approach in a smart home using IoT network to identify abnormal behavior. They employed passive network sniffing methods on a cloud application. Their approach was limited to detecting malicious behaviour in DNS and Telnet traffic.

H. Haddad Pajouh [27] et al. employed an RNN technique to find malware in the operation execution codes of ARM-based IoT apps (Opcodes). For IoT systems, A.A. Diro [28] et al. used a distributed attack detection approach based on DL. DL significantly increases classification accuracy for a challenging problem without using a feature selection approach. DL has the capacity to self-learn, provides excellent accuracy, works with large amounts of data, and boosts processing speed with GPU processors. It is also appropriate for networks with limited resources. Deep autoencoders were utilized by Yeir Meidan [29] et al. to identify botnet attacks at the network level. Deep encoders have the potential to learn complicated functions. The authors took a sample of benign IoT traffic to detect malicious behavior. If the benign snapshot is not built, they discover it to be a sign of a botnet. There were many studies done on host-based [29, 30] threat detection. When it comes to IoT devices, many have restricted access (for example, wearables) and lack the processing capability to execute complicated computations. The effectiveness of host-based threat detection is not appropriate for all IoT devices. In all networks, including IoT, a network-based threat detection technique is applied [31, 32]. Honey pots are used to monitor malware behaviour on a network. The key purposes of honey pots are to collect malware, identify their behaviours, characterize them, and track them. It is used for malware signature extraction and device emulation. It aids in understanding malware behaviour rather than forecasting botnets. A network-based [32] anomaly detection system continually analyses network behaviour by leveraging signature patterns from approaches (honeypot, sandbox) or a hybrid approach of Command & Control Server communication, network traffic, and data mining. D.H. Summerville et al. [33] employed network traffic flow data for anomaly-based malware detection which is suited for detecting compromised devices known as bot or botnet in all types of environments, including IoT.

The cloud [34] is utilized in various applications to store and analyze large amounts of data rapidly [35]. Because the cloud is centralized, when IoT devices connect to this cloud server, the quantity of data transfer is measured in Tbps (Terabits per second), resulting in massive traffic, high bandwidth use, and excessive latency. IoT devices transmit data continuously; however, uploading all information to the cloud is not essential, but periodic updates are required. At the same time, designers do not want any loss of essential data [36, 37]. To address this issue, edge and edge cloud enable that data to be processed close to the device and only essential data to be sent to the cloud. Edge computing reduces network traffic, bandwidth utilization, latency, and boosts availability [38]. Mollah MB et al. [39] suggested an effective data exchange and searching method for IoT employing edge with cloud assistance. Using edge computing, the authors addressed the significant challenges of data leakage, modification, integrity, and illegal access. Data sharing at the edge reduces the processing load on smart devices while assuring data integrity [40].

3. Materials and methods

3.1. Proposed intelligent threat detection framework

The Internet of Things network is described as several smart sensor embedded gadgets mounted in numerous locations. As a result, the intelligent threat detection system should be able to handle the malicious traffic created by these gadgets to deal with a rapid reaction in a robust way. In this condition, the central abnormality finding system behaves poorly in accurate detection. The proposed intelligent threat detection framework is planned to reduce the burden of centralized system processing by distributing the processing load at the end nodes. The edge computing process reduces the network traffic and latency.

Figure 1 represents the working functionality of the proposed framework. It comprises three parts: IoT device (things), edge service, and cloud storage. The IoT gadgets are interconnected through the edge gateway. Some of these devices have processor and memory constraints, and they cannot handle security events. The edge layer contains spark streaming process, network traffic monitoring devices that monitor malicious activity by the intelligence.

The edge layer is accountable for gathering data, processing, and securing the data. It will reduce the computational overhead on IoT devices. This layer

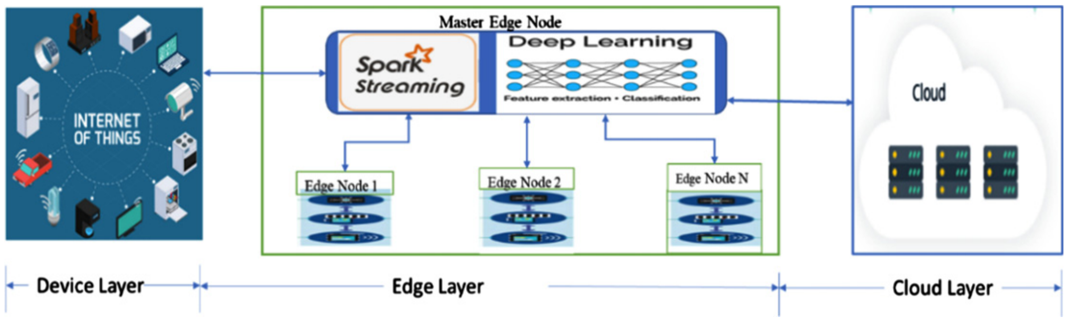


Fig. 1. Diagram of proposed intelligent threat detection framework.

is further responsible for improving service quality when data is sent to the cloud and lowering the computational overhead of IoT devices at the cloud. This layer is also in-charge of delivering critical and secure data to the cloud for further processing. The proposed framework detects anomalous behaviour in the IoT devices using the master edge node, and its detailed functionality is explained in the upcoming subsection. The Artificial Intelligence of Deep Learning methodology is used for training the proposed framework. The model is evaluated with three benchmark datasets created in the real-time network traffic at the IoT environment.

3.2. Methodology

3.2.1. Convolutional Neural Network (CNN)

CNN is a deep learning model used to learn high-level patterns in image processing. CNN's building blocks are made up of three layers: convolution with activation function ReLU (Rectified Linear Unit), max pooling, and completely connected [41, 42]. The first two layers will extract features, while the third layer will map the extracted features with output for classification. There will be multiple kernels in a convolutional layer. Each kernel works as a feature detector. The outcome of the convolution is frequently referred to as a feature map. Feature map displays where the kernel occurred in the signal. A collection of feature maps is the result of a set of kernels. This is a one-dimensional multichannel signal for one-dimensional data. The kernels in a layer are often a tiny exponent of two (i.e., 2,4,8 or 16). The convolution layer's mathematical action is a specialized linear operation. In this paper, a one-dimensional vectorized CNN is employed for malware detection. The performance of a model under specific kernels and weights has been determined with a weight vector via forward and backpropagation on a training sample, and learnable parameters, i.e., kernels and weights, are updated according to the loss value via the RMSprop optimization algorithm, which is an exponential moving average of the gradient.

