





Research Article

A Framework for Identification and Classification of IoT Devices for Security Analysis in Heterogeneous Network

Hafiz Muhammad Zahid,¹ Yasir Saleem,¹ Faisal Hayat,¹ Farrukh Zeeshan Khan ,² Roobaea Alroobaea ,³ Fahad Almansour,⁴ Muneer Ahmad ,⁵ and Ihsan Ali ⁶

¹Department of Computer Science & Engineering, University of Engineering and Technology, Lahore, Pakistan

²Department of Computer Science, University of Engineering and Technology, Taxila, Pakistan

³Department of Computer Science, College of Computers and Information Technology, Taif University, P. O. Box 11099, Taif 21944, Saudi Arabia

⁴Department of Computer Science, College of Sciences and Arts in Rass, Qassim University, Buraydah 51452, Saudi Arabia

⁵School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Sector H-12, 44000 Islamabad, Pakistan

⁶Department of Computer System and Technology, Faculty of Computer Science and Information Technology, Universiti Malaya, 50603 Kuala Lumpur, Malaysia

Correspondence should be addressed to Ihsan Ali; ihsanalichd@siswa.um.edu.my

Received 12 June 2021; Revised 22 December 2021; Accepted 19 March 2022; Published 26 April 2022

Academic Editor: Antonio Guerrieri

Copyright © 2022 Hafiz Muhammad Zahid et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Internet of Things (IoT) is a promising technology enabling physical devices like cameras, home appliances, and other devices to communicate and interoperate with each other. The next wave transforms our homes, society, enterprises, and cities with the massive presence of IoT devices. The devices in the Internet of Things (IoT) may exchange sensitive data, and an important issue for any organization is to get the data secured and protected. The preliminary requirement for this is a mechanism detecting and reporting anomalies automatically to some central controller. Therefore, this mechanism should be able to classify legit IoT devices from unauthorized ones. Malicious IoT devices, non-IoT devices, and other types of man-in-the-middle traffic sources must be quarantined for noncompliance. This helps formulate administrative policies and regulate/police traffic in the network for better QoS management. This work proposed a framework-based hierarchical deep neural network (HDNNs) to distinguish IoT devices from non-IoT devices using a feature set of IoT-specific traffic. A system has been designed based on HDNN that classifies IoT devices to their specific categories and identifies new entrants with reasonable accuracy. The results show that HDNN can distinguish IoT and non-IoT devices with higher accuracy and as well as classify IoT devices into the respective classes with the required accuracy.

1. Introduction

In the modern age, billions of devices such as home appliances, traffic lights, and lampposts are connected to the Internet, also known as the Internet of Things (IoT) [1]. These devices, called IoT devices, have several sensors that generate valuable data. Communication among them is done under different protocols such as Wi-Fi, Bluetooth, ZigBee, and Ethernet, which helps IoT devices increase their functionality using actuators [2, 3] sometimes.

IoT devices are usually part of a heterogeneous network providing valuable services to society. The communication between these devices also facilitates aggregating and processing data and reacting to the environment's changes automatically. These devices are sometimes involved in sharing enormous data, forming a network that shares data at extremely high rates throughout and in a continuous fashion [4].

According to the reports of the International Data Corporation (IDC), the number of IoT devices will reach 41 billion by 2020, according to the report of the International

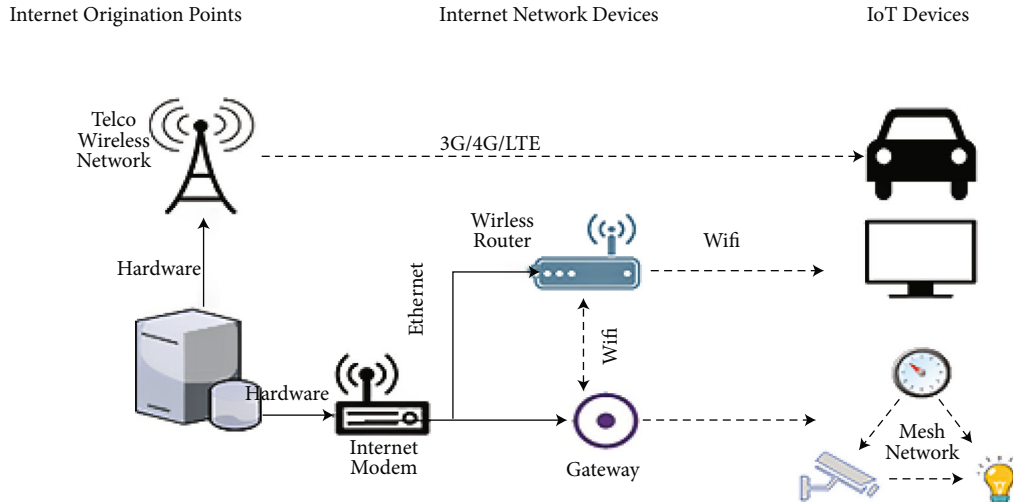


FIGURE 1: A heterogeneous network.

Data Corporation (IDC). The proliferation of IoT devices in IoT networks produces an operative challenge for administrators. The heterogeneous nature of IoT networks in big cities poses an additional asset management challenge as, in this case, the nature of IoT devices is diverse, and various departments install them. For instance, in such a heterogeneous network, light sensors can be fixed by the local council as shown in Figure 1. The local police division can install cameras, and different garbage and sewage sensors can be installed by the sanitation department. It is very difficult to identify the malfunctioning devices and their network location due to their invisibility [5].

