A Transfer Learning based Intrusion detection system for Internet of Things

Monika Vishwakarma
Central University of Rajasthan
Nishtha Kesswani (nishtha@curaj.ac.in)
Central University of Rajasthan

Research Article

Keywords: Internet of Things, Intrusion Detection System (IDS), Deep Learning (DL), Convolutional Neural Network (CNN), Transfer Learning.

Posted Date: May 23rd, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2930837/v1

Additional Declarations: No competing interests reported.

A Transfer Learning based Intrusion detection system for Internet of Things

Monika Vishwakarma · Nishtha Kesswani

the date of receipt and acceptance should be inserted later

Abstract There will be billions of gadgets emerging in the future. A few years ago, experts predicted that the Internet of Things (IoT) might soon be renamed the Internet of Everything (IoE) due to the widespread use of computing technologies in modern days. However, what happens if security issues are not addressed in today's IoT devices? Because of cyber security breaches, consumers and manufacturers of connected devices are at risk. Consequently, the number of cyber-attacks has skyrocketed across the networks. Machine learning-based techniques, particularly deep learning, have shown considerable promise in attack detection techniques. This article proposes a 1D Convolution Neural Network (CNN) based model to address anomaly detection in IoT environment. We looked at the capabilities of CNN to identify and categorize abnormalities in IoT networks. The ability of CNN to identify and categorize abnormalities in IoT networks using multiclass and binary classification via transfer learning was also assessed. The performance of the 1D CNN model is assessed using the Netflow-based NF-ToNIoT, NF-BoTIoT, NF-CSE-CICIDS2018, NF-UNSW-NB15, NF-UQ-NIDS, and CIC flowmeter-based IoT DS2, IoT Network Intrusion, MQTT-IoTIDS2020, CIC-ToNIoT datasets. The reason for selecting transfer learning is to reduce classification and run-time complexity. The training and testing times needed for classification are significantly decreased using the transfer learning approach. The proposed model successfully identifies 20 different attacks with an accuracy of 93.75% on the NF-UQ NIDS dataset. Additionally, we have verified our proposed model in real-time on an edge device with limited resources.

Keywords Internet of Things, Intrusion Detection System (IDS), Deep Learning (DL), Convolutional Neural Network (CNN), Transfer Learning.

Monika Vishwakarma
Central University of Rajasthan, Ajmer, India
E-mail: monika.vish98@gmail.com

Nishtha Kesswani
Central University of Rajasthan, Ajmer, India
E-mail: nishtha@curaj.ac.in

1 Introduction

Internet of Things (IoT) is a network of real-world objects, or "things," that are equipped with sensors, software, and other technologies to communicate and share data. Facilities using IoT technology are multiplying quickly. Connecting everyday objects to the Internet via embedded appliances, such as home appliances, thermostats, industries, and healthcare systems, has made continuous communication between people, procedures, and things possible.

According to the morder intelligent report [1], the IoT market is estimated to expand at a 10.53% Compound Annual Growth Rate (CAGR) during the projection period (2022-2027). Similarly, as per the Globe newswire report [3], the worldwide IoT market is calculated to reach USD 1,854.76 billion by 2028, extending at a 25.4% CAGR over the forecast period, and according to report [2], it is expected to rise from 478.36 billion dollars in 2022 to 2,465.26 billion dollars in 2029, a 26.4% CAGR during the forecast period.

Growing IoT technology affects many application areas. According to [20], applications may be categorized based on network availability, coverage, scalability, heterogeneity, repetition, user engagement, and effect. The Smart homes, cities, agriculture, transportation, wearable, and healthcare sectors are gaining popularity through IoT applications. There are multiple

facets that IoT incorporates into the functioning of a city: for instance, traffic control, pollution monitoring, resource management, parking solutions, infrastructure management, and disaster management.

Although smart cities simplify life and along with helping us monitor and regulate many factors, they increase complexity and interdependence. Connectivity increases the vulnerability of smart cities against security and privacy attacks [49]. In addition to the security concerns encountered by the Internet, mobile networks, and base stations, the IoT has unique security issues, such as privacy, authentication, administration, and information storage issues. As a result of these flaws and vulnerabilities, IoT apps provide a breeding ground for several cyber threats. Globally deployed IoT apps have been subject to a number of security and privacy breaches. IoT gadgets, which are low-powered and less secure, present an entry point for attackers to penetrate household and business networks and gain easy access to user data. Such gadgets may be targeted by adversaries to monitor the position of a specific person or fabricate data.

Intrusion detection Systems (IDS) play an important role in the detection of attacks. In order to effectively counter threats like targeted attacks, and data exfiltration, IDSs must evolve from basic correlation to detection, according to Nisioti *et al.* [40]. They uncover existing intrusion detection systems (IDSs) flaws and show how such systems need to improve. In addition to examining the works under discussion, they have contrasted them, pointing out their strengths and weaknesses. According to [22], security requirements and existing protection options for IoT networks have been investigated systematically. They also emphasized the flaws in current security measures. They discussed possible ML and DL-based IoT security research approaches.

Another article [11] explores many intrusion detection explanations for the IoT. Each solution aims to boost detection effectiveness in numerous different ways and decrease resource use by combining different architectures, detection algorithms, and particular threats that have been identified. The sorts of architecture and technology that can be identified are the key areas of attention in this study. A survey [38] that focuses on IoT risks sheds information on current research trends and technology requirements from various angles. The findings of this study offer a strong foundation for further inquiry. More study is needed to understand IoT-specific risks and their destructive effects fully.

1.1 Motivation

Several IDS techniques are available for detecting network anomalies. The role of IDS has evolved as crucial in safeguarding IoT networks by finding abnormalities because malicious behaviors in IoT networks must be instantly recognized and halted. Deep learning is attaining interest across multiple domains to address diverse problems. In this domain, CNN have proved to be excellent in voice recognition and image recognition. VGG16, VGG19, and ResNet50 are gaining popularity as they are pre-trained on an extensive dataset. Moreover, researchers use these models as transfer learning models for image classification and achieve better results.

Existing studies have assessed intrusion detection methods using an outdated KDD intrusion detection dataset. The KDD99 dataset does not consider many recent cyberattacks, and the IoT network was not taken into consideration while developing the KDD99 dataset. Thus, we have presented a CNN-based deep neural network-based IDS that is trained on the combination of the different datasets. The proposed system classifies 21 different kinds of intrusion and efficiently distinguishes them from normal network traffic using a CNN architecture in the multiclass classifier. Furthermore, this trained model is also used for transfer learning for the multiclass and binary class attack classification to improve efficiency. The novelty of this model is that proposed model can detect many malicious cyber attacks in an IoT network, and it can easily be deployed on an edge device due to its lightweight and low complexity.

The following are the key contributions of this article:

- We have proposed a lightweight CNN-based deep neural network that is trained on a combination of different intrusion-type datasets.
- We have also used a trained CNN model for transfer learning to improve the detection rate and reduce the model training complexity.
- We have also verified the proposed model in real time by deploying the model on an edge device.
- The findings of our experiment demonstrate how well our technology was able to classify 20 various IoT networking assaults and the typical patterns of networking data.

