RESEARCH PAPER
Future Research in AI & IoT, 2025, 1(2)
Homepage: https://frai.journals.umz.ac.ir/
DOI:

A Network Intrusion Detection System Based on Deep Learning Models in IoT Systems

Mahdi Mousavand[1], Payam Mahmoudi-Nasr[*1]
[1]*Department of Computer Engineering, University of Mazandaran, Mazandaran, Iran.*

Article Info
Received -----
Accepted -----
Available online -----

Keywords:
Internet of Things;
Network, Security;
Intrusion Detection System;
Botnet Detection;
Deep Learning;
Ensemble Learning.

Abstract:
The Internet of Things (IoT) has a vital role in the lives of people today. However, as the use of IoT devices becomes more widespread, there is a growing concern about security threats, like botnet attacks. Therefore, the use of inclusive solutions is required. Intrusion detection systems (IDS) can detect and mitigate attacks on IoT devices by analyzing network traffic and device behavior. This paper proposes an IDS that uses Deep Learning (DL) techniques. It is based on an ensemble learning model that employs diversity and F1-score as a performance metric to select the best transfer learning models. It also proposes 20 individual and hybrid DL models, including Convolution Neural Networks (CNN), Recurrent Neural Networks (RNNs), and Deep Neural Networks (DNN), to detect and classify regular and botnet attack classes. The proposed IDS engages a feature engineering method to reduce unnecessary computation. The Bot-IoT dataset used in this paper contained DDoS, DoS, Reconnaissance, and theft attack labels. The proposed IDS was compared with existing methods using the Bot-IoT dataset. Experimental results disclose a high performance of the proposed model for detecting and classifying various attack and regular labels.

***Corresponding Author:** p.mahmoudi@umz.ac.ir
Supplementary information: Supplementary information for this article is available at https://frai.journals.umz.ac.ir/
Please cite this paper as:

1. Introduction

The Internet of Things (IoT) is a complex and intelligent system for connecting and communicating data from numerous devices, without human involvement, to the Internet. This technology, as the fastest growing field, is expected to increase from 16 billion connected devices in 2015 to 30 billion in 2024 [1]. These devices can be used in medical and healthcare services, smart cities, smart homes, smart agriculture, and industrial applications [2]. IoT-based products and services are deployed and implemented using various technologies, such as IoT application software and cloud computing [3]. The benefits of IoT, which primarily encompass heightened comfort and convenience, are frequently offset by security risks. This inherent tradeoff poses a significant challenge in the development and deployment of IoT devices [4]. Due to the large-scale nature, component deployment abundance, limited power, and computing resources, most IoT devices are vulnerable to security and privacy threats. These vulnerabilities significantly increase with complex cyberattacks, such as Botnets. Attackers could use compromised IoT devices along with an IoT botnet, which induces more complex and large-scale attacks. The COVID-19 pandemic has accelerated the adoption of IoT devices. As people spend more time at home, the number of connected smart devices is increasing. Therefore, the number of malware attacks targeting devices increased from 32.7 million in 2018 to 2.9 billion in 2019. Likewise, the number of infected IoT devices increased from 16.17% in 2019 to 32.72% in 2020, as shown in Table 1 [5, 6]. According to Statista's report on "IoT attacks as a share of total worldwide malware activity" published in February 2022, the share of total worldwide malware activity attributed to IoT devices increased from 1% in 2018 to 19% in Q4 2020 [7]. Since the COVID-19 pandemic accelerated the growth of IoT devices, the importance of improved security measures has been highlighted to protect them from malware attacks. This research proposes a multiclass Network-based Intrusion Detection System (NIDS) to detect Botnet attacks in a realistic IoT network. A network traffic intrusion is defined as various types of attacks, such as theft, Reconnaissance, DDoS, and DoS in IoT. The proposed framework relies on

ISSN

the transfer and ensemble learning methods. The transfer learning models include 20 individual and hybrid DL models and investigate attack behavior patterns using temporal, spatial, and high-level learning features. Transfer learning models utilize feature engineering to decrease training time and memory cost by clearing (i) redundant and (ii) device-based features [8]. The ensemble learning model uses multiple transfer learning models to obtain the best prediction and compensate for the error rate. The ensemble learning model utilizes (i) the diversity of models to discover different attack patterns in network traffic [9, 10] (ii) the top-best transfer learning models according to their predictive performance [9, 10], and (iii) a grid search algorithm to compute the best weights based on the accuracy of the selected models. The proposed model was trained and tested on the Bot-IoT dataset [11], which has been employed in many studies [12-16].

Table 1. IoT share of Malware attacks and infected devices.

Year	Number of Malware attacks [5]	Growth of infected IoT devices [6]
2017	10.3 million	-
2018	32.7 million	-
2019	2.9 billion	16.17%
2020	-	32.72%

Bot-IoT is a reliable, up-to-date, and public botnet dataset containing realistic network traffic from five commercial IoT devices: motion-activated lights, a weather station, a smart thermostat, a smart fridge, and a remotely activated garage door in a smart home running four different attack scenarios: DDoS, DoS, Reconnaissance, and Theft. This paper is organized as follows. In Section 2, the state-of-the-art DL models proposed for IDS in an IoT network environment were reviewed. Section 3 provides an overview of the Deep Learning process. The Proposed Framework is presented in Section 4. In Section 5, the experimental setup and comparisons are presented. Lastly, Section 6 concludes the study and offers ideas for future work.

