Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.073577

ARTICLE

Scalable and Resilient AI Framework for Malware Detection in Software-Defined Internet of Things

Maha Abdelhaq[1], Ahmad Sami Al-Shamayleh[2], Adnan Akhunzada[3,*], Nikola Ivković[4] and Toobah Hasan[5]

[1]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh, 11671, Saudi Arabia

[2]Department of Data Science and Artificial Intelligence, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, 19328, Jordan

[3]College of Computing and Information Technology, Departement of Data & Cybersecurity, University of Doha for Science & Technology, Doha, 2444, Qatar

[4]Faculty of Organization and Informatics, University of Zagreb, Pavlinska 2, Varaždin, 42000, Croatia

[5]COMSAT University Islamabad (CUI), Islamabad, 45550, Pakistan

*Corresponding Author: Adnan Akhunzada. Email: adnan.adnan@udst.edu.qa

Received: 21 September 2025; Accepted: 24 November 2025; Published: 10 February 2026

ABSTRACT: The rapid expansion of the Internet of Things (IoT) and Edge Artificial Intelligence (AI) has redefined automation and connectivity across modern networks. However, the heterogeneity and limited resources of IoT devices expose them to increasingly sophisticated and persistent malware attacks. These adaptive and stealthy threats can evade conventional detection, establish remote control, propagate across devices, exfiltrate sensitive data, and compromise network integrity. This study presents a Software-Defined Internet of Things (SD-IoT) control-plane-based, AI-driven framework that integrates Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM) networks for efficient detection of evolving multi-vector, malware-driven botnet attacks. The proposed CUDA-enabled hybrid deep learning (DL) framework performs centralized real-time detection without adding computational overhead to IoT nodes. A feature selection strategy combining variable clustering, attribute evaluation, one-R attribute evaluation, correlation analysis, and principal component analysis (PCA) enhances detection accuracy and reduces complexity. The framework is rigorously evaluated using the N_BaIoT dataset under k-fold cross-validation. Experimental results achieve 99.96% detection accuracy, a false positive rate (FPR) of 0.0035%, and a detection latency of 0.18 ms, confirming its high efficiency and scalability. The findings demonstrate the framework's potential as a robust and intelligent security solution for next-generation IoT ecosystems.

KEYWORDS: AI-driven malware analysis; advanced persistent malware (APM); AI-powered malware detection; deep learning (DL); malware-driven botnets; software-defined internet of things (SD-IoT)

1 Introduction

The rapid advancement of technology and the proliferation of smart devices have driven exponential growth in IoT connectivity, with nearly 30.9 billion devices projected by 2025 [1]. While these developments enable intelligent communication, they also introduce significant cybersecurity challenges [2]. The heterogeneous, dynamic, and resource-constrained nature of IoT devices generates massive data and exposes networks to vulnerabilities, threats, and sophisticated attacks, including phishing, Man-in-the-Middle (MitM), side-channel exploits, application-layer intrusions, and large-scale malware-driven botnets.

Copyright © 2026 The Authors. Published by Tech Science Press.
This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Among these, malware-driven botnets are particularly severe, enabling DDoS, ransomware, credential theft, spam campaigns, and propagation of additional malware [3–5]. Early and efficient detection of such threats is critical for mitigation and prevention [6–8]. Software-defined IoT (SD-IoT) provides a promising approach by using centralized and programmable SDN control to manage heterogeneous networks, including IoT, Fog of Things (FoT), and Industrial IoT (IIoT) without overburdening devices [9]. By separating the control and data planes, SDN enables dynamic, adaptive, and automated detection mechanisms, supporting secure network orchestration, flexible device management, and scalable operation across large IoT ecosystems. This study is guided by four key research questions: (1) how the hybrid GRU–LSTM framework enhances malware botnet detection accuracy in IoT environments; (2) how effectively it addresses multiclass detection challenges such as class imbalance and dynamic traffic patterns; (3) how its performance compares with state-of-the-art intrusion detection models in terms of accuracy and generalization across datasets; and (4) whether it can maintain lightweight computational efficiency for real-time deployment in resource-constrained IoT networks. The main contributions of this paper are as follows:

- This study presents a scalable AI-driven Software-Defined Internet of Things (SD-IoT) framework designed to safeguard critical IoT infrastructures against sophisticated malware botnet attacks. The framework integrates Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM) framework for enhanced threat detection, while the centralized control plane enforces security policies efficiently without adding computational overhead to resource-constrained IoT devices.
- The proposed framework undergoes extensive evaluation using the well-established N-BaIoT dataset, ensuring a rigorous assessment against multivector botnet attacks through standard performance metrics.
- By integrating feature selection techniques, the framework effectively identifies and prioritizes critical features, reducing computational complexity while maintaining high detection accuracy.
- The framework's efficacy and robustness is validated through extended performance metrics and a 10-fold cross-validation approach, ensuring reliability in diverse deployment scenarios.
- Experimental results and analysis demonstrate superior performance in both detection accuracy and real-time processing, establishing a new benchmark for SDN-driven IoT security solutions.

The remainder of this paper is organized as follows: Section 2 reviews the background and related work; Sections 3 and 4 details the proposed methodology, including architecture, preprocessing, dataset, and evaluation metrics; Section 5 presents the experimental setup, results, and discussion; and Section 6 concludes the study with future directions.

2 Background and Related Work

This section briefly describes the background architecture of SD-IoV and given botnet attacks. Besides, the related work about botnet detection in IoT is discussed.

2.1 Background

Software-Defined Networking (SDN) enables centralized intelligence and programmability across the application, control, and data planes, facilitating efficient management of IoT and IIoT ecosystems [9]. However, the rise of sophisticated IoT botnets such as Mirai and Gafgyt, capable of large-scale DDoS and remote exploitation, has rendered traditional detection techniques inadequate, demanding advanced AI-driven behavioral and network-based defenses [10].