The rest of this paper is organized as follows: We have illustrated background details in Section 2. In Section 3, we have explained existing work on IDS. The section 4 elaborates the proposed work. Section 5 describes the experimental setup and findings, as well as the testbed. We have explained in Section 6, how the

suggested IDS tackles the issue of intrusion detection. Further, in the last Section 7, we summarize the article.

2 Background

2.1 Deep Convolution Neural Network (D-CNN)

CNN is an artificial neural network that uses deep feed-forward learning. Image recognition and categorization are two areas where it excels. Because of its enhanced image processing capabilities, this research offers a model based on CNN. In several domains, CNN gives appealing outcomes. Furthermore, convolutional neural network advantages may be completely leveraged by translating intrusion detection difficulties into image recognition challenges. CNN has an input layer, an output layer, and numerous hidden layers. Convolution layers and pooling layers are hidden layers in a CNN. Convolutional layers carry out convolution operations. CNN retrieves valuable data with little processing effort.

The main reason behind using CNN is its good performance and less computational complexity. However, converting the tabular networking data into images is worthless because of real-time deployment. So we have proposed one dimensional CNN model that can be used for transfer learning without reshaping the original data into an image. The process of normalization and polling are discussed below:

2.1.1 Batch Normalization (BN)

Normalizing the inputs to a layer may be accomplished using a generic strategy known as batch normalization [24]. It employs a transformation that keeps the output mean nearly zero while keeping the output standard deviation as close as possible to one. This forwards the data dispersed in standard normal distribution form for the next layer. To compute the mean of the hidden activation in batch input from layer L, we used Equation 1. Where n is the number of neurons in Layer L.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} L_i \tag{1}$$

The next step is to calculate the standard deviation explained in Equation 2.

$$\sigma = \left[\frac{1}{n}\sum(L_i - \mu)\right]^{\frac{1}{2}} \tag{2}$$

By subtracting the mean from each input and dividing the result by the overall standard deviation and the smoothing term (*epsilon*), these values will be used to normalize the hidden activation.

$$L_{i(normalize)} = \frac{(L_i - \mu)}{(\sigma + \epsilon)} \tag{3}$$

The last procedure consists of re-scaling and offsetting the input. γ and β are two BN algorithm components. These parameters are used to re-scale (γ) and shift (β) the vector storing the results of earlier operations.

$$L_i = \gamma L_{i(normalize)} + \beta \tag{4}$$

2.1.2 Average Pooling

The average value across the window given by pool size is used to downsample the input representation. Strides shift the window. When the "valid" padding option is used, the output that is generated has the form of:

$$output\_shape = \frac{(input\_shape - pool\_size + 1)}{strides} \tag{5}$$

In the next section, we discuss the related work.

3 Related Work

3.1 IDS based on Machine Learning Techniques

Mighan *et al.* [35] presented a hybrid stacked autoencoder SVM solution. The experiment utilized the ISCX dataset to test the model's performance utilizing Apache Spark and machine learning approaches. This research [9] proposes a three-layer, inventive, and intelligent IDS architecture. It serves three main purposes: Determine the specific IoT item connected to the network and profile it, identify wireless risks against connected IoT appliances, and classify the attack type that can be used. Similarly, Vishwakarma and Kesswani [53] established the two-stage IDS model. After initially using the naive Bayes classification technique, they employed the k-means method in the second step. Li *et al.* [30] researched a two-stage IDS that is based on software-defined technologies. An IDS was recommended by Murali *et al.* [37] to defend against the Sybil attack. They also described the Artificial Bee Colony (ABC) method on the RPL network, which was bio-inspired. The experiment was carried out using Contiki OS and the Cooja simulator.

The CorrAUC feature selection measure, which employs a wrapper technique to choose correct attributes for the ML algorithm selected using the AUC metric,

was introduced in [47]. Additionally, they integrated TOPSIS and Shannon Entropy with a bijective soft set to assess attributes for malicious traffic labeling in IoT networks. Similarly, this study [36] thoroughly examines a group of features. In order to recognize DDoS assaults, this study [29] presented a distributed intrusion detection platform established on fog computing. In the suggested work, features are chosen to utilize mutual information. This significantly raises overall detection performance. Researchers studied TCP/IP models, particularly the protocols MQTT, DNS, HTTP, and their flow IDs, to build an effective NIDS for spotting assaults that control IoT networks. Using DT, NB, and ANN methodologies, an AdaBoost ensemble approach was developed to improve overall performance. A hybrid classification model based on neural networks and SVM was introduced by Choudhary and Kesswani in their paper [14].

The real-time detection and prevention IDS proposed by Baykara *et al.* [10], uses the honeypots approach. A useful software tool was created for the proposed new system. Using a simulated campus network, they tested the built system in real-time. A framework was presented by Seth *et al.* [46] by merging several machine learning methods. The authors overcame the issue of an unbalanced class by using SMOTE and undersampling techniques. They used the CIC IDS 2018 dataset to conduct their studies. Kesswani and Agrawal [26] offer IDS-based SmartGuard, which has the ability to identify harmful potential in the network. Choudhary *et al.* [13] proposed the correlation and regression-based IDS for smart homes. Louk *et al.* [33] provide a dual ensemble strategy for identifying network anomalies that combine two distinct approaches: bagging and GBDT ensembles. To avoid overfitting, each GBDT ensemble model has undergone hyperparameter adjustment.

3.2 IDS based on Deep Learning Techniques

The research [41] suggests a deep learning-based intrusion detection methodology for enhancing the openness and robustness of IoT networks. They utilize the SHapley Additive exPlanations (SHAP) method to explain determinations caused by deep learning-based. In article [54], The authors initially suggest a drift detection strategy that uses the principal component analysis (PCA) technique to examine changes in the variance of the characteristics across attack detection data streams to discover data and concept drifts. They also discussed the online outlier identification method that finds outliers deviating from historical and recent data sets. They suggest an online deep neural network (DNN) that uses the Hedge weighting method to dynamically modify the sizes of the hidden layers in order to prevent these drifts and allow the model to continuously understand and adjust when new intrusion data arrives.

Shone *et al.* [48] introduces a deep autoencoder. This is a suggestion to abandon the encoder-decoder approach in favor of only employing the encoder phase. This is because applying good knowledge decreases computational and temporal overheads while preserving accuracy and efficiency. A distributed approach for identifying online intrusions from URLs was given by Tian *et al.* [51]. This technique protects several online applications in a distributed Edge of Things domain. Launching and detecting adversarial attacks against NIDS continues to be a significant research problem. In order to analyze the variations in adversarial learning against deep neural networks in NIDS, this study [21] examines the most current research on NIDS, adversarial assaults, and network defenses.

3.3 IDS based on Transfer Learning

Wang *et al.* [55] suggested a methodology to avoid gradient explosion or gradient disappearance based on a deep residual CNN. They used transfer learning to alter the residual CNN structure, which ensures the identification of unidentified assaults. They transformed the 1D ICS flow data into two-dimensional grayscale images. They used KDD Cup 99 dataset to assess the model's performance. ITL-IDS, a framework for IDS presented in this study [34], can begin discovering in a network with no past information. Without making any inferences about the nature of the attacks, an incremental clustering approach is utilized to determine the number and shape of clusters. The outcomes are appropriate for sharing with other ITL-IDS instances. With each cycle, transfer learning increases the amount of knowledge in the target settings. Using a transfer learning strategy, Idrissi *et al.* [23] provided a method for updating DL-IDS that enables to retrain and fine-tuning of previously learned models on tiny datasets with novel attack patterns. They constructed a CNN-based IDS in their trials using the BoT-IoT dataset, then updated it with little data from a labeled ToN-IoT dataset.

