The ISC Int'l Journal of Information Security

ISeCure

January 2026, Volume 18, Number 1 (pp. 19–33)

http://www.isecure-journal.org

Performance Evaluation of Deep Learning Models on Diverse IoT Datasets for Intrusion Detection

Abdullah Waqas [1,*]

[1] *Department of Electrical Engineering, National University of Technology, Islamabad, Pakistan*

ARTICLE INFO.

Article history:
Received: February 21, 2025
Revised: July 2, 2025
Accepted: September 15, 2025
Published Online: September 20, 2025

Keywords:
Deep Learning, Internet of Things, Neural Networks, Cybersecurity.

Type: Research Article

doi: 10.22042/isecure.2025.507948.1208

ABSTRACT

The Internet of Things (IoT) offers transformative potential across sectors like energy, defense, and healthcare, but its limited resources make it vulnerable to cyberattacks, necessitating robust security measures such as intrusion detection systems (IDS) to safeguard its infrastructure. This article presents a study that helps intrusion detection systems identify malicious and legitimate communications. To help the system make the best decisions possible, the sub-category of the attacked traffic is also classified. We trained the suggested models to be capable of binary and multiclass classification, targeting common attacks like denial of service (DoS), distributed denial of service (DDoS), reconnaissance, and information theft directed at IoT devices. Our methodology makes use of recently published IoT datasets, such as BoTIoT, ToNIoT, WUSTL-IIOT-20212021, and CiCIoT. To assess and contrast the performance of the proposed models on these datasets, we first applied stratified undersampling to convert the original imbalanced datasets into balanced subsets, which were then used for training and evaluation. Among the models evaluated, biLSTM achieved the highest accuracy of 99.66% and MCC of 0.99759 on the WUSTL-IIoT-2021 dataset. On the BoTIoT dataset, CNN with Dual Focal Loss reached 97.76% accuracy and 0.95536 MCC. For ToNIoT, LSTM achieved 97.01% accuracy with an MCC of 0.93643, while on the CiCIoT dataset, biLSTM obtained 96.23% accuracy and 0.96347 MCC. The results show that biLSTM and LSTM models give higher performance than FNN and CNN models in terms of precision, recall, F1 score, and MCC across all datasets, demonstrating improved performance for temporal IoT intrusion detection tasks.

1 Introduction

In Internet of Things (IoT) networks, where several objects are networked and interact independently, robust intrusion detection becomes essential. These networks are particularly vulnerable to various cyber threats because of their distributed architecture, diversity of communication protocols, lack of resources, and frequently insufficient security measures. The cyber security company Kespersky reports that millions of devices have been infiltrated globally in the last few years. The attacks on IoT devices in the Middle

* Corresponding author.
Email address: abdullah@nutech.edu.pk
ISSN: 2008-2045

East alone exceeded 330,000 incidents in 2022, highlighting the growing scale and geographic spread of IoT-targeted cyber threats [1]. IoT networks and the sensitive data they handle are seriously at risk from these threats, which range from malware infections and Distributed Denial of Service (DDoS) attacks to unauthorized access attempts and data breaches. With the exponential growth of IoT devices and the complexity of cyber threats, the deployment of robust IDS solutions is paramount to protect against potential vulnerabilities and malicious activities targeting IoT networks [2].

The classification of various attacks is an integral part of intrusion detection systems (IDS) within cybersecurity frameworks, as it enables IDS to distinguish between benign network activities and malicious behavior, thereby allowing security teams to focus on mitigating genuine threats and protecting network assets [3, 4]. This process enhances the accuracy, efficiency, and effectiveness of IDS in detecting, analyzing, and responding to cybersecurity incidents within IoT networks and other computing environments.

Artificial intelligence (AI) and deep learning (DL) have revolutionized IDS by providing advanced analytical capabilities to manage vast amounts of data generated by connected devices and improve decision-making [5, 6]. AI and DL-based IDS can detect unknown and zero-day threats, identify anomalies and abnormal patterns, and minimize false positives by differentiating between real network anomalies and benign deviations [7, 8]. However, their applicability in IoT environments must account for unique constraints such as limited computational resources, restricted memory, and the need for low-latency decision-making. For instance, while LSTM and biLSTM architectures can effectively model temporal dependencies in network traffic and improve detection accuracy [9], they require more memory and processing power, making them more suitable for deployment on edge or fog nodes. Conversely, lightweight models like FNNs and CNNs require less computational overhead and can be optimized for on-device inference, making them practical for real-time classification in constrained environments [10]. CNNs, in particular, have been shown to effectively capture spatial and local patterns in multivariate IoT traffic with significantly lower latency [11]. By systematically comparing these architectures under diverse datasets, this study identifies trade-offs in DL model selection for practical deployment across heterogeneous IoT settings.

Datasets used for intrusion detection in IoT networks play a crucial role in evaluating and benchmarking the performance of classifiers. Datasets such as CICIDS [12], IoT-23 [13], and NSL-KDD [14] are essential for evaluating and benchmarking intrusion detection systems (IDS) in IoT networks, offering a range of attack scenarios including DDoS, data exfiltration, and spoofing. These datasets, along with others like UNSW-NB15 [15] provide crucial resources for training and testing IDS models. However, a major limitation is that many of these datasets are outdated or were not originally designed for IoT environments, and later tailored, which may reduce their effectiveness in addressing current and evolving IoT-specific threats.

The objective of this study is to provide a systematic and comparative evaluation of four foundational deep learning architectures—FNN, CNN, LSTM, and biLSTM—across four diverse and recently published IoT datasets: BoTIoT, ToNIoT, CiCIoT, and WUSTL-IIoT-2021. Unlike prior studies that often focus on a single model or dataset, our work employs stratified undersampling to address class imbalance, utilizes diverse and recent datasets, and evaluates the performance of multiple deep learning models through comprehensive comparisons. This evaluation is necessary because IoT traffic characteristics and attack patterns evolve rapidly, and models trained on outdated datasets may not generalize well. Therefore, continuous benchmarking on new datasets is essential to assess the robustness and transferability of IDS models, making such comparative research valuable and timely.

The remainder of this paper is organized as follows: Section 2 provides a structured and thematic analysis of recent IoT IDS studies, highlighting key trends and gaps to position our contributions within the evolving landscape of IDS. Section 3 presents the system model, details the deep learning architectures employed, and discusses the mathematical formulation of the loss functions used to optimize model performance. Section 4 outlines the experimental design, including the selected datasets, preprocessing steps, and performance metrics used for evaluation. Section 5 presents the results and provides a detailed discussion on the performance of the models across different datasets and evaluation metrics. Section 6 concludes the study by summarizing key findings and outlining potential directions for future research.

2 Related Work

The integration of deep learning techniques in intrusion detection systems for IoT networks has gained significant attention from researchers due to its potential to enhance cybersecurity in dynamic environments. This section provides an in-depth analysis of recent studies and advancements in utilizing DL for IDS in IoT networks.

Early studies have explored deep learning models for IoT intrusion detection, such as FNN, CNN, LSTM, and bi-LSTM [16–21]. For instance, Almiani *et al.* [16] proposed a multi-layer deep recurrent neural network tailored for intrusion detection in IoT environments. Their model, evaluated on the NSL-KDD dataset, demonstrated strong detection performance, with detection rates of 98.27% for DoS, 97.35% for Probe, 64.93% for U2R, and 77.25% for R2L attacks. Vina *et al.* [22] experimented with DNN models having varying hidden layers and used a superparameter selection method to find the optimal number of layers. Vakili *et al.* [23] provide a detailed comparison of machine learning and deep learning models. Their findings suggest that ANN and CNNs often outperform LSTM models on feature-rich IoT datasets because they excel at capturing localized inter-feature patterns and hierarchical representations, while LSTMs depend on temporal continuity that may not be well-represented in intrusion detection data. However, most of these works rely on older or generalized datasets, which may limit their effectiveness against current IoT threats.