The following equations are used for updating the weights and bias for each layer.

$$y_i = \sum_{c=0}^{n_c-1} \sum_{k=-p}^{p} x_{c,j-k} w_{c,k} \; Convolution \tag{1}$$

$$\frac{\partial L}{\partial x_{c,i}} = \sum_{k=-p}^{p} \frac{\partial L}{\partial y_{i+k}} w_{c,k} \; input\ gradient \tag{2}$$

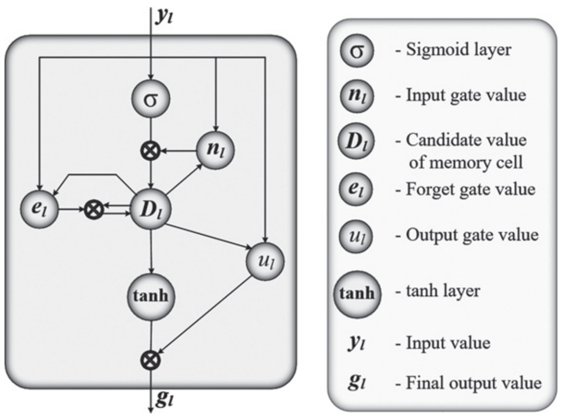
$$\frac{\partial L}{\partial w_{c,k}} = \sum_{j=0}^{m-1} \frac{\partial L}{\partial y_j} x_{c,j-k} \; parameter\ gradient \tag{3}$$

$$w_{t+1} = w_t - \propto V_t - \varepsilon \frac{\partial L}{\partial w}. \text{ RMSProp Optimization} \tag{4}$$

$$V_t = \beta V_{t+1} + 1 - \beta \left(\frac{\partial L}{\partial w}\right)^2 \tag{5}$$

$\frac{\partial L}{\partial w}$ - Gradient Component
$\propto$ - Learning rate
β - 0.9 (by default)
V_t - Initialized to zero
t - Time step
$W_t,\ W_{t-1}$ - Weight with respect to time t and $t-1$
ε - 10^{-6}

Where x, y are input and output, c- channel index, m-total number of instances, k- kernel size, p- half the kernel length, respectively. The Equation (1) is used to find the range of convolution, which is from minus infinity to plus infinity. The only non-zero values occur when the non-zero sections of the input value and the kernel overlap. Equation (3) determines



y_l
σ
n_l
e_l
D_l
u_l
tanh
g_l
σ - Sigmoid layer
n_l - Input gate value
D_l - Candidate value of memory cell
e_l - Forget gate value
u_l - Output gate value
tanh - tanh layer
y_l - Input value
g_l - Final output value

Fig. 2. The architecture of traditional Long Short-Term Memory (LSTM).

the partial derivative of the loss function for each of the parameters, i.e., each of the kernel element values. The input gradient is calculated using Equation (2) for each output. The weights are updated using Equations (4) & (5).

3.2.2. Long Short-Term Memory (LSTM)

In present days, the LSTM is utilized in every sequential modelling task, such as detecting motion, processing natural languages, and recognizinspeech. Figure 2 shows the block diagram of the traditional LSTM.

The memory cell and three multiplicative gating units such as input gate, output gate, and a forget gate are placed in the LSTM block. For the cells, continuous operations are provided by the recurrent integration among the cells and each gate. The cell controls the transmission of state values towards arbitrary time intervals. The writing, reading, and resetting of the operations for the cell is performed by every gate placed in the block [43].

The formula to calculate the LSTM block is given below. In that formula, when the input gate allows the input value to the block, the input value is protected in the state of the cell.

$$n_l = \sigma\left(X_n y_l + S_n g_{l-1} + a_n\right) \tag{6}$$

$$\tilde{D}_l = \text{ReLU}\left(X_D y_l + S_D g_{l-1} + a_D\right) \tag{7}$$

Where,

n_l => input value, σ => activation function

$\tilde{D}_l$ => candidate value of memory cells,

l => time steps, g_l => output value,

X, S, a => input vector, weight matrices, and bias, respectively.

The forget gate is managing the weight of the state unit, and the following equation is utilized to calculate the forget gate value,

$$e_l = \sigma(X_e y_l + S_e g_{l-1} + a_e) \tag{8}$$

After this calculation, the equation to calculate the new state of the memory cell is updated as a given blow,

$$D_l = n_l \times \tilde{D}_l + e_l \times D_{l-1} \tag{9}$$

With the help of the above calculation, the following equion is utilized to calculate the output value of the gate,

$$u_l = \sigma\left(X_u y_l + S_u g_{l-1} + P_u D_l + a_u\right) \tag{10}$$

And at last, the below-mentioned formula is used to find the final output value of cells,

$$g_l = u_l \times \text{Softmax}\left(D_l\right) \tag{11}$$

The output gate blocks the cell's output, and all gates use the sigmoid function for nonlinearity. The state unit also serves as an additional unit for the other gating units. They finally know that the long-term dependencies issue can be solved with minimal cost of computations with LSTM architecture.

3.2.3. Stacked LSTM

Graves pioneered Deep LSTM for voice detection. Layered LSTM is employed here to get depth in space, like how feedforward layers are layered in CNN. A stacked LSTM layer is consisting of multiple hidden LSTM layers, each with its own memory cell. Instead of a single output, an LSTM layer gives a sequence of output to the following hidden LSTM layer. Stacked LSTM has been shown to be stable for problems involving sequence prediction tasks that are difficult for other models. Network flow data is also in the form of a sequence of packets, which may be successfully handled using stacked LSTM for in-depth packet analysis.

3.3. The proposed deep learning model for intelligent threat detection

3.3.1. Data collection – IoT devices

The network traffic data collection is represented as $\mathrm{T_d} = [\mathrm{x_1},\ \mathrm{x_2}, \mathrm{x_3} \ldots \mathrm{x_n},\ \mathrm{L_c}]$, where $\mathrm{x_{1-n}}$ is the input features, and L is the label, c is the number of classes in the label. The network traffic dataset is represented

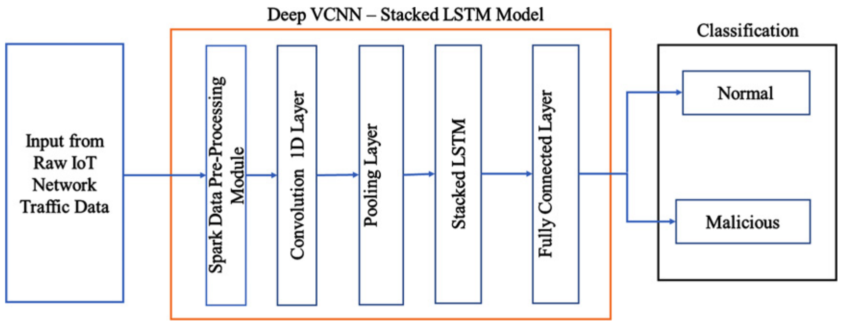

Deep VCNN – Stacked LSTM Model
Input from Raw IoT Network Traffic Data
Spark Data Pre-Processing Module
Convolution 1D Layer
Pooling Layer
Stacked LSTM
Fully Connected Layer
Classification
Normal
Malicious

Fig. 3. Block diagram of proposed hybrid stacked deep learning model.

