

Article

Hybrid CNN–BiLSTM–DNN Approach for Detecting Cybersecurity Threats in IoT Networks

Bright Agbor Agbor^{1,†} , Bliss Utibe-Abasi Stephen^{1,*,†}, Philip Asuquo^{1,†}, Uduak Onofiok Luke^{2,†} and Victor Anaga^{1,†}

¹ Department of Computer Engineering, University of Uyo, Uyo 520101, Nigeria; philipasuquo@uniuyo.edu.ng (P.A.)

² School of Sustainability, Civil and Environmental Engineering, University of Surrey, Guildford GU2 7XH, UK; u.luke@surrey.ac.uk

* Correspondence: blissstephen@uniuyo.edu.ng; Tel.: +234-813-485-2136

† These authors contributed equally to this work.

Abstract: The Internet of Things (IoT) ecosystem is rapidly expanding. It is driven by continuous innovation but accompanied by increasingly sophisticated cybersecurity threats. Protecting IoT devices from these emerging vulnerabilities has become a critical priority. This study addresses the limitations of existing IoT threat detection methods, which often struggle with the dynamic nature of IoT environments and the growing complexity of cyberattacks. To overcome these challenges, a novel hybrid architecture combining Convolutional Neural Networks (CNN), Bidirectional Long Short-Term Memory (BiLSTM), and Deep Neural Networks (DNN) is proposed for accurate and efficient IoT threat detection. The model's performance is evaluated using the IoT-23 and Edge-IIoTset datasets, which encompass over ten distinct attack types. The proposed framework achieves a remarkable 99% accuracy on both datasets, outperforming existing state-of-the-art IoT cybersecurity solutions. Advanced optimization techniques, including model pruning and quantization, are applied to enhance deployment efficiency in resource-constrained IoT environments. The results highlight the model's robustness and its adaptability to diverse IoT scenarios, which address key limitations of prior approaches. This research provides a robust and efficient solution for IoT threat detection, establishing a foundation for advancing IoT security and addressing the evolving landscape of cyber threats while driving future innovations in the field.

Keywords: Internet of Things; deep learning; cybersecurity threats; CNN; BiLSTM



Academic Editor: Zahid Akhtar

Received: 1 January 2025

Revised: 20 January 2025

Accepted: 24 January 2025

Published: 10 February 2025

Citation: Agbor, B.A.; Stephen, B.U.-A.; Asuquo, P.; Luke, U.O.; Anaga, V. Hybrid CNN–BiLSTM–DNN Approach for Detecting Cybersecurity Threats in IoT Networks. *Computers* **2025**, *14*, 58. <https://doi.org/10.3390/computers14020058>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The rapid increase in devices on the Internet of Things (IoT) has brought significant transformation to the industries by enhancing efficiency and enabling autonomous operations. The ability of the IoT to interconnect heterogeneous nodes, including people, devices, and infrastructure, has made daily life simpler, safer, and more productive, driving its adoption across diverse applications [1]. However, this widespread integration has also introduced significant cybersecurity concerns, which require robust actions to address emerging threats and vulnerabilities.

IoT networks integrate sensors, embedded computing, and communication technologies [2] and have demonstrated transformative potential across diverse fields. Its applications extend to hazardous industries, such as mining [3], as well as smart agriculture [4–6] and smart cities [7,8]. During the time of the COVID-19 outbreak, the IoT played

a pivotal role in health management by using interconnected devices to establish efficient and responsive healthcare networks [9]. By simplifying processes and improving efficiency, IoT technologies have improved life quality [10]. This impact is evident in areas such as predictive maintenance, supply chain optimization, livestock monitoring, energy management, smart grids, retail and e-commerce, wearable technology, finance and banking, defense and security, and IoT-based multimedia applications.

The rapid increase in connected devices, which is projected to exceed 29 billion by 2030 [11], has further amplified the vulnerabilities of IoT ecosystems to evolving cyber threats [12,13]. Key challenges in IoT ecosystems include quality of service, security requirements, proper management of ongoing services, accurate identification of services, information privacy, data confidentiality, and data encryption. These vulnerabilities expose IoT systems to significant security attacks, resulting in financial and reputational losses. Furthermore, as IoT devices generate and transmit vast amounts of data through sensors, actuators, transceivers, and wearables, these data become highly susceptible to threats and risks [14]. Addressing these pressing challenges requires the development of a robust security mechanism for combating attacks, mitigating vulnerabilities, and ensuring privacy and safety in IoT environments.

Despite significant advancements in the application of machine learning (ML) and deep learning (DL) for secure IoT, critical gaps remain unaddressed. A comprehensive review [13] of primary studies made public between 2018 and 2023 highlights that ML and DL approaches outperform traditional signature-based methods in the detection of IoT attacks. However, their effectiveness and reliability vary widely depending on factors such as the dataset used, the features selected, the algorithms used, and the evaluation metrics. Furthermore, developing resource-efficient models for IoT devices with constrained computational resources has not been a primary focus in many studies, thus resulting in a crucial need for effectively balancing detection accuracy with resource constraints.

The growing complexity and frequency of cybersecurity threats in IoT networks necessitate the development of more efficient and robust detection mechanisms. This paper proposes a novel security framework designed to detect IoT attacks by using a deep learning-based hybrid model. The objective is to address existing research gaps by providing an effective solution for identifying malicious devices in various attack scenarios. The proposed model uses a convolutional neural network (CNN) [15] to extract high-level feature representations from IoT traffic data. These features are then processed by a Bidirectional Long Short-Term Memory (BiLSTM) network [16–18], which identifies temporal dependencies in both directions, boosting the model's ability to recognize intricate attack patterns. Finally, a Deep Neural Network (DNN) layer is used for binary or multi-class classification of the processed output from the BiLSTM [19]. The proposed mechanism is evaluated on standard datasets, including IoT-23 and Edge-IIoTset, ensuring the model's robustness and generalizability in practical IoT settings. These widely used datasets have been employed in state-of-the-art works, including those in [20–23].

This study contributes to IoT cybersecurity by developing a novel hybrid neural network framework with enhanced detection capabilities. The key contributions are as follows:

- The proposed model integrates CNN, BiLSTM, and DNN into a hybrid deep learning architecture for improved detection.
- A dynamic analysis of common IoT attacks, which uses two different datasets to assess model performance across various scenarios.
- A comparative evaluation of the proposed model against other contemporary research studies for highlighting its advantages.

- Focus on developing a lightweight model optimized for real-time operation in IoT devices with constrained capabilities.

This paper is organized as follows: Section 2 provides a thorough examination of current research in IoT security, highlighting significant studies and existing strategies for mitigating attacks within IoT environments. Section 5 presents the results and offers an empirical evaluation of the proposed detection scheme. Following this, Section 6 provides a detailed discussion, comparing our findings with other contemporary research studies in the field of IoT cybersecurity. Finally, Section 7 summarizes the findings and recommends areas for future research, followed by the references section.

2. Related Work

The integration of IoT devices across various industries has significantly enhanced operational efficiency but has also introduced complex security challenges. Among these, cyber threats targeting IoT devices, including orchestrating compromised devices to launch large-scale cyberattacks, pose substantial threats to IoT ecosystems. These attacks exploit vulnerabilities in interconnected devices, necessitating advanced detection and mitigation strategies to ensure system security [12,13].

Conventional methods for IoT attack detection predominantly rely on signature-based intrusion detection systems (IDS). These approaches effectively identify known attack patterns but exhibit significant limitations in addressing the dynamic and evolving nature of IoT threats. As emphasized by Liao [14] and Zaman [1], the inability of signature-based systems to recognize zero-day attacks undermines their utility in modern IoT networks. Among traditional approaches, Zohourian et al. [24] proposed IoT-PRIDS, a lightweight, non-machine learning-based IDS. IoT-PRIDS employs packet header representations to identify abnormal network behaviors, eliminating the need for labeled datasets or high computational resources. This makes it ideal for IoT devices with limited resources. However, it faces challenges related to scalability, adaptability to advanced attack vectors, and a high rate of false positives in heterogeneous IoT environments. To overcome the limitations of traditional methods, ML techniques have emerged as promising solutions for cyberattack detection in IoT ecosystems. ML models can generalize from large datasets to identify novel attack patterns. Various approaches, such as supervised, unsupervised, and reinforcement learning, have been explored in this context. For instance, studies like Sahu et al. [25] introduced ensemble-based ML frameworks incorporating dimensionality reduction and feature engineering techniques to optimize detection accuracy. Similarly, Abdulkareem et al. [26] proposed the FI-SEL framework, leveraging Extremely Randomized Trees (ETC) for feature importance analysis, reducing data dimensionality by 86.9%. The framework employs stacking ensemble classifiers to achieve efficient classification with a balance between precision and recall, demonstrating superior performance on the Edge-IIoTset dataset. These approaches showcase ML's potential in optimizing resource utilization without compromising detection accuracy. Deep learning (DL) techniques, including CNNs, LSTMs, and hybrid models such as the CNN-LSTM used by Riaz et al. [27], Halbouni et al. [28], Altunay et al. [29], have demonstrated remarkable effectiveness in IoT security. Their ability to learn complex temporal and spatial patterns in traffic data enhances their detection capabilities.

The authors in [30] proposed a hybrid CNN-LSTM framework that utilized principal component analysis (PCA) for dimensionality reduction. Advanced optimization techniques, such as quantization and pruning, are integrated into the framework to facilitate efficient deployment in IoT environments with limited computational capacity, achieving high scalability and detection precision. Similarly, Sahu et al. [25] applied CNNs for extracting features and LSTMs for classification, achieving a detection accuracy of 96%

on data from Raspberry Pi-infected IoT devices. Bhandari et al. [31] introduced a DNN-based framework for malware detection, which demonstrated a detection accuracy of 93% and an F1-score of 92%, validating its feasibility for real-time implementation. Studies in IIoT environments further highlight the robustness of hybrid architectures. For example, Yazdinejad et al. [32] and Bukhari et al. [33] developed models combining CNN, LSTM, and GRU to achieve 100% accuracy in anomaly detection. Additionally, the integration of asynchronous federated learning ensures privacy and supports decentralized model training, making these solutions particularly suitable for large-scale IIoT systems.

The application of generative models has introduced new dimensions to IoT security. Abdalgawad et al. [34] explored Adversarial Autoencoders (AAE) and Bidirectional Generative Adversarial Networks (BiGAN) for zero-day attack detection. These models achieved an impressive F1-score of 0.99 on the IoT-23 dataset and demonstrated high efficacy in handling novel threats. Likewise, Zixu et al. [35] proposed a hierarchical framework for anomaly detection integrating Generative Adversarial Networks (GANs) and Autoencoders (AEs). By addressing data scarcity and privacy concerns, a sampling pool is reconstructed by the system at a central controller while training a global AE model. The evaluation using the UNSW Bot-IoT dataset [36] showcased its superior performance over conventional models, underscoring the effectiveness of combining GANs and AEs for distributed IoT ecosystems. Effective feature selection and handling class imbalance are critical for robust IoT intrusion detection. Alani and Miri [37] addressed these challenges using datasets such as TON_IoT [38] and IoT-23 [39], developing a universal feature selection method. This approach reduced prediction time by 70% while achieving an accuracy of 99.62%, demonstrating significant improvements in detection efficiency. In tackling class imbalance, Alfares and Banimelhem [40] used techniques such as Adaptive Synthetic Sampling (ADASYN), Synthetic Minority Oversampling (SMOTE), cost-sensitive learning, and the bagging technique. Similarly, Thiyam et al. [41] employed SMOTE and Tomek techniques for resampling, combined with a random forest-based feature shuffle method to identify the most impactful features. The resultant models enhanced detection performance, improving metrics like MCC and ROC values, and showcased resilience in imbalanced data scenarios. Recent advancements in specialized datasets, like Edge-IIoTset [42], CIC-DDoS2019 [43], and CIC-IoT22, have further propelled the development of high-performing IDS solutions. Studies like Ghenni et al. [44] and Mahdi et al. [45] leveraged these datasets to construct models achieving accuracies exceeding 99% with detection times under 0.3 s. The work by Bakhsh et al. [46] introduced a deep learning-based IDS, assessed using the CIC-IoT22 dataset, where models like FFNN, LSTM, and RandNN demonstrated high accuracy, with FFNN achieving 99.93%. These results highlight the utility of modern datasets in developing scalable and accurate IDS solutions tailored for IoT environments. The growing sophistication of IoT cyber threats demands robust, scalable, and efficient intrusion detection frameworks. This review underscores the evolution from traditional systems to advanced ML and DL techniques, showcasing their adaptability and potential to secure IoT ecosystems. From the reviewed literature, despite advancements in ML and DL for IoT attack detection, significant challenges persist. Many methods struggle to balance real-time processing with resource constraints and often lack transparency in decision-making. Reliance on labeled data and limited dataset diversity further impede the development of robust, universal solutions. While emerging techniques like hybrid and generative models show promise, they remain unoptimized for practical adoption. Additionally, comprehensive integration of ML and DL approaches is rare, leaving a critical gap within the field.