To overcome the limitations of single models, several studies proposed hybrid approaches that combine CNNs and RNNs. Nazir *et al.* [24] introduced CNN-LSTM hybrids to simultaneously capture spatial and temporal patterns, improving overall detection accuracy. Similarly, weighted LSTMs [25] and BiLSTMs [26] were used to enhance sequence learning and reduce overfitting. These models, while more powerful, introduce higher complexity, making them challenging to deploy on lightweight IoT devices.

Recent research has also explored the transformer-based models for intrusion detection [27]. For example, Fares *et al.* [28] demonstrated that self-attention architectures outperformed traditional RNNs on the BoTIoT dataset. Similarly, edge implicit weighting with an aggregated graph transformer architecture was proposed in [29] to enhance detection accuracy by leveraging both one-hop and two-hop neighbor information for contextual relational analysis. This method integrates multi-head attention to effectively capture patterns in highly dynamic traffic, and employs the Synthetic Minority Over-sampling Technique (SMOTE) to address class imbalance. While promising, these models require advanced hardware and hyperparameter tuning, making them less suitable for real-time IoT deployment. Our work complements these efforts by evaluating more interpretable and flexible models across varied IoT environments.

Handling class imbalance is another underexplored challenge in IoT IDS. While many studies use oversampling or SMOTE to artificially balance data, such methods may introduce noise or unrealistic traffic patterns [30]. In this work, we tested the performance of DL algorithms on balanced datasets obtained by applying stratified undersampling to maintain the originality of the datasets, contributing to a critical area of study.

Loss function engineering has also emerged as a promising research direction. Focal Loss (FL) [31] helps address imbalance by focusing on hard-to-classify examples. Hossain *et al.* [32] extended this by introducing DFL, which combines FL with Dual Cross Entropy (DCE) and applied it to improve the performance of image segmentation. Our work applies DFL in the context of IoT IDS, analyzing its effectiveness across multiple datasets and architectures.

Another limitation in existing literature is the reliance on outdated or single datasets, such as NSL-KDD and CICIDS2017. These datasets often lack diversity in attack types and fail to represent current IoT environments. More recent datasets—BoTIoT, ToNIoT, CiCIoT2022, and WUSTL-IIoT-2021—offer greater realism in traffic behavior and attack diversity. However, most of the studies does not conduct cross-dataset evaluations under diverse settings. Our work fills this gap by systematically comparing model behavior across four modern datasets using consistent preprocessing, loss functions, and evaluation metrics.

3 Proposed Solution

3.1 System Model

The proposed system is describes by a supervised learning setup where input data is represented as a matrix $\mathbf{X}$ with N samples, each sample being a vector $\mathbf{x}_i \in \mathbb{R}^D$ in a D-dimensional feature space. Associated with each sample is a label y_i, where $y_i \in \{1, \ldots, L\}$, indicating L distinct attack categories. The goal is to learn a function $f(\mathbf{x})$ that maps feature vectors to their corresponding labels. In supervised learning, this function is approximated by a learnable function $\hat{f}(\mathbf{x})$ using a labeled training dataset. The objective is to make $\hat{f}(\mathbf{x})$ as close as possible to the true underlying function $f(\mathbf{x})$ to accurately predict labels for new, unseen data points.

To ensure stable and efficient training across all architectures, we adopt a consistent set of hyperparameters guided by standard practices and preliminary experimentation. A learning rate of 0.001 is used to facilitate smooth convergence without large oscillations, while a batch size of 32 provides a balance between computational efficiency and stable gradient updates. Additionally, the training data is shuffled at the beginning of each epoch to prevent the model from overfitting to the input sequence and to encour-

age better generalization. The following section explains in detail the proposed DL models.

3.2 DL Models

In this study, we experimented with four DL models (FNN, CNN, LSTM, and biLSTM) as these represent four core categories of deep neural architectures. FNN serves as a baseline fully-connected model suitable for general-purpose, feature-based classification. CNN is utilized for its ability to learn local patterns and spatial hierarchies, making it effective for processing multivariate time-series data. LSTM is used to model long-range temporal dependencies, which are common in sequential network traffic and biLSTM extends this capability by processing data in both forward and backward directions, thereby capturing more context from the input sequence. These models were chosen for their collective ability to cover a range of representation capacities and temporal modeling strengths. Furthermore, the varying computational demands allow for comparative evaluation in terms of both performance and deployability in resource-constrained IoT environments.

Transformer-based architectures and Generative Adversarial Networks (GANs) have shown promising results in intrusion detection and related sequence modeling tasks. However, these models typically involve substantially higher computational complexity, larger memory footprints, and longer training and inference times compared to recurrent or convolutional architectures [33]. The datasets used in our study include devices such as temperature sensors, surveillance cameras, smart meters, and sensors and actuators. These devices operate on resource-constrained platforms such as Raspberry Pi boards or ARM Cortex-M-based microcontrollers, reflecting realistic computational and memory limitations. Therefore, given the high computational overhead and memory demands of Transformer-based and GAN-based architectures, we did not include them in our study.

3.2.1 FNN

Feedforward Neural Networks (FNNs) are versatile and widely used in applications such as pattern classification, clustering, regression, association, optimization, control, and forecasting. Typically, an FNN includes an input layer, an output layer, and several hidden layers. The weight matrices for the hidden and output layers are denoted as $\mathbf{W_h} \in \mathbb{R}^{Q\times P}$ and $\mathbf{W_o} \in \mathbb{R}^{N\times M}$, respectively, where Q represents the number of input neurons, P the number of hidden layer neurons, and M the number of output neurons. The hidden layer output is calculated as $H = f(XW_h + b_h)$, and the output layer's equation is $\hat{Y} = g(HWo + bo)$

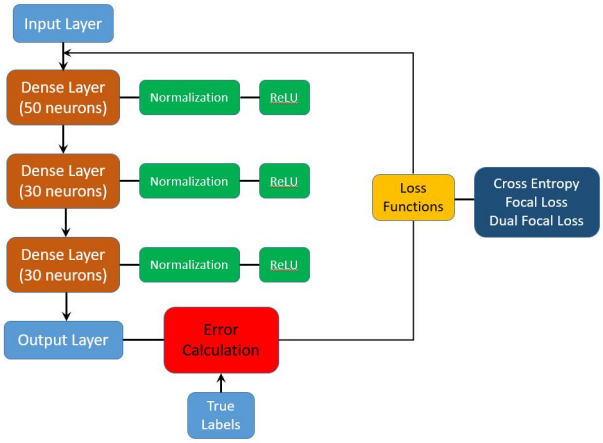

Figure 1. FNN architecture

with $f(.)$ and $g(.)$ as activation functions.

The FNN architecture adopted in this study follows the design proposed in [34], consisting of three fully connected hidden layers with a progressively decreasing number of neurons (50, 30, and 20), facilitating hierarchical feature abstraction and dimensionality reduction.

For classifying attacks in IoT systems, our proposed model uses an input layer for features followed by three fully connected layers with decreasing neurons (50, 30, and 20), each followed by layer normalization and ReLU activation for non-linearity. The final layer uses a softmax activation function to output probabilities for each attack class. Figure 1 shows the proposed architectures and parameters.