2 Related Works and Contributions

Most datasets illustrate that the current network traffic attacks are private, and few reliable, real-world IoT network traffic datasets are available [17]. The Bot-IoT dataset is an excellent test bed for DL-based IDS models because it provides a reliable, up-to-date, and realistic IoT network dataset [18]. Table 2 compares various IDS-based studies conducted on the Bot-IoT dataset in terms of the methods, challenges, and advantages. Ferrag et al.[19] investigated a review of DL-based approaches for IDS, datasets, and a comparative study. They described 35 well-known cyber datasets, including IoT traffic-based datasets, and assessed seven DL models, including DNN, RNN, CNN, Deep Belief Networks (DBN), and other DL models on two realistic datasets, including the Bot-IoT dataset. Ge et al. [20] developed a feed-forward neural network (FNN) for multiclass and binary classification on the Bot-IoT dataset and compared accuracy and runtime with an SVM model. They applied the Adam optimizer and sparse categorical cross-entropy as a loss function, was used. L1, L2, and dropout were used as regulators to avoid overfitting. In another study, Ge et al. [21] improved their FNN model by tuning its hyperparameters. Ferrag and Maglaras [22] Proposed an energy framework based on DL and Blockchain-based for smart grids, called DeepCoin. The DL-based scheme contains an Anomaly Detection System (ADS) that employs an RNN with Backpropagation Through Time (BPTT). It was evaluated on different datasets, including the Bot-IoT dataset, and the results were compared with those of SVM, Random Forest (RF), and Naive Bayes (NB) algorithms. To classify intrusions in the context of an IDS on the Bot-IoT dataset, Aldhaheri et al. [23] developed a hybrid model integrating Self-normalizing Neural Network (SNN) and the Dendritic Cell Algorithm (DCA) in order to select the features, where the information gain (IG) was used. Both the full features and the 10 best features subsets were used to train the model. They then contrasted the outcomes using Machine Learning (ML) algorithms. Alkadi et al. [24] mployed an IDS using a Bidirectional Long Short-Term Memory (BiLSTM) deep learning algorithm and evaluated it on two datasets, so that one of them includes the Bot-IoT dataset. NG BA et al. [25] Presented a Vector Convolutional Deep Learning (VCDL) approach, and Popoola et al. [26] developed a model that integrates a Deep Recurrent Neural Network (DRNN) and an LSTM Autoencoder (LAE) using the Synthetic Minority Oversampling Technique (SMOTE) on the Bot-IoT database. Ullah et al.[27] chose the CNN model based on its computational efficiency and due to its automatic key features classification capacity in the input data. They evaluated multiclass intrusions on four datasets, including the Bot-IoT dataset, and classified them using 1D, 2D, and 3D CNN models. Research study of Khraisat et al. [28] considered a hybrid method using a C5 decision tree classifier and a one-class SVM and employed an ensemble technique to improve prediction accuracy. They applied the IG method for feature selection to the Bot-IoT dataset. The results suggest that the detection accuracy can be improved by using the boosting approach for ensemble learning. Derhab et al.[29] developed an IDS for IoT networks that implements Temporal CNN (TCNN), which combines CNN with causal convolution and assesses it on the Bot-IoT dataset. They argued that the learning model performance will fail to spot minority classes on the Bot-IoT dataset as an unbalanced dataset. To solve this issue, they used the Synthetic Minority Oversampling Technique-Nominal Continuous (SMOTE-NC) to handle the unbalanced dataset and increase the minority classes to 100,000 samples each in the training subset. They also used data type conversion, label encoding, and unnecessary feature removal to reduce the feature space. They then evaluated the performance of

Table 2. A comparison of intrusion detection systems based studies on the Bot-IoT dataset.

Authors & Year	Methods	Challenges	Advantages	Dataset
Ferrag et al. [19], 2020	Review of DL approaches for IDS, the datasets, and a comparative study with seven DL models, including DNN, CNN, and RNN for binary and multiclass classification on two real traffic datasets.	Precision, F1-score, and recall were not used as performance metrics.	Deep discriminative models were more efficient than generative/unsupervised methods.	CSECIC-DS2018 and Bot-IoT
Ge et al. [20], 2020	FNN model for binary and multiclass classification was developed on 2% of the data extracted by the authors, and the results were compared with an SVM model.	Binary and multiclass classifiers have been confused with DoS over UDP and subcategories of attacks under the same category, respectively.	The DL model was more time-efficient than the ML model.	Bot-IoT
Ge et al. [21], 2021	Their earlier study [20] was improved to produce a more effective model.	RNN and its improved variants are preferable for processing time series data.	Enhanced model tweaking and the addition of a port-embedded layer.	Bot-IoT
Ferrag and Maglaras [22], 2019	It utilizes a Blockchain-based DL energy framework in Smart Grids. It uses an RNN with BPTT as an IDS. Results were contrasted with NB, RF, and SVM models.	They did not mention how they avoided overfitting.	Smart use of proposed NIDS for network attacks and fraudulent energy transactions. The DL model outperformed ML algorithms in terms of detection rate.	CICIDS2017, Power System, and Bot-IoT
Aldhaheri et al. [23], 2020	A hybrid model integrating DCA and SNN was proposed, evaluated, and compared with ML algorithms such as MLP, SVM, and NB using both the 10 Best and All features. Information Gain (IG) was used for feature selection.	Compared to MLP, the proposed method slightly improved time and complexity.	The proposed IG-selected features performed well with imbalanced classes.	Bot-IoT
Alkadi et al. [24], 2020	The authors proposed a deep Blockchain framework that provides secure and distributed intrusion detection using privacy-based Blockchain using smart contracts in IoT networks. BiLSTM was used as IDS, and the results were compared with ML classifiers such as SVM, RF, and NB.	Precision, F1-score and recall were not used as performance metrics.	Using RNN improved models to process sequenced lists of crypto-records as a series of time steps. The proposed BiLSTM model achieved a better false alarm rate compared to ML classifiers.	Bot-IoT and UNSW-NB15
NG BA et al. [25], 2020	The authors proposed a VCDL model consisting of two CNN layers. Training of IoT traffic was distributed, and computations were performed in Fog Nodes. It was performed for both binary and multiclass classification on all features and the 10 best feature subsets. Then they compared the results with MLP, RNN, and LSTM models.	The compared models needed more optimization.	Less time to detect anomalies in a distributed fog environment than in a centralized architecture.	Bot-IoT
Popoola et al. [26], 2021	The authors proposed the SMOTE-DRNN model, which uses a deep RNN and LAE model with SMOTE as an oversampling method, and compared it with other ML and DL models.	The main data set was manipulated using the oversampling method.	Using the LAE method for feature reduction resulted in less memory for data storage.	Bot-IoT
Ullah et al. [27], 2021	The authors proposed 1D, 2D, and 3D CNN models for binary and multiclass classification on four IoT network traffic datasets. They then compared the results with existing implementations of the DL models.	The EarlyStopping module could be used to avoid overtraining. As the complexity of the CNN models increases, the efficiency also an decreases.	Combine four datasets and create two new datasets with increased number of attacks. The proposed model performed well on these new datasets.	MQTT-IoT-IDS2020, IoT-23, BoT-IoT, IoT Network Intrusion
Khraisat et al. [28], 2019	The authors proposed an ensemble of one-class SVM and C5 classifiers with an IG-based feature selection method and compared the results with other ML and DL models.	Precision, F1-score, and recall were not used as performance metrics.	The ensemble learning model successfully improved intrusion detection and showed better accuracy compared to the other models.	Bot-IoT
Derhab et al. [29], 2020	The authors proposed a TCNN model using SMOTE-NC as an oversampling method and compared the result with the ML and DL models.	The main data set was manipulated using the oversampling method.	The proposed model offered a good compromise between effectiveness and efficiency.	Bot-IoT