Most of the existing models were evaluated on an outdated dataset such as KDD 19 and NSLKDD. Our literature research found that there is still room for improvement despite the outstanding detection accuracies achieved. The shortcomings are caused by inconsistencies and issues with the current datasets. It is still early in the region's growth. Most investigators focused on ML techniques and combined several methods to provide a more effective and genuine solution for a complex

dataset with few attackers. But still, they haven't put the suggested model to the test in actual situations. They tried the Contiki Simulator, but the real world is unlike the simulation. As a result, a model evaluated in a simulation environment may behave differently in a real-world setting. Because of this, we believe that the model and work provided in this article will be capable of producing accurate findings.

4 Proposed work

The proposed work is divided into two stages. In stage 1, we tested the work on benchmark datasets. Moreover, in stage 2, we tested the work in a real-time environment developed for smart homes. Figure 1 presents the system architecture for detecting malicious events on a smart home IoT network. First, all devices are connected to the local network to communicate with each other and share information. All the data passes through a router, and the Raspberry Pi module captures all the packets to analyze the packets transferring across the network. The features to be used in the trained model are extracted from the captured packets. Then the extracted features are passed to the proposed IDS system, where the trained model will classify the data into general or specific attack categories.

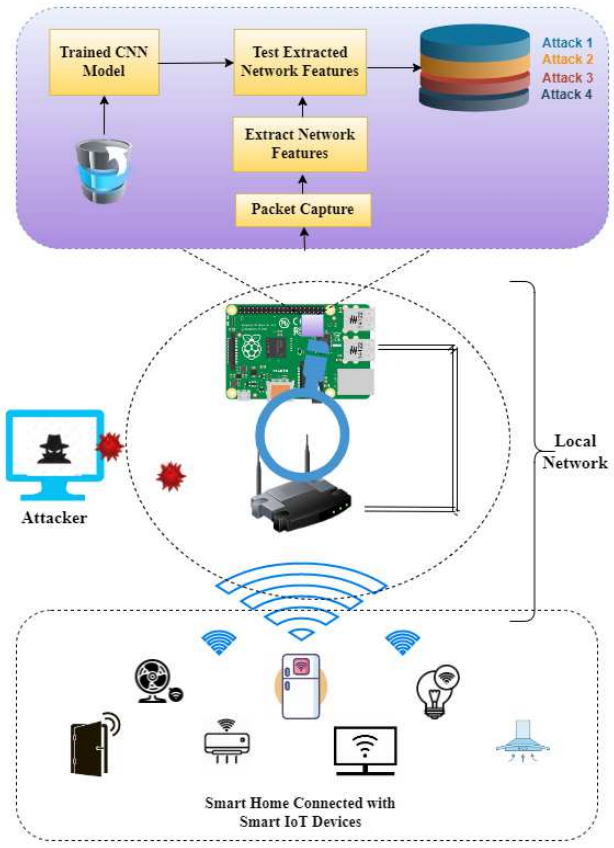

Fig. 1 Proposed System Architecture

4.1 Proposed 1D CNN Model

In this study, we develop a 1D CNN for IoT network anomaly detection. Figure 2 shows the proposed model's general structure, and Figure 3 shows the CNN block with layer description. An input layer, three convolution layer blocks, a flatten layer, two dense layers that are entirely connected, and an output layer are used to build this model. Each block contains two convolutional layers, a normalization layer, a pooling layer, and a dropout layer. In addition, each block includes a pooling layer. The reshaping approach may provide inputs to the input layer. The reshaping system converts the data from the network into a format that CNN can read.

Convolutional layer blocks are the primary constituents that go into the construction of a CNN. The convolution layer is responsible for extracting attributes from the input data and learning all information from these attributes. All of the information is brought up to a consistent standard by the normalization layer of the neural network. The normalization layer is responsible for standardizing the output of the convolution layer so that the average pooling layer may use it. The pooling layer improves information quality by concentrating features into sub-maps that already have prominent traits. The average pooling layer makes an estimate of the total number of updates that have occurred across the function map. This allows it to calculate the total number of features present in each patch. Overfitting is a risk when working with CNN; as a result, the parameters of the test dataset need to be adjusted with care. A dropout layer can reduce the likelihood of an overfitting problem by skipping over some neurons during training.

The first and second convolution layers employ 32 filters, kernel size 2, uniform kernel initializer, and the "same" padding parameter. The "same padding" refers to feature map sizes identical to the input feature maps (for stride 1). The normalization layer adjusts the activation of the previous layer for each sample in a batch independently. By summarising attributes in a feature map segment, the average pooling layer delivers a way to downsample feature maps. Pool size two is utilized in the average pooling layer. To regularise the training data model and reduce overfitting, we used a dropout layer with a dropout value of 0.05. Except for the filters, which are doubled in each consecutive block and kernel size 1 in the second and third blocks, each of the

three convolution blocks employs identical parameters. The model is flattened by applying the flatten layer, which transforms all the arrays from pooled characteristic maps into a single successive linear vector. The flattened layer is linked to a dense layer1 that is completely connected, and the dense layer2 is linked to the output layer. The output layer has neurons depending on the number of classes present in the dataset, whereas dense layer1 has 256 neurons and dense layer2 has 512 neurons.

4.2 Hyper-parameters setting

4.2.1 Loss Function

To calculate the loss function, we used sparse categorical cross-entropy [7]. When training, we want to minimize the loss between the expected and actual outputs, which is a function that compares the detected and actual output values. In a case where the number of classes is greater than two, we utilize categorical cross entropy [58] loss function defined in Equation 6.

$$L = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{c} y_{ij}\log p_{ij} \tag{6}$$

Here, n is the number of samples we have passed, y represents the label [0,1,2..], c is the number of classes, and p is the probability of that category. The probability that the provided input fits into each pre-set category, selecting the class with the highest probability as the final result.

4.2.2 Optimizer

Deep learning strategies often use several intrinsic optimizers, some of which are Gradient Descent (GD), Stochastic GD, AdaDelta, Adam, and others. In our model, we have implemented the Adaptive Gradient (Adagrad) optimizer [15] to minimize the loss function by modifying the network weights. Equation 7 demonstrates the calculation underpinning the Adagrad optimizer, which is used to compute weights for neurons:

$$w_t = w_{t-1} - {\eta_t}' \times \frac{\partial L}{\partial w_t} \tag{7}$$

w_t represents the weight of a neuron in the t^{th} iteration, whereas η' represents the learning rate. ∂L is the partial derivative of the Loss with respect to the ∂w_t. And the formula provided in Equation 8 is used to get η'.

$${\eta_t}' = \frac{\eta}{\alpha_t + \varepsilon} \tag{8}$$

η represents the constant learning rate of 0.01 with $1e-6$ decay, and ε represents a tiny positive value. Equation 9 is used to compute the value of α_t.

$$\alpha_t = \sum_{i=1}^{t}(\frac{\partial L}{\partial w_i})^2 \tag{9}$$

Here, α_t is computed by adding all previous t iterations of partial derivative $\frac{\partial L}{\partial w_t}$ by the square root operation. Therefore, when α_t increases, η_t' diminishes. Furthermore, while η_t' decreases, weights gradually decrease to meet the global minimum effectively.