3.2.2 CNN

CNNs have demonstrated exceptional performance across various domains [35, 36]. In CNNs, input features undergo transformation into meaningful information through convolutions, which are then utilized to build subsequent layers of the neural network. A convolutional layer in CNNs serves the purpose of feature extraction and performs linear operations. Convolutions are achieved by employing multiple kernels or filters. Mathematically, a convolution operation can be expressed as:

$$\hat{Y} = x * k + b \tag{1}$$

Here, the kernel k is of dimensions $n \times m$, where n and m represent the height and width of the kernel respectively. x denotes the input to the convolution operation, and b represents the bias term.

Our proposed model for classifying attacks in IoT systems leverages a sequence input layer to handle temporal IoT data, followed by two convolutional layers. In each convolutional layer, the number of filters

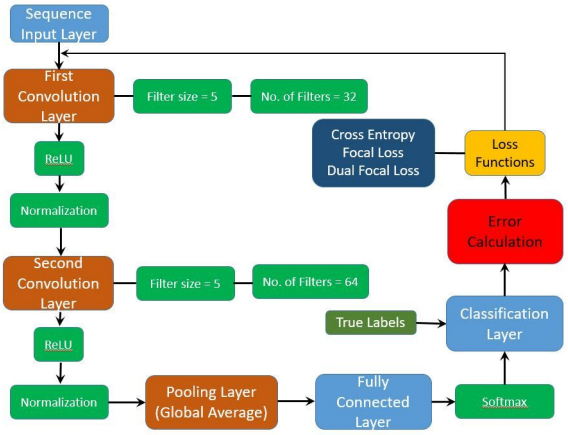

Figure 2. CNN architecture

is doubled compared to the previous layer, allowing for more complex feature extraction. This progressive increase in filter count has been shown to improve learning capacity in CNN-based IDS models, as demonstrated in [37], where deeper filters captured more complex attack signatures in time-dependent traffic. Each convolutional layer is followed by ReLU activation and layer normalization. A global average pooling layer is then applied to reduce feature dimensionality before feeding into the fully connected and softmax layers. We use causal padding to capture sequential dependencies effectively. The rectified linear unit (ReLU) activation function is applied after each convolutional layer. Additionally, layer normalization layers are incorporated to enhance convergence and stability during training. Global average pooling is then applied to aggregate features across time dimensions, reducing computational complexity and preventing overfitting. Subsequently, fully connected layers with ReLU activation and layer normalization are utilized for feature transformation and dimensionality reduction. The final layer comprises a fully connected layer with softmax activation, enabling multiclass classification of attacks. Figure 2 shows the proposed architectures and parameters.

3.2.3 LSTM and biLSTM

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to capture long-range dependencies and sequential patterns in data. At its core, LSTM consists of memory cells that store information over time and gates that regulate the flow of information into and out of these cells. The key components of an LSTM cell include the cell state (C_t), input gate (i_t), forget gate (f_t), and output gate (o_t). The cell state represents the long-term memory, the input gate determines how much new information to add, the forget gate controls how much information to discard, and the output gate regulates the output from the cell. The computations within an LSTM cell involve activation functions like the sigmoid function for gate outputs and the hyperbolic tangent (tanh) function for candidate cell state updates. The equations governing an LSTM cell's behavior are:

$$\begin{aligned} i_t &= \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \\ f_t &= \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ o_t &= \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \\ g_t &= \tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g) \\ C_t &= f_t \cdot C_{t-1} + i_t \cdot g_t \\ h_t &= o_t \cdot \tanh(C_t) \end{aligned} \tag{2}$$

Here, x_t represents the input at time step t, h_{t-1} is the previous hidden state, W and b are the weight matrices and bias terms respectively, and σ denotes the sigmoid activation function.

Our proposed model for classifying attacks in IoT systems begins with a sequence input layer tailored for temporal IoT data. Subsequently, three LSTM layers with varying numbers of hidden units are employed, each producing output based on the last time step. These LSTM layers capture long-term dependencies in sequential data, making them suitable for modeling temporal patterns in IoT attack data. A fully connected layer is then used for feature transformation and dimensionality reduction, followed by a softmax layer for multiclass classification of attacks. The model is trained using focal loss to address class imbalance by prioritizing hard-to-classify samples, thereby improving classification accuracy. In the subsequent experiment, we transitioned to using BiLSTM layers, which capture bidirectional dependencies by processing data in both forward and backward directions. This enables the model to better understand temporal patterns in IoT attack data, enhancing its ability to accurately classify attacks. The architecture uses increasing units in each LSTM and BiLSTM layer, with 64 units in the first layer, 128 in the second, and 128 in the final layer. The final layer is a dense layer with a softmax activation function for classification. Figure 3 shows the proposed architectures and parameters.

3.3 Loss Functions

3.3.1 Cross-Entropy Loss Function

The Cross-Entropy Loss function is a critical metric for evaluating the performance of deep learning models in multiclass classification problems. In such problems, the model predicts a probability distribution over all classes, and the Cross-Entropy Loss quantifies the divergence between this predicted distribution and the actual class distribution. Mathematically, for

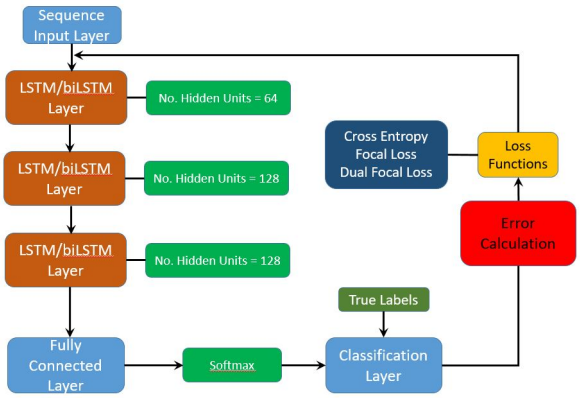

Figure 3. LSTM-biLSTM architecture

a classification problem with C classes, the Cross-Entropy Loss for a single sample is defined as:

$$\mathcal{L}_{CE}(Y, Z) = -\sum_{i=1}^{C} y_i \log(z_i) \tag{3}$$

where y_i is a binary indicator that equals 1 if class i is the correct class and 0 otherwise, and z_i is the predicted probability for class i produced by the softmax output layer.

The Cross-Entropy Loss penalizes incorrect high-confidence predictions more severely, encouraging the model to improve its accuracy and calibration, making it essential for evaluating the effectiveness of deep learning models in multiclass classification tasks.

3.3.2 Focal Loss Function

The Focal Loss (FL) function [31] extends Cross-Entropy Loss to better handle class imbalance—common in intrusion detection systems where some attack types occur far less frequently than others or normal traffic. In such cases, FL introduces a modulating factor that down-weights well-classified examples and focuses learning on hard examples. It is defined as:

$$\mathcal{L}_{FL}(Y, Z) = -\sum_{i=1}^{C} \alpha_i (1 - z_i)^{\gamma} y_i \log(z_i) \tag{4}$$

where Y is true label, Z is the predicted probability vector, C is the number of classes, α_i is a class weighting factor, and γ is a focusing parameter that determines the rate at which easy examples are down-weighted.

In IDS applications, y_i indicates whether the sample belongs to class i, and z_i is the predicted confidence. The term $(1 - z_i)^{\gamma}$ reduces the loss contribution from well-classified instances ($z_i \to 1$), concentrating learning on those the model finds difficult to classify correctly. By focusing more on challenging minority-class samples, FL improves the model's ability to detect rare and subtle attacks, making the IDS more robust and balanced across all traffic types.