2.2 Related Work

DL-driven architectures remain in their early stages for securing IoT networks [11]. In [12], an LSTM-based botnet detection model using CVUT traffic data achieved 99.9% accuracy. Similarly, reference [13] proposed a Bidirectional LSTM for IoT-botnet detection, attaining 96% accuracy on a Mirai-based dataset. Reference [14] combined CNN and RNN on CTU-13 and ISOT datasets, achieving 99.3%detection, while reference [15] applied LSTM to analyze content and metadata, yielding 98% accuracy. In [16], multiple DL models (LSTM, RNN, CNN) identified malicious domains with a 90% detection rate. Reference [17] emphasized deep learning's strength in multi-level abstraction and anomaly detection. Further, reference [18] utilized LSTM and SVM for detecting various attacks (DoS, DDoS, Port Scanning, etc.) in SDN-IoT using SDN-IoT and SDN-NF-TJ datasets, achieving 97% accuracy. Likewise, reference [19] presented a DNN-LSTM hybrid for fog-based SDN environments using N_BaIoT 2018, achieving 99.98% accuracy. Reference [20] proposed a GRU-LSTM hybrid for distributed attack detection using NSL-KDD, achieving 87.9% accuracy. In [21], an SDN-based IDS using LSTM achieved 91.4% on CSE-CIC-IDS2018, while reference [22] reached 99.9% with LSTM on N_BaIoT 2018 for binary botnet detection. Our proposed SDN-empowered system aims to enhance IoT security in scalability, performance, and control-plane protection. Unlike existing works, it addresses the lack of next-generation IoT datasets, limited training instances, and absence of comprehensive evaluations using standard and extended performance metrics. Table 1 summarizes the comparison of related work and key characteristics.

Table 1: Comparison of existing literature with characteristics

Ref.	Attack	Model	Dataset/Method	Strength	Limitation
Our	Mirai and Gafgyt Botnet	LSTM-GRU	N_BaIoT/Hybrid DL model	Implemented hybrid model (GRU-LSTM) for the detection of botnet attack in IoT	Detection time could be improved.
[13]	Botnet	LSTM	Self-generated dataset (CVUT)	Method for detecting the botnet attack	Did not consider hybrid algorithms.
[14]	UDP, ACK, DNS	BiLSTM	Botnet and self-generated benign records/DL algorithm with embedding word	Packet-level detection in IoTs and network	Bidirectional approach adds overhead and increases processing time.
[15]	DDoS, SPAM, P2P, IRC	CNN-RNN	CTU-13, ISOT/Botnet detection by modelling network traffic	Inspects statistical-based network flow features	Utilisation of updated dataset is meaningful.

(Continued)

Table 1 (continued)

Ref.	Attack	Model	Dataset/Method	Strength	Limitation
[16]	TCP, HTTP, UDP	LSTM	Cresci et al./Use the content and metadata by considering LSTM and synthetic minority oversampling	LSTM could detect Botnet behaviours that were significantly different from Normal.	Old-dated dataset has been considered for evaluation.
[17]	Botnet	CNN-LSTM, LSTM	Dataset of Alexa and 17-JGA/Analyze and find domain names using DL	Detect malware by specifying domains	Lack in presenting inner mechanics of DL model.
[18]	N/A	CNN, RNN	N/A/Deep Learning overview	Explain the significance of DL in numerous frameworks	No implementation.
[19]	Botnet	LSTM, SVM	SDN-IoT, SDN-NF-TJ/Network attack detection using SDN in IoT	Considered the power of SDN	Lack in presenting the 10-fold results.
[20]	Botnet	DNN-LSTM	N_BaIoT/Botnet detection in fog using SDN	Identify botnet attack in fog using SDN	Need to consider more data for experimentation.
[21]	Distributed attack	GRU-LSTM	NSL-KDD/Identify distributed attack in IoT	Use DL-based hybrid framework for attack detection	The detection rate is low to detect the advanced attacks.
[22]	Intrusion detection	LSTM	CSE-CIC-IDS2018/IDS system for detecting attack in IoT using SDN	SDN framework is efficient for detecting intrusion	Detection accuracy is 91.4% which is not impressive.
[23]	Botnet	LSTM	N_BaIoT/Botnet attack detection in IoT using SDN	SDN framework is implemented for botnet detection	The proposed work is for binary attack detection only.

3 Preliminaries

The algorithms and dataset used in this paper are detailed in this section.

3.1 Algorithms

The basic architectural description of proposed algorithms and dataset are discussed below.

3.1.1 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU), a newer variant of the recurrent neural network family [23], addresses the vanishing gradient problem by using two gates (update and reset) to control information flow and memory retention [24]. With fewer tensor operations and no separate cell state, GRU offers faster training while maintaining effective long-term dependency modeling [25].

3.1.2 Long Short-Term Memory (LSTM)

Long short-term Memory (LSTM) bridges the temporal gap to solve the vanishing gradient problem and has a similar control flow as a recurrent neural network for long term memory [26]. The forget gate is used by the recurrent neural network (RNN) to retain information across longer sequences. Back-propagation amplifies error signals, resulting in poor system performance and execution.

3.1.3 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a type of artificial neural network widely used for computer vision, classification, and object recognition. Its multilayered architecture, including convolutional, pooling, fully connected, flattening, and padding layers, enables real-time extraction of significant features from input data, with mathematical formulations detailed in [27].

3.1.4 Deep Neural Network (DNN)

Deep Neural Network (DNN) is a fully connected neural network designed to simulate human brain activity for pattern recognition and learning. It consists of input, hidden, and output layers, where neurons perform computations on incoming data and weights before passing information forward, with mathematical formulations detailed in [28].

4 Methodology

This section elaborates the methodology of the proposed framework, employed DL algorithms, dataset utilized, and pre-processing of data with feature selection techniques. A simplified overview of the control plane-enabled malware botnet detection framework is shown in Fig. 1 [22]. The proposed hybrid DL-driven framework is highly scalable, and efficient. Besides, the software defined orchestration does not put extra load on the IoT infrastructure; on the contrary it enables engineers to maintain and perform security operations in a more reliable and automated manner. To reduce extra overhead, we applied feature selection mechanisms to get the best features that optimize the accuracy and time complexity.