The absence of direct human involvement differentiates the Internet of Things (IoT) from the traditional Internet. An IoT device can generate information using changes in the environment around, analyze it, and act upon it autonomously, however, with a price considering the data privacy, security, and protection [6]. Researchers have warned of the prospective risk of large numbers of unprotected devices communicating on the Internet. Therefore, developers and manufacturers have been struggling to develop a robust security system for IoT networks. In 2013, a researcher at the dev environment in organizational security service discovered the first IoT malware. According to the above analysis, more than 25% of the malware consisted of devices except for computers, such as smart cameras, smart TVs, smartwatches, and other home appliances [7].

Another problem is that the manufacturers of IoT devices do not provide regular updates for their devices unless users initiate firmware updates, owing to constrained resources [8]. These devices cannot run full-fledged security mechanisms. Therefore, IoT devices are prone to attacks (e.g., their default login passwords and unpatched bugs) for more extended periods [9].

IoT devices work mainly in an unattended environment, so there is a fair chance that an intruder may intentionally gain physical access to them. Resultantly, intruders may gain important information through a communication channel by secretly listening to the conversation because most IoT

devices use wireless links. These devices do not incorporate robust security features because computation and power resources are limited [10]. Implementation of solid security mechanisms is not possible due to the limited resources available and untrustful interaction with the environment. Considering the possibility of vulnerable IoT devices in an IoT network, there must be a robust security solution based on patching the vulnerabilities from time to time [11].

Nowadays, different organizations also facilitate IoT device connectivity, which might obtrude security threats to their networks. Organizations must be capable of determining the devices connected to their networks. They should provide a mechanism for identifying whether the connected devices in their networks are legit and do not pose a risk or threat [12].

Analysis of real-time network traffic has been used in several proposals for the identification of devices in general and for the classification of legit devices from nonlegit ones [13]. We can state that network traffic traces have been proven to differentiate IoT devices from non-IoT devices as there is a substantial difference in the data flow pattern of non-IoT devices compared to IoT devices. An IoT device may work when some trigger occurs. For example, object-detecting sensors work only when someone is passing in front of the sensor. However, considering IoT device classification alone, it is mostly very difficult to classify the network traffic of a device into a fixed pattern and to create an invariant profile even for the same types of IoT devices such as Drop Camera and Withings Smart Baby Monitor that are both cameras from different vendors or manufacturers. However, the traffic generated by these two cameras is another pattern, as shown in Figure 2.

In another scenario, as shown in Figure 2, the traffic generated from a Netatmo Weather Station is similar to the traffic generated by the baby monitor. Therefore, for a better classification of devices communicating in the heterogeneous network, it is important to identify a pattern that may help to place the devices in their respective category even if the devices generate the same kind of data.

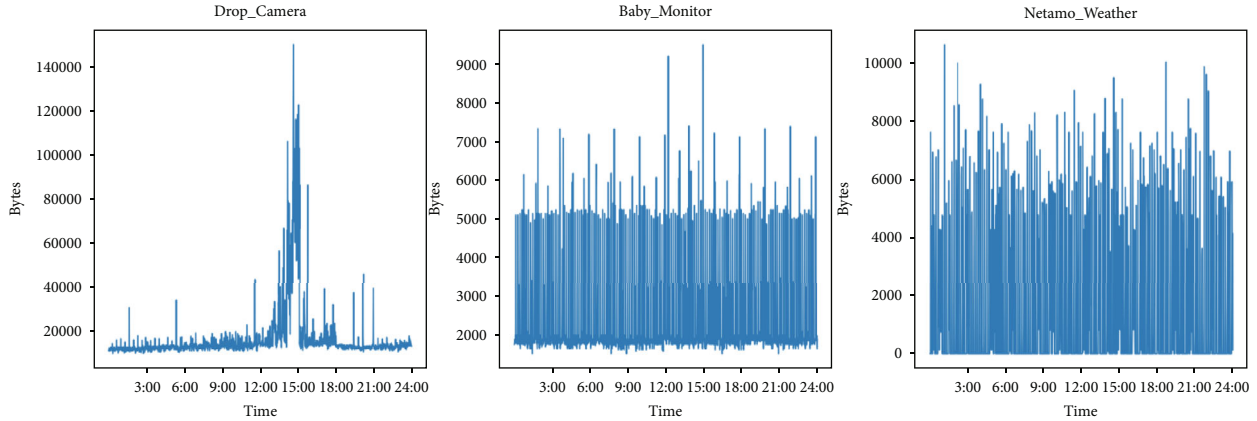


FIGURE 2: Daily traffic volume from three IoT devices [13].

IDE identification and classification of IoT devices in a heterogeneous network have been proposed based on port information, protocol, and MAC address [14]. By using the prefix (organization unique identifier (OUI)) of a MAC address, it is easy to identify the manufacturer of the device, but this information is not enough to guess the device functionality as many manufacturers produce diverse nature of devices. Additionally, manufacturers of different IoT devices may buy NICs from a third party, and it is impossible to know about IoT devices just by using the prefix (OUI) of the MAC address. Another aspect is this: The MAC address of IoT devices may even be spoofed [15].

In the recent past, multiple researchers have put forward many methods for automatic identification and classification of IoT devices with some constraints or rules [16]. These proposals mainly focused on the identification and classification of those IoT devices which have already been identified as part of the network under study. However, there is no proposal to identify and classify new devices entering the network. For example, when an unknown IP-based camera enters the network, it can be not easy to identify and classify it based on the previous IP base camera records.