3.3.3 Dual Focal Loss Function

The Dual Focal Loss (DFL) [32] extends Focal Loss by introducing a second modulating term to address overconfidence in incorrect predictions. The DFL for a single sample in a multiclass classification task is defined as:

$$\begin{aligned}\mathcal{L}_{DFL}(Y, Z) = &-\sum_{i=1}^{C} \alpha_i (1 - z_i)^{\gamma} y_i \log(z_i) \\ &-\sum_{i=1}^{C} \beta_i z_i^{\delta} (1 - y_i) \log(1 - z_i)\end{aligned} \tag{5}$$

Here:

- Y is ground truth label,
- Z is the predicted probability vector,
- C is the number of classes,
- α_i and β_i are class-specific weighting factors,
- γ and δ are focusing parameters for the two focal components.

The first term resembles the standard Focal Loss, applying a modulating factor $(1 - z_i)^{\gamma}$ to emphasize hard-to-classify true class instances. The second term introduces a new focal component, $(z_i)^{\delta}(1-y_i)\log(1-z_i)$, which increases the penalty for predictions where the model is overconfident but incorrect. This helps mitigate false negatives by discouraging overconfident misclassifications.

In IDS contexts, this dual mechanism ensures that the model learns effectively from both underrepresented attack classes and hard examples, while also penalizing overconfident predictions on incorrect classes. The flexibility is provided through α and β, γ and δ, which we set to 1.0 in our experiments, consistent with the defaults in [32].

4 Experiment Design

4.1 Datasets

Following is the details of datasets used in the experiments.

4.1.1 BoTIoT Dataset

The Bot-IoT dataset [38] was generated using multiple tools to create diverse botnet scenarios, and packet capture files were organized by attack types.

Additionally, simulated Internet of Things sensors were utilized. Statistical analysis of the dataset's features was conducted using Correlation Coefficient and Joint Entropy techniques. To enhance machine learning model performance and prediction effectiveness, more features were extracted and incorporated into the dataset [39]. It is suggested that these extracted features be labeled, including attack flow, categories, and sub-categories, to improve performance further.

The dataset encompasses four sub-components within the IoT testbed: simulation, networking platform, feature extraction, and forensics analytics. Alongside Normal traffic data, it includes data related to Denial of Service (DoS), Distributed Denial of Service (DDoS), Reconnaissance attack, and Theft. The dataset comprises multiple features and includes both training and testing datasets. The distribution of data items in these datasets is detailed in Table 1, highlighting the varying proportions of samples for different attack types. Notably, the training dataset has a significant representation of DDoS and DoS samples, accounting for 53% and 45%, respectively, while Normal and Theft classes have minimal representation with 0.012% and 0.002% samples, respectively.

Table 1. Class distribution and attack types in the BoTIoT dataset

Category Name	Original Records	Balanced Records
DDoS	38531238	1578
Normal	9063	1578
DoS	33003929	1578
Theft	1578	1578
Reconnaissance	1789474	1578

4.1.2 Class Distribution and Attack Types in the ToNIoT Dataset

The diverse data sources gathered from Telemetry datasets of IoT and IIoT sensors, Operating systems datasets of Windows 7 and 10, as well as Ubuntu 14 and 18 TLS and Network traffic datasets, are included in the datasets [40, 41]. The datasets were gathered from an authentic and expansive network constructed at the Australian Defence Force Academy (ADFA) Cyber Range and IoT Labs, the School of Engineering and Information Technology (SEIT), UNSW Canberra. For the industrial 4.0 network, which consists of IIoT and IoT networks, a new testbed network was established. To manage the interconnection between the three levels of IoT, Cloud, and Edge/Fog systems, the testbed was established utilising numerous virtual machines and hosts running Linux, Windows, and Kali operating systems. Numerous methods of attack, including denial-of-service (DoS), distributed denial-of-service (DDoS), and ransomware, are employed against web apps, Internet of Things gateways, and computer systems throughout the IoT/IIoT network. In order to collect several normal and cyber-attack events from network traffic, Windows audit traces, Linux audit traces, and telemetry data of IoT services, the datasets were gathered in parallel processing.

Throughout the IIoT network, various hacking techniques, including ransomware, DDoS, and denial-of-service attacks, were launched against computer systems, IoT gateways, and web applications.

In order to produce standard features and their labels, the dataset was filtered. The whole set of datasets was cleaned and prepared as CSV files that could be opened on any platform. Table 2 provides a thorough distribution of data items in these datasets, showing the differing proportions of samples for various assault types.

Table 2. Class distribution and attack types in the ToNIoT dataset

Category Name	Original Records	Balanced Records
normal	796380	7280
backdoor	508116	7280
dos	3375328	7280
ddos	6165008	7280
mitm	1052	-
injection	452659	7280
ransomware	72805	7280
password	1718568	7280
xss	2108944	7280
scanning	7140161	7280

4.1.3 CiCIot Dataset

The Canadian Institute for Cybersecurity (CIC) created the CICIoT dataset [42], which is a valuable resource for creating and testing deep learning-based intrusion detection systems (IDS) in Internet of Things contexts. In an IoT topology with 105 devices, 33 assaults are carried out while the dataset is being generated. The seven categories into which these assaults are divided are DDoS, DoS, Recon, Web-based, Brute Force, Spoofing, and Mirai. Lastly, all assaults are carried out by malevolent IoT devices that are directed towards other IoT devices.

The IoT lab at the CIC and its gadgets are depicted in Figure 4. IoT devices are, in fact, dispersed across the lab; some are positioned on the floor, others on the walls, and some on the table. The local network topology was used and the lab was equipped with multiple power connectors. To further organize the network and IoT devices, there were racks and storage rooms. The original records and records after balancing the CiCIoT datashet are shown in Table 3.

Figure 4. CiCIoT lab

Table 3. Balanced CiCIoT dataset

Category Name	Original Records	Balanced Records
BenignTraffic	1091170	12984
DNS_Spoofing	177786	12984
DDoS	33768100	12984
DictionaryBruteForce	12984	12984
MITM	305590	12984
DoS	8039344	12984
Recon	352244	12984
Mirai	2617378	12984
Webbased	24658	12984

4.1.4 WUSTL-IIoT-2021 Dataset

The WUSTL-IIoT-2021 dataset [43], developed by Washington University in St. Louis (WUSTL), is a comprehensive and specialized dataset designed specifically for research and development in the realm of Industrial Internet of Things (IIoT) security. This dataset is tailored to address the unique challenges and complexities associated with securing industrial systems and IoT devices in industrial settings.

The WUSTL-IIoT-2021 dataset encompasses a diverse range of data sources and attributes that are pertinent to IIoT security analysis. It includes telemetry data from various sensors commonly found in industrial environments, such as temperature sensors, pressure sensors, and motion sensors. Additionally, the dataset incorporates network traffic data, system logs, and other relevant information that reflects the operational aspects of industrial IoT systems.

One notable aspect of the WUSTL-IIoT-2021 dataset is its extensive feature set. It comprises a significant number of features extracted from the collected data sources, capturing different aspects of device behavior, network communications, and system states. These features are essential for building and training machine learning and deep learning models for various cybersecurity applications, including anomaly detection, intrusion detection, predictive maintenance, and system monitoring.

The WUSTL-IIoT-2021 dataset is organized into multiple categories or classes, representing different types of normal and abnormal behavior in industrial IoT systems. The dataset includes labeled data instances categorized into these classes, which enables researchers and practitioners to develop, validate, and evaluate their security algorithms and solutions effectively.