their TCNN model and compared it with two legacy ML algorithms and two DL models, namely LSTM and CNN. In our study, we found that some researchers used oversampling methods instead of the original data, whereas others utilized accuracy as the only performance metric in the imbalanced dataset. Considering this context, the main contributions of this study are as follows:

- An intrusion detection system is proposed in an IoT network environment based on an ensemble learning model that employs *(i)* diversity and *(ii)* F1-Score as the best performance metric in the imbalanced dataset to select the best transfer learning models.
- Twenty individual and hybrid DL models are proposed as transfer learning based on different *(i)* types, *(ii)* layers, and *(iii)* neurons to detect and classify normal and four botnet attack classes.
- The proposed method achieves better performance than other methods by proposing a feature engineering procedure that removes redundant features to reduce unnecessary complex computations.
- A detailed comparison of the proposed IDS with the best results in the literature using the Bot-IoT dataset.

3 METHODS & MATERIALS

DL is a sophisticated form of ML based on artificial neural networks (ANN), which resemble biological brain cells. It is a feed-forward neural network in which all layers are connected, and each neuron is connected to another layer, with no connection within a layer. DL algorithms are used in a variety of research areas and are renowned for their ability to identify valuable features in unprocessed data through subsequent nonlinear transformations [30]. It is possible to improve the accuracy of classification and prediction tasks using DL algorithms. IoT devices produce substantial amounts of data, making DL techniques a viable IDS solution. In this section, the DL techniques used in this study are discussed:

3.1 DNN

Deep neural networks (DNNs) consist of neurons ordered in a series of layers, where neurons use the neuron activations of the previous layer as input and utilize them to conduct a simple computation. Certain values are assigned to the neurons in the input layer and propagated to the neurons in the middle layer of the network. The weighted sums from one or more hidden layers are ultimately propagated to the output layer to produce a prediction or classification [31]. An example of a DNN architecture is shown in Figure 1. Three layers exist: input layer, hidden layer, and output layer. Each layer can contain one or more neurons. As the neurons in each layer are fully connected, data are transferred forward from one layer to the next.

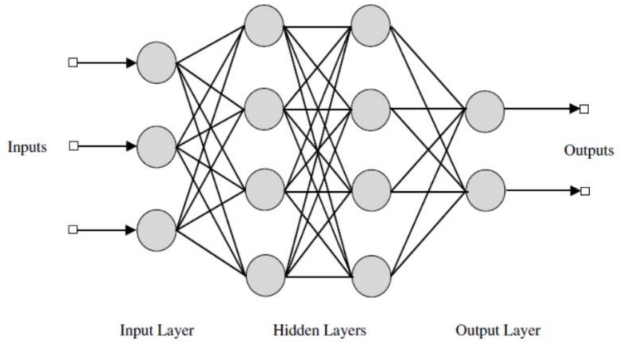

Figure 1. A Feedforward DNN Architecture.

Let $X = \{x_1, x_1, \ldots, x_m\}$ be the input vector, n be the size of the output vector $O(x)$, and $O: \mathbb{R}^m \times \mathbb{R}^n$. Each hidden layer H_i is mathematically calculated as follows:

$$H_i(x) = A(w_i^T x + b_i) \quad (1)$$

Where, b_i and w_i represent the bias and weight of the hidden layer i, and A is the nonlinear activation function. Softmax is a nonlinear activation function for issues involving multiclass classification [32, 33]:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad (2)$$

Where, x is the input.

3.1 CNN

Convolutional Neural Networks (CNNs) are a type of DNN that can be trained to recognize patterns and anomalies in network traffic because of their ability to extract local features and spatial information using multiple convolutional kernels. This enables the CNN to learn to detect suspicious behaviors, such as malware or other types of attacks. Researchers have applied CNN to time series because of their excellent spatial feature extraction [34]. Because the convolutional layer of the CNN convolves the data parameters with a series of equally sized filters, and its pooling layer uses maximum or average pooling to minimize the size of subsequent layers, which is similar to downsampling [35].

3.2 RNNs

Recurrent Neural Networks (RNNs) are a type of DNN that is suitable for processing sequential data, such as analyzing network traffic, which often occurs in a time-series fashion. The RNN architecture resembles a feedforward network (FFN) with an additional internal feedback loop that generates a cyclic graph. These loops serve as short-term memories for storing and retrieving historical data over time and performing temporal tasks [36]. LSTM and GRU are

both improved types of RNNs that attempt to remove the limitations of traditional RNNs, which have difficulty learning long-term dependencies in sequential data [37]. LSTM architectures allow for the storage of long-term contextual information and prevent the vanishing gradient difficulty. GRU and LSTM architectures are both similar, but the GRU is easier to compute and implement [38-40]. Since IDSs are based on detecting unusual patterns and trends in network traffic, DNN derivatives have been used for anomaly detection. Given that RNNs are appropriate for time-series data and that CNNs excel at abstract feature extraction and identification, hybrid models of CNNs and enhanced RNNs were employed.

3.3 Hybrid Models

This technique takes advantage of the CNN and RNN methods. A CNN can capture patterns in network traffic data as time series data through convolutional layers and construct a feature map. To capture long-term dependencies, the feature map is passed to improved RNN models such as LSTM or GRU [41].

3.4 Ensemble Learning

Another machine learning technique, named Ensemble Learning (EL), consists of multiple classifier models and uses them simultaneously to make decisions to increase the performance in estimating data output and achieve better results. The three major ensemble algorithms include bagging, boosting, and stacking.