We have tested several combinations of neuron layers. We precisely changed the size of the kernel, the number of filters, and the blocks that make up the convolutional layer. The results show that the CNN model functions more effectively. We chose Adagrad optimizer and modify the weights using a sparse categorical cross-entropy loss function. The learning rate determines the size of a model's steps during each iteration and is crucial to deep learning algorithms. In a series of experiments, we varied the Adagrad optimizer's learning rate (0.01, 0.001, 0.0001), and 0.01 was shown to be the optimal learning rate with the most elevated detection rate. We utilized an early halting strategy to prevent overfitting. The training operation will be terminated if the validation loss has not decreased after a specific number of iterations. For the maximum possible network output throughout the testing period, the number of epochs must be adjusted until the precision vs. epochs no extended rises. In our model, we used 20, 50, and 100 epochs. We consider 50 epochs the ideal number since 1D CNN models converge in 50 epochs or less.

A deep learning algorithm's activation function is crucial. Both the dense layer and the convolution layer use the ReLu activation function. The output layer uses softmax activation. Another essential hyperparameter for deep learning systems to modify is batch size. Expanding the batch size may enhance computation parallelization. Model training might be greatly faster as a consequence. Even though they result in equal training downfalls to smaller batch sizes, bigger batch sizes have been seen to generalize testing findings inadequately [25]. The distinction between train and test errors is referred to as the generalization gap. To determine the ideal batch size, we conducted trials with various batch sizes (16, 32, 64, 128, 256). The CNN model was trained and tested using a batch size of 128; this size was deemed to be ideal.

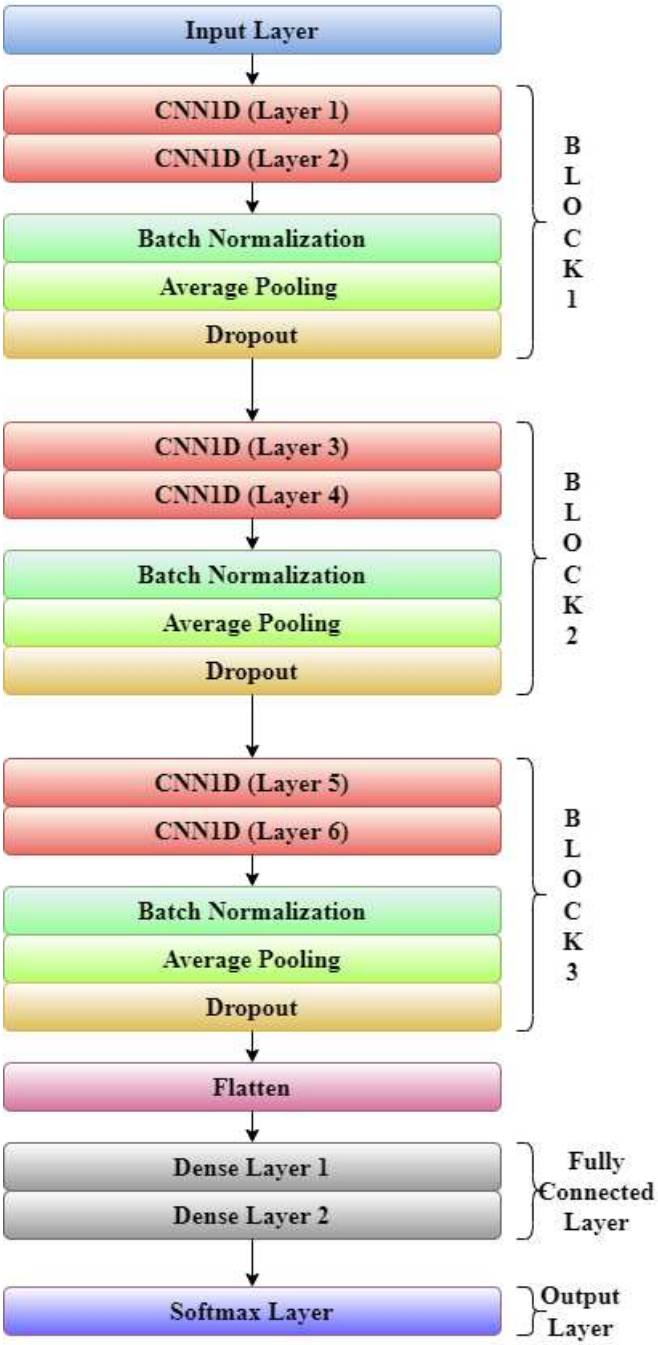

Fig. 2 Proposed CNN Model

4.3 Transfer Learning (TL) Technique

TL is a learning method where a model designed for one task is utilized as the foundation for another activity. As illustrated in Figure 4, we have deployed a pre-trained multiclass CNN model using transfer learning for multiclass and binary classification. We employed the NF-UQ-NIDS dataset [44] to train the CNN model described in the first phase. The NF-BoT-IoT, NF-CSE-CIC-IDS2018, NF-ToN-IoT, and NF-UNSW NB15 datasets were then classified using the same pre-trained model. We applied transfer learning ideas in multiclass and binary classification for these datasets since they are subsets of the NF-UQ-NIDS datasets.

The output layer of the pre-trained multiclass CNN model is removed for transfer learning. A new output layer with neurons depending on the attack categories in the dataset is added to the model. The dense and output layers are the only ones that remain active throughout the training phase of the model. All other layers of the pre-trained model are deactivated. Throughout the process of developing the classification model, the convolution, normalization, dropout, pooling, and flatten layers were all frozen, as seen in Figure 4. During the training period, it was only possible for the Dense and Output layers to acquire new knowledge.

5 Experiments and Results

We have tested the performance of our model on benchmark datasets and further on a real-time testbed.

5.1 Dataset

5.1.1 NF-BoT-IoT-Version-1 based Datasets

Due to the limited ability to practice ML-based Network Intrusion Detection Systems, there is a gap between academic investigation and experimental implementations. This article [44] solves this restriction by offering five NetFlow-based NIDS datasets with a standardized characteristic set as shown in Table 1. NF-UNSWNB15, NF-ToN IoT, NF-BoT IoT, and NF-CSE CICIDS2018 are the existing standard NIDS datasets utilized to create this dataset. The researchers altered the packet capture files from various datasets to the NetFlow format with a consistent attribute set, eliminating the problem of varying characteristics in separate datasets. A comprehensive NF UQ NIDS dataset that incorporates the data is shown in Figure 5. The recently published dataset illustrates the benefits of the standard characteristic of the dataset.