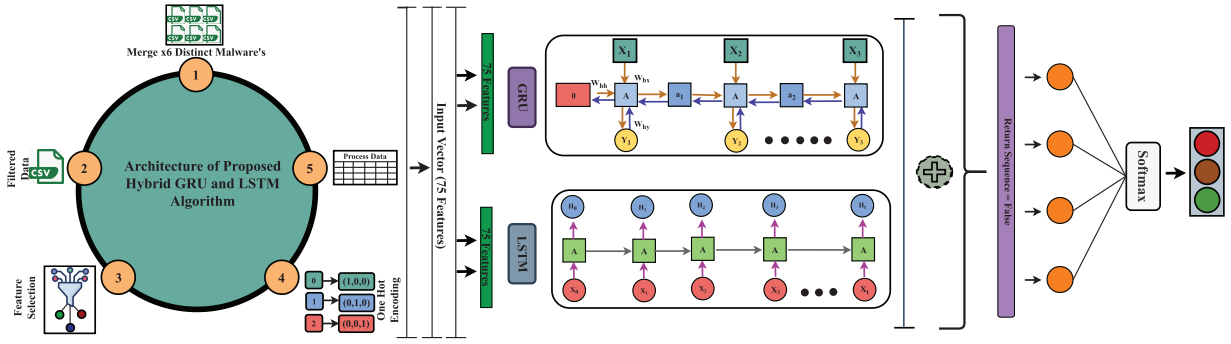


Figure 1: The architecture of proposed hybrid framework

4.1 Dataset

We utilized the N_BaIoT dataset, developed by the University of Negev, Israel, to address the limitations of outdated botnet datasets [29]. This dataset captures evolving IoT attack patterns, including Mirai and Gafgyt, across nine device types such as security cameras, webcams, doorbells, thermostats, and baby monitors. Table 2 summarizes the dataset, which comprises 7,062,611 records, of which 4,590,136 were used for experimentation, including 292,044 benign and 2,405,593 attack instances. The dataset contains 116 attributes, with 115 features and one label.

Table 2: Dataset description

Sr.	Name	Benign	Mirai	Gafgyt
1	Thermostat	13,113	512,134	310,631
2	PT 737E Security Camera	62,154	436,011	330,095
3	PT 838E Security Camera	98,514	429,338	309,040
4	SNH 1011 Web Cam	52,150	–	323,072
5	XCS7 1002 WHT Security Camera	46,585	513,249	303,223
6	XCS7 1003 WHT Security Camera	19,528	514,861	316,438
	Total Records	**292,044**	**2,405,593**	**1,892,499**

Although the experiments were conducted solely on the N_BaIoT dataset, its rich diversity of benign and malicious traffic traces collected from multiple IoT devices under real-world conditions supports strong model generalization. The dataset encompasses dynamic traffic behaviors, device heterogeneity, and a wide range of attack signatures (e.g., Mirai and Gafgyt), making it representative of modern IoT network environments. Consequently, the proposed framework can be readily adapted to other IoT and IIoT datasets with minimal retraining, ensuring scalability and applicability to real-world deployments.

4.2 Pre-Processing

The dataset utilized in this paper contains records of various IoT device. This dataset needs to pre-processed for removing anomalies and the extraction of useful features from data. The preprocessing of the N_BaIoT2018 dataset has been completed. Records having missing, nan, or infinite values were first removed from the dataset. To enhance the quality of the dataset, the normalization is likewise accomplished, which incorporates scaling all values from 0–1 range by using the MinMaxScalar function. The preprocessing of the dataset has been done by importing all CSV's files. After that, the 6 distinct CSV's files merged together and labels are encoded against each sample. At last, the merged CSV file is saved.

4.2.1 Handling Categorical Data

In N_BaIoT dataset, target classes are strings (i.e., Benign, Mirai, Gafgyt), therefor; One-hot-Encoding is used to change categorical data to integer data to be used subsequently for training of algorithm properly.

4.2.2 Feature Scaling

The dataset comprised of various features having the immense variation between the minimum and maximum integer values. For the stable convergence of weights and to avoid gradient ascend and descend, we have normalized all the dataset features utilizing MinMaxScaler to map values in the range of 0 and 1.

4.3 Feature Selection

Feature selection is applied prior to classification to enhance malware detection accuracy while reducing the computational complexity of deep learning model training. Recent studies highlight the benefits of combining multiple feature filtration methods, as individually weak features can collectively improve classifier performance. In this work, features were extracted using five techniques, Variable Clustering Attribute, One-R Attribute Evaluation, Attribute Evaluation, Correlation, and Principal Component Analysis (PCA), as summarized in Table 3. The less common feature selection techniques include the Variable Clustering Attribute method, which groups correlated features to minimize redundancy, and the One-R Evaluation, which employs simple rule-based accuracy to enhance interpretability and computational efficiency. Experiment is performed on Weka [30] for feature filtration. For experimentation, total number of samples are 4,590,136 out of which 292,044 benign and 4,298,092 botnet signatures. From botnet signatures, 1,892,499 Gafgyt, and 2,405,593 are Mirai attack. To achieve efficient and accurate results, we have performed feature filtration techniques to get significant features using 5 distinct techniques, and finally we applied majority voting mechanism to get 74 best features. Besides, 75th feature is a label representing a total of 75 features.