4 Proposed Framework

In this section, we present several algorithms and explain the DL model development process, as illustrated in Figure 2. The goal is to find an optimal method for detecting intrusions in IoT networks, which involves data preprocessing, feature engineering, and model design. To achieve this, we chose the Bot-IoT dataset as a standard testbed to analyze the performance of our approach.

4.1 Bot-IoT Dataset

Koroniotis et.al [11] designed and simulated a realistic IoT testbed environment in the Research Cyber Range Lab at UNSW Canberra and published it as Bot-IoT dataset in 2019. After simulating IoT devices on multiple network platforms, data features were extracted through forensic analysis and statistical and ML models to determine abnormal and normal instances. They then extracted a subset containing 5% of the original dataset with approximately 3,700,000 records, with All and 10 Best features using the joint entropy and correlation coefficient to ease the handling of the dataset. All features and 10 Best features are listed in Table 3. The original Bot-IoT dataset includes approximately 72,000,000 records represented by 43 features, with each record labeled as attack or normal. These four types of attacks are Information Theft, Denial of Service (DoS), Distributed Denial of Service (DDoS), and Reconnaissance. The taxonomy and numbers of attack and benign samples used in this study are summarized in Table 4. In this study, the introduced extracted subset with All and 10 Best features are employed for training, validating, and testing models.

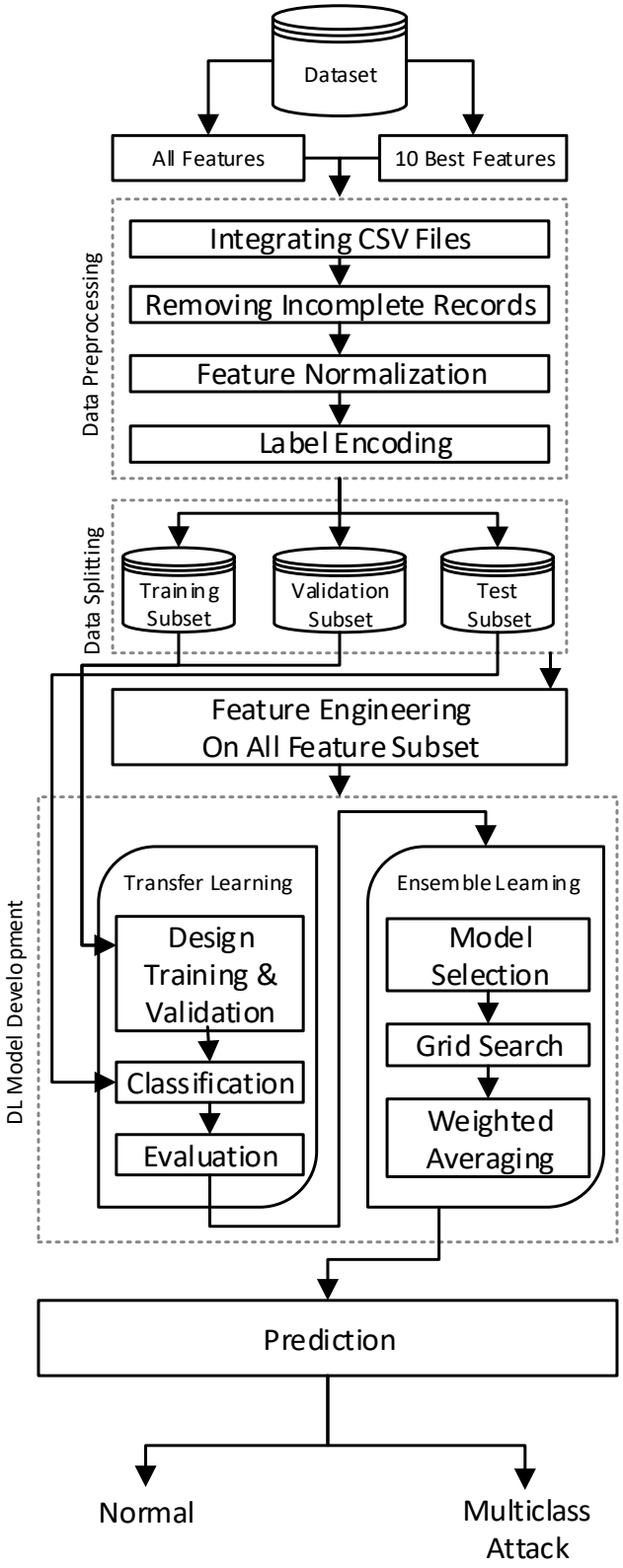

Figure 2. The Proposed Framework Architecture.

Table 3. The Features of Bot-IoT Dataset

Features	Feature Names
Best 10	state_number, seq, drate, srate, N_IN_Conn_P_SrcIP, min, mean, N_IN_Conn_P_DstIP, , max, stddev
All	TnBpDstIP, max, Pkts_P_State_P_Protocol_P_SrcIP, saddr, rate, proto_number, N_IN_Conn_P_SrcIP, dbytes, dur, stddev, pktSeqID, bytes, stddev, spkts, AR_P_Proto_P_Sport, seq, srate, min, Pkts_P_State_P_Protocol_P_DestIP, max AR_P_Proto_P_SrcIp, TnBPSrcIp, dport, psize_mean, dpkts AR_P_Proto_P_DstIP, state_number, sum, flgs, TnP_PDstIP, Pkts_P_State_P_Protocol_P_DstIP, mmax, proto_number N_IN_Conn_P_DstIP, flgs_number, stddev, mean, psize, TnP_perProto, Pkts_P_State_P_Protocol_P_SrcIP, pcount, spkts, pvelocity, stddev, daddr, sbytes, rate, max, ptime, TnP_PSrcIP Pkts_P_State_P_Protocol_P_DestIP.

Table 4. Bot-IoT Dataset Attack Taxonomy and Data Distribution.