Table 4. Balanced WUSTL-IIoT-2021 dataset

Category Name	Original Records	Balanced Records
Attacked	87016	87016
Normal	1107448	87016

We used three datasets in our experiment and test the results on CNN, FNN, LSTM, and biLSTM architecture.

4.2 Data Preparation and Pre-processing

While the original datasets are available as CSV files with multiple features and records, we prepared data as per the requirements of the experiments. First of all, we removed the columns (features) having all missing values. Afterwards, we removed the records with missing values. Although other options such as duplication, interpolation, average value replacement are possible, but we aim to maintain the originality of the data therefore, removing the records with missing value is chosen over other options. Then, the text features such as upd packet, source ip, destination ip, are converted into numeric values using one-hot-encoding technique. It is important to note that in this study, we did not apply custom pre-processing to restructure input features into sequential or temporal formats. All deep learning models were trained on the available feature vectors provided by the dataset publishers.

Data imbalance is a common challenge in many machine learning tasks, particularly in domains like intrusion detection where certain classes of data are significantly underrepresented compared to others. To address this issue and ensure fair model training, we employed a data balancing technique known as undersampling in our study. Undersampling involves reducing the number of samples in the majority class to achieve a more balanced distribution across all classes. Unlike oversampling methods that generate synthetic data or duplicate existing samples, undersampling directly removes samples from the majority class. This approach was chosen to preserve the originality of the dataset while improving class balance, as it ensures that the remaining samples in the majority class are still authentic and representative of the actual data distribution.

In our study, we retained all samples from minority classes to preserve rare but critical instances. For majority classes, we applied stratified undersampling by selecting samples proportionally from each sub-class. This step was crucial to prevent bias and maintain

the integrity of the dataset. While undersampling helped address the challenge of class imbalance, it also comes with considerations. One such consideration is the potential loss of information due to the removal of samples, which could impact the model's ability to capture complex patterns and nuances in the data. In the context of deep learning techniques, data balancing through undersampling plays a vital role in improving model performance, reducing overfitting, and ensuring fair representation of all classes during training. By including this data balancing approach in our methodology, we aimed to enhance the robustness and accuracy of our deep learning-based intrusion detection system. The samples against train and test after balancing are shown in Table 5.

Table 5. Post-Balancing sample distribution

Dataset	Training Samples	Testing Samples
WUSTL-IIoT-2021	139226	34806
CiCIoT	93484	23372
BoTIoT	6312	1578
TonIoT	52416	13104

4.3 Performance Metrics

Several important indicators were taken into account when assessing the performance of our intrusion detection models in order to determine how well they detected and classified attacks. These indicators offer perceptions into several facets of model performance, enabling a thorough assessment of the suggested intrusion detection system based on deep learning.

4.3.1 Accuracy

A key indicator of the general accuracy of the model's predictions is accuracy. It is a representation of the proportion of correctly classified occurrences to all instances within the dataset. If the dataset is unbalanced, a higher accuracy suggests that the model is producing more accurate predictions, but it might not give a whole picture.

4.3.2 Recall

Recall, which is sometimes referred to as sensitivity or true positive rate, measures how well a model can identify instances of a given class among all of the real instances of that class in the dataset. It is especially crucial in situations where there is a significant cost associated with missing a positive instance, such identifying an attack.

4.3.3 F1 Score

The F1 score is a balanced metric that takes into account both precision and recall. It provides a single figure that strikes a compromise between the trade-off between false positives and false negatives. It is computed as the harmonic mean of precision and recall. Recall and precision are better balanced when the F1 score is higher.

4.3.4 MCC (Matthews Correlation Coefficient)

False positives, false negatives, true positives, and true negatives are all taken into consideration by the Matthews Correlation Coefficient (MCC). It offers a fair assessment of the model's effectiveness, particularly when there is an imbalance between the classes. The MCC is a number between -1 and 1, where 1 denotes ideal forecasts, 0 represents random predictions, and -1 denotes total discrepancy between predictions and actual results.

5 Results and Discussions

In this section, we demonstrate and explain the performance of CNN, FNN, LSTM, and biLSTM models presented in Section 3.2 on the datasets discussed in above sections. For all of these models, three loss functions are tested; cross entropy, focal loss, and dual focal loss.

5.1 Analysis of DL Models for BoTIoT Dataset

Distinct patterns emerge when comparing the results of various loss functions across all deep learning (DL) models. We can see from Table 6 that Cross Entropy Loss consistently achieves the highest overall performance for all models. For the CNN, LSTM, and biLSTM models, the performance of FL and DFL is comparable to Cross Entropy Loss. However, the FNN model consistently exhibits significantly lower performance compared to the CNN, LSTM, and biLSTM models across all loss functions. This degradation is primarily due to the shallow architecture of the FNN, which lacks sufficient depth and the inability to model temporal dependencies. FL and DFL emphasize harder-to-classify samples by down-weighting the loss from easier samples, which benefits deeper networks capable of learning complex features. However, FNN fails to leverage this due to limited representational capacity and insufficient depth.

Comparing these results with the findings of [34], where random oversampling (CNN-Rand and FNN-Rand) and synthetic data generation (CNN-STAGSamp and FNN-CTAGSamp) were used to balance the dataset, reveals further insights. Random oversampling results in decreased performance for the FNN model. Specifically, there is a noticeable decline in recall, precision, F1 score, and MCC, despite

Table 6. DL model performance with different loss functions on BoTIoT balanced dataset

DL Model	Loss Func	Acc	Recall	Prec	F1 Score	MCC
biLSTM	Cross	96.2%	0.965	0.963	0.9632	0.954
	FL	95%	0.949	0.953	0.949	0.939
	DFL	96.6%	0.967	0.971	0.968	0.96
LSTM	Cross	96.3%	0.965	0.966	0.965	0.956
	FL	96.7%	0.967	0.969	0.9675	0.959
	DFL	96.8%	0.969	0.971	0.968	0.962
CNN	Cross	98.8%	0.989	0.988	0.9884	0.986
	FL	98.3%	0.984	0.983	0.983	0.979
	DFL	98.4%	0.985	0.9844	0.9843	0.981
FNN	Cross	86.3%	0.860	0.867	0.859	0.829
	FL	83.7%	0.833	0.843	0.833	0.796
	DFL	77.7%	0.777	0.8112	0.7631	0.7321
FNN-Rand [34]	FL	86.1%	0.499	0.499	0.528	0.762
FNN-CTGANSamp [34]	FL	88.1%	0.865	0.499	0.554	0.789
CNN-Rand [34]	FL	80.8%	0.784	0.535	0.568	0.655
CNN-CTGAN [34]	FL	81.68%	0.799	0.430	0.454	0.688

a higher accuracy compared to the downsampling method. The higher accuracy in FNN-Rand is attributed to the model's exposure to a larger dataset, including repeated samples from minority classes, which aids generalization. However, recall, precision, F1 score, and MCC are lower because upsampling can lead to overfitting on the duplicated minority class samples.

In the case of FNN-CTGANSamp, there is an improvement in recall, but precision, F1 score, and MCC are not significantly affected. Synthetic data generation enhances recall by increasing the number of minority class samples in the training set, enabling the model to better identify all relevant instances. However, precision remains low because synthetic data can introduce noise. If synthetic samples do not accurately represent true minority class instances, the model may incorrectly classify more instances as positive, increasing false positives and reducing precision.