Table 3: List of best feature with their corresponding feature selection mechanism

Techniques	Features
Attribute evaluation	115,39,38,40,29,41,37,36,35,34,31,32,33,42,43,44,51,53,54,55,52,50,45,49,46,47,48,30,28 57,10,11,27,12,8,7,81,79,71,72,6,5,2,3,4,13,76,77,78,14,15,22,24,25,26,23,21,16,20,17,112, 56,1,58,114,97,96,98,87,99,95,94,93,92,89,90,91,84,75,73,80,100,101,102,109,62,63,61,82, 113,110,108,103,107,104,105,83,88,86,59,68,67,69,85,70,74,66,65,64,63,60,106,112,18,19, 111,9
Variation clustering attribute	44,57,51,64,74,58,65,77,107,80,3,18,6,21,2,67,70,109,9,24,73,29,76,12,2,79,45,72,4,52,38 69,31,66,85,78,90,47,92,93,43,49,85,59,95,30,88,81,7,1,22,19,16,10,25,13,28,84,91,71,20 98,34,40,41,105,48,37,68,55,82,89,62,96,32,103,39,110,46,94,33,36,83,53,60,17,75,5,108, 23,8,11,50,26,14,102,42,100,113,99,63,106,101,111,104,97,87,56,114,112,61,54,86,1,35
One R-Attribute	30,15,76,27,12,14,29,73,79,70,67,11,26,23,8,25,10,28,13,5,20,17,2,22,7,24,9,48,40,43,101, 41,60,34,53,75,86,94,87,93,83,52,46,98,91,21,84,6,105,112,63,77,78,59,61,39,56,19,80,18, 3,32,72,54,74,45,110,96,103,89,82,16,1,49,64,90,37,44,92,85,109,102,57,65,50,47,113,111, 106,95,58,38,69,104,42,66,31,99,71,35,97,114,107,51,36,115,33,68,88,81,108,100,55,62,4
Correlation	13,28,4,19,1,16,30,15,7,22,10,25,29,14,26,11,9,24,61,79,27,12,73,67,70,23,68,21,20,5,111, 54,17,3,18,62,71,98,51,52,33,112,104,55,58,105,6,48,91,84,41,34,72,45,38,69,31,65,66,63, 47,53,103,60,46,96,78,110,39,89,32,82,40,56,75,44,37,87,107,114,97,90,113,83,49,106,42, 35,99,92,85,77,80,95,88,50,81,102,57,43,109,74,36,86,64,93,100,94,101,115,108,2,8
PCA	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18

4.4 Proposed DL-Framework

We implemented three deep learning frameworks: Hybrid GRU–LSTM, Hybrid CNN (2D–3D), and Hybrid GRU–DNN. Their detailed architectures, including layers, neurons, activation and loss functions, epochs, batch size, and optimizer, are summarized in Table 4. All models were trained using categorical cross-entropy, a batch size of 128, learning rate of 0.001, and the Adam optimizer, with no dropout layers employed. Training and cross-validation are outlined in Algorithm 1, which describes the execution flow for merging two classifiers. During training, “T” denotes the training dataset, “t” the testing dataset, and “W”

the weights for GRU (G) and LSTM (L). Nested loops iterate over epochs (“E”) and batches (“B”) to compute loss and update weights, followed by evaluation on the test set.

Table 4: Description of the proposed framework

Algorithm	Layer	Neurons	B.S	Epoch	Optimizer	A.F	L.F
GRU-LSTM	GRU Layer (3)	(450, 400, 350)	128	5	Adam	ReLU	CC-E
	LSTM Layer (3)	(450, 400, 350)				ReLU	
	Dense Layer	350				–	
	Output Layer	3				Softmax	
	Dropout	–				–	
GRU-DNN	GRU Layer (3)	(450, 400, 350)	128	5	Adam	ReLU	CC-E
	Dense Layer (3)	(450, 400, 350)				ReLU	
	Dense Layer	350				–	
	Output Layer	3				Softmax	
	Dropout	–				–	
CNN2D-3D	2D Layer (3)	(450, 400, 350)	128	5	Adam	ReLU	CC-E
	3D Layer (3)	(450, 400, 350)				ReLU	
	Dense Layer	350				–	
	Output Layer	3				Softmax	
	Dropout	–				–	

Notes: CC-E = Categorical Cross-Entropy; B.S = Batch Size; A.F = Activation Function; L.F = Loss Function.

The computational complexity of proposed algorithm (i.e., Hybrid GRU-LSTM) is dependent upon two different deep learning algorithms, i.e., (GRU and LSTM) due to its hybrid nature. Both of the parallel algorithms belong to the same Recurrent Neural Network (RNN) family of deep learning models, therefore they have same complexity which is represented in Eq. (1).

$$O = (W) \tag{1}$$

In Eq. (1), W is the number of the weights. RNN model is local in space and time as a result, the input size has no effect on network storage space. The time complexity per weight for each time step is O(1).

Algorithm 1: Training of GRU & LSTM

Require: Data set (D-ŝt), Dataset Training (D-t̂r), Dataset Testing (D-t̂t), GRU Layers (G-l̂y), LSTM Layers (L-l̂y), Folds (F-d̂), Epochs (E-p̂), Batch-size (B-ŝ), Weights (W-ê), Loss Function (CategoricalCrossentropy) (L-f̂), Optimizer (Adam) A-d̂, Results R-ŝt, Results for each Fold (R f̂d)

Ensure: Training, Testing and Saving Results R

1: **function** HYBRID GRU-LSTM(D-t̂r)
2: Normalize D-t̂r(0 ~ 1)
3: *R*eshape (D-ŝt)
4: *I*nitialize W-ê for (G-l̂y)(L-l̂y),
5: *I*nitialize W for CL & D
6: *Hybrid* = ⊕ ((G-l̂y)(L-l̂y))
7: **for** $Fold \leftarrow 1$ to K **do**

(Continued)

Algorithm 1 (continued)

```
        Split (D-ŝt) into (D-t̂r) and (D-t̂t)
        for Epoch ← 1 to (E-p̂) do
            for Sample ← 1 to ... (D-t̂r) do
                Train w.r.t (B-ŝ) and A-d̂
            end for
            Pred = Predict on (D-t̂r)
            LossVector = Calculate (L-f̂) w.r.t (D-t̂r) for each Pred
            if (LossVector > intended Results) then
                Update (W-ê) using backpropagation through time (BPTT) for GRU & LSTM branches;
    gradients are aggregated at the fusion layer and optimized with Adam
            end if
        end for
        Rf̂d = Calculate R-ŝt for (F-d̂)
        R.append(Rf̂d)
    end for
    Result = Predict on C
    Save R
end function
```

5 Experimental Setup, Evaluation Metrics and Results

This section provide details regarding the experimentation setup and evaluation of our proposed mechanism in terms of its performance. The performance evaluation metrics are also discussed in this section.