Based on the discussion above, there is a need to build a holistic security solution for IoT networks. The preliminary requirement for this is the identification and classification of connected and incoming devices. In this paper, the challenge of identification and classification of IoT devices in the heterogeneous network has been taken up. We have used real-time network traffic and developed a method to differentiate IoT devices from non-IoT devices as well as classify IoT devices into their respective categories. IP addresses have not been used as a parameter. The reason is this: IP addresses can be spoofed very easily. We have used network traffic statistics and necessary metadata instead of relying on the deep packet inspection and proposed a hierarchical deep neural network (HDNN) framework that can discriminate between IoT devices from non-IoT devices and also identify the type of IoT devices with reasonable accuracy.

2. Literature Review

IoT is a network in which physical devices with sensors, software, and some other technologies act as tiny computers. These devices can collect data from the environment and process the data and then send this data to other devices with the help of internet connectivity. The exponential growth of IoT devices like smartwatches, smart cameras, lights, sensors, energy management devices, and other different types of devices in the intelligent environment has gained a ballistic interest of researchers to develop improved frameworks using ML and DL approaches to provide better services to human-kind in their lives. Researchers from different fields such as computer science, electrical, and other disciplines have paid great efforts to deal with this exponential growth. They are concerned about finding automated approaches for the security of IoT systems. In this section, we have covered closely related work done by different researchers on IoT for different categories, including IoT device identification, time-series data classification, and network traffic analysis using the manual as well as ML and DL approaches.

A lot of work has been done by researchers for characterizing internet traffic from 2005 to 2012. This work is primarily focused on web application detection such as Skype, mail, web browsing, and peer-to-peer applications but characterized IoT device traffic commonly known as mobile to mobile communication (M2M).

2.1. Machine Learning Approaches for Traffic Analyses. Much work has been done over the last few years using different types of machine learning and deep learning algorithms to classify traffic applications for specific computers and identify malware and botnets in the networks.

Lopez-Martin et al. [17] proposed a framework for a network traffic classifier (NTC) using a combination of DL models, CNN, and RNN. This framework was used to classify traffic flows like HTTP, SMTP, Telnet, YouTube, QUIC, and Office365 with six different features of traffic such as source port number, port number of destination, payload volume, transport control protocol (TCP) window size, interarrival time, and direction of traffic, and these features

were derived from first twenty packets of traffic flows. It was shown that the proposed framework with the combination of both CNN and RNN was better in the perspective of detection than other alternative algorithms without the need for any feature extraction or selection, which is mostly required when using different machine learning models.

In [18, 19], the authors used a combination of flow level features with different types of packet-level aspects such as packet size, byte payload distribution, packet interarrival times, and TLS handshake metadata (cipher suite code) to expand their work on the detection of malicious or illegal behavior on the network. Extraction of feature tools from the network was developed by researchers and was launched as an open-source tool for feature extraction.

2.2. Machine Learning Approaches for IoT Identification and Classification. Falk and Fries [18] proposed different types of authentication methods as a source of device identification and whitelisting (list of authorized devices). These methods were implemented for whitelisting in industrial automation control systems (IACS). Researchers found that in the IACS environment, the devices used in this domain were engaged in a communication relation that is already known. Therefore, the whole complexity of the system can be fixed. Authors noticed that large-scale enterprise environments are dynamic where new types of devices were frequently introduced. Thus, in this case, these methods can be failed.

Meidan et al. [19] applied the random forest (a machine learning approach) to extract features from network traffic data using feature extraction techniques as explained in [25] to identify an unauthorized device from many devices based on a single TCP flow in intelligent environments. Researchers collected data from 27 different IoT devices of nine different types and manually labelled the traffic data to train and evaluate a multiclass classifier for every device type. It was shown that it correctly discovered the unauthorized ninth device type and identified the remaining eight types of devices as a particular type on the list of authorized (white list) device types. This multiclass classifier used approximately 300 features (packet level and flow level). Among them, the essential attributes are lifetime minimum (TTL), median and average packets, the ratio of total bytes transferred and received, the total number of packets with reloading tag settings (RST), and Alexa server rank. This experiment's limitations were that researchers classified devices with specific device types, but there were many device types with a single device in its category. In this way, it cannot be generalized. The second drawback of this experiment was that the devices were identified with each other, but it was not for complex mixed real-time traffic.

Sivanathan et al. [5] proposed an approach for the classification of IoT and non-IoT devices using network traffic data collected over 3 weeks. The authors applied a random forest multiclass classifier to 12 attributes extracted from network traffic such as protocols, packet length, and port number and obtained a good accuracy for classification. This method has the drawback that it must be trained for each device using network traces, and this is not a practical approach for a large number of IoT devices in the commercial market.

Pêgo and Nunes [20] developed an application to discover the properties of a new device that can be used to decide the class of a device. This application automatically creates an interface and the required integration drivers for the new device. This paper's key concern was to identify the devices interacting within a network using the data exchanged by IoT devices. Researchers found the accuracy of different ML techniques for device discovery in the IoT smart environment. This forwarded a step towards automation of IoT devices in the IoT environment and reduced traditional device integration problems for platforms that bundle possible different IoT devices in an intelligent environment. The authors collected communication data by listening to smart environment traffic. This communication data (communication files with XML format) was converted to a database with information about each device in a smart environment using an application developed in the iPhone operating system (iOS) which applied different machine learning algorithms like the Levenshtein distance algorithm, TF-IDF tables, synonyms match, and finally and multi-property matching to discover the device that communicated in the IoT network correctly.

Ferrando and Stacey [21] described the issues and challenges to secure IoT devices. The authors proposed an approach for security detection applied to data streams and classified threats in the early stages. This approach is a step towards the novelty of securing IoT devices because this technique can classify the traffic generated by sensors and determine the diverse set of network anomalies. Researchers evaluated the method as anomaly detection based on data generated from a network device because most of the anomalies in network traffic data share-related attributes. The hypothesis was that noticing the distribution of features in network traffic was acceptable as examining the distributions of diagnostic power in the form of detection and classification of large categories of anomalies.