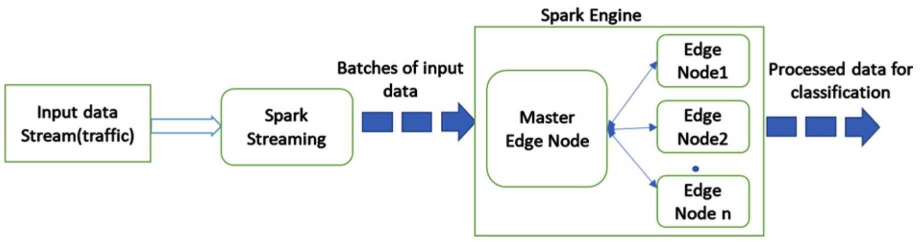


Fig. 4. Diagram of pre-processing steps used in spark.

in two-dimensional matrix T_{DS} using Equation (12).

$$T_{DS} = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_k^1 \\ x_1^2 & x_2^2 & \dots & x_k^2 \\ \vdots & \vdots & & \vdots \\ x_1^n & x_2^n & \dots & x_k^n \end{bmatrix} \tag{12}$$

3.3.2. *Spark – data pre-processing module*

Spark is an open-source big-data processing engine. It does the job in parallel with the help of the master node and worker nodes. It provides faster computing speed on a huge volume of data. In this work, PySpark version 3 is used in the google colab to process the IoT traffic data. The process of scaling and normalization is done in parallel with spark. The traffic data's normalization process is performed using the Equation (13), is mentioned below:

$$f_k^n\,(new) = \frac{x_k^n - \min\left(x_k^n\right)}{\max\left(x_k^n\right) - \min\left(x_k^n\right)} \tag{13}$$

Where the range of normalization is [0,1]. x_k^n is the raw traffic data; after the normalization input feature of the dataset is symbolized in Equation (14), the input features for the proposed hybrid stacked deep learning model.

$$NT_{DS} = \begin{bmatrix} f_1^1 & f_2^1 & \dots & f_k^1 \\ f_1^2 & f_2^2 & \dots & f_k^2 \\ \vdots & \vdots & & \vdots \\ f_1^n & f_2^n & \dots & f_k^n \end{bmatrix}. \tag{14}$$

3.3.3. *HSDL model*

The structure of the proposed model is based on two levels: Deep VCNN and SLSTM, shown in Fig. 3. Deep VCNN is constructed with three layers of CNN which are used for feature extraction. One Max-pooling layer is used for dimensionality reduction of the input shape, followed by a stacked LSTM layer, which drops some input features to avoid overfitting. The output of the stacked LSTM layer passed to a fully connected layer to find the malware classification on IoT traffic. A fully connected layer calculated the loss values for the output label. The number of nodes in the fully connected and output layers is equivalent to the number of classes in the dataset.

Deep VCNN performs the convolution operation with zero padding to find the new input features. For zero padding, zeros are added at the righand left end of the data. The kernel size and filters are fixed to find the most relevant features. The activation function used in hidden layers is ReLU (Rectified Linear Unit) that is calculated using the Equations (15).

$$f(y) = \max(0, f) \tag{15}$$

The pooling layer is used to perform the downsampling by using a pool size of 2 and stride size of 3. The output of the deep VCNN is a reduced form of input vectors, given as input to the SLSTM. SLSTM has three layers of LSTM; it trains the pre-trained input vectors to find the network anomaly in the given input using the Equations (6–11). The learning rate of the model is fixed as 0.005 after many trials. It is one of the tunning parameters for optimizing the DL algorithms. Further improving the model's performance, hyperparameters should be tuned.

The RMSProp optimizer is used in this model to lower the cost function (loss), which aids in reaching the global minima in the shortest period. However, cost functions are not always convex, so it will not converge to global minima; instead settles in local minima. The RMSProp is used to attain global minima by varying the learning rate and momentum to address this convergence problem. Equations (4 & 5) used for calculating RMSProp. The binary cross entropy's loss function is calculated using the following Equation (16).

$$Loss = -\frac{1}{K}\sum_{i=1}^{K}\sum_{j=1}^{M} Y_{ij}\log(P_{ij}). \tag{16}$$

Where K is the number of records in traffic data, Y_{ij} is the true probability distribution, $\log(P_{ij})$ is the predicted probability distribution. Binary cross-entropy is used to calculate the loss value in the network, and its output is a probability bween 0 and 1. It is worth noting that binary cross-entropy loss grows when the anticipated valudeviates from the actual class label. The learning process continues still loss reaches the defined threshold. The learning model is constructed at the master edge node to detect anomalous behaviour in the network.

3.3.4. Edge-based distributed learning

It consists of one master edge node and worker edge nodes for distributed learning. The edge nodes are used to handle the data collected from various IoT devices. The load is uniformly distributed across the existing worker edge nodes in the proposed method. The distributed learning process solves scalability issues. Once the training process is completed then the results are recorded in the master edge node and used to identify abnormalities in elusive IoT traffic flow.

3.3.5. Botnet detection in IoT traffic

"Robot Network" refers to a botnet, which is made up of malware-infected devices that are managed from a remote location by a botmaster. These compromised devices were used to carry out a variety of exploits and attacks. The first IoT-based botnet assault, known as the Mirai attack, occurred in 2016, in which around 1 million IoT devices were transformed into bots that generated 1Tbps of network traffic. The master edge node detects this sort of irregularity in IoT traffic. The features are efficiently obtained using the deep VCNN by aggregating the IoT traffic data. The SLSTM is used to differentiate between benign and malicious IoT device communications. The SoftMax function is used to detect the flow of traffic categorization. The binary classification approach uses just two network nodes in the SLSTM's output layer, whereas the multiclass classification strategy uses a number of nodes according to the number of class labels in the IoT traffic dataset. Algorithm 1 is used in the IoT network traffic to identify unexpected behaviour. It examines IoT devices traffic to determine if it is benign or malicious.

```
Algorithm 1: Proposed Traffic Flow Based Intelligent Threat Detection
Input: IoT network traffic data
Output: Normal/Malicious

START:
Step 1: Data Pre-processing
    For all traffic data spark streaming do
        For all batch data processing (parallel) do
            Normalization (13)
        End for
    End for
Step 2: Feature Selection
    For each VCNN do
```

```
        Calculate Convolution using (1)
        Calculate Gradient using ((2) & (3))
    End For
    Feature Selection using Max Pooling
Step 3: Training

For SLSTM do
    For each LSTM do
        Update weight and bias using (9)
        Calculate ReLU Activation (6)
        Drop weights
    End For
Calculate SoftMax Activation (11)
Optimize hyper parameter using (4)
Build Model
End For
Step 4: Testing
    For all traffic testing data do
        Test Model – Compute Class
        Evaluate Model – Metrics (Accuracy, F1 Score, Precision, Recall)
    End For
End
```

4. Result analysis with performance metrics

This section presents the evaluation of the proposed framework. It is evaluated using three benchmark datasets (NBaIoT, UNSW_NB15, UNSW_BOT_IoT). These datasets are generated from the different IoT network environments. The IoT network traffic data were collected from different smart devices used here to build the deep learning model based on anomaly detection for detecting malicious activity in the IoT network.

4.1. Dataset description

The dataset NBaIoT [44, 45] was developed by the University of California, Irvine, School of Information and Computer Science. This dataset was created from nine viable IoT devices in a real-time environment. Mirai and BASHLITE (Botnets) viruses are used to infect the IoT dataset, and the infection strategies of network traffic are collected in real-time. This dataset is originally designed to differentiate the malicious and benign traffic data for anomaly detection in the IoT network. It consists of 11 different classes, which means ten different attacks carried out by two botnets and one class of benign. The ground truth training and testing datasets are concatenated, around 5,79,782 & 2,48,478 records, respectively, on different forms as attack & normal. The dataset is in the form of binary classification

The dataset UNSW-NB 15 [46, 47] was developed by the Australian Cyber Security Centre (ACCS) cyber range lab using the IXIA PerfectStrom tool. The tool is used here to produce a mixture of modern regular operations and synthetic attack behaviors. TCP dump tool used to detect raw traffic of 100 GB data (e.g., Pcap files). Around nine types of attacks were included in this dataset, namely Worms, Fuzzers, DoS, Generic, backdoors, Reconnaissance, Analysis, Exploits & shellcode. The Argus and Bro-IDS programs were used to build the class label to produce 49 features. The ground truth training and testing datasets are concatenated, which are around 1,75,341 & 82,332 records, respectively, on different forms as attack & normal. The dataset is in the form of binary classification.