IoT Traffic		Samples	
Class	Subclass	Training	Testing
DDoS	TCP HTTP UDP	1540576	386048
DoS	TCP HTTP UDP	1320794	329466
Reconnaissance	OS Fingerprint Service Scan	73016	18066
Normal	Normal	367	110
Theft	Data Exfiltration Keylogging	64	15

4.2 Data Preprocessing

It is necessary to transform the raw and voluminous network traffic data in the Bot-IoT dataset into a format that can be processed easily and effectively. Data preprocessing includes the following steps:

(a) *Integrating CSV Files:* The dataset is present in several Comma-Separated Values (CSVs) and is obtained from Pcap files, which belong to the virtualized setup, containing both normal and attack traffic. Therefore, it is necessary to integrate the separated values *.csv* data files into a single data-frame file.

(b) *Removing Incomplete Records*: Incomplete data leads to inconsistencies in the data and incorrect calculations.

(c) *Feature Normalization*: To enhance the distribution of the model and to preserve the original distribution of network traffic features, the feature elements were scaled to a range of [0,1] using the min-max transformation method:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (3)$$

Where x, x_{min}, and x_{max} are a network traffic feature vector and its minimum and maximum values, respectively.

(d) *Label Encoding*: The labels in the dataset, containing *Reconnaissance, Dos, DDoS, Theft,* and *Normal*, should be converted to a numeric form so that DL algorithms perform better in performance metrics.

(e) *Data Splitting:* In the final step, all network traffic data should be divided indiscriminately into training, testing, and validation subsets. The validation subset was used for cross-validation and to avoid overfitting and underfitting during the training process.

4.3 DL Prediction Models

4.3.1 *Transfer Learning*

This is used as a starting point for ensemble learning in some primary models. It accelerates training and improves learning performance. *Transfer learning* aims to develop models that perform the best classification of the test data. For this purpose, training data were used to train the models, and using validation data resulted in their performance improvement.

Transfer learning models contain 14 *pre-trained individual* and 6 *hybrid* models. These models consist of different layers, neurons in each layer, and hyperparameters. Table 5 presents the specifications of each model.

(a) *Individual models:* These models were selected so that they contained the top basic deep learning models in different contextual patterns. They include DNNs as basic models, CNNs as spatial feature extraction models, RNNs, and GRUs for time-series data.

(b) *Hybrid models*: These models are comprised of CNN and RNN derivations. Three CNN+LSTM and three CNN+GRU models were included. The CNN model excels in learning features from IoT raw data, and the RNN model handles sequential IoT data efficiently [42, 43].

4.3.2 *Ensemble Learning*

In the first step, the best transfer learning models based on the best predictive performance and diversity [9] were chosen. Next, the Weighted Averaging technique was employed as a combination method in ensemble learning. The grid Search optimization algorithm was used for finding the weights $w_i \in [0, 0.5]$ in steps of 0.1, considering the best accuracy value. In the last step, the weighted values are combined as follows:

$$H(x) = \sum_{i=1}^{T} w_i h_i(x) \quad (4)$$

Where $h_i(x) \in$ □ is the output value of the transfer learning model; (h_i) for instance x, T is the number of selected transfer learning models $\{h_1, \ldots, h_T\}$, and the maximum value output $(H(x))$ is selected for the final prediction (P).

$$P = MAX_{i=1}^{T}\left(H_i(x)\right) \quad (5)$$

Ensemble learning prediction is described in Algorithm 1.

Table 5. The proposed individual and hybrid models

	Model Name	Hidden Layers	Neurons in Each Layer
1	DNN(1)	1 layer	1024
2	DNN(2)	2 layers	1024-768
3	DNN(3)	3 layers	1024-768-512
4	DNN(4)	4 layers	1024-768-512-256
5	DNN(5)	5 layers	1024-768-512-256-128
6	CNN(1)	1 layer	64
7	CNN(2)	2 layers	64-64
8	CNN(3)	4 layers	64-64-128-128
9	RNN(1)	4 layers	(30)-60-80-90
10	RNN(2)	4 layers	(90)-60-80-90
11	RNN(3)	4 layers	(90)-512-256-128
12	GRU(1)	4 layers	(30)-60-80-90
13	GRU(2)	4 layers	(90)-60-80-90
14	GRU(3)	4 layers	(90)-512-256-128
15	CNN+LSTM(1)	1 layer	64-(70)-128
16	CNN+LSTM(2)	2 layers	64-64-(70)-128
17	CNN+LSTM(3)	3 layers	64-64-128-128-(70)
18	CNN+GRU(1)	1 layer	64-(70)-128
19	CNN+GRU(2)	2 layers	64-64-(70)-128
20	CNN+GRU(3)	4 layers	64-64-128-128-(70)-128

Algorithm 1 Ensemble Learning with Weighted Averaging

Require: Training dataset X with n instances and d features, labels y for training dataset, a list of $transfermodels$ pre-trained transfer learning models, and $gridsearchsteps$ steps for grid search optimization algorithm

Ensure: A weighted ensemble of selected transfer learning models $ensemblemodel$

Sort $transfermodels$ by descending F1-Score
Sort $transfermodels$ by descending diversity
Select the top 3 models based on F1-Score and diversity, add them to $ensemblemodels$
Initialize $maxaccuracy$ to 0 and $bestweights$ to empty list
Perform a grid search for weights in ensemble model
Iterate over all combinations of weights $w_1, w_2, ..., w_n$ such that each $transfermodels$ can only have to participate less than or equal to 0.5
For each combination of weights, calculate the ensemble prediction using weighted averaging
Evaluate the accuracy of the ensemble prediction
If the accuracy is greater than $maxaccuracy$, update $maxaccuracy$ and $bestweights$
Create an ensemble model with the selected transfer learning models and weights
Assign $ensembleweights$ to be the $bestweights$ from step 5
Assign $ensemblemodels$ to be the top 3 models from step 3
Train the ensemble model using the weighted averaging technique
Evaluate the ensemble model on Bot-IoT dataset

5 RESULTS

The details of the evaluation metrics, experimental setup, and evaluation parameters are provided in this section.

5.1 Performance Metrics

The confusion matrix, which displays the convergence between the detected and real values, is a crucial tool for assessing the effectiveness and performance of classification tasks. Each row shows the actual samples and each column shows the predicted samples, of a class, which is presented in Table 6. The confusion matrix and the following metrics were used to evaluate and compare the performance of the proposed transfer models:

(a) Accuracy= (TP+TN)/(TP+TN+FP+FN)
(b) Precision= TP/(TP+FP)
(c) Recall= TP/(TP+FN)
(d) F1-Score=2× (Precision×Recall)/(Precision +Recall)

Where TP, TN, FP, and FN denote the true positives, true negatives, false positives, and false negatives, respectively.