Shen et al. [22] explained how different supervised machine learning techniques could be applied to analyze data collected by listening to intelligent environment traffic and correctly identifying unauthorized IoT devices to protect the private information of an organization. Researchers trained and evaluated a multiclass classifier on the collected and manually labeled dataset from network traffic data of twenty-seven IoT devices of nine different types. They examined that it accurately identifies the ninth type as unknown and the remaining belonged to authorized devices.

Suárez and Salcedo [23] applied different classification techniques such as K -means and ID3 on the dataset collected from twelve different devices such as cameras, lights, sensors, and fridges. They used twelve features extracted from network communication data of IoT devices such as the capacity of the battery, size of memory, internet bandwidth required, gateway, Bluetooth enabled, etc. and determined four classes of devices using ML algorithms with the help of similar features of these devices. K -mean was tested on three, four, and five clusters and grouped the devices into four categories such as mobile orchestrators, fixed orchestrators, fixed followers, and dummy followers.

Lopez-Martin et al. [28] used deep learning approaches to classify the application layer protocols by using the features extracted from the data of packets captured at layer 3. Researchers tested the classification with many different sets of features, including both ports in some sets, win size, and payload size. In this paper, it has been shown the possibility of using the traffic rate to classify and identify information from network traffic.

Miettinen et al. [24] proposed a system capable of identifying the types of IoT devices automatically connected in the IoT smart environment. The authors used fingerprint classification that enabled enforcement of protocols and constraints to overcome damage as a result of unauthorized access to the network. The proposed system imposes some filtering traffic rules in the network to protect devices communicating in the smart environment due to threats originating from other highly risky devices in the network. The designed method was attempted by researchers to separate traffic for the IoT devices which were already seen in the network. This method is impractical because many IoT devices are being released every year. They have not used the method for mixed traffic generated from non-IoT devices. This method provides a way to generate and collect data.

Cvitić et al. [25] proposed a novel technique for the detection of distributed denial of services DDoS traffic generated by IoT devices, and this approach worked as a conceptual network model for anomaly detection. This model was based on the device classes and respective classes are totally dependent on the traffic generated by these devices separately.

In the last few years, different researchers worked on IoT device identification based on port information and MAC address. Nmap is an open-source tool that has robust functionality used to detect 2600 different versions of operating systems, but it is very difficult to guess the IoT device based on port information when IoT devices use HTTP or HTTPS ports as communication sources. Therefore, there must be a robust framework to identify and classify IoT devices and their categories based on traffic patterns generated from different devices in heterogeneous networks [26].

In this work, we have presented a framework that can discriminate between IoT devices from non-IoT devices and identify the type of IoT devices with the required accuracy according to a given traffic session or sequence of sessions.

3. Methodology

This research employs deep learning with a collection of different algorithms such as DBN, convolutional neural network, and DNN, inspired by the brain's functionality and structure. We have proposed a robust framework for IoT device identification and classification based on hierarchical deep neural networks using the Keras framework [27]. A type of artificial neural network with one input layer for input variables and one hidden layer and one output layer is known as a shallow neural network [28]. DNN is similar to the shallow neural network, but there is more than one hidden layer of neurons that process the inputs. All neurodes

of hidden layers are fully connected to all neurodes of input layers. Similarly, all neurodes of output layers are fully connected to all neurodes of previously hidden layers. Hidden layers are normally used for feature extraction or feature selection from features fed in the input layer. Neurodes in hidden layers act as feature detectors, and the number of hidden layers is increased; then hidden layers will be more optimal and more important features to the output layer for identification and Classification.

In ANN, there are three layers which are as follows:

- (i) Input layer (all inputs (features) provided to the model through this layer)
- (ii) Hidden layer (maybe more than one depending upon the problem and used for processing the inputs received from the input layer)
- (iii) Output layer (for prediction)

The input layer is used for communication with the external environment that provides the pattern to the neural network. This layer works with independent variables, and the neurons in this layer take decisions and fed them to the next hidden layer. The input layer must show the situation for which we have been training the neural network. Every neuron in the input layer represents independent variables with influence on the target variable in a neural network.

This hidden layer has a collection of neurons with different activation functions that can be applied to it, which can be found in between the input and output layers. It deals with the input layer's processed input, and its responsibility is just to extract the required features from the input data. There can be more than one hidden layer in DNN. The model's accuracy can be increased by increasing the number of neurons in the network and additional layers are useful up to a limit of 9-10. Accuracy may be constant or may be decreased as their predictive power can be declined. However, 3 to 10, mostly hidden layers, are being used nowadays. The number of neurons should be considered in each network, as the number of neurons depends on the problems' complexity. If there are unnecessary neurons in the network, then the model will lead to overfitting. If there are few neurons in the network, then these few neurons adequately detect the signal in the complex dataset.

The machine learning and deep learning models contain two types of parameters (hyperparameters and model parameters). Model parameters indicate how the input data can be used to get desired output by learning at training time whereas hyperparameters tell how our model can be defined at the start of training like how many hidden layers can be used in ANN. These hyperparameters can be decided as a judgment of an expert and can be changed concerning time for optimization, and similarly, model parameters like weights can be updated during backpropagation network for strong relationship or better accuracy.