5.1 Experimental Setup

The construction of our proposed hybrid deep learning model includes initialization of two parallel deep learning models, i.e., GRU and LSTM following the use of a merge layer for continuation. Consequently, ending the neural network with a simple dense layer as an output layer using the "*softmax*" activation function. The details of the proposed and other contemporary models are presented in Table 4. The software libraries used for experimentation and evaluation include TensorFlow and sklearn, respectively. An efficient, user-friendly high-level python library famously known as Keras is also utilized. The detail of the software, along with versions and hardware specifications for our experimentation, is elaborated in Table 5.

Table 5: Details of experimental setup

Component	Specification	Library	Version
CPU	Core i7-8750H (2.21 GHz), RAM 16 GB	Numpy	1.8.2
Core/Thread	6/12	TensorFlow	1.1.4
GPU	NVIDIA GTX 1050Ti	Scikit-Learn	0.15.2
OS	Windows 10 (64-bit)	Pandas	1.3.4
Tool	Anaconda Spyder	Keras	2.6.2
Language	Python 3.6		

5.2 Evaluation Metrics

Deep learning is considered as optimum if it entails a high detection rate of accuracy to correctly identify anomalies with low false alarms. We evaluated the performance of our proposed and comparative models, Hybrid GRU-LSTM, Hybrid CNN2D-CNN3D, and Hybrid GRU-DNN, using standard metrics (accuracy, precision, recall, F1-score) and extended metrics including FNR, FPR, FDR, FOR, NPV, TNR, ROC curve, and MCC.

5.3 Results

The deep learning framework is designed to classify three classes: benign, Mirai, and Gafgyt attacks. Experiments were conducted using the proposed hybrid GRU–LSTM model and comparative algorithms (GRU–DNN and CNN2D–CNN3D). We employed 10-fold cross-validation to ensure unbiased evaluation, with results for standard metrics (precision, recall, F1-score, and accuracy) reported in Table 6. Confusion matrices for all models are presented in Fig. 2, demonstrating that the proposed GRU–LSTM effectively distinguishes between benign and attack classes.

Table 6: 10-Fold results of GRU-LSTM, GRU-DNN, and CNN2D-CNN3D

Metrics	Technique	1	2	3	4	5	6	7	8	9	10
Accuracy	*GRU-LSTM*	99.96	99.95	99.97	99.96	99.95	99.95	99.98	99.95	99.95	99.98
	GRU-DNN	99.93	99.96	99.92	99.93	99.96	99.94	99.94	99.94	99.95	99.93
	CNN2D-CNN3D	99.93	99.91	99.90	99.90	99.91	99.93	99.89	99.89	99.93	99.91
Precision	*GRU-LSTM*	99.78	99.76	99.76	99.76	99.77	99.78	99.76	99.79	99.77	99.77
	GRU-DNN	99.76	99.64	99.85	99.15	99.76	99.74	99.50	99.71	99.71	99.64
	CNN2D-CNN3D	99.65	99.65	99.60	99.59	99.61	99.61	99.61	99.64	99.61	99.61
Recall	*GRU-LSTM*	99.66	99.64	99.65	99.68	99.63	99.63	99.63	99.83	99.65	99.63
	GRU-DNN	99.22	99.73	98.97	99.76	99.73	99.44	99.64	99.40	99.57	99.39
	CNN2D-CNN3D	99.56	99.51	99.52	99.53	99.53	99.56	99.54	99.54	99.54	99.52
F1-score	*GRU-LSTM*	99.66	99.64	99.65	99.68	99.63	99.63	99.63	99.83	99.65	99.63
	GRU-DNN	99.22	99.73	98.97	99.76	99.73	99.44	99.64	99.40	99.57	99.39
	CNN2D-CNN3D	99.56	99.51	99.52	99.53	99.53	99.56	99.54	99.54	99.54	99.52