In the first step, to identify the best models, the accuracy metric was applied. Thereafter, the precision, recall, and F1-score were used as effective metrics to select the optimal model.

Table 6. Confusion Matrix.

	Predicted Attack	Predicted Normal
Actual Attack	True Positive (TP)	False Negative
Actual Normal	False Positive	True Negative

5.2 Experimental Setup

To evaluate and implement the proposed models on the Bot-IoT dataset, experiments were conducted on an Asus laptop running Windows 10 with 16 GB of RAM, an Intel Core I7-10510U processor, and an NVIDIA GeForce GTX 1650 Max-Q as an integrated GPU. Model training, validation, and testing were implemented using MiniConda, Keras API, and TensorFlow frameworks developed in the Python programming language (version 3.9).

5.3 Experimental Parameters

Figure 3 depicts the process of training, validating, and optimizing the transfer models:

(i) During the training process, the decrease in the loss function on the validation set was tracked using the early stopping feature of the Keras API. If there was no improvement after several epochs, the training was halted. This approach helps in avoiding overfitting and prevents unnecessary consumption of computational resources. It is important to note that the overall training duration, the number of iterations, and the model's accuracy can vary due to the inherent randomness of the training process.
(ii) The batch size was set to 128 for the DNN, CNN, and hybrid models; 256 for the GRU models; and 512 for the RNN models. By reducing the batch size, the model became more accurate as the training time increased.
(iii) The dropout is set to 0.10 for DNN, GRU, and hybrid models, 0.20 for RNN models, and 0.50 for CNN models.
(iv) The ReLU activator function was used in the Dense, Conv1D, and MaxPool1D layers; the Tanh activator function is used in the GRU, SimpleRNN, and LSTM layers; and the Softmax activator function is used in the final layer for multi-class classification.
(v) Weights and biases were randomly selected.

(vi) A Sparse Categorical Cross-entropy was used as the loss function.

(vii) An Adam optimizer, with a learning rate of 0.001.

Table 7. Results of 20 proposed models trained and tested on the Bot-IoT dataset.

Model Name	Best Epoch		Time (sec)		Accuracy (%)		Precision (%)		Recall (%)		F1-Score (%)	
	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.
DNN(1)	30	13	3095	2115	99.95	97.363	95.3	90	98.4	96.9	96.7	92.3
DNN(2)	14	23	3332	3686	99.96	97.971	99.8	95	97.6	95.4	98.6	94.9
DNN(3)	21	16	5570	2973	99.96	97.865	96.1	95.7	98.9	91.5	97.3	93.3
DNN(4)	17	22	5272	4202	99.9	97.879	99.9	93.5	87.1	83.7	91.5	84.3
DNN(5)	10	25	3812	5057	99.93	97.821	86	78.8	89.8	62.4	86.8	64.6
CNN(1)	23	13	4480	2302	99.86	97.11	99.9	75.9	83.3	66	88.1	68.7
CNN(2)	10	16	2957	2956	99.98	98.06	99.3	96.5	98.3	81.9	98.8	87.2
CNN(3)	10	18	3962	4197	99.98	98.18	99.6	94.1	93.3	94.9	96.1	94.5
RNN(1)	12	15	2377	2738	97.69	96.79	76.6	74.7	71.8	59.5	73.6	60.2
RNN(2)	11	23	2299	2247	97.93	97.08	78.8	73.1	76	64.7	77.2	66.8
RNN(3)	7	12	1940	2118	97.8	97.01	90.1	74.1	82.4	67.1	84.5	69.4
GRU(1)	29	35	4096	3386	99.83	98.3	95.1	92.6	91	94.8	92.9	93.6
GRU(2)	16	35	2534	3834	99.89	98.07	97.5	91.4	88.5	88.1	92.3	89.3
GRU(3)	24	16	3711	1801	99.97	98.04	94.6	95	97.4	91.3	95.8	93
CNN+LSTM(1)	19	21	7105	5407	99.98	98.21	99.6	98.6	86.1	96.2	89.5	97.4
CNN+LSTM(2)	10	18	5805	5272	99.98	98.26	98.2	96.8	99.3	94.8	98.7	95.7
CNN+LSTM(3)	11	9	5817	3670	99.97	98	98.2	97.3	99.1	81.2	98.6	85.3
CNN+GRU(1)	24	20	6277	4589	99.74	98.14	99.9	97	98.6	93.7	99.2	95.2
CNN+GRU(2)	8	8	2213	1889	99.6	97.92	99.3	89.8	99.1	95.1	99.2	91.3
CNN+GRU(3)	8	14	2687	3711	99.97	98.12	99.8	95.5	90.3	89.1	94	92

Table 8. Weights and Performance Metrics of Transfer and Ensemble Learning Models.

Model	Weights		Accuracy		Precision		Recall		F1-Score	
	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.	All Feat.	10 Best Feat.
DNN	0.20	0.10	99.950	97.925	97.92	95.19	98.09	97.35	98.00	96.05
CNN+GRU	0.30	0.40	99.955	98.187	98.28	95.80	98.10	95.10	98.19	95.28
CNN+LSTM	0.20	0.50	99.973	98.330	96.09	97.48	98.71	95.53	97.24	96.47
Proposed	-	-	99.985	99.331	99.28	96.80	99.80	96.83	99.54	96.76

Table 9. Comparative Analysis Between Related Works for Bot-IoT Dataset.