The output layer receives the input from the hidden layer and executes it for identification and classification. It will check the predicted output with actual outputs, and if the difference between predicted and actual output is very

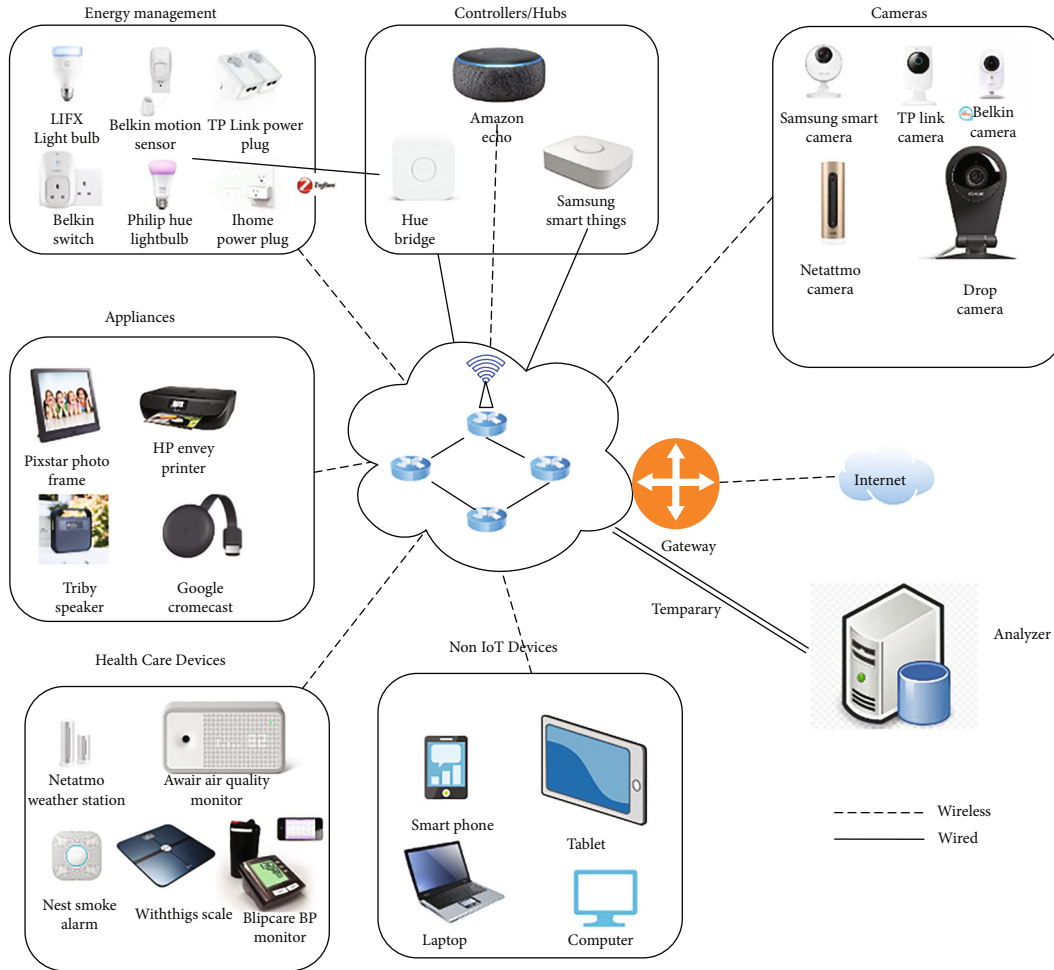


FIGURE 3: Testbed architecture for IoT and non-IoT devices.

high, then this layer traces the information back to the input layer and adjusts the weights by using a backpropagation network for every epoch. The number of neurons in the output layer must be according to the problem that is to be processed.

In this part of the paper, we have covered how pcap files (Wireshark files) are converted into executable (CSV) format. The dataset has been collected at the request of the University of New South Wales, Sydney (UNWS), Australia, as self-generated data in 28 pcap files collected from the testbed as a smart environment that has a number of IoT devices and non-IoT devices. The same dataset was employed by Bai et al. [13]. In the testbed, they used 28 IoT and non-IoT devices for communication, and their data were recorded under a synthesized network traffic trace for 6 months period.

Figure 3 shows the devices IoT and non-IoT used in the testbed.

As a first step, we converted the pcap files into an executable format (CSV format). For that purpose, the pcap files were transformed into CSV files using Python queries with 83 generalized features followed by data labeling for IoT device classification in a heterogeneous network. To label

our dataset with the specific device, we have used SQL server management studio. The dataset contains information about the network traffic stream of the devices that were used in the testbed for the collection of the dataset in a smart environment.

Using the large range of feature values in the data might lead to less accurate results and problems with the training. Hence, we have decided to use the built-in MinMaxScaler of the sklearn library in Python. This scaler can be used to perform min-max scaling, which will lead to the state that every value in the dataset is in the range (0, 1). We noticed that after performing the feature scaling, the test set results were better because the accuracy can be increased after the scaling of features.

Feature selection is one of the most important parts considered in IoT traffic for machine learning algorithms because this technique gives the most important and relevant features for target variables. Hence, the accuracy of the model can be increased. Feature selection also helps to reduce “the curse of dimensionality” that is well-known and might cause the model to overfit or perform poorly. Different machine learning algorithms do not often require feature selection like decision tree (DT) and random forest

TABLE 1: Optimal features after recursive feature elimination.