This investigation focuses on addressing this gap by offering an integrated neural-network-based framework, validated on IoT-23 and Edge-IIoTset datasets, to advance IoT

threat detection. By leveraging insights from the reviewed literature, we propose a robust, adaptable solution designed to counter evolving threats effectively. As IoT ecosystems grow, so do their vulnerabilities; our work seeks to ensure secure and resilient architectures. A detailed summary of existing literature is presented in Table 1.

Table 1. Summary of leading research in IoT ecosystem.

Year	Study	Solution Proposed	Architecture	Dataset
2021	Sahu et al. [25]	Internet of Things attack detection	CNN+LSTM	IoT-23
2022	Alani and Miri [37]	Explainable universal feature set for IoT intrusion detection		TON_IoT, IoT-23, & IoT-ID
2022	Riaz et al. [27]	Malware detection in Internet of Things (IoT) devices	CNN+LSTM	IoBT
2022	Abdalgawad et al. [34]	Generative deep learning for detecting cyberattacks	AAE & BiGAN	IoT-23
2023	Bhandari et al. [31]	Deep neural-network-based middleware for cyber-attack detection	DNN	IoT-23, EdgeIIoTset
2023	Yazdinejad et al. [32]	Ensemble deep learning model for cyber threat hunting	Deep LSTM AE	GP, SWaT
2023	Altunay et al. [29]	Deep learning-based intrusion detection system	CNN, LSTM, CNN+LSTM	UNSW-NB15, X-IIoTID
2024	Nazir et al. [30]	Deep learning-based threat detection system	CNN–LSTM	IoT-23, N-BaIoT, CICIDS2017
2024	Abdulkareem et al. [26]	Lightweight SEL for attack detection	FI-SEL	Edge-IIoTset
2024	Bukhari et al. [33]	An asynchronous federated learning approach with a deep hybrid detection model	Asynchronous FL–CNN–GRU–LSTM	Edge-IIoTset
2024	Gheni et al. [44]	Two-step data clustering for improved intrusion detection system	Clustering	CICIoT2023

3. Proposed Framework

This section presents a framework designed to protect IoT networks from various attacks. The core of this framework given in Figure 1 is a powerful combination of three neural network models: CNN, BiLSTM network, and a fully connected DNN. These models work together to create a robust CNN–BiLSTM–DNN model. The extracted features from the CNN model, as shown in Figure 2, will be input to the BiLSTM model for further processing.

The BiLSTM model’s output will then be sent to the DNN for final classification. If the DNN identifies the feature as malicious, the system will alert the monitor with the device’s

identity and address. This will enable the system to take preventive measures such as isolating the malicious device and its activities. In our proposed model, the BiLSTM model introduces an extra layer of temporal understanding to the feature extraction process, which helps to identify temporal patterns present in the data, which means it can learn and remember patterns that occur over time and also allows the model to enhance the identification and response to evolving threats. This helps the DNN make more accurate classifications, especially when dealing with malicious activities that involve a series of events.

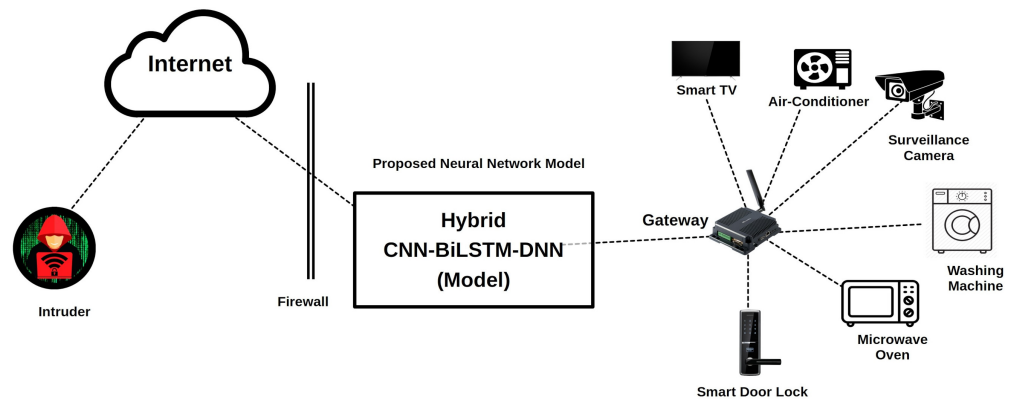


Figure 1. Proposed architecture.

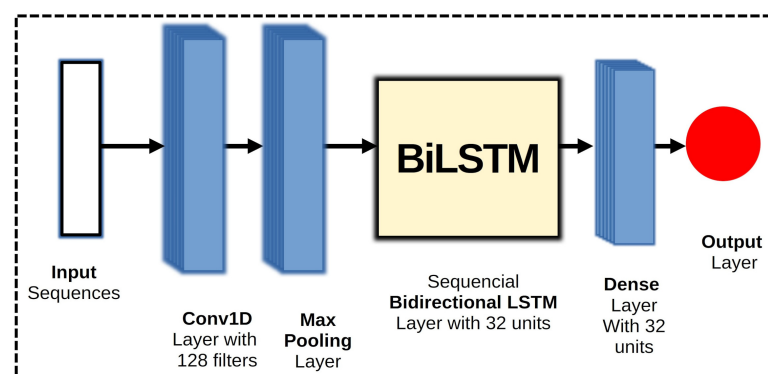


Figure 2. Hybrid model structure.

The BiLSTM model's output will then be sent to the DNN for final classification. If the DNN identifies the feature as malicious, the system will alert the monitor with the device's identity and address. This will enable the system to take preventive measures, such as isolating the malicious device and its activities. In our proposed model, the BiLSTM model adds a layer of temporal understanding to the feature extraction process, which helps to capture temporal dependencies within the data, which means it can learn and remember patterns that occur over time and also allows the model to improve the identification and response to new threats. This helps the DNN make more accurate classifications, especially when dealing with malicious activities that involve a series of events.

3.1. Dataset Preparation

In this research, the IoT-23 dataset from Aposemat, as detailed in Garcia et al. [39], and the Edge-IIoTset Ferrag et al. [42] dataset were used to experimentally evaluate the proposed framework. The IoT-23 dataset contains 20 malware scenarios captured from malicious devices and benign scenarios that captured the network traffic from a Philips HUE smart LED lamp, an Amazon Echo smart assistant, and a Somfy smart door lock, all categorized as IoT devices. The IoT's network traffic in this dataset was captured in

the Stratosphere Laboratory, AIC group, FEL, CTU University, Czech Republic. The Edge-IIoTset dataset, available at [<http://ieee-dataport.org/8939>] (access on 1 January 2025), was developed through a dedicated IoT/IIoT testbed designed to emulate a comprehensive and varied environment. This testbed features a wide range of devices, protocols, sensors, and cloud/edge configurations. Over ten kinds of IoT devices were used to collect data, encompassing cost-effective digital sensors for humidity and temperature monitoring, ultrasonic sensors, pH meters, water level detectors, heart rate monitors, soil moisture sensors, and flame sensors, among others.

3.2. Input Layer

This layer takes raw data and prepares it for processing by subsequent layers. The preprocessed IoT network traffic data received at this layer is structured into time-series sequences representing extracted network traffic features such as packet size, flow duration, and protocol types. Each sequence has a fixed length determined by the number of attributes.

The model's input is represented as a time-series matrix of the following shape:

$$X \in \mathbb{R}^{N \times T \times F} \quad (1)$$

where sample size is represented by N , T is the time steps in each sequence, and F represents the number of features (e.g., packet size, flow duration, protocol types). Each input sequence is represented as follows:

$$X_i = \{x_{i1}, x_{i2}, \dots, x_{iT}\}, \quad x_{it} \in \mathbb{R}^F \quad (2)$$

3.3. 1D Convolution Layer

The convolutional layer is used for extracting spatial features from localized areas of the input data. A single convolutional layer is used in this study. In the Conv1D layer, 128 filters with a kernel size of 5 and ReLU activation are used. The ReLU activation function added at the Conv1D layers helps to introduce non-linearity for deciphering complex data patterns. For a given input X , the output of the Conv1D layer after applying the ReLU activation function is given by the following:

$$Y_{i,j,k} = \text{ReLU} \left(\sum_{l=0}^{F-1} X_{i,j+l} \cdot W_{l,k} + b_k \right) \quad (3)$$

where $Y_{i,j,k}$ is the output at batch index i , position j , and filter k , $\text{ReLU}(z) = \max(0, z)$ is the activation function, applied element-wise to introduce non-linearity, $\sum_{l=0}^{F-1}$ summation over the filter size F , where l indexes the filter positions, $X_{i,j+l}$, the input value at batch i , position $j+l$ of the sequence, $W_{l,k}$, the weight at the l -th position of the k -th filter and b_k is the bias term for the k -th filter.

3.4. Pooling Layers

A pooling layer is employed after the Conv1D layer to downsample the feature maps generated from the Conv1D layers. The inclusion of pooling layers helps to reduce the spatial dimensionality while retaining the most significant features. In this study, MaxPooling1D layers have been used to reduce the dimension of the feature maps, prevent overfitting, and further improve the computational efficiency of the model.

3.5. Bilstm Layer

The BiLSTM in this model, with 32 units, processes the data sequence in both forward and reverse paths, capturing temporal dependencies from both past and future data. This bidirectional processing further strengthens the ability of the model to identify complex

patterns and anomalies, notably in IoT intrusion detection, where understanding both past and future events is critical for precise detection. As shown in Figure 3, the BiLSTM model consists of two LSTM units. One unit processes the input sequence forward, while the other processes it backward. The ‘return_sequences = False’ parameter ensures that only the output from the final time step is passed to the next layer, allowing for more concise feature extraction. The process is mathematically represented by the following equations:

$$\mathbf{z}_t^{(f)} = \text{LSTM}^{(f)}(\mathbf{x}_t, \mathbf{z}_{t-1}^{(f)}, \mathbf{q}_{t-1}^{(f)}) \quad (4)$$

$$\mathbf{z}_t^{(b)} = \text{LSTM}^{(b)}(\mathbf{x}_t, \mathbf{z}_{t+1}^{(b)}, \mathbf{q}_{t+1}^{(b)}) \quad (5)$$

$$\mathbf{z}_t = [\mathbf{z}_t^{(f)}; \mathbf{z}_t^{(b)}] \quad (6)$$

where $\mathbf{z}_t^{(f)}$ and $\mathbf{z}_t^{(b)}$ denote the hidden states in both forward and backward paths at a specific time step t , respectively. $\mathbf{q}_{t-1}^{(f)}$ and $\mathbf{q}_{t+1}^{(b)}$ denote the forward and backward cell states at a specific time step t , respectively. $\mathbf{x}_t$ denotes the input vector at a specific time step t , and $\mathbf{z}_t$ denotes the combination of the forward and backward hidden states into a single vector.