Reference	Best Model	Task	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Koroniotis *et al.* [11]	LSTM	Multiclass	98.057	99.99	98.05	-
Ge *et al.* [20]	FNN	Multiclass	98.09	-	-	-
Ferrag and Maglaras [22]	RNN with BPTT	Multiclass	98.20	-	-	-
Ferrag *et al.* [19]	CNN	Multiclass	98.37	-	-	-
Aldhaheri *et al.* [23]	SNN	Multiclass	98.73	99.17	98.36	98.77
Alkadi et al. [24]	BLSTM	Multiclass	98.91	-	-	-
NG BA et al. [25]	VCDL	Binary	99.74	99.99	99.75	-
Ge *et al.* [21]	FNN	Binary	99.79	-	-	-
Popoola et al. [26]	LS-DRNN	Multiclass	99.93	96.87	99.75	98.22
Ullah et al. [27]	CNN1D	Multiclass	99.97	-	-	-
Khraisat et al. [28]	Ensemble Learning	Multiclass	99.97	-	-	-
Derhab et al. [29]	TCNN	Multiclass	99.998	99.997	97.49	98.66
Proposed Model	**Ensemble Learning**	**Multiclass**	**99.985**	**99.28**	**99.80**	**99.54**

Data Splitting

The total network traffic data was divided indiscriminately into training (80%) and test (20%) sets. For cross-validation and to avoid overfitting or underfitting in the training process, 15% of the test set was used as the validation set.

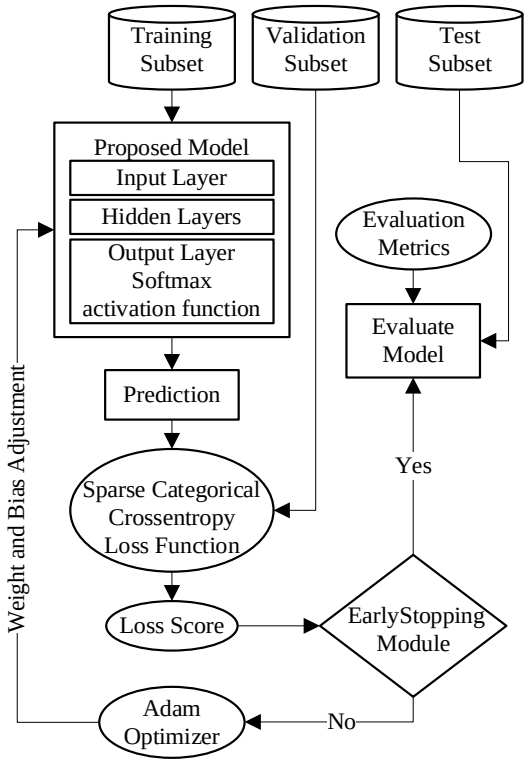

Figure 3. Architecture for Training, Validation, and Optimization of the Proposed Models.

5.4 Ensemble Learning

Table 7 shows the computed performance results of all the proposed transfer models. Conventional deep learning models, including DNNs, CNNs, and RNNs as individual models, and CNN+GRU and CNN+LSTM models were chosen as hybrid models that take advantage of these transfer models. CNN models can capture patterns in network traffic data as time-series data through convolutional layers and construct a feature map. To learn long-term temporal dependencies, they were passed to improved RNN models: LSTM and GRU. The best transfer models are selected according to (i) the best F1-score as the best performance metric for the imbalanced dataset [10, 44] and (ii) having diversity in models to find different patterns in the dataset and improve the prediction results [9]. Table 8 shows the selected transfer models, their computed weights, and the calculated performance metrics.

6 DISCUSSION

According to the results shown in Tables 7 and 8, it can be concluded that the proposed transfer learning models outperformed similar studies in terms of the F1-score, which was the best performance metric for imbalanced datasets. This improvement can be attributed to the integration of individual and hybrid deep learning models, selected based on their ability to capture different patterns in IoT network traffic data. In addition, the ensemble learning approach using the weighted averaging technique provides a further boost in performance by combining the outputs of the best

transfer models with the optimized weights. It should be noted that this study was conducted using the Bot-IoT dataset as a standard testbed. The proposed method involves several important steps, such as data and feature preprocessing and model design, each of which contributes to the overall success of the proposed method. Furthermore, the proposed transfer learning models were carefully selected based on their diversity and ability to capture the different patterns in the dataset. This approach led to the implementation of a more accurate and robust intrusion detection system that can effectively identify various types of attacks, such as Theft, DoS, DDoS, Reconnaissance, and Normal. Compared with other similar studies, as shown in Table 9, the proposed model demonstrated superior performance in accurately detecting intrusions in IoT networks. It has been demonstrated that the integration of individual and hybrid deep learning models, along with ensemble learning techniques, is a viable approach for performance improvement of intrusion detection systems in IoT networks. The results of this research provide valuable insights into the development of more efficient and reliable intrusion detection methods to secure IoT networks against cyber threats. The proposed method demonstrated promising results and can serve as a basis for future research in the field of intrusion detection in IoT networks. By continuing to refine and optimize these techniques, we can improve the security of IoT systems and devices, thereby protecting them from potential cyberattacks.

7 CONCLUSION

The proposed approach demonstrated promising results in detecting intrusions in IoT networks by combining data preprocessing, feature engineering, and model design. The Bot-IoT dataset was utilized as a standard testbed to evaluate the performance of the proposed approach. Data preprocessing was performed to transform the raw and voluminous network traffic data into a format that could be processed easily and effectively. Ensemble learning was employed to combine the best transfer learning models based on their predictive performance and diversity. The transfer learning models used in this study included individual models, such as DNNs, CNNs, and RNNs, and hybrid models, such as CNN+GRU and CNN+LSTM. The ensemble learning technique employed in this study, using the Weighted Averaging method, resulted in improved accuracy and F1-score for the detection of intrusions in IoT networks. Overall, the proposed approach makes a valuable contribution to the field of intrusion detection in IoT networks. The combination of data preprocessing, feature engineering, and model design with transfer learning and ensemble learning techniques holds great potential for improving the accuracy and effectiveness of intrusion detection systems in IoT networks. Further research is required to explore the efficacy of this approach in real-world scenarios and develop more sophisticated models that can adapt to the evolving nature of cybersecurity threats in IoT networks. This research presents a valuable contribution towards improving the accuracy and effectiveness of intrusion detection systems in IoT networks and lays the groundwork for future research in this field.

REFERENCES

[1] S. Smith, "IoT Connections To Reach 83 Billion By 2024, Driven By Maturing Industrial Use Cases," *Accessed: Apr*, vol. 10, 2020. [Online]. Available: https://www.juniperresearch.com/press/iot-connections-to-reach-83-bn-by-2024.