Features	Description
Source MAC	MAC address of the source
Source_port	Port number of source
Dest_MAC	MAC address of destination
Dest port	Destination port number
Flow ID	ID of network flow
Protocol	Protocols for communication (6 and 17)
Tot forw pack	Total forward packets
Flow duration	Total duration of network flow
Tot Back pack	Total number of packets backward
TotLen Bwd_Pkts	Length of all backward packets
TotLen Fwd_Pkts	Length of all forward packets
Forward_Pkt Len max	Max length of a forward packet from forward packets
Forward_Pkt Len min	Min length of a forward packet from forward packets
Forward_Pkt Len mean	Mean of forward packets
Forward_Pkt Len Std	STD of forward packets length
Backword_Pkt Len max	Greater length of a backward packet from all backward packets
Backward_Pkt Len_mean	Mean(average)length of backward packets
Backward_Pkt Len_Std	STD of backward packets length
Flow_Byts/s	Traffic flow in bytes/second
Flow_Pkts/s	Packets flow/second
Forward_Header Len	Length of header of forward packets
Backward_Header Len	Length of header of backward packets

(RF). The reason is that the feature selection process is being done on the fly due to the way these models are being trained (the “best” feature is selected at each split of the tree). However, some models may need feature selection to be performed to reach better results. In this work, we have used a hierarchical deep neural network, so we have to perform feature selection for that purpose. We have used different feature selection methods, but the recursive feature elimination (RFE) selected important and optimal features which we have used in our training dataset for the proposed framework. The features gave better results instead of other features.

Table 1 shows the most important and optimal features.

3.1. Experimental Evaluation

3.1.1. Identification and Classification of Devices. To identify and classify IoT devices for security analysis in heterogeneous networks, we have proposed an end-to-end robust framework based on hierarchical deep neural networks. Our proposed method is actually in two stages. In the first stage, we have used a deep neural network with one input layer, 4 hidden layers, and one output layer, all layers are fully connected, and it is used to distinguish between IoT devices from non-IoT devices such as laptops, MacBook, and Samsung Galaxy Tab. In the second stage, we have used a second deep neural network with 4 hidden layers with a different number of neurons, and it is used to classify IoT devices into their respective classes.

The mathematical model of the proposed deep neural network is given below [29]:

$$(x)^{(j)} = f \left(b + \sum_{i=1}^n x_i^{(j-1)} \times w_i \right), \quad (1)$$

$$P_t = \text{SoftMax} \left(x^{(j)} \right), \quad (2)$$

where b is bias, x is a vector that has input to the input layer, w is weights, n is the number of previous layer inputs, j is the number of hidden layers, P is a vector which has probability after SoftMax, and t represents the number of values in the P vector, if $P_1 > P_2$.

We find the total input for each neuron of the hidden layer and squash the total net input using the sigmoid activation function or logistic activation function. For the hidden layer, we have used the logistic activation function, and this same process will be repeated for all neurons of hidden layers and the output layer neurons. We can simplify Equation (1) as follows:

$$\text{net}_{h1} = \sum_{i=1}^n w_i \times x_i + b. \quad (3)$$

After the net input of each hidden layer, the neuron then squashes this net input using the logistic activation function to find the output of each neuron of the hidden layer, this