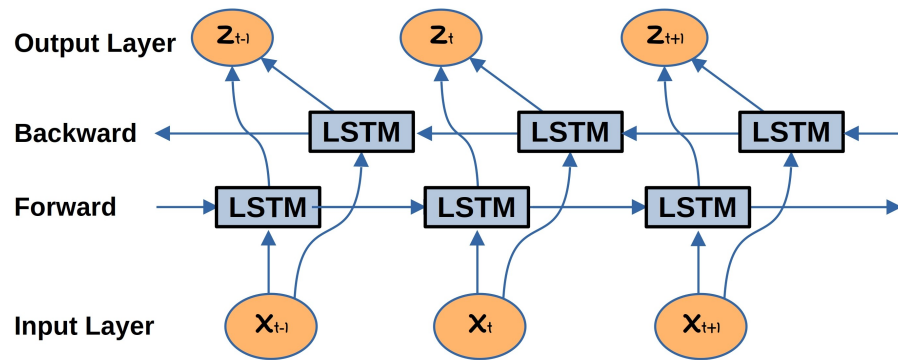


Figure 3. BiLSTM architecture.

3.6. Dense Layer

The fully connected (dense) layers process the spatial and temporal features learned by the CNN and BiLSTM layers. In our proposed model, a dense layer with 32 units and ReLU activation is used to transform features into a higher-dimensional space for decision-making.

3.7. Output Layer

The final dense layer in the model comprises a single neuron equipped with a sigmoid activation function. This layer outputs a probability score, which estimates the likelihood that the input sequence belongs to the malicious (class 1) or benign (class 0) category. Mathematically, the sigmoid function is given in Equation (7).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (7)$$

For multi-class classification tasks, the softmax activation function is used. The neurons in this layer correspond to the number of classes (i.e., the different types of attacks considered). The softmax function is given in Equation (8):

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (8)$$

where z_i is the output of the i -th neuron, and the total number of classes is denoted by K .

3.8. Data Flow Within Framework

The data flow within the proposed framework follows a multi-stage process. The flow starts by feeding the IoT network traffic data into the input layer, followed by using the 1D convolutional layer to extract localized spatial patterns such as abnormal flow behaviors. This step is followed by the downsampling of feature maps, ensuring the retention of key information and improved efficiency. Flattened features are passed to the BiLSTM layer, which analyzes sequential dependencies in both forward and backward paths to detect temporal patterns (e.g., repeated attack sequences). Further, the Dense layers consolidate spatial and temporal features into high-level representations, which leads to the output layer, which further produces a binary classification (malicious or benign) or multi-class classification.

4. Methods

This research focuses on a thorough examination of traffic patterns in data generated by IoT devices, leveraging advanced deep learning methodologies to detect and classify threats within IoT ecosystems. With the increase in IoT devices, there has been a corresponding surge in malicious activities, requiring strong and adaptable security frameworks. The IoT-23 and Edge-IIoTset datasets serve as the foundation for both experimental analysis and evaluation of the proposed model, which is designed to uncover concealed malicious signatures within IoT network traffic.

The methodology, outlined in the flowchart in Figure 4, incorporates a systematic approach encompassing data preprocessing, model training, and performance evaluation. The datasets are first acquired and preprocessed before being divided into training, validation, and testing subsets. Deep learning models are iteratively trained and optimized with the training and validation sets. The final trained model is employed to classify the test data into two categories: *Malicious* and *Benign*. This rigorous pipeline ensures the proposed framework delivers actionable insights and demonstrates its efficacy in enhancing real-world IoT security. Algorithm 1 gives a comprehensive explanation of the methods used in the hybrid CNN-BiLSTM-DNN model.

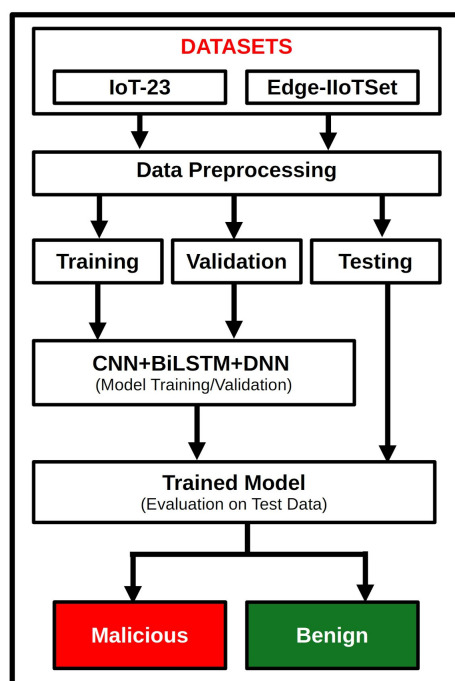


Figure 4. Binary classification flowchart.

Algorithm 1: Proposed Hybrid CNN–BiLSTM–DNN Model**Data:** IoT data sequences X **Result:** Trained model parameters Φ **Initialization:** Define model architecture with the following components:

- **Conv1D:** Apply 1D convolution with 128 filters, kernel size 3, with ReLU.
- **MaxPooling1D:** Pool with size 2 to reduce dimensions.
- **Dropout:** Apply dropout with rate 0.1.
- **Conv1D:** Apply 1D convolution with 64 filters, kernel size 3, with ReLU.
- **MaxPooling1D:** Pool with size 2 to reduce dimensions.
- **Dropout:** Apply dropout with rate 0.1.
- **BatchNormalization:** Apply batch normalization.
- **Bidirectional LSTM:** Process features with a BiLSTM layer with 32 units and L2 regularization (0.01).
- **Dropout:** Apply dropout with rate 0.25.
- **BatchNormalization:** Apply batch normalization.
- **Dense Layers:** Introduce a fully connected layer with 32 neurons using ReLU activation and L2 regularization (0.01), followed by Dropout with a rate of 0.25, BatchNormalization, and an output layer with one neuron using the sigmoid activation function and L2 regularization (0.01).

while Training epochs $i = 1$ to E **do** **Forward Pass:**

1. Extract features using Conv1D and MaxPooling:
 $F_{\text{CNN}} = \text{Conv1D} + \text{MaxPooling1D}$
2. Process features using BiLSTM: $F_{\text{BiLSTM}} = \text{BiLSTM}(F_{\text{CNN}})$
3. Compute Dense layer outputs: $F_{\text{DNN}} = \text{Dense}(F_{\text{BiLSTM}})$

Loss Calculation: Compute binary cross-entropy loss L . **Backpropagation:**

- Compute gradients: $\nabla L = \text{Gradient}(L, \Phi)$.
- Update parameters: $\Phi \leftarrow \Phi - \eta \nabla L$, where η denotes the learning rate.

end**Testing Phase:** Process test data using the trained model and classify predictions based on the sigmoid activation threshold.**return** Trained model parameters Φ

4.1. Dataset Statistics

Datasets are integral to research and provide the foundational data required for analysis, hypothesis testing, model development, and pattern recognition. For this study, the IoT-23 and the Edge-IIoTset dataset serve as the foundation for both experimental analysis and validation of the proposed framework, which is designed to uncover concealed malicious signatures within IoT network traffic.

4.1.1. Iot-23 Dataset

The IoT-23 dataset Garcia et al. [39] consists of different network traffic data collected: twenty malware-compromised Raspberry Pi devices and three benign IoT devices. The dataset categorizes traffic into ten distinct labels corresponding to nine attack scenarios and one benign scenario. The Benign label denotes the absence of any suspicious or malicious activities within a connection, while the nine malicious labels represent various attack types, including Attack (encompassing brute force, command injection, and other methods), C&C, DDoS, FileDownload, HeartBeat, Mirai, Okiru, PartOfAHorizontalPortScan, and Torii. A detailed examination of the dataset reveals significant class imbalances. The three most prevalent malicious labels are PartOfAHorizontalPortScan (213,852,924 flows), Okiru (47,381,241 flows), and DDoS (19,538,713 flows).

4.1.2. Edge-Iiotset Dataset

The Edge-IIoTset dataset, introduced by [42], serves as an extensive and realistic cybersecurity dataset for IoT and IIoT applications, termed Edge-IIoTset. It simulates different types of cyberattacks, divided into five primary categories: (1) DoS/DDoS; (2) Malware; (3) Injection; (4) Man-in-the-Middle (MITM); (5) Information Gathering.

This dataset was utilized in our study to develop a system capable of classifying network traffic into two categories: *benign* and *malicious* using neural networks. For simplicity, all attack samples were grouped under the “Malicious” class, while normal traffic was labeled as “Benign”. Table 2 presents a comprehensive analysis of the sample distribution across the classes within the Edge-IIoTset dataset.

Table 2. Overview of data samples in Edge-IIoTset dataset classes.

Classes	No. of Samples
Benign traffic	500,000
DDoS TCP SYN Flood	267,327
Password attack	179,170
DDoS HTTP Flood	15,795
Port scanning	9987
Uploading attack	9516
SQL Injection	8755
Vulnerability scanner	5424
XSS	2352
Backdoor	900
Ransomware	639
OS Fingerprinting	135

4.2. Dataset Refinement and Preprocessing

We began by preprocessing the data, a critical foundation for any data-driven research that ensures precise and dependable analyses. We subjected the IoT-23 dataset to an extensive series of refinement procedures. This section outlines the comprehensive preprocessing steps applied to the IoT-23 and Edge-IIoTset datasets.

Standardization:

$$z = \frac{x - \mu}{\sigma} \quad (9)$$

with mean

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i \quad (10)$$

and standard deviation

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2} \quad (11)$$

4.2.1. Iot-23 Dataset Preprocessing

We preprocessed the IoT-23 dataset to prepare it for modeling. To address severe class imbalance, we combined 500,000 benign samples with 44,662 samples each from the *PartOfAHorizontalPortScan*, *Okiru*, and *DDoS* classes, along with low-frequency labels, as shown in Table 3. We converted the *id_orig_h* and *id_resp_h* features to integers, then floats, and replaced missing values in history with *unknown* while imputing other categorical features with their mean. We one-hot encoded the predictors, label-encoded and one-hot encoded the target variable for binary and multiclass classification, respectively, and split the dataset into training, validation, and testing sets (80%, 10%, 10%). Finally, we standardized all features to have a mean of 0 and a standard deviation of 1 using Equation (9).

Table 3. Overview of data samples in IoT-23 Dataset Classes.

Classes	No. of Samples
Benign	500,000
PartOfAHorizontalPortScan	144,662
Okiru	144,662
DDoS	144,662
C&C-HeartBeat	34,507
C&C	21,995
Attack	9398
C&C-FileDownload	53
C&C-Torii	30
FileDownload	18
C&C-HeartBeat-FileDownload	11
C&C-Mirai	2

4.2.2. Edge-Iiotset Dataset Preprocessing

The preprocessing phase for the Edge-IIoTset dataset involved several key steps. First, we converted the *Timestamp* feature into datetime format and derived *Day*, *Hour*, and *Time elapsed*. Next, we transformed the *Src_IP* and *Dst_IP* features into integer representations and removed redundant features such as *Timestamp* and *Flow ID*. We separated the target variable from the predictors, using the *Attack_label* feature for binary classification, where benign traffic and malicious traffic were encoded as 0 and 1, respectively. For multiclass classification, we applied one-hot encoding to the *Attack_label*. The dataset was split into training, validation, and test sets in a two-step process: first, 80% of the data were allocated to training and 20% to a temporary set. Then, the temporary set was divided equally into test and validation sets, resulting in 80% of the main dataset for training, 10% for testing, and 10% for validation. Finally, we normalized the features using the *StandardScaler* class from Scikit-Learn.