5.1.2 CICFlowmeter based Datasets

The CIC-Flowmeter is an open-source flow creation platform that uses pcap data to build network features. We have used CIC Flowmeter-based dataset [52], which contains 80 distinct features. We have taken IoT-DS2, MQTT-IoT-IDS, and IoT Network Intrusion Dataset

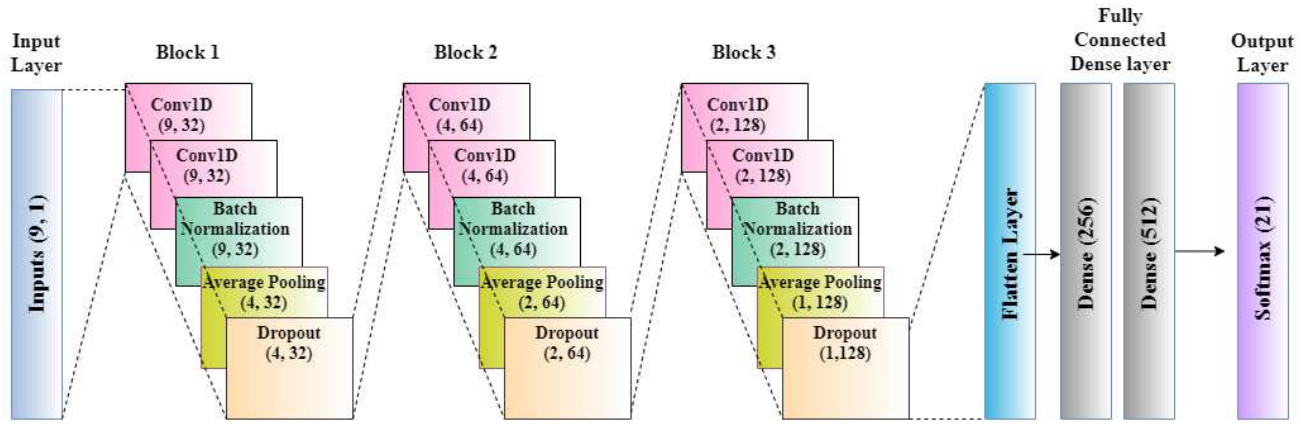

Fig. 3 Proposed CNN model with the number of neurons at each layer

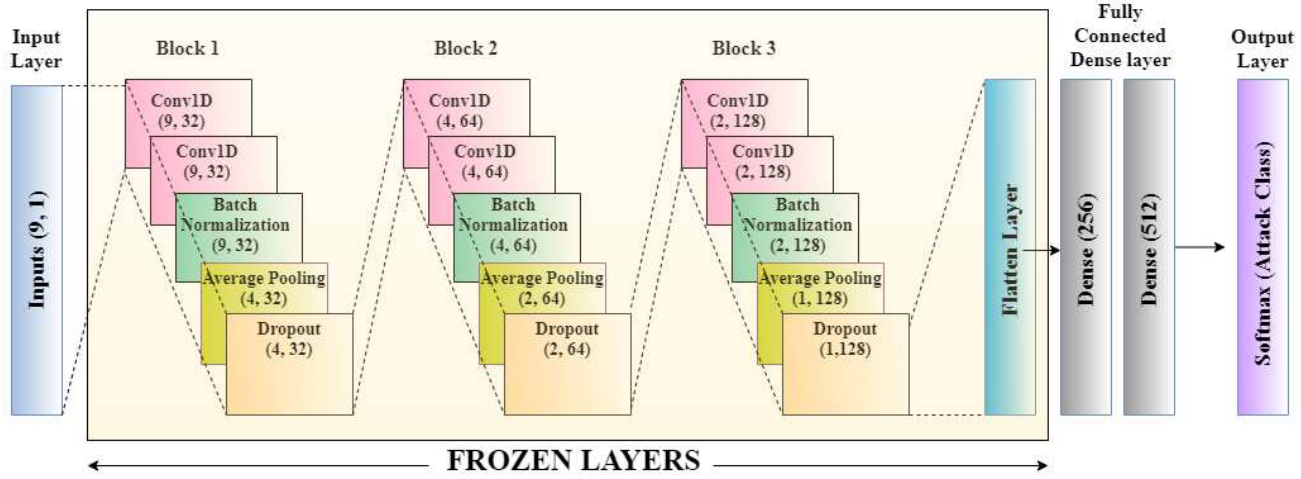

Fig. 4 Layers of the proposed model using transfer learning

Table 1 Netflow based Dataset Characterization

Datasets	Total Dataset	Train Dataset	Test Dataset	No. of classes	Features
NF-BoT-IoT	600100	480080	120020	5	L4 SRC PORT, L4 DST PORT, PROTOCOL, TCP FLAGS, L7 PROTO, IN BYTES, OUT BYTES, IN PKTS, OUT PKTS
NF-ToN-IoT	1379274	1103419	275855	10	
NF-UNSW-NB15	1623118	1298494	324624	10	
NF-CSE-CIC-IDS2018	8392401	6713920	1678481	15	
NF-UQ-NIDS	11994893	9595914	2398979	21	

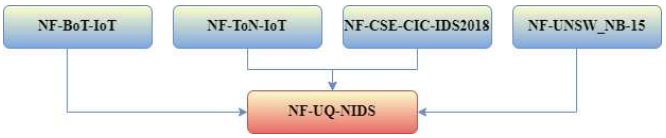

Fig. 5 Data Concatenation

that are available on [4] and CIC-ToN-IoT dataset available on [45]. To pass the dataset into the proposed model, we have selected the best nine features from the 83 features using the random forest's feature importance [6] method. It calculates the importance of a feature using Gini importance. Table 3 shows the selected features of the datasets and other related information regarding the dataset.

5.2 Results on benchmark datasets

We evaluated our CNN-based model's performance in binary and multiclass classification. Our suggested model was trained using a GPU and 12GB of RAM on Google Colab. Table 4 and Table 5 display the outcomes of the binary and multiclass classifications for the Netflow-based dataset, respectively. Similar to this, Table 6 dis-

Table 2 Experimental Setup Environment

Model Training Environment	
System Configuration	**Model**
Operating System	Windows 11
Working Environment	Python
RAM	8GB
Processor	Intel(5) 10th Gen
Libraries	Numpy Scikit-learn TensorFlow Keras
Model Testing Environment	
Device	Raspberry Pi 3
Operating System	Raspbian
RAM	1GB
Libraries	Numpy Scikit-learn TensorFlow Lite Pyshark

plays the outcomes of the multiclass classification using the CICflow-based dataset.

5.3 Testbed for Real-time Intrusion Detection using Proposed Model

We have tested our trained model by deploying it on an edge device, i.e., Raspberry pi3. The benefit of edge devices is that they usually have low power consumption. Deep learning models need many resources; thus, deployment on edge devices must be lightweight. Typically, inferences are formed on the edge device to do away with the necessity for downloading and uploading to a distant server. As a result, inference time is less. So, after the training phase, we converted the trained model into a lightweight model using the TensorFlow Lite module. In the model optimization phase, we converted data type from float64 to float32 or can be converted into integer type that uses low memory. After the optimized model, we have the model with the tflite extension. Finally, the lightweight model is ready for deployment on the edge device. Figure 6 shows the complete workflow of model conversion from train model to lightweight model for the deployment on constrained devices.