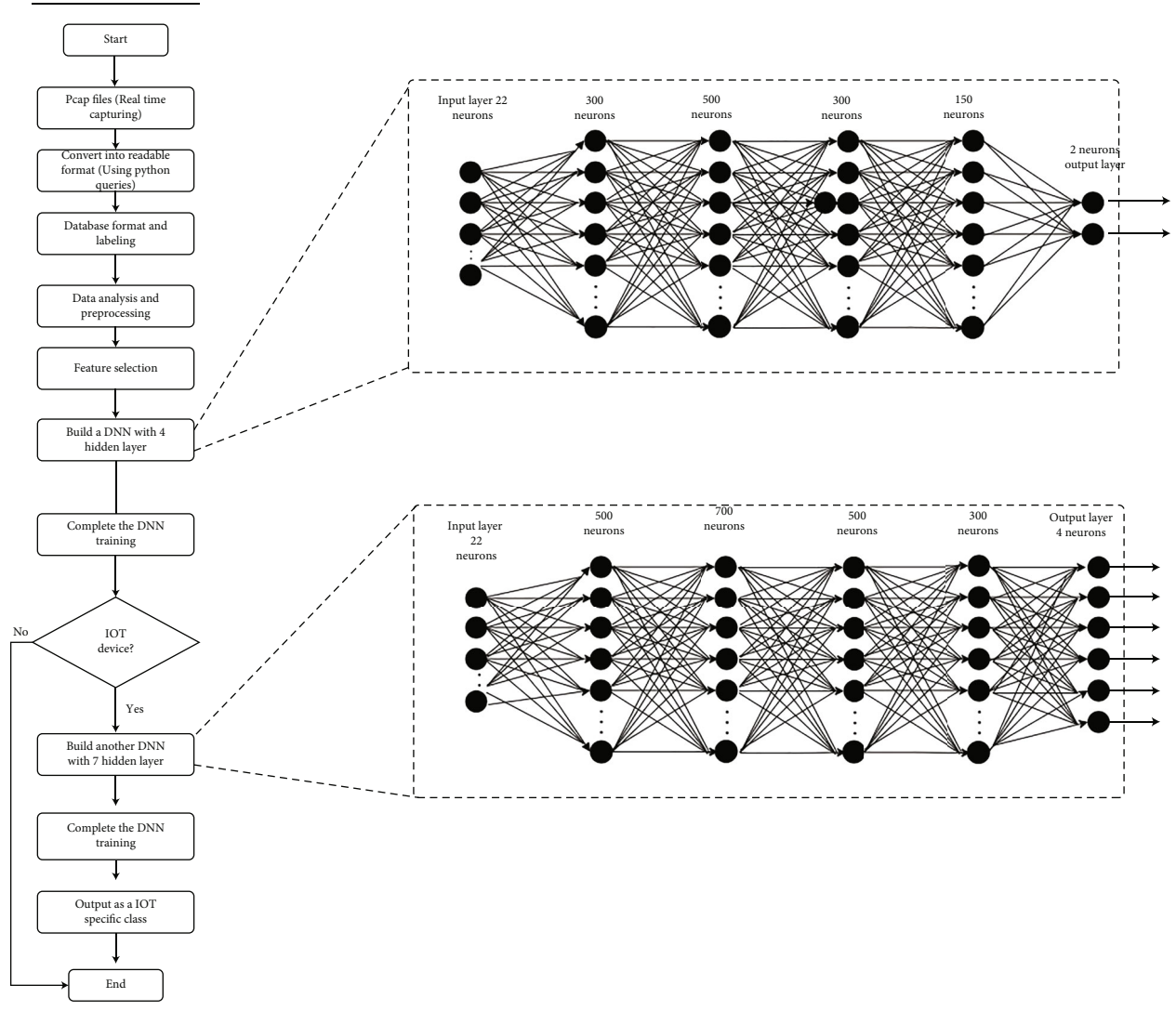


FIGURE 4: Proposed methodology with two DNNs.

same process can be used for all neurons of the output layer as well, and outputs of hidden layer neurons become the inputs of the output layer and at the output layer; we have used sigmoid function which can map value between 0 and 1.

$$out_{h1} = \frac{1}{1 + e^{-net_{h1}}}. \quad (4)$$

The error can be computed for each output neuron by using the squared error function and summed to get the total error:

$$E_{Total} = \sum \frac{1}{2} (target - actual)^2. \quad (5)$$

After getting the target value from the output of the output layer, we can calculate the error to adjust weights by using a backpropagation network.

Figure 4 shows the proposed methodology with two deep neural networks for IoT device classification.

The second deep neural network is also expressed as

$$(x)^{j1} = f \left(b1 + \sum_{i=1}^n x_i^{(j1-1)} \times w_i \right), \quad (6)$$

$$P_{t1} = SoftMax(x^{(j1)}),$$

where $b1$ is bias, x is a vector that has input to the input layer, w is weights, n is the number of previous layer inputs, j is number of hidden layers, P is a vector which has probability after SoftMax, and $t1$ represents the number of values in the P vector.

We find the total input for each neuron of hidden layer and squash the total net input using the sigmoid activation function or logistic activation function. For the hidden layer, we have used the logistic activation function, and this same

TABLE 2: List of devices with labelled data.

Category	Device name	Label	Connection type
Controllors and hubs	Samsung smart things	1	Wired connection
	Amazon Echo		Wireless connection
	Netatmo camera		Wireless connection
Cameras	Belkin camera	2	Wireless
	Samsung smart camera		Wireless
	Drop camera		Wireless
	TP-Link camera		Wireless
Switches and triggers	iHome power plug	3	Wireless
	Philip hue light bulb		Wireless
	Belkin switch		Wireless
	TP-link power plug		Wireless
Healthcare devices	Netatmo weather station	4	Wireless
	Awair air quality monitor		Wireless
	Nest smoke alarm		Wireless
	With things scale		Wireless
Electronics	Google Chromecast	4	Wireless
	HP envy printer		Wireless
Router	Tribby speaker	5	Wireless
	Bridge LAN (gateway) TP-Link router		6

TABLE 3: Nonoptimal hyperparameter list.

<i>First deep neural network</i>	
Input layer with tan h activation function	22 neurons
Total hidden layers	3 hidden layers
First unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	200 neurons
Second unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	300 neurons
Third first unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	100 neurons
Output layer with SoftMax activation function	2 neurons
Learning rate (LR)	0.01
Decay, momentum	$1e - 6, 0.9$
Loss, optimizer	mean_squared_error, sgd
Epochs	99
Batch_size	35
<i>Second deep neural network</i>	
Input layer with tan h activation function	22 neurons
Total hidden layers	3 hidden layers
First unseen layer using a best activation function, Rectified Linear Unit (ReLU)	100 neurons
Second unseen (hidden) layer using a best activation function, Rectified Linear Unit (ReLU)	200 neurons
Third first unseen (hidden) layer using a best activation function, Rectified Linear Unit (ReLU)	100 neurons
Output layer with SoftMax activation function	6 neurons
Learning rate (LR)	0.01
Decay, momentum	$1e - 6, 0.9$
Loss, optimizer	categorical_crossentropy, sgd
Epochs	99
Batch_size	35

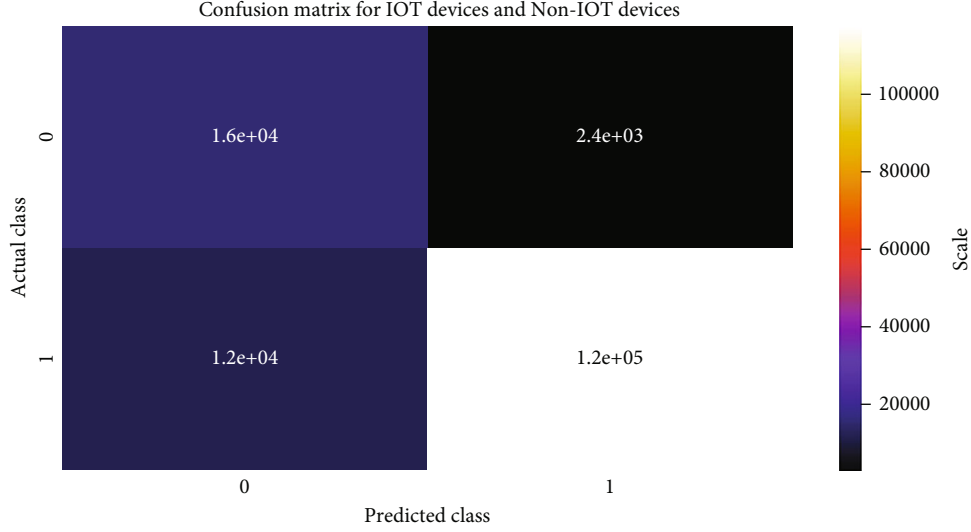


FIGURE 5: Confusion matrix of IoT vs. non-IoT devices.