4.3. Architecture of Hybrid Cnn–Bilstm–Dnn Model

In this study, we leveraged the strengths of CNN for capturing localized patterns in input sequences, combined with BiLSTMs, for processing the reduced sequence and capturing both forward and backward temporal dependencies. This hybrid model effectively merges the benefits of these two powerful networks. We extensively experimented with and evaluated the proposed model on two diverse datasets: IoT-23 and Edge-IIoTset. The intricate design of the hybrid framework proposed is detailed in Table 4. Throughout this section, we delve into the hybrid CNN–BiLSTM–DNN architecture, exploring its design and the rationale behind its selection for our analysis.

Table 4. Proposed hybrid model architecture with layer details.

Layer Type	Output Shape	Param #
Conv1D	(None, 35, 128)	512
MaxPooling1D	(None, 17, 128)	0
Dropout	(None, 17, 128)	0
Conv1D	(None, 15, 64)	24,640
MaxPooling1D	(None, 7, 64)	0
Dropout	(None, 7, 64)	0
BatchNormalization	(None, 7, 64)	256
Bidirectional	(None, 64)	24,832
Dropout	(None, 64)	0

Table 4. *Cont.*

Layer Type	Output Shape	Param #
BatchNormalization	(None, 64)	256
Dense	(None, 32)	2080
Dropout	(None, 32)	0
BatchNormalization	(None, 32)	128
Dense	(None, 1)	33

4.3.1. Cnn Layers

The CNN (Conv1D) layers are the starting point of the model, designed to capture localized patterns within the input sequence. As shown in Table 4, the first and second CNN layers apply 128 and 64 filters, respectively, with a filter width of 3 across the sequence. These layers identify temporal dependencies in the data. The ReLU activation is applied to each layer, introducing non-linearity and enabling the model to recognize more intricate patterns. These steps act as an automated feature extraction process, reducing the need for manual engineering.

4.3.2. Pooling Layers

The Pooling (MaxPooling1D) layers are strategically applied after each CNN layer to down-sample the input representation, retaining only the most significant activations within each window of the pooling operation. Within this architecture, the pooling layers help to effectively reduce the dimensionality and computational complexity of the data while preserving essential features.

4.3.3. Bilstm Layer

The BiLSTM layer with 32 units processes the compressed sequence by handling the input in both forward and reverse directions. By considering both past and future contexts, it captures bidirectional temporal dependencies, which are critical for tasks where the sequence context in both directions matters. The output of this layer, summarized into a fixed-size representation, encapsulates the temporal relationships learned.

4.3.4. Dnn Layer

The DNN layers in the model aggregate the features learned from the previous layers and map them to the binary target output. The first dense layer, consisting of 32 neurons with ReLU activation, refines the extracted features into higher-level abstractions. For the final output, a sigmoid activation is used for binary classification, and a softmax activation is used in the multi-class classification. In binary classification, the sigmoid activation function provides a single output value between 0 and 1, representing the chance that the input is in the positive class.

4.3.5. Dropout and Batchnormalization Layers

Dropout layers, as shown in Table 4, are applied after Conv1D, BiLSTM, and Dense layers to reduce overfitting by randomly deactivating neurons during training, while BatchNormalization layers are used to normalize activations, improving training stability and convergence speed of the model.

4.4. Flow of Data Within the Framework

The proposed framework employs a strategic approach designed for handling IoT security data. Each phase builds upon the previous one, adding layers of analysis and insight crucial for accurately identifying and categorizing threats. This progression of data, from its raw state to final classifications, encapsulates the core of our methodology. Table 5

provides a detailed overview of all stages involved in the comprehensive flow of data within the framework.

Table 5. Flow of data within the framework.

Phase	Overview	Next Phase
Data Preparation	Collecting and organizing raw data	Data Preprocessing
Data Preprocessing	Cleaning and normalizing data	Input Layer
Input Layer	Feeding data into the model	Convolution Layer (1D)
Convolution Layer (1D)	Extracting local patterns from data	Pooling Layer
Pooling Layer (1D)	Reducing spatial dimensions	Dropout Layer
Dropout Layer	Preventing overfitting	Convolution Layer (1D)
Convolution Layer (1D)	Extracting deeper patterns	Pooling Layer
Pooling Layer (1D)	Further dimension reduction	Dropout Layer
Dropout Layer	Mitigating overfitting	Normalization Layer
Normalization Layer	Stabilizing activations	BiLSTM Layer
BiLSTM Layer	Capturing sequential dependencies	Dropout Layer
Dropout Layer	Regularizing after BiLSTM	Normalization Layer
Normalization Layer	Normalizing activations	Dense Layer
Dense Layer	Fully connected feature processing	Dropout Layer
Dropout Layer	Preventing overfitting	Normalization Layer
Normalization Layer	Normalizing dense layer outputs	Output Layer
Output Layer	Generating final classification	–

4.5. Training Specifications Summary for the Hybrid Cnn–Bilstm Model

The specifications for training the proposed hybrid model involve several key hyperparameters for enhancing performance and preventing overfitting. The choice of hyperparameters is driven by empirical evaluation and domain knowledge. The number of filters in the Conv1D layers, kernel size, and activation functions are selected to capture intricate patterns in the data. Pooling layers are used to reduce dimensionality and prevent overfitting, while the BiLSTM layer is employed to learn temporal dependencies effectively. Dense layers with Sigmoid and Softmax activations have been configured for binary and multiclass classification, respectively, to improve classification performance. The key specifications used in this model are outlined below based on the specific architecture used.

1. **Optimization Algorithm:** The model utilizes the *adam optimizer* with a learning rate of 0.001. Adam is an adaptive gradient method that integrates the advantages of RMSProp and SGD with momentum, adjusting the learning rate for each parameter individually and ensuring efficient convergence.
2. **Learning Rate Scheduler:** A ReduceLROnPlateau learning rate scheduler is employed to dynamically reduce the learning rate in response to a plateau in validation loss. This helps to fine-tune the model toward the end of training, preventing overfitting while ensuring a stable convergence.
3. **Early Stopping:** EarlyStopping is used to halt training when the validation loss stops improving. This prevents the model from overfitting by ensuring it does not train unnecessarily after reaching optimal performance. The best weights are restored when training stops early.
4. **Dropout:** Implemented at various layers are shown in Table 4 with the first two dropout layers applying a rate of 0.1 and the last two layers applying a rate of 0.25.
5. **L2 Regularization:** Incorporated in the LSTM and Dense layers of the model, L2 regularization with a factor of 0.01 penalizes larger weights in order to further reduce overfitting and improve the model's generalizability.

By utilizing these techniques, the model is trained to accurately recognize intricate patterns in IoT data while preserving robustness and minimizing overfitting.

4.6. Advanced Model Compression Techniques for Efficiently Deploying Hybrid Cnn–Bilstm–Dnn Model in Iot Ecosystem

This section presents an overview of the pruning and quantization processes applied to the proposed hybrid CNN–BiLSTM–DNN model, tailored for IoT environments. These techniques aim to boost performance while reducing resource usage, all without compromising accuracy. Notably, pruning and quantization can be employed separately or in combination [47]. The step-by-step workflow for optimizing the hybrid model through pruning and quantization is illustrated in Figure 5.

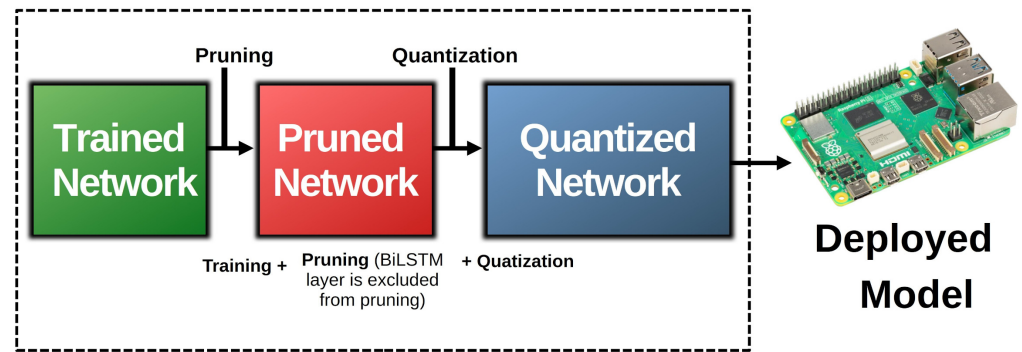


Figure 5. Workflow for optimizing the hybrid model through pruning and quantization.

Pruning in neural networks encompasses techniques designed to minimize the size of a trained model by removing redundant parameters, connections, or structures while maintaining its performance [48]. In our model, we implemented pruning by zeroing out small weight values, guided by a polynomial decay schedule as described in the Equation (12). As highlighted in [30], models that are pruned require less bandwidth for updates, making them better suited for on-device learning, a critical feature for addressing the dynamic and evolving nature of IoT security threats. Notably, in this study, we have pruned all layers except the BiLSTM layer.

$$S(t) = S_{\text{final}} + (S_{\text{initial}} - S_{\text{final}}) \times \left(1 - \frac{\min(t, t_{\text{end}})}{t_{\text{end}} - t_{\text{begin}}}\right)^3 \quad (12)$$

where $S(t)$ represents the sparsity at time t , which corresponds to the current training step. The parameter S_{initial} denotes the initial sparsity, typically set to a value such as 50%, while S_{final} indicates the final sparsity, often set to a value like 80%. The variable t refers to the current training step, with t_{begin} marking the training step at which pruning starts and t_{end} signifying the training step where pruning concludes.

The structured pruning process, represented algorithmically in Algorithm 2, leverages TensorFlow’s model optimization toolkit to enhance the hybrid CNN–BiLSTM–DNN model. Initially, pruning parameters and schedules are defined using a polynomial decay function to control the pruning rate. A pruning wrapper is applied to all layers except the BiLSTM layer, where *PruneBidirectional* is used to exclude it from pruning due to its critical role. The pruned model is then fine-tuned to restore performance metrics, ensuring that accuracy and efficiency are retained. The final output is a simplified and optimized model ready for quantization and deployment on resource-constrained IoT devices.

Algorithm 2: Proposed Model Pruning Process

1. **Input:** Pre-trained hybrid Model, Defined Pruning Parameters
2. Configure the pruning schedule using a polynomial decay function.
3. Apply the pruning wrapper to all layers, excluding the BiLSTM layer by leveraging **PruneBidirectional** to bypass pruning for this specific layer.
4. Fine-tune the pruned model to restore and optimize performance metrics.
5. **Output:** Simplified and Efficiently Optimized Model for Quantization

Alongside pruning, we further optimized and compressed the proposed model through post-training quantization. As explained in [49], a strategy for minimizing a model's computational and storage requirements is to use reduced bit widths for calculations and tensor storage, a process called quantization, in contrast to traditional floating-point precision [50–52].

Quantization involves converting higher-precision weights (e.g., using a 32-bit floating point) into lower-precision representations (e.g., using 8-bit integers), with the process mathematically represented as follows:

$$w_q = \text{round}\left(\frac{w}{\Delta}\right) \cdot \Delta \quad (13)$$

where w represents the original weight value, while w_q denotes the quantized weight value. The parameter Δ is the quantization step size, which is computed using the formula provided in Equation (14).

$$\Delta = \frac{\max(w) - \min(w)}{2^b - 1} \quad (14)$$

where b is the number of bits (e.g., $b = 8$ for 8-bit quantization).

Quantization reduces memory and computational demands by approximating model weights while preserving a satisfactory level of accuracy. These mathematical principles underscore how pruning and quantization collaboratively optimize neural networks for efficient deployment in IoT environments.