UNSW_IoT_Botnet [48] was created by the Cyber Range Lab of UNSW Canberra. This dataset was created for building a realistic IoT network environment with primary components of network architecture, fabricated IoT facilities, and feature extraction techniques. The Argus tool is used here to extract the necessary data features from the dataset. There are over 72 million records and 46 attributes in the generated dataset. The attribute assessment approach

Table 1
Statistics of three dataset

Dataset	Training Instances	Testing Instances	Classes
NBaIoT	579782	248478	11
UNSW-NB 15	1778033	762014	10
UNSW_BOT_IoT	204174	87502	4

Table 2
Name of classes and corresponding instance of three dataset

Data Set	NBaIoT		UNSW_NB15		UNSW_BOT_IoT	
S.No.	Class Name	Instances	Class Name	Instances	Class Name	Instances
1	benign	15538	Normal	2218764	DDoS	1541315
2	gafgyt_combo	15345	Generic	215481	DoS	1320148
3	gafgyt_junk	15449	Exploits	44525	Recon	72919
4	gafgyt_scan	14648	Fuzzers	24246	Normal	370
5	gafgyt_tcp	15676	DoS	16353		
6	gafgyt_udp	15602	Reconnaissance	13987		
7	mirai_ack	15138	Analysis	2677		
8	mirai_scan	14517	Backdoor	2329		
9	mirai_syn	16436	Shellcode	1511		
10	mirai_udp	15625	Worms	174		
11	mirai_udpplain	15304				

Table 3
Features of three dataset

Dataset Name	Name of the attribute	Number of Features
NBaIoT	Proto, saddr, sport, daddr, dport, seq, drate, srate, max, attack category& port, jitter, statistics of L1, L3, L5(mean, std, weight, variance, radius, covariance, pcc) of each IP	115
UNSW_NB 15 (NB15)	id,dur,proto,service,state,spkts,dpkts,sbytes,dbytes,rate,sttl,dttl,sload,dload,sloss, dloss,sinpkt,dinpkt,sjit,djit,swin,stcpb,dtcpb,dwin,tcprtt,synack,ackdat,smean, dmean,trans_depth,response_body_len,ct_srv_src,ct_state_ttl,ct_dst_ltm,ct_src_ dport_ltm,ct_dst_sport_ltm,ct_dst_src_ltm,is_ftp_login,ct_ftp_cmd,ct_flw_http_mthd, ct_src_ltm,ct_srv_dst,is_sm_ips_ports,Attack_category(Label)	43
UNSW_BOT_IoT (BOT_IoT)	proto, saddr, sport, daddr, dport, seq, stddev, N_IN_Conn_P_SrcIP, min, state_number, mean, N_IN_Conn_P_DstIP, drate, srate, max, attack_category (Label)	16

is used to reduce dimensionality, which aids in improving the intelligent threat detection system's performance in terms of time and space. The dataset is in the form of binary and multi-class classification.

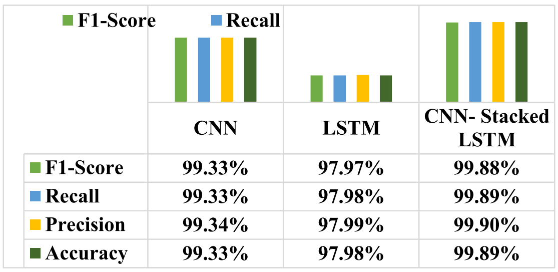
Table 2 contains information for each class instance of three separate datasets. The NBaIoT dataset is balanced, implying that the multiple class instances are almost equal in the count. The other two datasets are unbalanced, implying that the number of cases in each class varies significantly. The proposed model will be validated using both balanced and unbalanced datasets. Table 3 gives information about the input vector. BOT IoT has 15 input vectors, NBaIoT has 115, NB15 has 42, and BOT_IoT has 15 input vectors. The most common vector in all datasets is source IP, destination IP, source port, destination port, protocol, sequence, number of input connections between source to destination IP, jitter, and statistical information of network flow. which is contributing more for finding the malicious activity in the network.

4.2. Experimental setup

The experiments were performed on an Intel(R) Core (TM) i7-9750HF CPU @ 2.60 GHz, 2592 Mhz, 6 Core(s), 12 Logical Processor(s) with 32 GB RAM, under Windows and virtual machine of Ubuntu. The proposed framework has Apache Spark for preprocessing the massive amount of IoT traffic data in a distributed and parallel way. Deep Learning models were implemented with Tensorflow and Keras package. The master edge node experimentation was conducted on a single system. The performance of the DL models was evaluated using three datasets that are represented in Table 1.

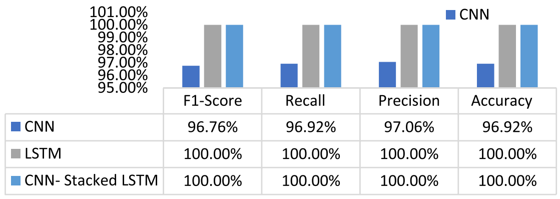
4.3. Experimental result analysis

The proposed intelligent threat detection system is evaluated using performance metrics of accuracy, precision, recall, and F1 score and confusion matrix performance metrics, calculated using the following



	CNN	LSTM	CNN- Stacked LSTM
🟩 F1-Score	99.33%	97.97%	99.88%
🟦 Recall	99.33%	97.98%	99.89%
🟨 Precision	99.34%	97.99%	99.90%
🟩 Accuracy	99.33%	97.98%	99.89%

Fig. 5. Performance metrics of UNSW_BOT_IoT dataset.



	F1-Score	Recall	Precision	Accuracy
🟦 CNN	96.76%	96.92%	97.06%	96.92%
⬜ LSTM	100.00%	100.00%	100.00%	100.00%
🟦 CNN- Stacked LSTM	100.00%	100.00%	100.00%	100.00%

Fig. 7. Performance metrics of UNSW_NB 15 dataset.

Equations (17 to 20).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{17}$$

$$Precision = \frac{TP}{FP + TP} \tag{18}$$

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{20}$$

Figures 5 & 7 display the classification performance of the proposed model with other two DL models are assessed on the imbalanced dataset of UNSW_BOT_IoT & UNSW_NB15, and Figs. 6 & 8 depicts the confusion matrix of the respective dataset. The experimental outcomes of training and testing are ouerformed wh nearly 100 percent accuracy. Even though the data set is unbalanced, the model better predicts all types of malicious activity in the IoT network.