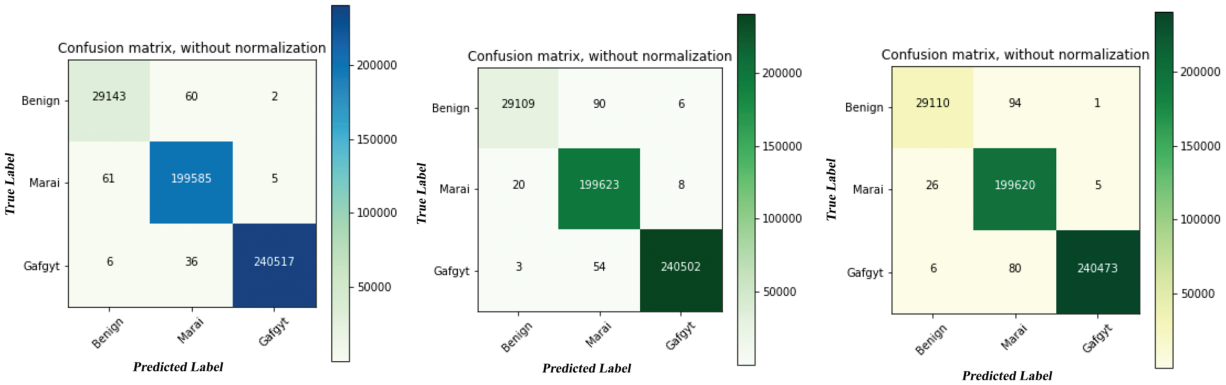
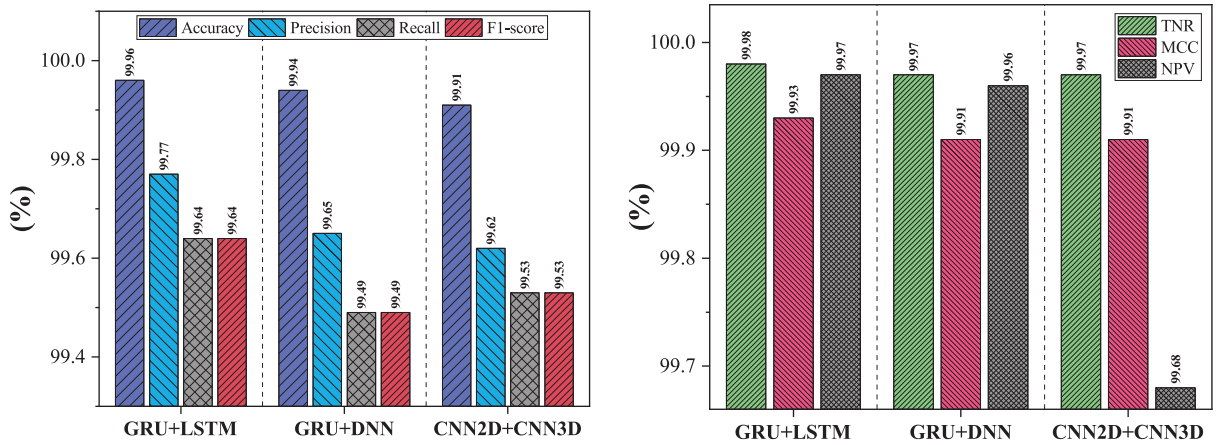


Figure 2: Confusion matrices of proposed GRU-LSTM, GRU-DNN and CNN2D-CNN3D

To evaluate the performance of the proposed algorithms, accuracy, precision, recall, and F1-score were computed. As shown in Fig. 3, the hybrid GRU–LSTM achieved the highest performance with 99.96% accuracy, 99.77% precision, 99.64% recall, and 99.64% F1-score, outperforming GRU–DNN (99.94%, 99.65%, 99.49%, 99.49%) and CNN2D–CNN3D (99.91%, 99.62%, 99.53%, 99.53%). ROC analysis is presented in Fig. 4. AUC values appear as 1.0 due to rounding, while the actual scores are 0.998163 (GRU–LSTM), 0.997339 (GRU–DNN), and 0.997881 (CNN2D–CNN3D), indicating robust performance without overfitting and effective separation of benign and attack classes. Additionally, True Negative Rate (TNR), Matthews Correlation Coefficient (MCC), and Negative Predictive Value (NPV) were evaluated to assess anomaly detection capability (Fig. 5). The proposed GRU–LSTM achieved 99.98% TNR, 99.93% MCC, and 99.97% NPV, surpassing GRU–DNN (99.97%, 99.90%, 99.96%) and CNN2D–CNN3D (99.97%, 99.90%, 99.68%), demonstrating superior detection of malicious samples and overall classification reliability.



(a) Accuracy, Precision, Recall, and F1-Score **(b)** TNR, MCC, and NPV

Figure 3: Performance metrics of the proposed algorithms. (**a**) Accuracy, Precision, Recall, and F1-Score (**b**) TNR, MCC, and NPV

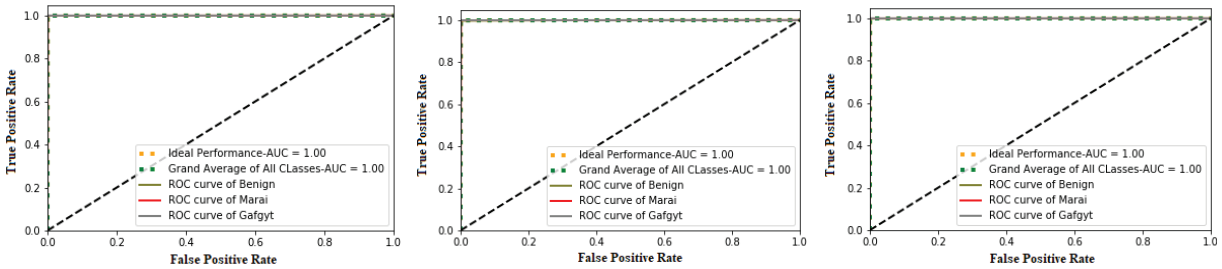
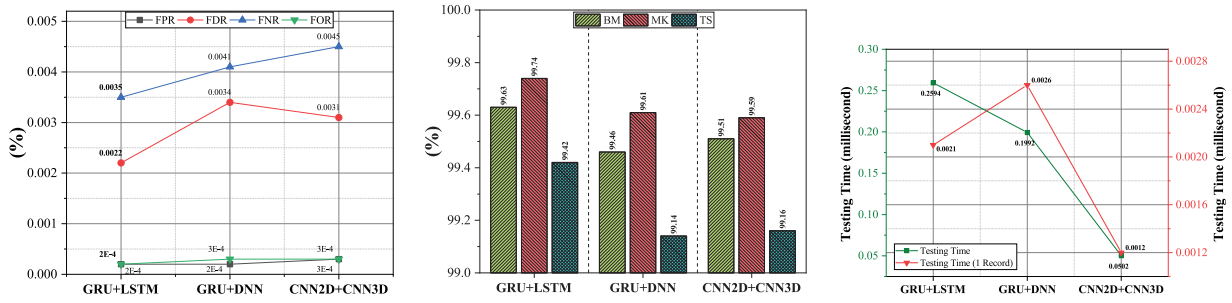


Figure 4: ROC-Curve of proposed GRU-LSTM, GRU-DNN and CNN2D-CNN3D

An effective model achieves low False Positive Rate (FPR), False Negative Rate (FNR), False Discovery Rate (FDR), and False Omission Rate (FOR). FPR indicates the proportion of benign samples misclassified as attacks, FNR the fraction of attacks misclassified as benign, FDR the proportion of false positives among predicted positives, and FOR the proportion of false negatives among predicted negatives. As shown in Fig. 5, the proposed GRU–LSTM achieved 0.0002% FPR, 0.0022% FNR, 0.0035% FDR, and 0.0002% FOR, outperforming GRU–DNN (0.0002%, 0.0041%, 0.0034%, 0.0003%) and CNN2D–CNN3D (0.0003%, 0.0045%, 0.0031%, 0.0003%), demonstrating high detection efficiency.