The post-training quantization process, represented algorithmically in Algorithm 3, uses TensorFlow Lite to make the pruned hybrid CNN–BiLSTM–DNN model more efficient for IoT devices. First, unnecessary pruning data are removed from the pruned model using TensorFlow's optimization tools. The cleaned model is then converted using TensorFlow Lite Converter. The result is a smaller, faster, and more efficient model, ready for deployment on resource-constrained IoT devices.

Algorithm 3: Enhancing the Proposed Hybrid Model via Post-Training Quantization

1. **Input:** Pruned Hybrid Model
2. Strip pruning artifacts from the pruned model using TensorFlow Model Optimization Toolkit.
3. Initialize TensorFlow Lite Converter with the stripped model.
4. Apply post-training quantization using the TensorFlow Lite Converter with default optimizations and supported TensorFlow Lite operations.
5. **Output:** Optimized TensorFlow Lite Model with reduced size and enhanced efficiency for IoT deployments.

4.7. Performance Measures

This section focuses on evaluating the proposed security framework's performance using metrics extracted from the confusion matrix. The assessment methodology is critical for analyzing performance and guiding the development of the classifier, as discussed in [53].

4.7.1. Confusion Matrix Components

The confusion matrix in Figure 6 is used to calculate several key components: *True Positive (TP)* refers to positive samples accurately classified as positive, while *True Negative (TN)* represents negatively labeled samples correctly classified as negative. Additionally, *False Positive (FP)* represents negative samples that have been wrongly classified as positive, and *False Negative (FN)* denotes positively labeled samples incorrectly identified as negative.

		Predicted Class	
		Positive Class	Negative Class
Actual/True Class	Positive Class	True Positive (TP)	False Negative (FN)
	Negative Class	False Positive (FP)	True Negative (TN)

Figure 6. Confusion matrix.

4.7.2. Performance Metrics

The evaluation of the model's performance is based on the following confusion matrix-derived metrics:

1. **Accuracy:** The accuracy given in Equation (15) is computed as the ratio of correctly classified samples, which is the sum of True Positives (TP) and True Negatives (TN), to the total sample size, which includes both Positive (P) and Negative (N) samples Vujovic [54]. The formula for accuracy is the following:

$$\text{Accuracy} = \frac{TP + TN}{P + N} \quad (15)$$

The highest achievable accuracy is 1.0, while the lowest is 0.0.

2. **True Positive Rate (TP Rate)-Sensitivity (or Recall):** The True Positive Rate is calculated by dividing the number of correctly predicted positive instances (TP) by the total number of actual positive instances (P) Vujovic [54]. Mathematically, the TP rate is given by the following:

$$\text{TP Rate} = \frac{TP}{TP + FN} \quad (16)$$

3. **Precision:** The ratio of correctly predicted positive instances to the total predicted positives, which is calculated as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (17)$$

4. **F1-Score:** The harmonic mean of precision and recall, which offers a balanced measurement, is especially useful in class-imbalanced datasets. It is computed as follows:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (18)$$

These metrics provide an exhaustive evaluation of the proposed model's ability to classify instances accurately, highlighting its robustness and effectiveness.

5. Results

5.1. Experimental Settings

The framework proposed is implemented with TensorFlow on Google Colaboratory, an open-source, cloud-based Jupyter notebook environment provided by Google [55]. Additionally, we utilized a GPU for training our proposed model to enhance computational efficiency and accelerate the training process. This combination of TensorFlow and GPU resources ensures that our model training is both robust and time-efficient, leveraging advanced hardware capabilities available in Google Colaboratory.

5.2. Evaluation of the Proposed Hybrid Model on the Iot-23 Dataset

The outcomes of the hybrid model applied to the IoT-23 dataset are discussed in this section, using various performance metrics and visualizations. The hybrid model, which combines CNN, BiLSTM, and DNN, was employed to classify network traffic as either malicious or benign. The IoT-23 dataset, characterized by its heterogeneous nature, presents unique challenges for IoT attack detection.

Based on Figure 7, the model demonstrates rapid convergence during the initial epochs, with training accuracy stabilizing as the training progresses, indicating effective optimization. The figure also shows a significant decrease in training loss during the early stages, while validation loss remains consistent, confirming early convergence without signs of overfitting.

The confusion matrices in Figures 8 and 9 further validate the model's strong performance in both binary and multi-class classification tasks, with high diagonal values reflecting accurate predictions. This highlights the hybrid CNN–BiLSTM–DNN model's ability to generalize effectively.

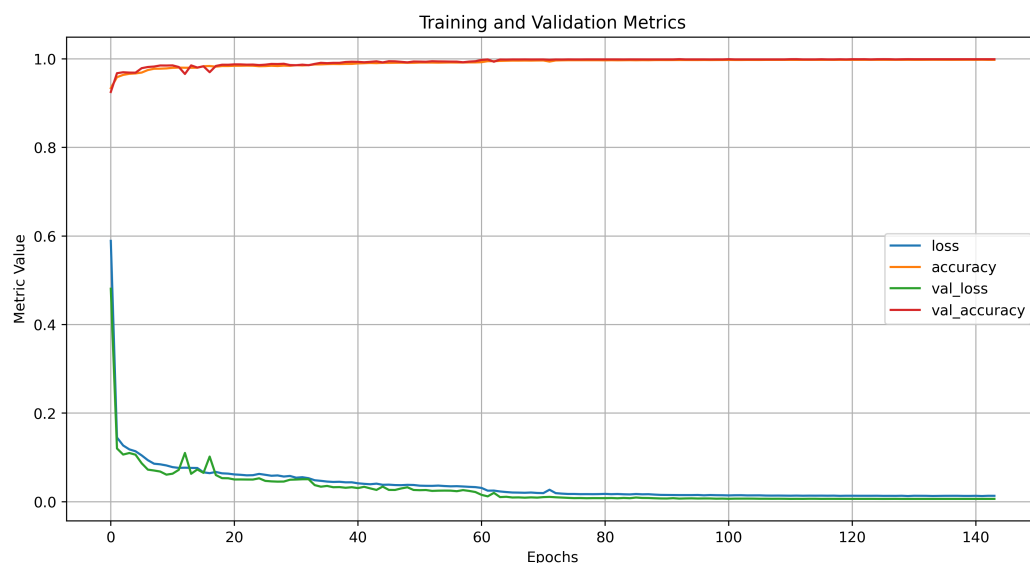


Figure 7. Training and validation metrics for the IoT-23 dataset over 30 epochs.

As summarized in Tables 6 and 7, the model achieves outstanding performance metrics on the IoT-23 dataset for both binary and multi-class tasks. These results emphasize the

robustness of the proposed model in detecting IoT threats, with the visualizations offering valuable insights into the model's behavior and potential areas for fine-tuning.

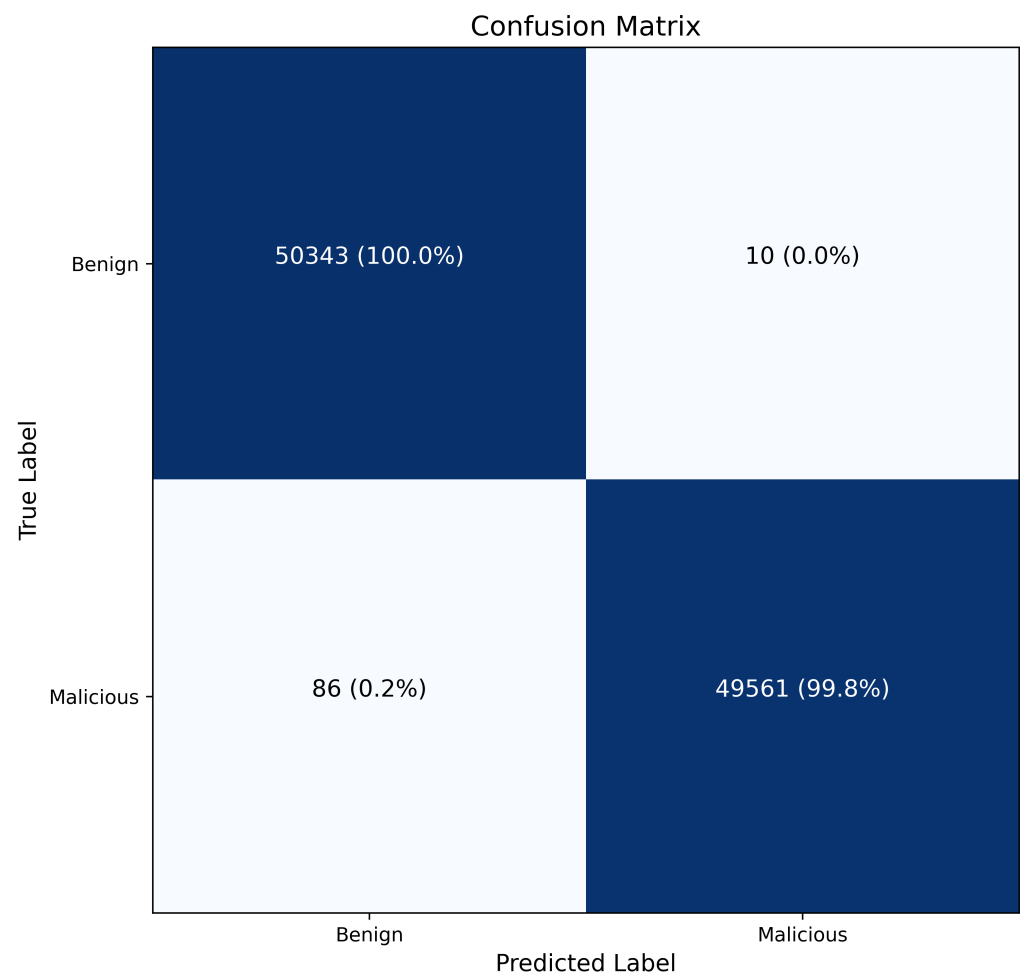


Figure 8. Binary classification confusion matrix generated using the IoT-23 dataset.

Table 6. Performance evaluation of the CNN–BiLSTM–DNN model for binary classification using the IoT-23 dataset.

Class	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Support
Benign	99.90	99.98	99.83	99.90	50,353
Malicious	99.90	99.83	99.98	99.90	49,647

Table 7. Performance evaluation of the CNN–BiLSTM–DNN model for multi-class classification using the IoT-23 dataset.

Class	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Support
Benign	99.97	99.99	99.95	99.97	50,535
DDoS	99.97	99.88	99.99	99.93	14,504
Okiru	99.97	100.0	100.0	100.0	14,255
Other Attacks	99.97	99.86	99.97	99.92	6514
PortScan	99.97	100.0	100.0	100.0	14,374

PortScan → PartOfAHorizontalPortScan.