Figure 9 demonstrates the proposed model's classification performance compared to the other two DL models on the NBaIoT balanced dataset, and Fig. 10 represents the classification of each class in the confusion matrix. The findings reveal that, apart from the miraisyn assault, the suggested technique correctly classifies all attacks. Miraisyn is accurately identified by 77% of respondents, with the remaining 23% misidentifying it as miraiudpscan. Because it did not categorize the attack as normal, the performance in binary classification was 100 percent. The CNN model has misclassified many classes as normal rather than attack. The LSTM model also outperforms other models in categorization across all types of assaults. The suggested method performs better in predicting all sorts of network attacks in an IoT network.

5. Comparison of result

5.1. Performance analysis with time

The suggested distributed technique's anomaly detection timing is compared to those of the existing approach. Table 4 compares the detection times of the

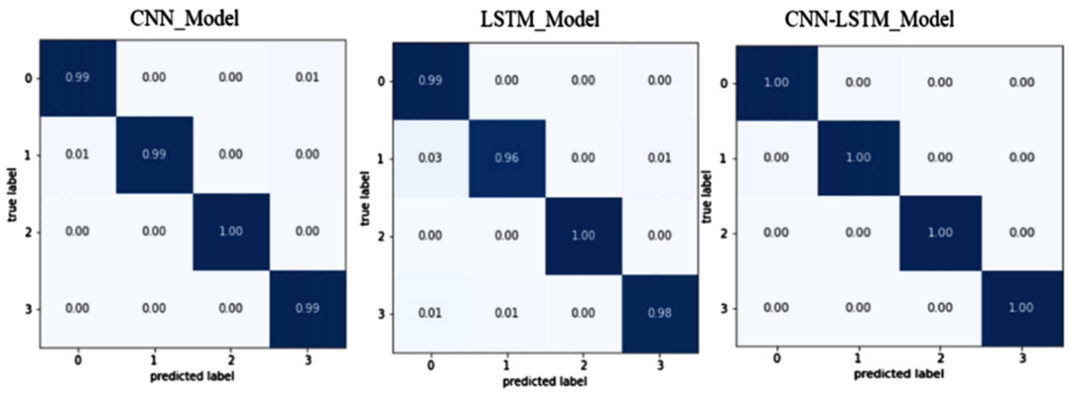


Fig. 6. Confusion matrix of UNSW_BOT_IoT dataset.

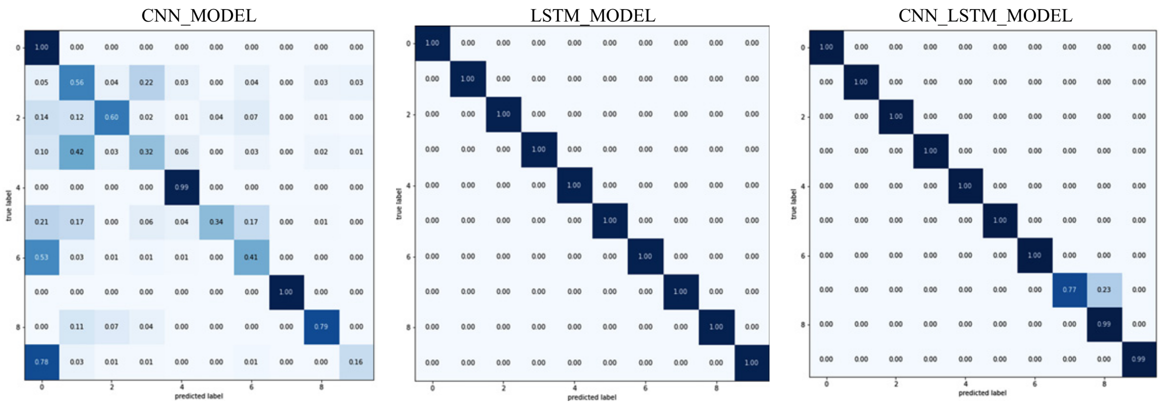
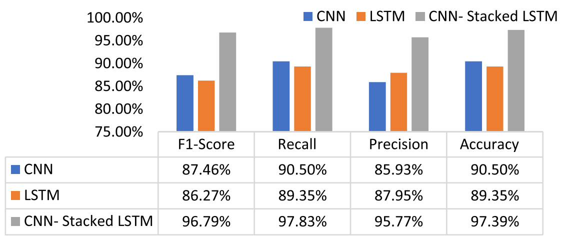


Fig. 8. Confusion matrix of UNSW_NB15 dataset.



	F1-Score	Recall	Precision	Accuracy
■ CNN	87.46%	90.50%	85.93%	90.50%
■ LSTM	86.27%	89.35%	87.95%	89.35%
■ CNN- Stacked LSTM	96.79%	97.83%	95.77%	97.39%

Fig. 9. Performance metrics of NBaIoT dataset.

CNN and LSTM techniques for different amounts of IoT traffic. If The number of input features increases, then the time required to process the model also increases in the unbalanced datasets, where input features are 16 in BOT IoT and 42 in UNSW NB 15. Even though the number of features is more, the suggested method requires less time to train the model in the balanced dataset (NBaIoT).

5.2. *Performance analysis with metrics*

The proposed approach's performance is evaluated using the metrics of accuracy, precision, recall, and F1-score, which are stated in Table 5. Accuracy is the rate of accurately classified attack as attack and normal as normal among total classification, which is achieved 99.98% & 100% in imbalanced data sets and 97.39% in the balanced dataset. When the costs of false positives are high, precision is employed to determine. Precision in the network threat detection implies that real attack traffic is forecasted as normal traffic. These devices may be attacked with malware. The cost of false negatives is calculated using recall. If malicious network traffic is anticipated as regular network flow in traffic flow instances,

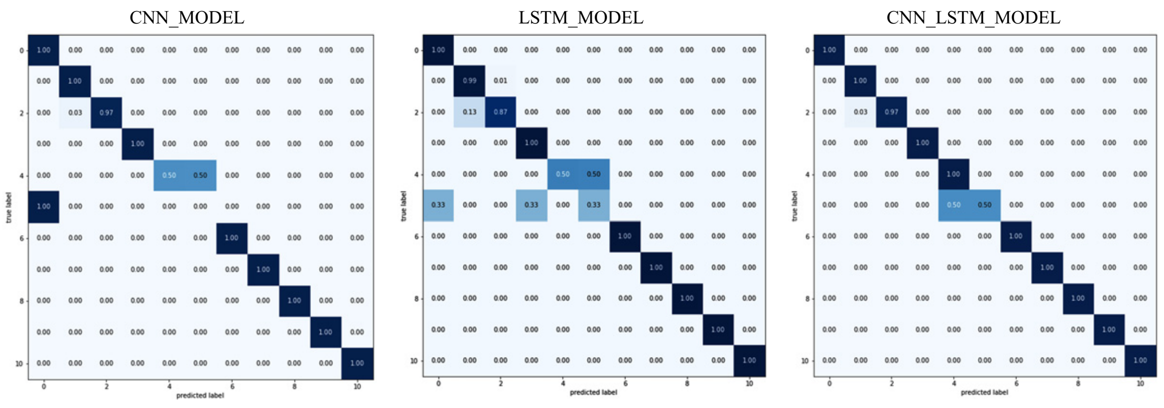


Fig. 10. Confusion matrix of NBaIoT dataset.