Here, we have used Raspberry pi 3 to check the performance of the proposed model. We have written a script in python in which we capture packets. For capturing the live packets, we have used pyshark [5] open source python module. After capturing the packets, we extract those packets that we used to train the model and detect the intrusions using the trained model. Furthermore, we created an IoT-based fire alarm system and smart room light system to verify the malicious packets as a real-time testbed. Both devices use

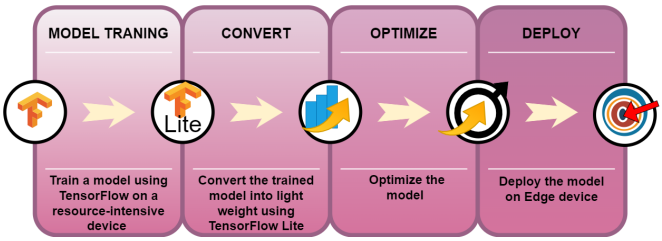

Fig. 6 Deep TensorFlow model conversion into light weight model workflow

an ESP8266 NodeMCU microcontroller. Data was sent from the NodeMCU using a WiFi module to a nearby Raspberry Pi 3 server, which uses the packet capture method to continually sniff the network and collect packets in real-time. In order to evaluate the effectiveness of the model in real time, we used the script and Ettercap (kali) on two separate computers. We attacked the same network using a Denial of Service assault and a MITM attack. Both attacks were carried out from separate systems. The testbed architecture is shown in Figure 7.

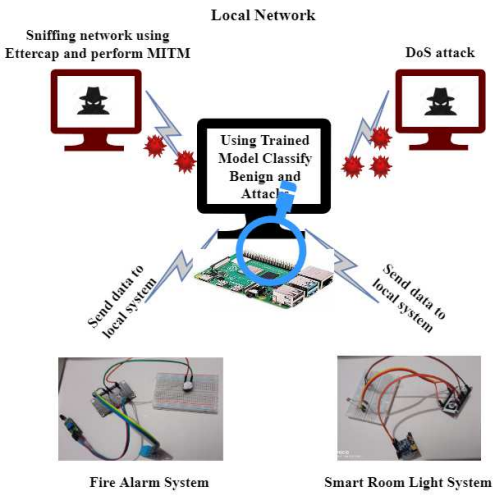

Fig. 7 Testbed Architecture

Moreover, we have also compared our proposed model with the existing models as shown in Table 8, and a comparison with the existing CNN model is shown in Table 9. Additionally, we have verified our proposed real-time model on an edge device with limited resources, i.e., Raspberry pi3.

5.4 Results of Testbed

We have tested our proposed model on Raspberry pi3. We write a python script that is continuously run and sniffs the packets. The captured packets are transformed

Table 3 CICflowmeter based Dataset Characterization

Datasets	**Total Dataset**	**Train Dataset**	**Test Dataset**	**No. of Classes**	**Features**
IoT DS2	1438157	1150525	287632	17	'Bwd_Pkt_Len_Std', 'Bwd_IAT_Min', 'Init_Fwd_Win_Byts', 'Bwd_IAT_Mean', 'TotLen_Bwd_Pkts', 'Bwd_IAT_Tot', 'Subflow_Fwd_Byts', 'Dst_Port', 'Src_Port'
IoT Network Intrusion	625783	500626	125157	5	
MQTT-IoTIDS2020	3654006	2923204	730802	5	
CIC-ToN-IoT	5351760	4281408	1070352	10	

Table 4 Binary Class Classification of NetFlow based Datasets

Classification Models	Measurements	Datasets				
		NF-ToN-IoT	NF-BoT IoT	NF-UNSW NB15	NF-CSE CIC IDS2018	NF-UQ NIDS
Sarhan *et al.* [44]	Accuracy	99.66%	93.82%	98.62%	95.33%	98.34%
	Precision	99.67%	93.70%	90.70%	94.71%	95.66%
	Recall	–	–	–	–	–
	F1-score	100%	97%	85%	83%	94%
Proposed	Accuracy	99.19%	98.79%	98.32%	98.99%	98.75%
	Precision	99.20%	98.69%	98.24%	98.98%	98.75%
	Recall	99.19%	98.79%	98.32%	98.99%	98.75%
	F1-score	99.19%	98.67%	98.26%	98.98%	98.74%

Table 5 Multi Class Classification of Netflow based Datasets

Classification Models	Measurements	Datasets				
		NF-ToN-IoT	NF-BoT IoT	NF-UNSW NB15	NF-CSE CIC IDS 2018	NF-UQ NIDS
Sarhan *et al.* [44]	Accuracy	–	–	–	–	–
	Precision	56.34%	73.58%	97.62%	71.92%	70.81%
	Recall	–	–	–	–	–
	F1-score	60%	77%	98%	80%	79%
Proposed	Accuracy	70.54%	83.67%	97.21%	97.53%	93.74%
	Precision	59.91%	87.13%	96.59%	96.62%	92.75%
	Recall	70.54%	83.67%	97.21%	97.53%	93.74%
	F1-score	62%	83%	96%	96%	92%

Table 6 Multi Class Classification of CICFlowmeter Based Datasets

Classification Models	Measurements	Datasets			
		IoT DS2	IoT Network Intrusion	MQTT-IoT-IDS2020	CIC-ToN-IoT
Ullah *et al.* [52]	Accuracy	99.70%	86.28%	99.92%	-
	Precision	99.74%	88.84%	99.90%	-
	Recall	99.70%	86.28%	99.92%	-
	F1-score	99.74%	87.54%	99.91%	-
Proposed	Accuracy	97.17%	96.20%	95.33%	86.96%
	Precision	97.19%	96.13%	96.07%	83.49%
	Recall	97.17%	96.20%	95.33%	86.96%
	F1-score	96.56%	96.13%	95.23%	81.75%

through the proposed model, which classifies the incoming packet belonging to benign or anomaly. To check the performance, we applied intense scanning of the network through the Zenmap [8] using the IP address of the Raspberry pi3, and we successfully identified malicious events on the network. We have taken 100 packets to analyze the result, in which the model detected 19 times as an anomaly and 81 times benign. The confusion matrix is shown below:

6 Discussion

This section compares the outcomes of CNN models with other existing research investigations. Our suggested models were noticeably better at spotting irreg-

ularities in IoT environments. This paper looked at the possible use of a CNN for the problem of anomaly identification in IoT networks. We examined the capability of CNN to locate and classify anomalies. The ability of CNN to identify and categorize anomalies in IoT networks utilizing binary and multiclass classification via transfer learning was evaluated.

The pre-trained multiclass CNN paradigm takes advantage of transfer learning for multiclass and binary classification. According to our best knowledge, the only study [52] that has employed a transfer learning technique for abnormality detection, where a pre-trained multiclass model is reused for binary and multi-class anomaly detection. However, they were not tested their model in real-time. The main reason why transfer learning is used is to reduce classification and run-time complexity. In addition to this, transfer learning is an added advantage of our proposed model. The training, validation, and testing times needed for classification are significantly decreased using the transfer learning approach. Table 4 shows the results of our proposed work in binary class classification. In which one class belongs to the anomaly behavior of the network and another one belongs to the normal behavior of the network. Our proposed work is also compared with other existing work and the result shows the Table 5 shows the precision of transfer learning is 87.13%, 59.91%, 96.62%, for NetFlow-based NF-BoT IoT, NF-ToN-ToN, NF-CSE CICIDS2018, is better than Sarhan *et al.* model which is 73.58%, 56.34%, 71.92%, respectively. The comparison outcome demonstrates that our suggested model's performance is superior to the existing model.