This trend is more pronounced in CNN models, where the results indicate that the originality of samples is crucial for capturing true features. Random sampling and synthetic data generation compromise data originality, leading to reduced performance. Thus, we can conclude that maintaining the originality of the dataset while balancing is more critical than merely increasing the amount of data for training.

5.2 Analysis of DL Models for CiCIoT Dataset

CiCIoT dataset is a novel and extensive IoT attack dataset to foster the development of security analytics applications in real IoT operations. To accomplish this, 33 attacks are executed in an IoT topology composed of 105 devices. Table 7 presents the performance analysis of various DL models for different loss function on CiCIoT dataset.

For the CNN model, DFL achieved the highest overall performance with an accuracy of 0.7033, recall of 0.70348, precision of 0.72331, F1 score of 0.70181, and MCC of 0.66727. Cross Entropy Loss and FL also performed well, with Cross achieving an accuracy of 0.69622 and FL slightly better at 0.69815. The metrics

Table 7. DL model performance with different loss functions on CiCIoT balanced dataset

DL Model	Loss Func	Acc	Recall	Prec	F1 Score	MCC
biLSTM	Cross	71.2%	0.711	0.735	0.710	0.678
	FL	71%	0.712	0.737	0.7098	0.6783
	DFL	70.7%	0.7073	0.7356	0.7053	0.674
LSTM	Cross	71.8%	0.717	0.744	0.7174	0.686
	FL	69.2%	0.693	0.714	0.694	0.656
	DFL	71.2%	0.711	0.7335	0.7092	0.6764
CNN	Cross	69.6%	0.696	0.715	0.694	0.658
	FL	69.8%	0.698	0.709	0.698	0.659
	DFL	70.3%	0.704	0.723	0.701	0.667
FNN	Cross	46.2%	0.462	0.503	0.414	0.377
	FL	36.9%	0.370	0.459	0.355	0.309
	DFL	37.9%	0.382	0.480	0.363	0.325

for recall, precision, F1 score, and MCC were similarly close between these two loss functions, indicating that while all three loss functions are effective, DFL provides a slight edge.

In contrast, the FNN model showed a significant decline in performance with FL and DFL compared to Cross Entropy Loss. Cross Entropy Loss achieved an accuracy of 0.46156, recall of 0.46216, precision of 0.50346, F1 score of 0.4136, and MCC of 0.37744. Both FL and DFL resulted in lower metrics, with FL having the lowest accuracy at 0.36924 and DFL slightly better but still lower than Cross Entropy at 0.37911. This indicates that the FNN model does not benefit as much from FL and DFL as it does from Cross Entropy Loss.

The LSTM model showed strong performance across all loss functions, with Cross Entropy Loss yielding the highest accuracy at 0.71769 and MCC at 0.68571. FL and DFL, while slightly lower in accuracy and MCC, still demonstrated competitive performance. FL achieved an accuracy of 0.69252 and an MCC of 0.6564, while DFL had an accuracy of 0.71264 and an MCC of 0.67643. These results suggest that LSTM can effectively utilize different loss functions, but Cross Entropy provides the best balance of accuracy and MCC.

Similarly, the biLSTM model performed well with all three loss functions. Cross Entropy Loss achieved the highest accuracy at 0.71249 and MCC at 0.67785. FL and DFL showed comparable results, with FL achieving an accuracy of 0.71004 and MCC of 0.67826, while DFL had an accuracy of 0.707 and MCC of 0.67384. The metrics for recall, precision, and F1 score were also high across all loss functions, indicating the robustness of the biLSTM model to different loss functions.

Table 8. DL model performance with different loss functions on WUSTIIoT balanced dataset

DL Model	Loss Func	Acc	Recall	Prec	F1 Score	MCC
biLSTM	Cross	99.5%	0.99505	0.99499	0.995	0.99004
	FL	99.84%	0.99839	0.99839	0.99839	0.99678
	DFL	99.88%	0.9988	0.99879	0.99879	0.99759
LSTM	Cross	99.93%	0.99928	0.99928	0.99928	0.99856
	FL	99.85%	0.9985	0.99851	0.99851	0.99701
	DFL	99.91%	0.99914	0.99914	0.99914	0.99828
CNN	Cross	99.6%	0.9962	0.9963	0.99629	0.9926
	FL	99.63 %	0.9963	0.9964	0.9963	0.9928
	DFL	99.601%	0.99599	0.99604	0.99601	0.99204
FNN	Cross	73.4%	0.73313	0.8184	0.7141	0.545
	FL	73.1%	0.7293	0.8181	0.7105	0.5401
	DFL	72.99%	0.7308	0.8154	0.7107	0.5396

5.3 Analysis of DL Models for WUSTL-IIOT-2021 Dataset

A comprehensive analysis of presented DL models utilizing different loss functions evaluated across multiple performance metrics on WUSTL-IIOT-2021 dataset is shown in Table 8.

It can be seen from the table that the CNN model shows good performance across all metrics with minimal variation among the loss functions. This can be attributed to the simpler binary classification task and the cleaner, less noisy nature of the WUSTL dataset, which allows the LSTM to effectively capture temporal dependencies. In contrast, datasets like CiCIoT are more complex and noisy, limiting the model's generalization capability.

In contrast, the FNN model displays significantly lower performance metrics compared to CNN, LSTM, and biLSTM, particularly in accuracy and MCC. Despite using different loss functions, the FNN model consistently achieves similar precision scores, indicating a fundamental limitation in its ability to generalize compared to more complex models. The slight variations in recall and F1 score among the loss functions indicate minor improvements, but overall, the FNN model's performance remains moderate.

The LSTM model stands out with exceptionally high scores across all metrics, particularly with Cross Entropy Loss, achieving excellent performance. The slightly lower but still high performance with FL and DFL suggests that while these loss functions are effective, Cross Entropy Loss may be better suited for the LSTM model's architecture and the nature of the dataset, providing superior generalization capabilities.

The biLSTM model also demonstrates strong performance, especially with FL and DFL, achieving high metrics across the board. The slightly lower scores with Cross Entropy Loss indicate that biLSTM benefits more from the specialized handling of class imbalance provided by Focal Loss and DFL. This highlights the biLSTM model's sensitivity to the choice of loss function, which can significantly impact its classification performance.

5.4 Analysis of DL Models for ToNIoT Dataset

In Table 9, we present the performance comparison of proposed DL algorithm for different loss function on the ToNIoT dataset.

Similar to the other datasets, the CNN model demonstrates solid performance across all metrics and loss functions. Using Cross Entropy Loss, CNN

Table 9. DL model performance with different loss functions on ToNIoT balanced dataset

DL Model	Loss Func	Acc	Recall	Prec	F1 Score	MCC
biLSTM	Cross	96.5%	0.9649	0.967	0.965	0.961
	FL	95.9%	0.959	0.9588	0.9584	0.9535
	DFL	96.7%	0.9668	0.9671	0.967	0.963
LSTM	Cross	95.6%	0.955	0.958	0.956	0.951
	FL	94.1%	0.941	0.942	0.940	0.933
	DFL	91.93%	0.9189	0.9253	0.9200	0.9113
CNN	Cross	95.2%	0.9517	0.9521	0.9515	0.9457
	FL	94.6%	0.9462	0.9471	0.9461	0.9397
	DFL	95.8%	0.95798	0.9584	0.9581	0.9529

achieves high accuracy, precision, and F1 score. Focal Loss and DFL results in slightly lower metrics. Despite its theoretical merits, the performance improvement of DFL over FL is modest in our experiments. This is because the datasets were already balanced through undersampling, which reduced the effect of sample imbalance. Therefore, the added benefit of dynamic focusing (in DFL) becomes less pronounced when class distributions are already equalized.