Table 4
Training time of CNN, LSTM, proposed model based on IoT botnet datasets

Dataset	Model/ Training Time	Per Step (msec)	Per Epoch (sec)	Iteration	Total (mins)	Batch Size
UNSW_ BOT_ IoT	CNN	6	14	6	1.4	1000
	LSTM	22	4	50	3.3	1000
	Proposed	16	3	42	2.1	1000
UNSW-NB 15	CNN	37	29	20	9	5000
	LSTM	770	440	14	52	5000
	Proposed	435	140	18	42	10000
NBaIoT	CNN	18	2	18	0.6	500
	LSTM	1	115	20	40	500
	Proposed	360	41	20	14	500

Table 5
Comparison of performance metrics

DATA SET	Metrics/ Model	CNN	LSTM	Proposed
UNSW_BOT_IoT	F1-Score	99.78%	97.97%	99.88%
	Recall	99.78%	97.98%	99.89%
	Precision	99.78%	97.99%	99.90%
	Accuracy	99.78%	97.98%	99.89%
UNSW_NB 15	F1-Score	96.76%	100.00%	100.00%
	Recall	96.92%	100.00%	100.00%
	Precision	97.06%	100.00%	100.00%
	Accuracy	96.92%	100.00%	100.00%
NBaIoT	F1-Score	87.46%	86.27%	96.79%
	Recall	90.50%	89.35%	97.83%
	Precision	85.93%	87.95%	95.77%
	Accuracy	90.50%	89.35%	97.39%

the result might be disastrous for the entire network system. There is a need to balance precision and recall. When there is an unequal class distribution, the F1-Score could be a preferable metric to utilize in large-scale data. Our proposed approach obtains high accuracy, recall, and F1- score in both balanced and unbalanced datasets, demonstrating that our model outperforms at predicting anomalous attacks in IoT networks.

5.3. *ROC (Receiver-operating characteristic) analysis*

The true positive rate (TPR) is plotted against the false positive rate (FPR) to create the receiver-operating characteristic (ROC) curve. TPR and FPR values can range from 0 to 1. As a result, the ROC's maximum area is 1. The accuracy of the model increases as the area under the curve increases. Figures 11–13 represents the loss and accuracy curve of ROC of three datasets; it observed that our proposed approach achieves maximum value for predicting the malware.

6. Conclusion

This research helps to establish effective anomaly detection in real-time at the edge system and have efficacy in real-time data processing, scaling, distributing, and effectively detecting malware at an early point. The proposed framework helps in reducing communication cost and bandwidth utilization and boosts data availability. The spark distributed processing is used for raw network data processing and normalization, which is distributed at the edge nodes to analyze parallel network traffic. The VCNN module is used to select model features while training, which decreases computing overhead. The dropout and hidden LSTM layers are used to avoid model overfitting, and RMSProp optimizers are used to improve the proposed model's accuracy. The proposed model is tested using two imbalanced datasets of UNSW NB15 & UNSW BOT-IoT and one balanced dataset of NBaIoT. The experimental results show that the proposed framework of DL – based intelligent threat detection is effective in terms of detecting malicious behaviour early in an edge

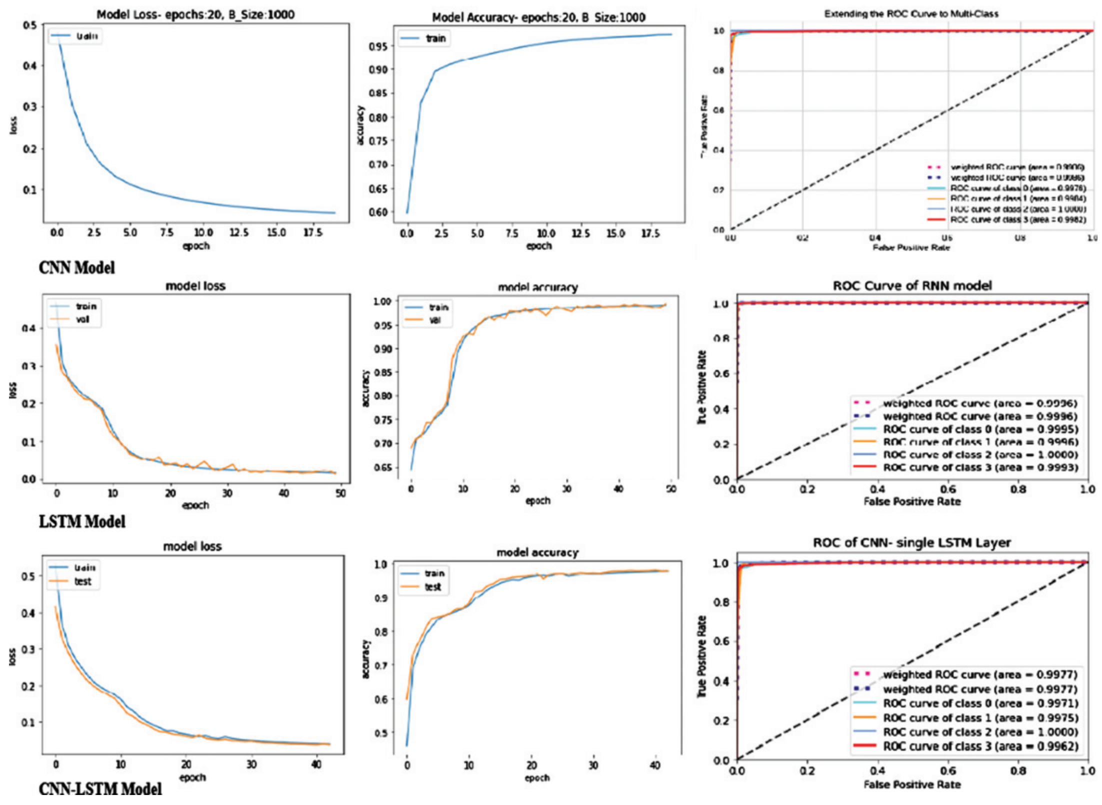


Fig. 11. Roc Curve of binary and multi-class classification of UNSW_BOT_IoT dataset.