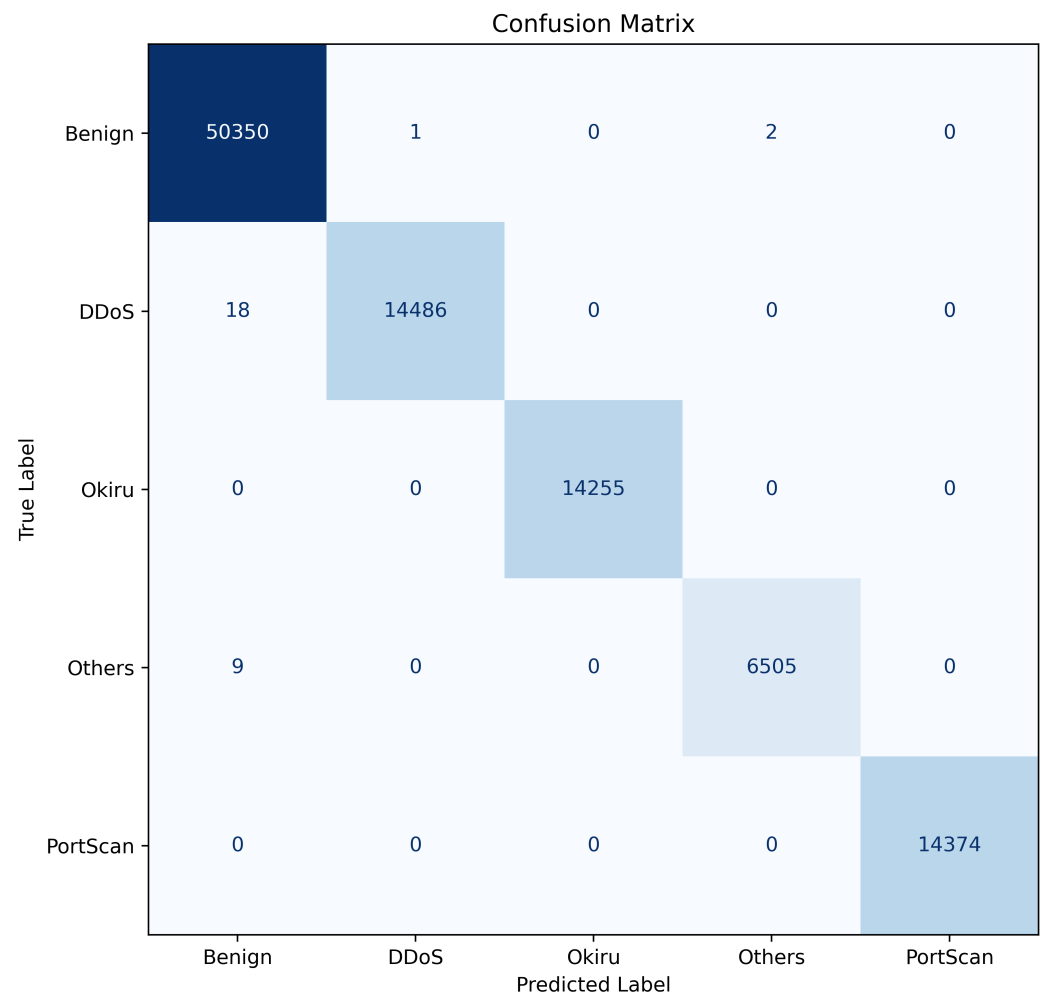


Figure 9. Confusion matrix for multi-class classification based on the IoT-23 dataset.

5.3. Evaluation of the Proposed Model on Edge-Iiotset Dataset

The ever-evolving landscape of cybersecurity necessitates advanced solutions to address the growing complexity of threat detection and classification in the IoT ecosystem. The proposed hybrid architecture integrating CNN, BiLSTM, and DNNs has also shown great promise in tackling both binary and multi-class classification problems using the Edge-IIoTset.

Figures 10 and 11 showcase the confusion matrices depicting the performance of the model in binary and multi-class classification, respectively, on the Edge-IIoTset dataset. The hybrid architecture demonstrates exceptional accuracy in correctly classifying the threat categories, confirming its effectiveness in accurately identifying genuine network traffic. The accuracy curve in Figure 12 highlights the model's effectiveness across multiple classes. Most data points cluster near the top-right corner, reflecting an optimal balance between precision (minimizing false positives) and recall (minimizing missed true positives). This shows that the model performs well in detecting threats while minimizing misclassification. The loss curve, as shown in Figure 13, illustrates the model's optimization progress over epochs, demonstrating its ability to minimize error and improve performance during training and validation phases. Tables 8 and 9 summarize the performance metrics of the model for binary and multiclass classification, respectively, across various classes.

Figure 12 also highlights the rapid convergence of the model during the early epochs, with training accuracy stabilizing as epochs progressed, indicating effective optimization. Meanwhile, Figure 13 shows a significant decrease in training loss during the initial

epochs, while validation loss remained relatively stable, reinforcing the observation of early convergence.

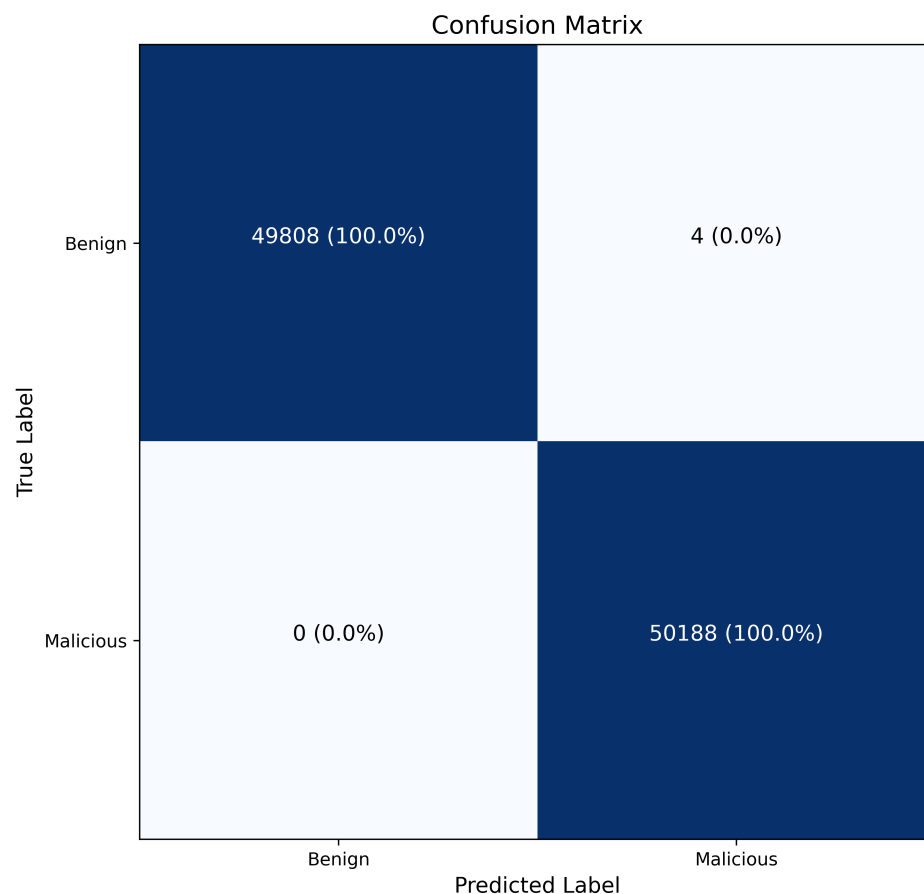


Figure 10. Confusion matrix for binary classification based on the Edge-IIoTset dataset).

Table 8. Performance evaluation of a CNN–BiLSTM–DNN model for binary classification using the Edge-IIoTset dataset.

Class	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Support
Benign	100.0	99.99	100.0	99.99	49,812
Malicious	100.0	100.0	99.99	99.99	50,188

Table 9. Performance evaluation of a multi-class CNN–BiLSTM–DNN model using the Edge-IIoTset dataset.

Class	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)	Support
Benign Traffic	99.99	100.0	100.0	100.0	74
DDoS TCP	99.99	100.0	100.0	100.0	49,812
Uploading Attack	99.99	100.0	99.94	99.97	1602
XSS	99.99	100.0	100.0	100.0	26,906
DDoS HTTP	99.99	100.0	100.0	100.0	14
Password Attack	99.99	100.0	99.99	99.99	17,897
Backdoor	99.99	100.0	100.0	100.0	1040
OS	99.99	96.10	100.0	98.01	77
Fingerprinting	99.99	99.65	100.0	99.82	856
Port Scanning	99.99	99.68	99.48	99.57	937
Vulnerability Scanner	99.99	99.68	99.48	99.57	937
Ransomware	99.99	100.0	99.45	99.73	544
SQL Injection	99.99	97.93	98.74	98.33	241

DDoS TCP → DDoS TCP SYN Flood; DDoS HTTP → DDoS HTTP Flood.

The hybrid CNN–BiLSTM–DNN architecture emerges as a robust contender for multi-class classification in cybersecurity. Its application to the IoT-23 and Edge-IIoTset datasets highlights its capacity to achieve high accuracy across a variety of threat types, with notable strengths in recall, precision, and F1-score.

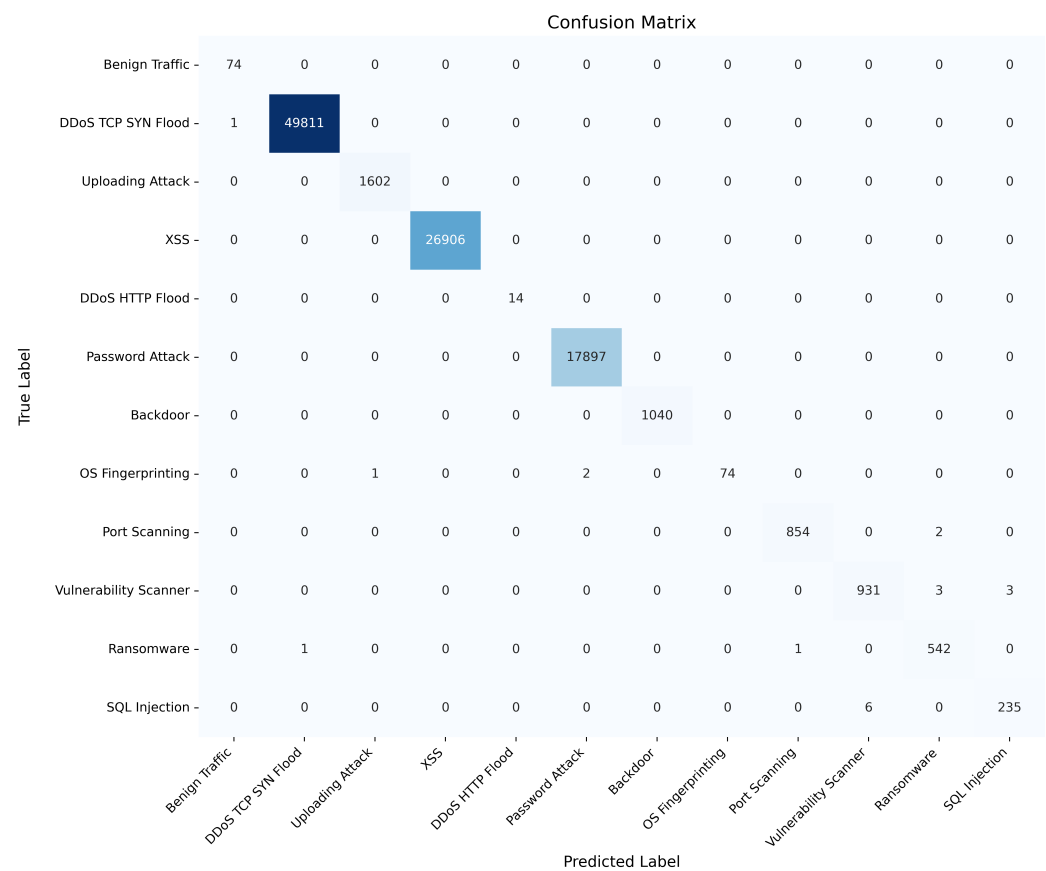


Figure 11. Confusion matrix depicting the outcomes of multi-class classification on the Edge-IIoTset dataset.