Table 7 UNSWNB-15 and CSE-CICIDS2018 multiclass classification comparison

References	Datasets	Accuracy	Proposed 1D CNN
[57]	UNSWNB-15	90.21%	97.21%
[42]	UNSWNB-15	92.39%	
[39]	UNSWNB-15	74%	
[59]	UNSWNB-15	86.11%	
[17]	CSE CICIDS2018	90.25%	97.53%
[18]	CSE CICIDS2018	97.38%	
[16]	CSE CICIDS2018	96%	

Table 8 UNSWNB-15 Binary classification comparison

References	Datasets	Accuracy	Proposed 1D CNN
[43]	UNSWNB-15	95.71%	98.32%
[50]	UNSWNB-15	97%	

Table 9 Multiclass Classification Comparison with CNN model

Refernces	Model	Accuracy	Precision	Recall	F1-score
[31]	CNN	86.95%	89.56%	87.25%	88.41%
[28]	CNN	98.02%	97.71%	98.39%	98.05%
[32]	CNN	95.86%	–	–	–
[56]	CNN	97.34%	–	–	–
[12]	CNN	92.53%	–	–	–
[27]	CNN	94.32%	94.89%	94.32%	94.33%
[19]	FNN	98.09%	98.88%	98.88%	98.88%
Proposed	1D CNN	97.53%	96.62%	97.53%	96%

7 Conclusion and Future Work

Deep learning techniques have proven to be effective in classifying anomalies across a wide range of domains. In order to identify and categorize binary and multiclass abnormalities, this paper presents an anomaly detection system for IoT networks. We provide a method for identifying unusual behavior in IoT networks using NetFlow and CICFlowmeter-based datasets derived from an older dataset. Using 1D convolutional neural network models, we categorize various abnormalities. The model is trained on a large dataset along with transfer learning to improve the dataset's performance and reduce the model's training complexity. The performance of the model is also compared with the existing work. The results show 97.21% and 97.53% accuracy with the UNSWNB-15 and CSE CICIDS2018 datasets, which is quite good in the results compared with the existing model. Similarly, the proposed CNN is compared with the existing CNN model, and the results show the model's efficiency. The proposed model is also tested on a real testbed by deploying a fire alarm and a smart room light system. The Raspberry pi3 is an edge device where both devices send their data. The proposed IDS model was deployed on the Raspberry pi3 and monitored all incoming and outgoing data.

In the future, we will also define rules for network safety by analyzing the network packet behavior and ensemble with an intelligent deep learning-based model to improve the detection performance and reduce the false positive rate.

Ethical Statemenet: The manuscript in part or in full has not been submitted or published anywhere.

References

1. Home — mordor intelligence. https://www.mordorintelligence.com/. (Accessed on 10/27/2022).
2. Internet of things [iot] market size, share & trends, 2029. https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307. (Accessed on 10/27/2022).
3. Internet of things (iot) market to exhibit 25.4% cagr till. https://www.globenewswire.com/news-release/2021/08/23/2284666/0/en/Internet-of-Things-IoT-Market-to-Exhibit-25-4-CAGR-till-2028- Critical-Need-to-Virtually-Monitor-Operations-to-Boost-Growth-Fortune-Business-Insights.html. (Accessed on 10/27/2022).
4. Ontario tech university. https://sites.google.com/view/iotdataset1.
5. pyshark · pypi. https://pypi.org/project/pyshark/. (Accessed on 09/01/2022).
6. sklearn.ensemble.randomforestclassifier — scikit-learn 1.2.2 documentation. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. (Accessed on 05/12/2023).
7. tf.keras.metrics.sparsecategoricalaccuracy tensorflow v2.12.0. https://www.tensorflow.org/api_docs/python/tf /keras/metrics/SparseCategoricalAccuracy. (Accessed on 05/12/2023).
8. Zenmap - official cross-platform nmap security scanner gui. https://nmap.org/zenmap/. (Accessed on 10/18/2022).
9. Eirini Anthi, Lowri Williams, Małgorzata Słowińska, George Theodorakopoulos, and Pete Burnap. A supervised intrusion detection system for smart home iot devices. *IEEE Internet of Things Journal*, 6(5):9042–9053, 2019.
10. Muhammet Baykara and Resul Das. A novel honeypot based security approach for real-time intrusion detection and prevention systems. *Journal of Information Security and Applications*, 41:103–116, 2018.
11. Elhadj Benkhelifa, Thomas Welsh, and Walaa Hamouda. A critical review of practices and challenges in intrusion detection systems for iot: Toward universal and resilient systems. *IEEE Communications Surveys & Tutorials*, 20(4):3496–3509, 2018.
12. Seok-Jun Bu and Sung-Bae Cho. A convolutional neural-based learning classifier system for detecting database intrusion via insider attack. *Information Sciences*, 512:123–136, 2020.
13. Sarika Choudhary, Apurba Dey, and Nishtha Kesswani. Crids: Correlation and regression-based network intrusion detection system for iot. *SN Computer Science*, 2(3):1–7, 2021.
14. Sarika Choudhary, Nishtha Kesswani, et al. A hybrid classification approach for intrusion detection in iot network. *Journal of Scientific and Industrial Research (JSIR)*, 80(09):809–816, 2021.
15. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.
16. Laurens D'hooge, Tim Wauters, Bruno Volckaert, and Filip De Turck. Inter-dataset generalization strength of supervised machine learning methods for intrusion detection. *Journal of Information Security and Applications*, 54:102564, 2020.
17. Rawaa Ismael Farhan, Abeer Tariq Maolood, and NidaaFlaih Hassan. Performance analysis of flow-based attacks detection on cse-cic-ids2018 dataset using deep learning. *Indones. J. Electr. Eng. Comput. Sci*, 20(3):1413–1418, 2020.
18. Mohamed Amine Ferrag, Leandros Maglaras, Sotiris Moschoyiannis, and Helge Janicke. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, 50:102419, 2020.
19. Mengmeng Ge, Xiping Fu, Naeem Syed, Zubair Baig, Gideon Teo, and Antonio Robles-Kelly. Deep learning-based intrusion detection for iot networks. In *2019 IEEE 24th pacific rim international symposium on dependable computing (PRDC)*, pages 256–25609. IEEE, 2019.
20. Alexander Gluhak, Srdjan Krco, Michele Nati, Dennis Pfisterer, Nathalie Mitton, and Tahiry Razafindralambo. A survey on facilities for experimental internet of things research. *IEEE Communications Magazine*, 49(11):58–67, 2011.
21. Ke He, Dan Dongseong Kim, and Muhammad Rizwan Asghar. Adversarial machine learning for network intrusion detection systems: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 2023.
22. Fatima Hussain, Rasheed Hussain, Syed Ali Hassan, and Ekram Hossain. Machine learning in iot security: Current solutions and future challenges. *IEEE Communications Surveys & Tutorials*, 22(3):1686–1721, 2020.
23. Idriss Idrissi, Mostafa Azizi, and Omar Moussaoui. Accelerating the update of a dl-based ids for iot using deep transfer learning. *Indones. J. Electr. Eng. Comput. Sci.*, 23(2):1059–1067, 2021.

24. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.
25. Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.
26. Nishtha Kesswani and Basant Agarwal. Smartguard: an iot-based intrusion detection system for smart homes. *International Journal of Intelligent Information and Database Systems*, 13(1):61–71, 2020.
27. Izhar Ahmed Khan, Nour Moustafa, Dechang Pi, Karam M Sallam, Albert Y Zomaya, and Bentian Li. A new explainable deep learning framework for cyber threat discovery in industrial iot networks. *IEEE Internet of Things Journal*, 2021.
28. Kannan Krithivasan, S Pravinraj, Shankar Sriram VS, et al. Detection of cyberattacks in industrial control systems using enhanced principal component analysis and hypergraph-based convolution neural network (epca-hg-cnn). *IEEE Transactions on Industry Applications*, 56(4):4394–4404, 2020.
29. Prabhat Kumar, Randhir Kumar, Govind P Gupta, and Rakesh Tripathi. A distributed framework for detecting ddos attacks in smart contract-based blockchain-iot systems by leveraging fog computing. *Transactions on Emerging Telecommunications Technologies*, 32(6):e4112, 2021.
30. Jiaqi Li, Zhifeng Zhao, Rongpeng Li, and Honggang Zhang. Ai-based two-stage intrusion detection for software defined iot networks. *IEEE Internet of Things Journal*, 6(2):2093–2102, 2018.
31. Yanmiao Li, Yingying Xu, Zhi Liu, Haixia Hou, Yushuo Zheng, Yang Xin, Yuefeng Zhao, and Lizhen Cui. Robust detection for network intrusion of industrial iot based on multi-cnn fusion. *Measurement*, 154:107450, 2020.
32. Xiang Liu, Ziyang Tang, and Baijian Yang. Predicting network attacks with cnn by constructing images from netflow data. In *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, pages 61–66. IEEE, 2019.
33. Maya Hilda Lestari Louk and Bayu Adhi Tama. Dual-ids: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. *Expert Systems with Applications*, 213:119030, 2023.
34. Ehsan Mahdavi, Ali Fanian, Abdolreza Mirzaei, and Zahra Taghiyarrenani. Itl-ids: Incremental transfer learning for intrusion detection systems. *Knowledge-Based Systems*, 253:109542, 2022.
35. Soosan Naderi Mighan and Mohsen Kahani. A novel scalable intrusion detection system based on deep learning. *International Journal of Information Security*, 20(3):387–403, 2021.
36. Nour Moustafa, Benjamin Turnbull, and Kim-Kwang Raymond Choo. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. *IEEE Internet of Things Journal*, 6(3):4815–4830, 2018.
37. Sarumathi Murali and Abbas Jamalipour. A lightweight intrusion detection for sybil attack under mobile rpl in the internet of things. *IEEE Internet of Things Journal*, 7(1):379–388, 2019.
38. Nataliia Neshenko, Elias Bou-Harb, Jorge Crichigno, Georges Kaddoum, and Nasir Ghani. Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys & Tutorials*, 21(3):2702–2733, 2019.
39. Phong Cao Nguyen, Dong Hai Duong, Thinh Truong Nguyen, Luan Anh Luong, Huong Hoang Luong, Hai Thanh Nguyen, et al. An intrusion detection approach for small-sized networks. In *Inventive Computation and Information Technologies*, pages 899–913. Springer, 2022.
40. Antonia Nisioti, Alexios Mylonas, Paul D Yoo, and Vasilios Katos. From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. *IEEE Communications Surveys & Tutorials*, 20(4):3369–3388, 2018.
41. Ayodeji Oseni, Nour Moustafa, Gideon Creech, Nasrin Sohrabi, Andrew Strelzoff, Zahir Tari, and Igor Linkov. An explainable deep learning framework for resilient intrusion detection in iot-enabled transportation networks. *IEEE Transactions on Intelligent Transportation Systems*, 2022.
42. Olugbemiga Solomon Popoola, Ikechukwu Ignatius Ayogu, Olamatanmi Josephine Mebawondu, Chukwuemeka Christian Ugwu, and Adebayo Olusola Adetunmbi. An evolutionary approach towards achieving enhanced intrusion detection system. In *2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON)*, pages 1–5, 2022.
43. 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.
44. Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. Netflow datasets for machine learning-based network intrusion detection systems. In *Big Data Technologies and Applications*, pages 117–135. Springer, 2020.
45. Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. An explainable machine learning-based network intrusion detection system for enabling generalisability in securing iot networks. *arXiv preprint arXiv:2104.07183*, 2021.
46. Sugandh Seth, Kuljit Kaur Chahal, and Gurvinder Singh. A novel ensemble framework for an intelligent intrusion detection system. *IEEE Access*, 9:138451–138467, 2021.
47. 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.
48. Nathan Shone, Tran Nguyen Ngoc, Vu Dinh 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.
49. Mehdi Sookhak, Helen Tang, Ying He, and F Richard Yu. Security and privacy of smart cities: a survey, research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(2):1718–1743, 2018.
50. Qiao Tian, Jingmei Li, and Haibo Liu. A method for guaranteeing wireless communication based on a combination of deep and shallow learning. *IEEE Access*, 7:38688–38695, 2019.
51. Zhihong Tian, Chaochao Luo, Jing Qiu, Xiaojiang Du, and Mohsen Guizani. A distributed deep learning system for web attack detection on edge devices. *IEEE Transactions on Industrial Informatics*, 16(3):1963–1971, 2019.
52. Imtiaz Ullah and Qusay H Mahmoud. Design and development of a deep learning-based model for anomaly detection in iot networks. *IEEE Access*, 9:103906–103926, 2021.
53. Monika Vishwakarma and Nishtha Kesswani. A two-stage intrusion detection system (tids) for internet of things. In *Advances in Deep Learning, Artificial Intelligence and Robotics*, pages 89–97. Springer, 2022.
54. Omar Abdel Wahab. Intrusion detection in the iot under data and concept drifts: Online deep learning approach. *IEEE Internet of Things Journal*, 9(20):19706–19716, 2022.
55. Weiping Wang, Zhaorong Wang, Zhanfan Zhou, Haixia Deng, Weiliang Zhao, Chunyang Wang, and Yongzhen Guo. Anomaly detection of industrial control systems based on transfer learning. *Tsinghua Science and Technology*, 26(6):821–832, 2021.
56. Hongyu Yang and Fengyan Wang. Wireless network intrusion detection based on improved convolutional neural network. *Ieee Access*, 7:64366–64374, 2019.
57. Yanqing Yang, Kangfeng Zheng, Chunhua Wu, Xinxin Niu, and Yixian Yang. Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. *Applied Sciences*, 9(2):238, 2019.
58. Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.
59. Ruijie Zhao, Guan Gui, Zhi Xue, Jie Yin, Tomoaki Ohtsuki, Bamidele Adebisi, and Haris Gacanin. A novel intrusion detection method based on lightweight neural network for internet of things. *IEEE Internet of Things Journal*, 2021.