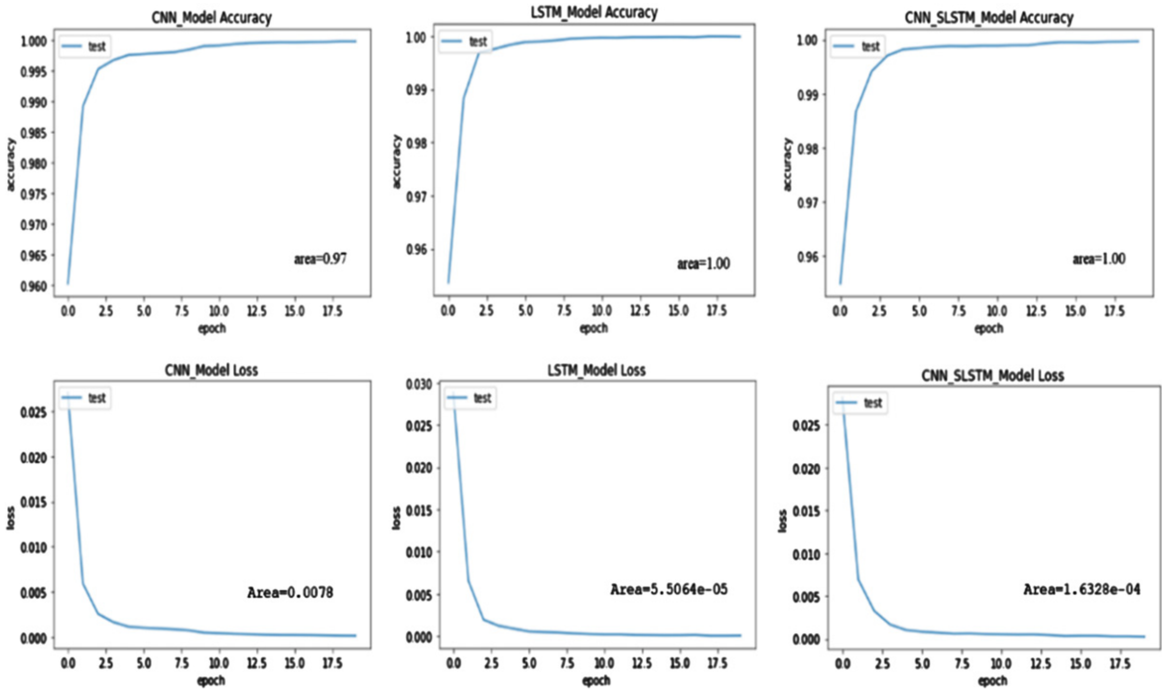


Fig. 12. ROC curve of binary classification on UNSW-NB15 dataset.

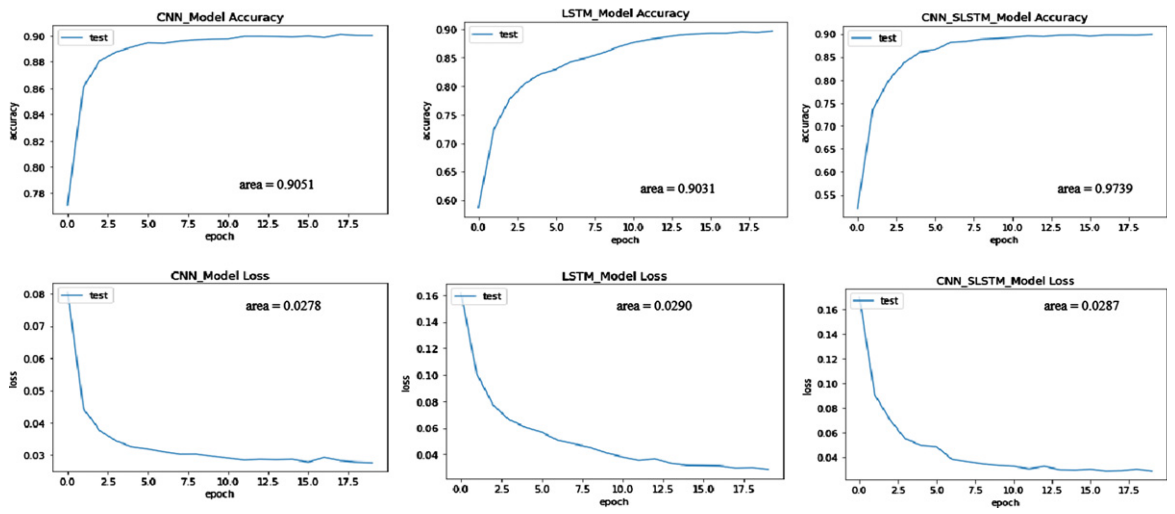


Fig. 13. ROC curve of binary classification on NBaIoT dataset.

Table 6
Comparison of accuracy with state of art model

Year	Dataset	Methodology	Accuracy
2019 [48]	UNSW_BOT_IoT	LSTM	98.05%
2020 [26]		Cloud-based (IoT_Mon)	98.84%
Proposed		HSDL	99.89%
2021 [50]	UNSW_NB15	CNN-LSTM	97.17 %
Proposed		HSDL	100.00%
2021 [49]	NBaIoT	CNN & LSTM	90.88%
Proposed		HSDL	97.39%

environment. According to Table 6, the suggested technique significantly outperformed state-of-the-art intelligent threat detection systems and traditional DL models. Furthermore, ROC analysis reveals that the suggested framework outperformed accuracy by a maximum of 1(100%) in imbalanced datasets.

References

[1] H. Belkhiri, A. Messai, M. Belaoued and F. Haider, Security in the internet of things: Recent challenges and solutions, in *International Conference on Electrical Engineering and Control Applications*, Constantine, Algeria, 2019, pp. 1133–1145.

[2] Palo alto networks, “2020 unit 42 IoT threat report,” 2020, https://unit42.paloaltonetworks.com/iot-threat-report-2020/.

[3] M. Antonakakis, T. April, M. Bailey et al., Understanding the mirai botnet, in *26th USENIX Security Symposium (USENIX Security17)*, Vancouver, BC, Canada, 2017, pp. 1093–1110.

[4] S. Fadilpasic, Researchers discover iot botnet capable of launching various ddos attacks, 2020. https://www.itpro portal.com/news/researchers-discover-iot-botnet capable-of-launching-various-ddos-attacks/.

[5] J. Vijayan, New malware family assembles iot botnet, 2020. https://www.darkreading.com/iot/new-malware-familyassembles-iot-botnet–/d/d-id/1337578.

[6] A. Derhab, M. Guerroumi, A. Gumaei, et al., Blockchain and random subspace learning-based ids for sdn-enabled industrial iot security, *Sensors* **19**(14) (2019), 3119.

[7] M. Imran, M.H. Durad, F.A. Khan and A. Derhab, Toward an optimal solution against denial of service attacks in software defined networks, *Future Generation Computer Systems* **92** (2019), 444–453.

[8] B. Du, H. Peng, S. Wang, et al., Deep irregular convolutional residual lstm for urban traffic passenger flows prediction, *IEEE Transactions on Intelligent Transportation Systems* **21**(3) (2020), 972–985.

[9] F.A. Khan and A. Gumaei, A comparative study of machine learning classifiers for network intrusion detection, in: *In International Conference On Artificial Intelligence and Security*, Springer, Cham, 2019, pp. 75–86.

[10] H. Chen, O. Engkvist, Y. Wang, M. Olivecrona and T. Blaschke, The rise of deep learning in drug discovery, *Drug Discov Today* **23**(6) (2018), 1241–1250.

[11] Z. Ning, K. Zhang, X. Wang, et al., Intelligent edge computing in internet of vehicles: A joint computation offloading and caching solution, *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[12] E. Bou-Harb, M. Debbabi and C. Assi, Big data behavioral analytics meet graph theory: On effective botnet takedowns, *IEEE Network* **31**(1) (2017), 18–26.

[13] E.M.B. Karbab, M. Debbabi, A. Derhab and D. Mouheb, Scalable and robust unsupervised android malware fingerprinting using community-based network partitioning, *Computers & Security* **96** (2020), article 101932.

[14] M. Marjani, F. Nasaruddin, A. Gani, et al., Big IOT data analytics: Architecture, opportunities, and open research challenges, *IEEE Access* **5** (2017), 5247–5261.