(a) FPR, FDR, FNR, FOR **(b)** BM, MK, TS **(c)** Testing Time

Figure 5: Performance metrics of the proposed hybrid algorithms. (**a**) FPR, FDR, FNR, FOR (**b**) BM, MK, TS (**c**) testing time

Global performance measures, including Bookmaker Informedness (BM), Markedness (MK), and Threat Score (TS), further validate model performance. As depicted in Fig. 5, GRU–LSTM achieved 99.63% BM, 99.74% MK, and 99.42% TS, outperforming GRU–DNN (99.46%, 99.61%, 99.14%) and CNN2D–CNN3D (99.50%, 99.59%, 99.16%), confirming its superior discriminative power and overall efficiency.

The testing time of the proposed algorithms is shown in Fig. 5. Single-sample inference required 0.0021 ms for GRU–LSTM, 0.0026 ms for GRU–DNN, and 0.0012 ms for CNN2D–CNN3D, measured using optimized TensorFlow GPU execution on a NVIDIA GTX 1050 Ti. These times correspond to single-sample inference and may vary for larger batches or full-pipeline evaluation. Table 7 provides a comparison of our work with the current state-of-the-art in terms of detection accuracy and time efficiency. This table clearly states the difference between existing work and our work by providing significant details that are adequate for comparison.

Table 7: The comparison of our results with others

Parameters	Proposed work	Meidan [29]	McDermott [13]	Parra [31]
Dataset	N_BaIoT	N_BaIoT	Self Generated	N_BaIoT
Algorithm	Hybrid GRU-LSTM	Autoencoder	BLSTM-RNN	LSTM, DCNN
Binary_class	–	–	–	–
Multi_class	✓	✓	✓	✓
10-fold	✓	–	–	✓
Accuracy	99.96	80	98.33	94.80
Precision	99.77	98.80	86.65	90.30
Recall	99.64	98.92	99.00	99.87
F1-score	99.64	98.92	99.00	97.28
FPR	0.0002	0.03	–	0.0001
Testing time	0.2594 (ms)	–	–	–

Note: ✓ denotes a 'Yes' response

6 Conclusion

The continued growth of IoT systems faces significant challenges due to the increasing sophistication and advancement of cyber attacks. Advanced malware-driven botnets have the potential to severely disrupt or even paralyze IoT ecosystems, while also serving as platforms for the propagation of additional malware across compromised networks. These threats underscore the urgent need for robust, adaptive, and intelligent

security mechanisms to safeguard the integrity, availability, and privacy of IoT infrastructures. The authors present an efficient and highly effective hybrid DL framework to accurately identify multi-variant malware-driven botnet attacks in an IoT ecosystem. Besides, we have evaluated the proof of concept of employing numerous feature selection mechanisms that also contribute to the overall performance without provoking any real-time cost. Moreover, the proposed mechanism has been thoroughly tested with current hybrid DL benchmarks. Further, the devised scheme has been cross validated using standard k-fold validation mechanism to explicitly show unbiased performance. Our proposed mechanism shows promising results in terms of detection accuracy with a trivial trade-off in time complexity. In future work, we plan to extend the model to large-scale, heterogeneous IoT networks and incorporate transformer-based and agentic AI frameworks for improved scalability, while evaluating it on diverse datasets to validate its effectiveness in detecting, mitigating, and securing emerging IoT ecosystems.

Acknowledgement: Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2025R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Funding Statement: This research was supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2025R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Author Contributions: Study conception and design: Maha Abdelhaq, Tooba Hasan, Adnan Akhunzada; Data collection: Maha Abdelhaq, Tooba Hasan; Analysis and interpretation of results: Maha Abdelhaq, Tooba Hasan, Adnan Akhunzada; Draft manuscript preparation: Maha Abdelhaq; Supervision, review, and editing: Adnan Akhunzada; Validation: Ahmad Sami Al-Shamayleh, Nikola Ivković; Funding acquisition: Nikola Ivković, Maha Abdelhaq, Ahmad Sami Al-Shamayleh. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The dataset utilized in this study are publicly available and accessible at: https://research.unsw.edu.au/projects/bot-iot-dataset (accessed on 04 August 2025).

Ethics Approval: Not applicable.

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

Supplementary Materials: The supplementary material is available online at https://www.techscience.com/doi/10.32604/cmc.2025.073577/s1.

References

1. Saadouni C, Jaouhari SE, Tamani N, Ziti S, Mroueh L, Bouchti KE. Identification techniques in the internet of things: survey, taxonomy and research frontier. IEEE Communicat Surv Tutor. 2025. doi:10.1109/COMST.2025.3541165.
2. Beltrán-López P, Pérez MG, Nespoli P. Cyber deception: taxonomy, state of the art, frameworks, trends, and open challenges. IEEE Communicat Surv Tutor. 2025. doi:10.1109/COMST.2025.3594788.
3. Garg U, Mishra P, Gupta N, Pilli ES. IoT botnets unveiled: architectural analysis, threat vectors, and cutting-edge detection techniques. Cluster Comput. 2025;28(15):945. doi:10.1007/s10586-025-05633-1.
4. Maaz M, Ahmed G, Sami Al-Shamayleh A, Akhunzada A, Siddiqui S, Hussein Al-Ghushami A. Empowering IoT resilience: hybrid deep learning techniques for enhanced security. IEEE Access. 2024;12:180597–618. doi:10.1109/access.2024.3482005.
5. Farooq MJ, Zhu Q. Modeling, analysis, and mitigation of dynamic botnet formation in wireless IoT networks. IEEE Transact Inform Foren Secur. 2019;14(9):2412–26. doi:10.1109/tifs.2019.2898817.
6. Shao S, Gu T, Nie Y, Ji Z, Wu F, Ba Z, et al. An active defense adjudication method based on adaptive anomaly sensing for mimic IoT. IEEE Transact Serv Comput. 2025;18(1):57–71. doi:10.1109/tsc.2024.3436673.