The LSTM model shows commendable performance with Cross Entropy Loss, achieving high accuracy, recall, precision, and F1 score. Focal Loss slightly under-performs compared to Cross Entropy Loss, suggesting that while LSTM is robust as compared to CNN, it still struggle with balanced data when FL and DFL loss function are used. DFL, however, shows a noticeable drop in performance, with lower accuracy and precision, indicating that DFL might not be as effective for LSTM in this specific context.

The biLSTM model stands out with exceptional performance, especially with DFL. Using Cross Entropy Loss, biLSTM achieves high accuracy (0.96489) and precision (0.96711). Focal Loss provides slightly lower yet still impressive metrics. However, with DFL, biLSTM reaches the highest accuracy (0.96657) and precision (0.96713), suggesting that the biLSTM model benefits significantly from DFL, which helps in refining its predictive capabilities by focusing more on hard-to-classify instances.

The CNN and biLSTM models exhibit robust performance across all loss functions, with DFL providing a slight edge in precision and accuracy. The LSTM model performs exceptionally well with Cross Entropy Loss. The choice of loss function is crucial, with DFL generally enhancing the performance of more complex models like CNN and biLSTM, while Cross Entropy remains a strong performer for LSTM.

5.5 Performance Comparison with Conventional ML Models

In this section, we compare the performance of the proposed deep learning (DL) models against conventional machine learning (ML) algorithms—namely, Support Vector Machine (SVM), Random Forest (RF), and K-Nearest Neighbors (KNN)—across multiple industrial IoT (IIoT) benchmark datasets. The results are summarized in Table 10.

The comparison between conventional ML models and presented DL approaches reveals that while traditional ML methods such as Random Forest, SVM, and KNN achieve excellent results on small, well-balanced datasets like BoT-IoT, they struggle to maintain this performance on larger and more complex datasets. For instance, Random Forest achieves an impressive 99.93% accuracy on the BoT-IoT dataset, which contains only 1,578 samples of each class. This high performance is largely due to the simplicity and limited scale of the dataset, where traditional ML models are effective in capturing patterns without the need for deep feature extraction or complex temporal modeling.

However, the performance of ML models declines significantly on larger datasets such as CiCIoT and ToNIoT which includes a far greater number of samples and exhibits higher complexity in class distributions and feature relationships. On CiCIoT, SVM accuracy drops to 56.06%, with similarly reduced scores for other metrics. This sharp contrast highlights the limitations of ML models when faced with high-dimensional and noisy data typical of real-world IoT environments. In contrast, DL models—especially those based on CNNs and LSTMs—offer more consistent performance across datasets due to their capacity to learn hierarchical and temporal features. These strengths make DL models more suitable for deployment in dynamic, data-rich IoT scenarios, where adaptability and generalization are critical. It is important to note that, all evaluations were conducted on a single stratified train-test split to enable consistent comparison across models and loss functions. However, this limits statistical generalizability and may introduce partition-specific bias. Future work will incorporate repeated evaluations using k-fold cross-validation and confidence-based statistical testing to validate robustness across different data splits.

6 Conclusions and Future Work

In this study, we conducted a comprehensive performance analysis of four deep learning models: CNN, FNN, LSTM, and biLSTM. We evaluated these models using three distinct loss functions: cross-entropy, focal loss, and dual focal loss. Our experiments utilized four diverse datasets—BoTIoT, CiCIoT, ToNIoT, and WUSTL-IIOT-2021—to assess the impact of various feature sets and dataset characteristics on the models and loss functions. The datasets were balanced through undersampling to preserve their origi-

Table 10. Performance comparison of traditional ML models on balanced datasets

Dataset	Model	Accuracy (%)	Recall	Precision	F1-Score	MCC
BoTIoT	SVM	95.03	0.9514	0.9535	0.9498	0.9395
	Random Forest	99.93	0.9993	0.9994	0.9993	0.9992
	KNN	99.35	0.9935	0.9934	0.9935	0.9918
CiCIoT	SVM	56.06	0.5606	0.5630	0.5446	0.4939
	Random Forest	81.34	0.8134	0.8137	0.8135	0.7869
	KNN	78.44	0.7844	0.7998	0.7826	0.7581
WUSTL-IIoT-2021	SVM	99.55	0.9955	0.9955	0.9955	0.9910
	Random Forest	99.99	0.9999	0.9999	0.9999	0.9997
	KNN	99.87	0.9987	0.9987	0.9987	0.9975
ToNIoT	SVM	83.32	0.8332	0.8564	0.8063	0.8057
	Random Forest	99.76	0.9976	0.9976	0.9976	0.9973
	KNN	99.76	0.9976	0.9976	0.9976	0.9973

nal characteristics. Our findings indicate that data balancing via undersampling, which maintains the integrity of the original samples, perfroms better than balancing methods using oversampling and synthetic data generation across all deep learning models. Furthermore, the results consistently demonstrate that CNN, LSTM, and biLSTM models surpass FNN models in performance on all datasets. This superior performance is attributed to their enhanced ability to capture complex features and temporal dependencies, which are critical in the context of IoT-based intrusion detection systems. Despite promising results, this study is limited by its use of balanced datasets, lack of evaluation on zero-day attacks, and the absence of analysis on computational efficiency for real-time deployment in resource-constrained environments. In future work, we plan to extend our findings by evaluating the proposed models on highly imbalanced and streaming datasets to better simulate real-world deployment conditions. This includes analyzing model performance under varying degrees of class imbalance and applying imbalance-aware techniques such as cost-sensitive learning, class-weighted loss functions, and oversampling strategies like SMOTE. These approaches will help us assess the robustness and generalizability of our models beyond balanced training environments. In addition, we will incorporate k-fold cross-validation and statistical testing (e.g., t-tests and confidence intervals) to ensure that reported performance trends are statistically significant and not due to dataset partitioning bias. In parallel, we aim to explore the integration of transformer-based and attention-enhanced architectures to improve temporal representation and model interpretability.

References

[1] Kaspersky Lab. Kaspersky blocked over 330 thousand attacks on iot devices in the middle east in 2022, 2023. URL https://me-en.kaspersky.com/about/press-releases/kaspersky-blocked-over-330-thousand-attacks-on-iot-devices-in-the-middle-east-in-2022. Accessed: 2025-05-15.

[2] Alyazia Aldhaheri, Fatima Alwahedi, Mohamed Amine Ferrag, and Ammar Battah. Deep learning for cyber threat detection in iot networks: A review. *Internet of Things and Cyber-Physical Systems*, 2023.

[3] Durgesh Srivastava, Rajeshwar Singh, Chinmay Chakraborty, Sunil Kr Maakar, Aaisha Makkar, and Deepak Sinwar. A framework for detection of cyber attacks by the classification of intrusion detection datasets. *Microprocessors and Microsystems*, 105:104964, 2024.

[4] Azar Abid Salih and Adnan Mohsin Abdulazeez. Evaluation of classification algorithms for intrusion detection system: A review. *Journal of Soft Computing and Data Mining*, 2(1):31–40, 2021.

[5] Umamah bint Khalid, Muddasar Naeem, Fabrizio Stasolla, Madiha Haider Syed, Musarat Abbas, and Antonio Coronato. Impact of ai-powered solutions in rehabilitation process: Recent improvements and future trends. *International Journal of General Medicine*, pages 943–969, 2024.

[6] Antonio Coronato, Muddasar Naeem, Giuseppe De Pietro, and Giovanni Paragliola. Reinforcement learning for intelligent healthcare applications: A survey. *Artificial Intelligence in Medicine*, 109:101964, 2020.