[15] A. Aldweesh, A. Derhab and A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues, *Knowledge-Based Systems* **189** (2020), article 105124.

[16] M.A. Ferrag, L. Maglaras, S. Moschoyiannis and H. Janicke, Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study, *Journal of Information Security and Applications* **50** (2020), article 102419.

[17] S. Mahdavifar and A.A. Ghorbani, Application of deep learning to cybersecurity: A survey, *Neurocomputing* **347** (2019), 149–176.

[18] Z. Ning, K. Zhang, X. Wang, et al., Intelligent edge computing in internet of vehicles: A joint computation offloading and caching solution, *IEEE Transactions on Intelligent Transportation Systems* (2020).

[19] T. Wang, G. Zhang, A. Liu, M.Z.A. Bhuiyan and Q. Jin, A secure iot service architecture with an efficient balance dynamics based on cloud and edge computing, *IEEE Internet of Things Journal* **6**(3) (2018), 4831–4843.

[20] X. Wang, Z. Ning and S. Guo, Multi-agent imitation learning for pervasive edge computing: A decentralized computation offloading algorithm, *IEEE Transactions on Parallel and Distributed Systems* **32**(2) (2020), 411–425.

[21] A. Derhab, M. Belaoued, M. Guerroumi and F.A. Khan, Two-factor mutual authentication offloading for mobile cloud computing, *IEEE Access* **8** (2020), 28956–28969.

[22] A. Boulemtafes, A. Derhab and Y. Challal, A review of privacy-preserving techniques for deep learning, *Neurocomputing* **384** (2020), 21–45.

[23] C.D. McDermott, F. Majdani and A.V. Petrovski, Botnet detection in the internet of things using deep learning approaches, In *2018 international joint conference on neural networks (IJCNN)*, IEEE, pp. 1–8.

[24] J. Kim, M. Shim, S. Hong, Y. Shin and E. Choi, Intelligent detection of IoT botnets using machine learning and deep learning, *Applied Sciences* **10**(19) (2020), 7009.

[25] S. Homayoun, M. Ahmadzadeh, S. Hashemi, A. Dehghantanha and R. Khayami, BoTShark: A deep learning approach for botnet traffic detection. *In Cyber Threat Intelligence*, Springer, Cham, 2018, pp. 137–153.

[26] M. Hammoudeh, J. Pimlott, S. Belguith, G. Epiphaniou, T. Baker, A.S. Kayes, B. Adebisi and A. Bounceur, Network traffic analysis for threat detection in the internet of things, *IEEE Internet of Things Magazine* **3**(4) (2020), 40–45.

[27] H. HaddadPajouh, A. Dehghantanha, R. Khayami and K.-K.R. Choo, A deep recurrent neural network based approach for internet of things malware threat hunting, *Future Generation Computer Systems* **85** (2018), 88–96.

[28] A.A. Diro and N. Chilamkurti, Distributed attack detection scheme using deep learning approach for Internet of Things, *Future Generation Computer Systems* (2017), 1–5.

[29] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai and Y. Elovici N-BaIoT: Network-based detection of IoT botnet attacks using deep autoencoders, *IEEE Pervasive Computing, Special Issue - Securing the IoT* (2018).

[30] I. Butun, B. Kantarci and M. Erol-Kantarci, Anomaly detection and privacy preservation in cloud-centric internet of things, In *2015 IEEE International Conference on Communication Workshop (ICCW)*, IEEE, 2015, pp. 2610–2615.

[31] D. Midi, A. Rullo, A. Mudgerikar and E. Bertino, Kalis A System for Knowledge-Driven Adaptable Intrusion Detection for the Internet of Things, in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS) IEEE*, 6 2017, pp. 656–666.

[32] Y.M.P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama and C. Rossow, IoTPOT: A novel honeypot for revealing current IoT threats, *Journal of Information Processing* **24**(3) (2016), 522–533.

[33] D.H. Summerville, K.M. Zach and Y. Chen, Ultra-lightweight deep packet anomaly detection for Internet of Things devices, In *2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC)*, IEEE, 2015, pp. 1–8.

[34] M. Satyanarayanan, A brief history of cloud offload: A personal journey from odyssey through cyber foraging to cloudlets, *GetMobile: Mobile Computing and Communications* **18**(4) (2015), 19–23.

[35] R. Ranjan, B. Benatallah, S. Dustdar and M.P. Papazoglou, Cloud resource orchestration programming: Overview, issues, and directions, *IEEE Internet Comput* **19**(5) (2015), 46–56.

[36] A. Jonathan, M. Ryden, K. Oh, A. Chandra and J. Weissman, Nebula: Distributed edge cloud for data intensive computing, *IEEE Transactions on Parallel and Distributed Systems* **28**(11) (2017), 3229–3242.

[37] G. Tanganelli, C. Vallati and E. Mingozzi, Edge-centric distributed discovery and access in the internet of things, *IEEE Internet of Things Journal* **5**(1) (2017), 425–438.

[38] J. Pan and J. McElhannon, Future edge cloud and edge computing for internet of things applications, *IEEE Internet of Things Journal* **5**(1) (2017), 439–449.

[39] M.B. Mollah, M.A. Azad and A. Vasilakos, Secure data sharing and searching at the edge of cloud-assisted internet of things, *IEEE Cloud Computing* **4**(1) (2017), 34–42.

[40] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, et al., Edge analytics in the internet of things, *IEEE Pervasive Computing* **14** (2015), 24–31.

[41] W. Jo, S. Kim, C. Lee and T. Shon, Packet preprocessing in CNN-based network intrusion detection system, *Electronics* **9**(7) (2020), 1151.

[42] B. Alotaibi and M. Alotaibi, A stacked deep learning approach for IoT cyberattack detection, *Journal of Sensors* **2020** (2020).

[43] H. Chung and K.S. Shin, Genetic algorithm-optimized long short-term memory network for stock market prediction, *Sustainability* **10**(10) (2018), 3765.

[44] Y. Mirsky, T. Doitshman, Y. Elovici and A. Shabtai, Kitsune: An ensemble of autoencoders for online network intrusion detection, in *Network and Distributed System Security (NDSS) Symposium*, San Diego, CA, USA, 2018.

[45] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai and Y. Elovici, N-BaIoT: Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders, *IEEE Pervasive Computing, Special Issue - Securing the IoT*, 2018.

[46] N. Moustafa and J. Slay, UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), *Military Communications and Information Systems Conference (MilCIS)*, IEEE, 2015.

[47] N. Moustafa and J. Slay, The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 dataset and the comparison with the KDD99 dataset, *Information Security Journal: A Global Perspective* (2016), 1–14.

[48] N. Koroniotis, N. Moustafa, E. Sitnikova and B. Turnbull, Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset, *Future Generation Computer Systems* **100** (2019), 779–796.

[49] H. Alkahtani and T.H. Aldhyani, Botnet attack detection by using CNN-LSTM model for internet of things applications, *Security and Communication Networks* **2021** (2021).

[50] M.M. Hassan, A. Gumaei, A. Alsanad, M. Alrubaian and G. Fortino, A hybrid deep learning model for efficient intrusion detection in big data environment, *Information Sciences* **513** (2020), 386–396.