7. Taheri R, Shojafar M, Arabikhan F, Gegov A. Unveiling vulnerabilities in deep learning-based malware detection: differential privacy driven adversarial attacks. Comput Secur. 2024;146:104035. doi:10.1016/j.cose.2024.104035.
8. Eslamnejad M, Taheri R, Shojafar M, Bader-El-Den M. Federated learning-based robust android malware detection: label-flipping attacks and defenses. Neural Comput Appl. 2025;37(32):27057. doi:10.1007/s00521-025-11656-x.
9. Akhunzada A, Al-Shamayleh AS, Zeadally S, Almogren A, Abu-Shareha AA. Design and performance of an AI-enabled threat intelligence framework for IoT-enabled autonomous vehicles. Comput Elect Eng. 2024;119(1):109609. doi:10.1016/j.compeleceng.2024.109609.
10. Al-Shurbaji T, Anbar M, Manickam S, Hasbullah IH, Alfriehat N, Alabsi BA, et al. Deep learning-based intrusion detection system for detecting IoT botnet attacks: a review. IEEE Access. 2025;13:11792–822. doi:10.1109/access.2025.3526711.
11. Benson T, Chandrasekaran B. Sounding the bell for improving internet (of things) security. In: Proceedings of the 2017 Workshop on Internet of Things Security and Privacy. New York, NY, USA: ACM; 2017. p. 77–82.
12. Torres P, Catania C, Garcia S, Garino CG. An analysis of recurrent neural networks for botnet detection behavior. In: 2016 IEEE biennial congress of Argentina (ARGENCON). Piscataway, NJ, USA: IEEE; 2016. p. 1–6. doi:10.1109/argencon.2016.7585247.
13. McDermott CD, Majdani F, Petrovski AV. Botnet detection in the internet of things using deep learning approaches. In: 2018 International Joint Conference on Neural Networks (IJCNN). Piscataway, NJ, USA: IEEE; 2018. p. 1–8.
14. Pektaş A, Acarman T. Botnet detection based on network flow summary and deep learning. Int J Netw Manag. 2018;28(6):e2039. doi:10.1002/nem.2039.
15. Kudugunta S, Ferrara E. Deep neural networks for bot detection. Inform Sci. 2018;467:312–22.
16. Vinayakumar R, Soman K, Poornachandran P, Sachin Kumar S. Evaluating deep learning approaches to characterize and classify the DGAs at scale. J Intell Fuzzy Syst. 2018;34(3):1265–76.
17. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436. doi:10.1038/nature14539.
18. Chaganti R, Suliman W, Ravi V, Dua A. Deep learning approach for SDN-enabled intrusion detection system in IoT networks. Information. 2023;14(1):41. doi:10.3390/info14010041.
19. Sattari F, Farooqi AH, Qadir Z, Raza B, Nazari H, Almutiry M. A hybrid deep learning approach for bottleneck detection in IoT. IEEE Access. 2022;10:77039–53. doi:10.1109/access.2022.3188635.
20. Diro AA, Chilamkurti N. Distributed attack detection scheme using deep learning approach for Internet of Things. Future Generat Comput Syst. 2018;82:761–8. doi:10.1016/j.future.2017.08.043.
21. Wani A, Khaliq R. SDN-based intrusion detection system for IoT using deep learning classifier (IDSIoT-SDL). CAAI Transact Intell Technol. 2021;6(3):281–90. doi:10.1049/cit2.12003.
22. Hasan T, Akhunzada A, Giannetsos T, Malik J. Orchestrating SDN control plane towards enhanced IoT security. In: 2020 6th IEEE Conference on Network Softwarization (NetSoft). Piscataway, NJ, USA: IEEE; 2020. p. 457–64.
23. Chung J, Gulcehre C, Cho K, Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555. 2014.
24. Dey R, Salemt FM. Gate-variants of gated recurrent unit (GRU) neural networks. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS). Piscataway, NJ, USA: IEEE; 2017. p. 1597–600.
25. Kostadinov S. Understanding GRU Networks [Internet]. 2017 [cited 2025 Nov 1]. Available from: https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be.
26. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997;9(8):1735–80. doi:10.1162/neco.1997.9.8.1735.
27. Jacovi A, Shalom OS, Goldberg Y. Understanding convolutional neural networks for text classification. arXiv:1809.08037. 2018.
28. Montavon G, Samek W, Müller KR. Methods for interpreting and understanding deep neural networks. Digital Signal Process. 2018;73:1–15. doi:10.1016/j.dsp.2017.10.011.

29. Meidan Y, Bohadana M, Mathov Y, Mirsky Y, Shabtai A, Breitenbacher D, et al. N-BaIoT—network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervas Comput. 2018;17(3):12–22. doi:10.1109/mprv.2018.03367731.
30. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The WEKA data mining software: an update. ACM SIGKDD Explorat Newsletter. 2009;11(1):10–8. doi:10.1145/1656274.1656278.
31. Parra GDLT, Rad P, Choo KKR, Beebe N. Detecting Internet of Things attacks using distributed deep learning. J Netw Comput Appl. 2020;163:102662. doi:10.1016/j.jnca.2020.102662.