[2] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information systems frontiers*, vol. 17, pp. 243-259, 2015.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645-1660, 2013.

[4] I. H. Sarker, A. I. Khan, Y. B. Abushark, and F. Alsolami, "Internet of things (iot) security intelligence: a comprehensive overview, machine learning solutions and research directions," *Mobile Networks and Applications*, pp. 1-17, 2022.

[5] M. Injadat, A. Moubayed, and A. Shami, "Detecting botnet attacks in IoT environments: An optimized machine learning approach," in *2020 32nd International Conference on Microelectronics (ICM)*, 2020: IEEE, pp. 1-4.

[6] N. Nokia, "Threat intelligence report 2020," *Comput. Fraud Secur*, 2020.

[7] "Global share of IoT attacks 2021." Statista. https://www.statista.com/statistics/1321250/worldwide-internet-of-things-attacks/ (accessed March 6, 2023.

[8] T. Verdonck, B. Baesens, M. Óskarsdóttir, and S. vanden Broucke, "Special issue on feature engineering editorial," *Machine Learning*, pp. 1-12, 2021.

[9] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[10] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International journal of pattern recognition and artificial intelligence*, vol. 23, no. 04, pp. 687-719, 2009.

[11] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779-796, 2019, doi: 10.1016/j.future.2019.05.041.

[12] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1-22, 2019.

[13] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN computer science*, vol. 2, no. 3, p. 160, 2021.

[14] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big data*, vol. 7, pp. 1-29, 2020.

[15] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242-3254, 2020.

[16] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433-442, 2020.

[17] R. Vijayanand, D. Devaraj, and B. Kannapiran, "Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection," *Computers & Security*, vol. 77, pp. 304-314, 2018.

[18] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for IoT big data and streaming analytics: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2923-2960, 2018.

[19] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020, doi: 10.1016/j.jisa.2019.102419.

[20] M. Ge, X. Fu, N. Syed, Z. Baig, G. Teo, and A. Robles-Kelly, "Deep learning-based intrusion detection for IoT networks," in *IEEE 24th Pacific Rim International Symposium on Dependable Computing*

(PRDC): IEEE, 2020, pp. 256-25609, doi: 10.1109/PRDC47002.2019.00056.

[21] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Towards a deep learning-driven intrusion detection approach for Internet of Things," *Computer Networks*, vol. 186, p. 107784, 2021, doi: doi.org/10.1016/j.comnet.2020.107784.

[22] M. A. Ferrag and L. Maglaras, "DeepCoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1285-1297, 2019.

[23] S. Aldhaheri, D. Alghazzawi, L. Cheng, B. Alzahrani, and A. Al-Barakati, "Deepdca: novel network-based detection of iot attacks using artificial immune system," *Applied Sciences*, vol. 10, no. 6, p. 1909, 2020.

[24] O. Alkadi, N. Moustafa, B. Turnbull, and K.-K. R. Choo, "A deep blockchain framework-enabled collaborative intrusion detection for protecting IoT and cloud networks," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9463-9472, 2020.

[25] B. A. NG and S. Selvakumar, "Anomaly detection framework for Internet of things traffic using vector convolutional deep learning approach in fog environment," *Future Generation Computer Systems*, vol. 113, pp. 255-265, 2020.

[26] S. I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, and A. A. Atayero, "Memory-efficient deep learning for botnet attack detection in IoT networks," *Electronics*, vol. 10, no. 9, p. 1104, 2021.

[27] I. Ullah and Q. H. Mahmoud, "Design and development of a deep learning-based model for anomaly detection in IoT networks," *IEEE Access*, vol. 9, pp. 103906-103926, 2021, doi: 10.1109/ACCESS.2021.3094024.

[28] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A novel ensemble of hybrid intrusion detection system for detecting internet of things attacks," *Electronics*, vol. 8, no. 11, p. 1210, 2019.

[29] A. Derhab, A. Aldweesh, A. Z. Emam, and F. A. Khan, "Intrusion detection system for internet of things based on temporal convolution neural network and efficient feature engineering," *Wireless Communications and Mobile Computing*, vol. 2020, 2020.

[30] I. Idrissi, M. Azizi, and O. Moussaoui, "IoT security with Deep Learning-based Intrusion Detection Systems: A systematic literature review," in *2020 Fourth international conference on intelligent computing in data sciences (ICDS)*, 2020: IEEE, pp. 1-10.

[31] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295-2329, 2017.

[32] K. Simran, S. Sriram, R. Vinayakumar, and K. Soman, "Deep learning approach for intelligent named entity recognition of cyber security," in *International Symposium on Signal Processing and Intelligent Recognition Systems*, 2020: Springer, pp. 163-172.

[33] Z. Ahmad *et al.*, "Anomaly detection using deep neural network for IoT architecture," *Applied Sciences*, vol. 11, no. 15, p. 7050, 2021.

[34] C. Yin, S. Zhang, J. Wang, and N. N. Xiong, "Anomaly detection based on convolutional recurrent autoencoder for IoT time series," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 112-122, 2020, doi: 10.1109/TSMC.2020.2968516.

[35] L. Aversano, M. L. Bernardi, M. Cimitile, and R. Pecori, "A systematic review on Deep Learning approaches for IoT security," *Computer Science Review*, vol. 40, p. 100389, 2021.

[36] R. Vinayakumar, K. Soman, and P. Poornachandran, "Evaluation of recurrent neural network and its variants for intrusion detection system (IDS)," *International Journal of Information System Modeling and Design (IJISMD)*, vol. 8, no. 3, pp. 43-63, 2017.

[37] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157-166, 1994.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[39] A. G. Felix, S. Jürgen, and C. Fred, "Learning to forget: Continual prediction with LSTM," *Neural computation*, vol. 12, no. 10, pp. 2451-2471, 2000.

[40] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2016: IEEE, pp. 324-328.

[41] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017: IEEE, pp. 1222-1228.

[42] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for internet of things (IoT) security," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1646-1685, 2020.

[43] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of IDS for IoT: recent updates, security issues, and challenges," *Archives of Computational Methods in Engineering*, vol. 28, pp. 3211-3243, 2021.

[44] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, pp. 1-50, 2016.