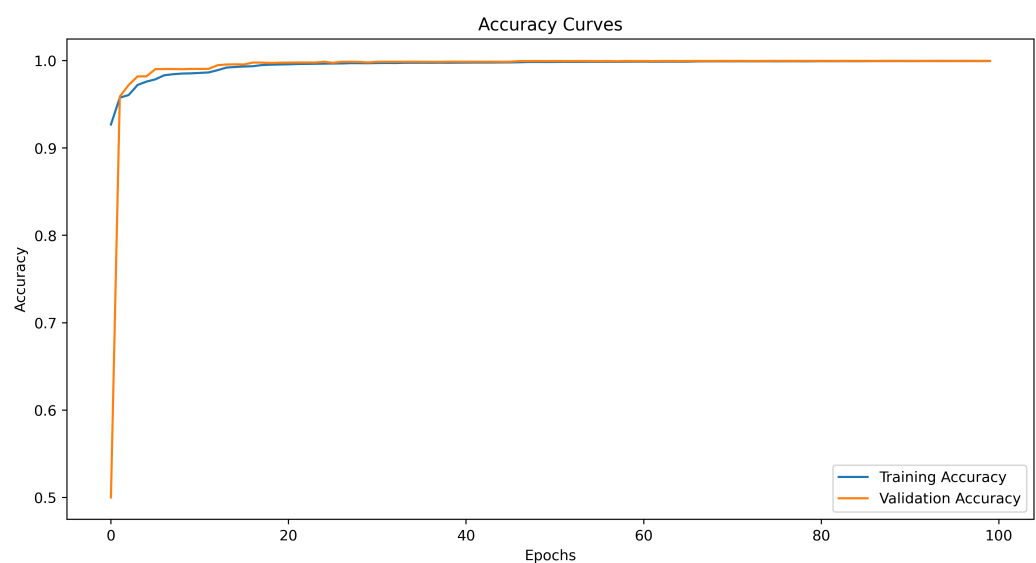


Figure 12. Accuracy curve for multi-class classification based on the Edge-IIoTset dataset.

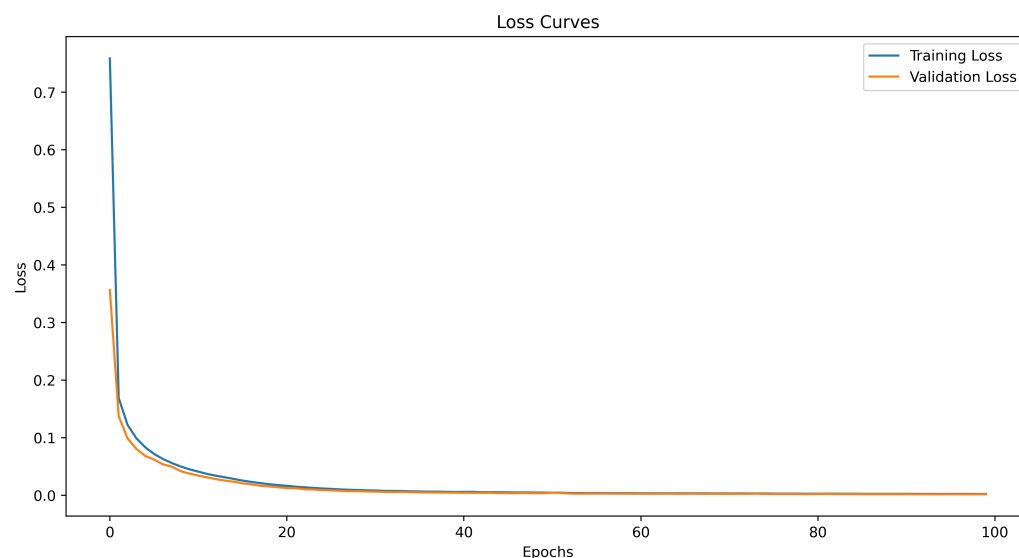


Figure 13. Loss curve for multi-class classification based on the Edge-IIoTset dataset.

6. Discussion

The performance results presented in Section 5 offer a comprehensive analysis of our proposed model's efficacy in detecting cybersecurity threats within IoT networks. These results underscore the model's performance across various metrics, utilizing the IoT-23 and Edge-IIoTset datasets, which encompass a wide array of attack types and scenarios. Table 10 presents a detailed comparison of our research with key studies in IoT cybersecurity. This comparison includes an examination of the datasets used, the types of cyberattacks addressed, and the accuracy rates achieved by different models. The table also illustrates the incorporation of advanced model optimization techniques in our study, such as model pruning and quantization, which have significantly enhanced the performance and efficiency of our proposed model. This comparison not only showcases the advancements made by our research but also positions our model as a leading solution in the evolving landscape of IoT security.

Table 10. Comparison of our model with leading research in IoT security.

Study	Dataset	Model	Accuracy (%)	Advanced Optimization
Bhandari et al. [31]	IoT-23	DNN	93	No
	Edge-IIoTset		94	
Sahu et al. [25]	IoT-23	CNN and LSTM	96	No
Nazir et al. [30]	IoT-23	CNN–LSTM	95	Yes
	N-BaIoT		99	
	CICIDS2017		99	
Proposed model	IoT-23 Edge-IIoTset	CNN–BiLSTM–DNN	99	Yes

Table 10 showcases the performance of our proposed security mechanism alongside four related studies. The findings demonstrate that our model achieves higher attack detection accuracy, validated using the IoT-23 and Edge-IIoTset datasets. The hybrid CNN–BiLSTM–DNN model we propose is capable of detecting a wider range of attacks, including DDoS, PartOfAHorizontalPortScan, Okiru, and other attacks categorized as miscellaneous.

The studies summarized in Table 10 highlight high accuracy rates achieved by IoT security models utilizing techniques such as CNN–LSTM, CNN and LSTM, and DNN.

However, a notable limitation in the works of Sahu et al. [25] and Bhandari et al. [31] is the lack of pruning and quantization techniques in their implementations. These techniques are crucial for optimizing deep learning models for IoT devices, as they significantly reduce model size, computational overhead, and energy consumption. Without these optimizations, the efficiency of these models on resource-constrained IoT devices is compromised, limiting their practical deployment in real-world applications. Addressing this gap could be an essential focus for future research in the field.

While Nazir et al. [30] integrated pruning and quantization into their model, its performance on the IoT-23 dataset remains inferior compared to our proposed model.

Additionally, our model's evaluation across two diverse datasets demonstrates its adaptability and robustness in various IoT applications. By integrating advanced optimization techniques, our approach ensures compatibility with the diverse computational and energy constraints of the IoT ecosystem, making it a strong candidate for real-world deployment.

7. Conclusions

This research offers a detailed and in-depth analysis of the IoT-23 and Edge-IIoTset datasets, focusing on the performance of the proposed hybrid CNN-BiLSTM-DNN model for both binary and multi-class attack detection. Comprehensive comparisons of leading research solutions were conducted to establish the model's relevance within the larger scope of IoT and IIoT cybersecurity. The findings indicate that the Edge-IIoTset and IoT-23 datasets are particularly suitable for binary classification tasks. Additionally, specific traffic scenarios, including Benign, DDoS, Okiru, and PartOfAHorizontalPortScan, were selectively sampled from the IoT-23 dataset for evaluation. Future work will focus on incorporating the full IoT-23 and Edge-IIoTset datasets and considering other deep learning approaches to enhance model performance. Plans also include deploying the proposed model on actual IoT devices to validate its real-world applications. Moreover, the significance of employing advanced data preprocessing techniques, such as data augmentation and feature selection, will be examined to further optimize the model. The research also aims to expand the model's adaptability to datasets, featuring a wider range of attack scenarios. The insights from this study are projected to be of great value to researchers, data scientists, academics, and practitioners in the IoT and IIoT cybersecurity domains. This work contributes to the ongoing efforts to address critical challenges and develop effective solutions for improving network security in IoT environments.

Author Contributions: Conceptualization, P.A. and B.A.A.; methodology, B.U.-A.S.; software, U.O.L.; validation, V.A. and P.A.; formal analysis, B.A.A.; investigation, U.O.L.; resources, U.O.L.; data curation, V.A.; writing—original draft preparation, V.A. and B.U.-A.S.; writing—review and editing, P.A. and B.A.A.; visualization, V.A.; supervision, B.U.-A.S.; project administration, B.U.-A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data supporting the reported results can be found in the following sources: for the Aposemat IoT-23 dataset, visit (<https://www.stratosphereips.org/datasets-iot23>) (accessed on 6 November 2024), and for the Edge-IIoTset dataset, refer to the links provided in reference [42].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
CNN	Convolutional Neural Network
BiLSTM	Bidirectional Long Short-Term Memory
DNN	Deep Neural Network
IDS	Intrusion Detection System
DL	Deep Learning
ML	Machine Learning
PCA	Principal Component Analysis
SMOTE	Synthetic Minority Over-sampling Technique
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
F1-Score	Harmonic Mean of Precision and Recall
SGD	Stochastic Gradient Descent
RF	Random Forest
GP	Gaussian Process
AE	Auto Encoder
GRU	Gated Recurrent Unit
FL	Federated Learning
AAE	Adversarial Autoencoders
BiGAN	Bidirectional Generative Adversarial Networks

References

1. Zaman, S.; Alhazmi, K.; Aseeri, M.A.; Ahmed, M.R.; Khan, R.T.; Kaiser, M.S.; Mahmud, M. Security Threats and Artificial Intelligence Based Countermeasures for Internet of Things Networks: A Comprehensive Survey. *IEEE Access* **2021**, *9*, 94668–94690. [\[CrossRef\]](#)
2. Swamy, S.N.; Kota, S.R. An Empirical Study on System Level Aspects of Internet of Things (IoT). *IEEE Access* **2020**, *8*, 188082–188134. [\[CrossRef\]](#)
3. Molaei, F.; Rahimi, E.; Siavoshi, H.; Afrouz, S.G.; Tenorio, V. A Comprehensive Review on Internet of Things (IoT) and its Implications in the Mining Industry. *Am. J. Eng. Appl. Sci.* **2020**, *13*, 499–515. [\[CrossRef\]](#)
4. Quy, V.K.; Hau, N.V.; Anh, D.V.; Quy, N.M.; Ban, N.T.; Lanza, S.; Randazzo, G.; Muzirafuti, A. IoT-Enabled Smart Agriculture: Architecture, Applications, and Challenges. *Appl. Sci.* **2022**, *12*, 3396. [\[CrossRef\]](#)
5. Kim, W.S.; Lee, W.S.; Kim, Y.J. A Review of the Applications of the Internet of Things (IoT) for Agricultural Automation. *J. Biosyst. Eng.* **2020**, *45*, 385–400. [\[CrossRef\]](#)
6. Kassim, M.R.M. IoT Applications in Smart Agriculture: Issues and Challenges. In Proceedings of the 2020 IEEE Conference on Open Systems (ICOS), Kuala Lumpur, Malaysia, 17–19 November 2020; pp. 19–24. [\[CrossRef\]](#)
7. González-Zamar, M.D.; Abad-Segura, E.; Vázquez-Cano, E.; López-Meneses, E. IoT Technology Applications-Based Smart Cities: Research Analysis. *Electronics* **2020**, *9*, 1246. [\[CrossRef\]](#)
8. Kanellopoulos, D.; Sharma, V.K.; Panagiotakopoulos, T.; Kameas, A. Networking Architectures and Protocols for IoT Applications in Smart Cities: Recent Developments and Perspectives. *Electronics* **2023**, *12*, 2490. [\[CrossRef\]](#)
9. Singh, R.P.; Javaid, M.; Haleem, A.; Suman, R. Internet of Things (IoT) applications to fight against COVID-19 pandemic. *Diabetes Metab. Syndr. Clin. Res. Rev.* **2020**, *14*, 521–524. [\[CrossRef\]](#) [\[PubMed\]](#)
10. Nivzetić, S.; Šolić, P.; López-de-Ipiña González-de-Artaza, D.; Patrono, L. Internet of Things (IoT): Opportunities, issues and challenges towards a smart and sustainable future. *J. Clean. Prod.* **2020**, *274*, 122877. [\[CrossRef\]](#)
11. Kaspersky. Kaspersky Unveils an Overview of IoT-Related Threats in 2023. Kaspersky Press Releases. Available online: <https://www.kaspersky.com/about/press-releases/kaspersky-unveils-an-overview-of-iot-related-threats-in-2023> (accessed on 6 December 2024).
12. Khanna, A.; Kaur, S. Internet of Things (IoT), Applications and Challenges: A Comprehensive Review. *Wirel. Pers. Commun.* **2020**, *114*, 1687–1762. [\[CrossRef\]](#)