[7] Zahedi Azam, Md Motaharul Islam, and Mo-

hammad Nurul Huda. Comparative analysis of intrusion detection systems and machine learning based model analysis through decision tree. *IEEE Access*, 2023.

[8] Francisco L Loaiza, John D Birdwell, George L Kennedy, and Dale Visser. *Utility of artificial intelligence and machine learning in cybersecurity*. JSTOR, 2022.

[9] Yanxia Sun and Zenghui Wang. Intrusion detection in iot and wireless networks using image-based neural network classification. *Applied Soft Computing*, page 113236, 2025.

[10] Nathan Shone, Tran N Ngoc, Vu D Phai, and Qi Shi. A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1): 41–50, 2018.

[11] Ahmad Javaid, Qussai Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pages 21–26, 2016.

[12] Iman Sharafaldin, Arash Habibi Lashkari, Ali A Ghorbani, et al. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116, 2018.

[13] Alberto Parmisano, Sebastian Garcia, and M Jose Erquiaga. A labeled dataset with malicious and benign iot network traffic. *Stratosphere Laboratory: Praha, Czech Republic*, 2020.

[14] Yahya Al-Hadhrami and Farookh Khadeer Hussain. Real time dataset generation framework for intrusion detection systems in iot. *Future Generation Computer Systems*, 108:414–423, 2020.

[15] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 Military Communications and Information Systems Conference (MilCIS)*, pages 1–6, 2015. .

[16] Muder Almiani, Alia AbuGhazleh, Amer Al-Rahayfeh, Saleh Atiewi, and Abdul Razaque. Deep recurrent neural network for iot intrusion detection system. *Simulation Modelling Practice and Theory*, 101:102031, 2020.

[17] Rasheed Ahmad, Izzat Alsmadi, Wasim Alhamdani, and Lo'ai Tawalbeh. A comprehensive deep learning benchmark for iot ids. *Computers & Security*, 114:102588, 2022.

[18] Yazan Otoum, Dandan Liu, and Amiya Nayak. Dl-ids: a deep learning–based intrusion detection framework for securing iot. *Transactions on Emerging Telecommunications Technologies*, 33 (3):e3803, 2022.

[19] Muhammad Almas Khan, Muazzam A Khan, Sana Ullah Jan, Jawad Ahmad, Sajjad Shaukat Jamal, Awais Aziz Shah, Nikolaos Pitropakis, and William J Buchanan. A deep learning-based intrusion detection system for mqtt enabled iot. *Sensors*, 21(21):7016, 2021.

[20] Bipraneel Roy and Hon Cheung. A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network. In *2018 28th international telecommunication networks and applications conference (ITNAC)*, pages 1–6. IEEE, 2018.

[21] Hai-Viet Le, Quoc-Dung Ngo, and Van-Hoang Le. Iot botnet detection using system call graphs and one-class cnn classification. *Int. J. Innov. Technol. Explor. Eng*, 8(10):937–942, 2019.

[22] Ravi Vinayakumar, Mamoun Alazab, K Padannayil Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[23] Meysam Vakili, Mohammad Ghamsari, and Masoumeh Rezaei. Performance analysis and comparison of machine and deep learning algorithms for iot data classification. *arXiv preprint arXiv:2001.09636*, 2020.

[24] Ahsan Nazir, Jingsha He, Nafei Zhu, Saima Siraj Qureshi, Siraj Uddin Qureshi, Faheem Ullah, Ahsan Wajahat, and Muhammad Salman Pathan. A deep learning-based novel hybrid cnn-lstm architecture for efficient detection of threats in the iot ecosystem. *Ain Shams Engineering Journal*, 15(7):102777, 2024.

[25] Meryem Amar and Bouabid El Ouahidi. A weighted lstm deep learning for intrusion detection. In *International Conference on Advanced Communication Systems and Information Security*, pages 170–179. Springer, 2019.

[26] Abdullah Alqahtani. Optimized deep autoencoder and bilstm for intrusion detection in iots-fog computing. *Multimedia Tools and Applications*, 84(8):4907–4943, 2025.

[27] Uday Chandra Akuthota and Lava Bhargava. Transformer based intrusion detection for iot networks. *IEEE Internet of Things Journal*, 2025.

[28] Ibrahim A Fares, Ahmed G Abdellatif, Mohamed Abd Elaziz, Mansour Shrahili, Adham Elmahallawy, Rana Muhammad Sohaib, Mahmoud A Shawky, and Syed Tariq Shah. Deep transfer learning based on hybrid swin transformers with lstm for intrusion detection systems in iot environment. *IEEE Open Journal of the Communications Society*, 2025.

[29] C Karpagavalli and M Kaliappan. Edge implicit weighting with graph transformers for robust

intrusion detection in internet of things network. *Computers & Security*, 150:104299, 2025.

[30] Bashar Hamad Aubaidan, Rabiah Abdul Kadir, and Mohamad Taha Ijab. A comparative analysis of smote and cssf techniques for diabetes classification using imbalanced data. *J. Comput. Sci*, 20:1146–1165, 2024.

[31] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[32] Md Sazzad Hossain, John M Betts, and Andrew P Paplinski. Dual focal loss to address class imbalance in semantic segmentation. *Neurocomputing*, 462:69–87, 2021.

[33] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6):1–28, 2022.

[34] Ayesha S Dina, AB Siddique, and D Manivannan. A deep learning approach for intrusion detection in internet of things using focal loss function. *Internet of Things*, 22:100699, 2023.

[35] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4293–4302, 2016.

[36] Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data*, 8:1–74, 2021.

[37] Yulin Wang, Jinheng Wang, and Honglin Jin. Network intrusion detection method based on improved cnn in internet of things environment. *Mobile Information Systems*, 2022(1):3850582, 2022.

[38] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, 100:779–796, 2019.

[39] Muhammad Shafiq, Zhihong Tian, Ali Kashif Bashir, Xiaojiang Du, and Mohsen Guizani. Corrauc: a malicious bot-iot traffic detection method in iot network using machine-learning techniques. *IEEE Internet of Things Journal*, 8(5):3242–3254, 2020.

[40] Nour Moustafa. A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. *Sustainable Cities and Society*, 72:102994, 2021.

[41] Tim M Booij, Irina Chiscop, Erik Meeuwissen, Nour Moustafa, and Frank TH Den Hartog. Ton_iot: The role of heterogeneity and the need for standardization of features and attack types in iot network intrusion data sets. *IEEE Internet of Things Journal*, 9(1):485–496, 2021.

[42] Euclides Carlos Pinto Neto, Sajjad Dadkhah, Raphael Ferreira, Alireza Zohourian, Rongxing Lu, and Ali A Ghorbani. Ciciot2023: A real-time dataset and benchmark for large-scale attacks in iot environment. *Sensors*, 23(13):5941, 2023.

[43] M Zolanvari, M. A Teixeira, L Gupta, K. M. Khan, and R Jain. Wustl-iiot-2021 dataset for iiot cybersecurity research, Washington University in St Louis USA, October 2021. URL http://www.cse.wustl.edu/~jain/iiot2/index.html.

Abdullah Waqas received his Ph.D degree from Quaid-I-Azam University, Islamabad, Pakistan in 2018. Currently, he is working as Assistant Professor in the Department of Electrical Engineering, National University of Technology, Islamabad, Pakistan. His research interests include Machine Learning, Deep Learning, IoT, Wireless Communication, Network Protocols, Game Theory, and Ad hoc and Sensor Networks.