13. Mishra, N.; Pandya, S. Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review. *IEEE Access* **2021**, *9*, 59353–59377. [\[CrossRef\]](#)
14. Liao, B.; Ali, Y.; Nazir, S.; He, L.; Khan, H.U. Security Analysis of IoT Devices by Using Mobile Computing: A Systematic Literature Review. *IEEE Access* **2020**, *8*, 120331–120350. [\[CrossRef\]](#)
15. Muhammad, K.; Hussain, T.; Tanveer, M.; Sannino, G.; de Albuquerque, V.H.C. Cost-effective video summarization using deep CNN with hierarchical weighted fusion for IoT surveillance networks. *IEEE Internet Things J.* **2020**, *7*, 4455–4463. [\[CrossRef\]](#)
16. Khan, M.; Wang, H.; Riaz, A.; Elfatyany, A.; Karim, S. Bidirectional LSTM-RNN-Based Hybrid Deep Learning Frameworks for Univariate Time Series Classification. *J. Supercomput.* **2021**, *77*, 7021–7045. [\[CrossRef\]](#)
17. Hua, Y.; Mou, L.; Zhu, X.X. Recurrently Exploring Class-Wise Attention in a Hybrid Convolutional and Bidirectional LSTM Network for Multi-Label Aerial Image Classification. *ISPRS J. Photogramm. Remote Sens.* **2019**, *149*, 188–199. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Algethami, S.A.; Alshamrani, S.S. A Deep Learning-Based Framework for Strengthening Cybersecurity in Internet of Health Things (IoHT) Environments. *Appl. Sci.* **2024**, *14*, 4729. [\[CrossRef\]](#)
19. Omarov, B.; Sailaukyzy, Z.; Bigaliyeva, A.; Kereyev, A.; Naizabayeva, L.; Dautbayeva, A. One Dimensional Conv-BiLSTM Network with Attention Mechanism for IoT Intrusion Detection. *Comput. Math. Methods Eng.* **2023**, *77*, 3765–3781. [\[CrossRef\]](#)
20. Al Nuaimi, T.; Al Zaabi, S.; Alyilieli, M.; AlMaskari, M.; Alblooshi, S.; Alhabsi, F.; Yusof, M.F.B.; Al Badawi, A. A Comparative Evaluation of Intrusion Detection Systems on the Edge-IIoT-2022 Dataset. *Intell. Syst. Appl.* **2023**, *20*, 200298. [\[CrossRef\]](#)
21. Bovenzi, G.; Aceto, G.; Ciuonzo, D.; Montieri, A.; Persico, V.; Pescapé, A. Network Anomaly Detection Methods in IoT Environments via Deep Learning: A Fair Comparison of Performance and Robustness. *Comput. Secur.* **2023**, *128*, 103167. [\[CrossRef\]](#)
22. Kikissagbe, B.R.; Adda, M.; Célicourt, P.; Haman, I.T.; Najjar, A. Machine Learning for DoS Attack Detection in IoT Systems. *Procedia Comput. Sci.* **2024**, *241*, 195–202. [\[CrossRef\]](#)
23. Ahli, A.; Raza, A.; Ovaz, K.; Akpinar, M. Binary and Multi-Class Classification on the IoT-23 Dataset. In Proceedings of the 2023 Advances in Science and Engineering Technology International Conferences (ASET), Dubai, United Arab Emirates, 20–23 February 2023; pp. 1–7. [\[CrossRef\]](#)
24. Zohourian, A.; Dadkhah, S.; Molyneaux, H.; Pinto Neto, E.C.; Ghorbani, A.A. IoT-PRIDS: Leveraging Packet Representations for Intrusion Detection in IoT Networks. *Comput. Secur.* **2024**, *146*, 104034. [\[CrossRef\]](#)
25. Sahu, A.K.; Sharma, S.; Tanveer, M.; Raja, R. Internet of Things Attack Detection Using Hybrid Deep Learning Model. *Comput. Commun.* **2021**, *176*, 112–122. [\[CrossRef\]](#)
26. Abdulkareem, S.A.; Foh, C.H.; Carrez, F.; Moessner, K. A Lightweight SEL for Attack Detection in IoT/IoT Networks. *J. Netw. Comput. Appl.* **2024**, *230*, 103980. [\[CrossRef\]](#)
27. Riaz, S.; Latif, S.; Usman, S.M.; Ullah, S.S.; Algarni, A.D.; Yasin, A.; Anwar, A.; Elmannai, H.; Hussain, S. Malware Detection in Internet of Things (IoT) Devices Using Deep Learning. *Sensors* **2022**, *22*, 9305. [\[CrossRef\]](#)
28. Halbouni, A.; Gunawan, T.S.; Habaebi, M.H.; Halbouni, M.; Kartiwi, M.; Ahmad, R. CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System. *IEEE Access* **2022**, *10*, 99837–99849. [\[CrossRef\]](#)
29. Altunay, H.C.; Albayrak, Z. A Hybrid CNN+LSTM-Based Intrusion Detection System for Industrial IoT Networks. *Eng. Sci. Technol. Int. J.* **2023**, *38*, 101322. [\[CrossRef\]](#)
30. Nazir, A.; He, J.; Zhu, N.; Qureshi, S.S.; Qureshi, S.U.; Ullah, F.; Wajahat, A.; Pathan, M.S. A Deep Learning-Based Novel Hybrid CNN-LSTM Architecture for Efficient Detection of Threats in the IoT Ecosystem. *Ain Shams Eng. J.* **2024**, *15*, 102777. [\[CrossRef\]](#)
31. Bhandari, G.; Lyth, A.; Shalaginov, A.; Grønli, T.-M. Distributed Deep Neural-Network-Based Middleware for Cyber-Attacks Detection in Smart IoT Ecosystem: A Novel Framework and Performance Evaluation Approach. *Electronics* **2023**, *12*, 298. [\[CrossRef\]](#)
32. Yazdinejad, A.; Kazemi, M.; Parizi, R.M.; Dehghantanha, A.; Karimipour, H. An Ensemble Deep Learning Model for Cyber Threat Hunting in Industrial Internet of Things. *Digit. Commun. Netw.* **2023**, *9*, 101–110. [\[CrossRef\]](#)
33. Bukhari, S.M.S.; Zafar, M.H.; Houran, M.A.; Qadir, Z.; Moosavi, S.K.R.; Sanfilippo, F. Enhancing Cybersecurity in Edge IIoT Networks: An Asynchronous Federated Learning Approach with a Deep Hybrid Detection Model. *Internet Things* **2024**, *27*, 101252. [\[CrossRef\]](#)
34. Abdalgawad, N.; Sajun, A.; Kaddoura, Y.; Zualkernan, I.A.; Aloul, F. Generative Deep Learning to Detect Cyberattacks for the IoT-23 Dataset. *IEEE Access* **2022**, *10*, 6430–6441. [\[CrossRef\]](#)
35. Zixu, T.; Liyanage, K.S.K.; Gurusamy, M. Generative Adversarial Network and Auto Encoder-Based Anomaly Detection in Distributed IoT Networks. In Proceedings of the GLOBECOM 2020—2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020; pp. 1–7. [\[CrossRef\]](#)
36. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [\[CrossRef\]](#)
37. Alani, M.M.; Miri, A. Towards an Explainable Universal Feature Set for IoT Intrusion Detection. *Sensors* **2022**, *22*, 5690. [\[CrossRef\]](#) [\[PubMed\]](#)

38. Moustafa, N. ToN_IoT Datasets. *IEEE Dataport* **2019**. . [CrossRef]
39. Garcia, S.; Parmisano, A.; Erquiaga, M.J. IoT-23: A Labeled Dataset with Malicious and Benign IoT Network Traffic (Version 1.0.0) [Data set]. *Zenodo* **2020**. [CrossRef]
40. Alfares, H.; Banimelhem, O. Comparative Analysis of Machine Learning Techniques for Handling Imbalance in IoT-23 Dataset for Intrusion Detection Systems. In Proceedings of the 11th International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Dubai, United Arab Emirates, 4–6 February 2006; pp. 112–119. [CrossRef]
41. Thiyyam, B.; Dey, S. Efficient Feature Evaluation Approach for a Class-Imbalanced Dataset Using Machine Learning. *Procedia Comput. Sci.* **2023**, *218*, 2520–2532. [CrossRef]
42. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. [CrossRef]
43. Sharafaldin, I.; Lashkari, A.H.; Hakak, S.; Ghorbani, A.A. Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy. In Proceedings of the IEEE 53rd International Carnahan Conference on Security Technology, Chennai, India, 1–3 October 2019; pp. 1–8. [CrossRef]
44. Gheni, H.Q.; Al-Yaseen, W.L. Two-Step Data Clustering for Improved Intrusion Detection System Using CICIoT2023 Dataset. *e-Prime-Adv. Electr. Eng. Electron. Energy* **2024**, *9*, 100673. [CrossRef]
45. Mahdi, Z.; Abdalhussien, N.; Mahmood, N.; Zaki, R. Detection of Real-Time Distributed Denial-of-Service (DDoS) Attacks on Internet of Things (IoT) Networks Using Machine Learning Algorithms. *Comput. Mater. Contin.* **2024**, *80*, 2139–2159. [CrossRef]
46. Bakhsh, S.A.; Khan, M.A.; Ahmed, F.; Alshehri, M.S.; Ali, H.; Ahmad, J. Enhancing IoT Network Security Through Deep Learning-Powered Intrusion Detection System. *Internet Things* **2023**, *24*, 100936. [CrossRef]
47. Liang, T.; Glossner, J.; Wang, L.; Shi, S.; Zhang, X. Pruning and Quantization for Deep Neural Network Acceleration: A Survey. *arXiv* **2021**, arXiv:2101.09671. [CrossRef]
48. Bibi, U.; Mazhar, M.; Sabir, D.; Butt, M.F.U.; Hassan, A.; Ghazanfar, M.A.; Khan, A.A.; Abdul, W. Advances in Pruning and Quantization for Natural Language Processing. *IEEE Access* **2024**, *12*, 139113–139128. [CrossRef]
49. Zeeshan, M. *Efficient Deep Learning Models for Edge IoT Devices—A Review*; Tech. Rep.; Department of Electrical Engineering, Indian Institute of Technology Patna: Patna, India, August 2024. Available online: https://d197for5662m48.cloudfront.net/documents/publicationstatus/216349/preprint_pdf/fd256b0205683d43c9f61e8327fec378.pdf (accessed on 6 December 2024).
50. Courbariaux, M.; Bengio, Y.; David, J.-P. BinaryConnect: Training Deep Neural Networks with Binary Weights During Propagations. *Adv. Neural Inf. Process. Syst.* **2015**, *12*, 1–9. [CrossRef]
51. Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv* **2016**, arXiv:1510.00149.
52. Zhuang, B.; Shen, C.; Tan, M.; Liu, L.; Reid, I. Towards Effective Low-Bitwidth Convolutional Neural Networks. *arXiv* **2017**, arXiv:1711.00205.
53. Tharwat, A. Classification Assessment Methods. *Appl. Comput. Inform.* **2021**, *17*, 168–192. [CrossRef]
54. Vujovic, Z.D. Classification Model Evaluation Metrics. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 6. [CrossRef]
55. Sukhdeve, D.S.R.; Sukhdeve, S.S. *Google Cloud Platform for Data Science: A Crash Course*; Apress Data Science and Machine Learning Series; Springer: Berlin/Heidelberg, Germany, 2023; pp. 11–34. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.