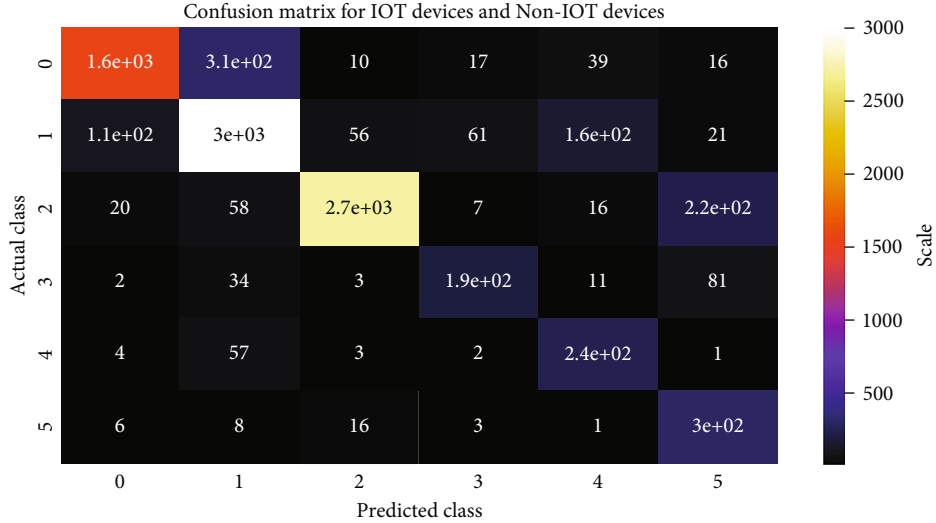


FIGURE 6: Confusion matrix for the 2nd phase of the framework.

process will be repeated for all neurons of hidden layers and the output layer neurons. We can simplify Equation (1) as follows:

$$net_{h1} = \sum_{i=1}^n w_i \times x_i + b. \quad (7)$$

After the net input of each hidden layer, neuron squashed this net input using the logistic activation function to find the output of each neuron of the hidden layer, this same process can be used for all neurons of the output layer as well, and the outputs of hidden layer neurons become the inputs of the output layer; and at the output layer, we have used sigmoid function which can map value between 0 and 1.

$$out_{h1} = \frac{1}{1 + e^{-net_{h1}}}. \quad (8)$$

The error can be computed for each output neuron by using the squared error function and summed to get the total error [30]:

$$E_{Total} = \sum \frac{1}{2} (target - actual)^2. \quad (9)$$

After getting the target value from the output of the output layer, we can calculate the error to adjust weights by using a backpropagation network.

Table 2 shows the devices with their specific categories and labels that we have used in our proposed method. This figure has not used the light bulb category in our proposed model because there is only one device in this category. Therefore, we have used only 6 categories for IoT devices labeled 1-6.

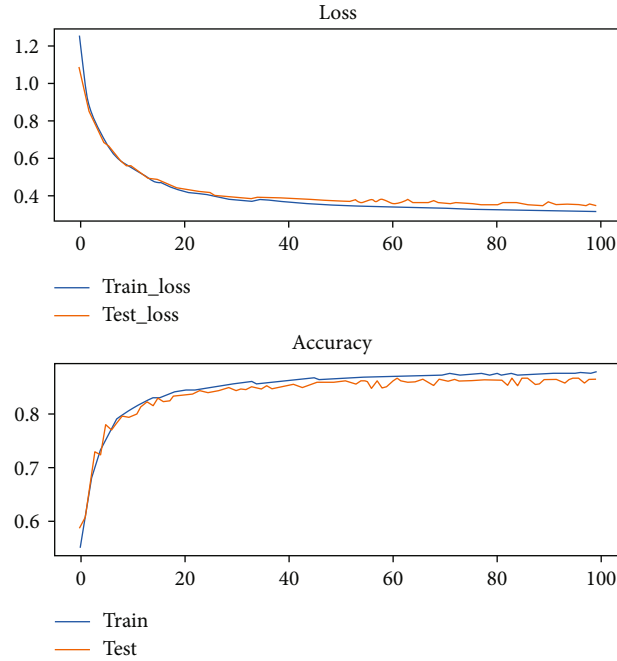


FIGURE 7: Accuracy and loss of the proposed framework for 99 epochs.

TABLE 4: Optimal hyperparameters list values.

<i>First deep neural network</i>	
Input layer with tan h activation function	22 neurons
Total hidden layers	4 hidden layers
First unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	300 neurons
Second unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	500 neurons
Third first unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	150 neurons
Fourth unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	300 neurons
Output layer with SoftMax activation function	2 neurons
Learning rate (LR)	0.0001
Decay, momentum	$1e-6$, 0.9
Loss, optimizer	mean_squared_error, sgd
Epochs	3800
Batch_size	15
<i>Second deep neural network</i>	
Input layer with tan h activation function	22 neurons
Total hidden layers	4 hidden layers
First unseen layer using the best activation function, Rectified Linear Unit (ReLU)	300 neurons
Second unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	500 neurons
Third first unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	700 neurons
Fourth unseen (hidden) layer using the best activation function, Rectified Linear Unit (ReLU)	300 neurons
Output layer with SoftMax activation function	6 neurons
Learning rate (LR)	0.00001
Decay, momentum	$1e-6$, 0.9
Loss, optimizer	categorical_crossentropy, sgd
Epochs	3800
Batch_size	30

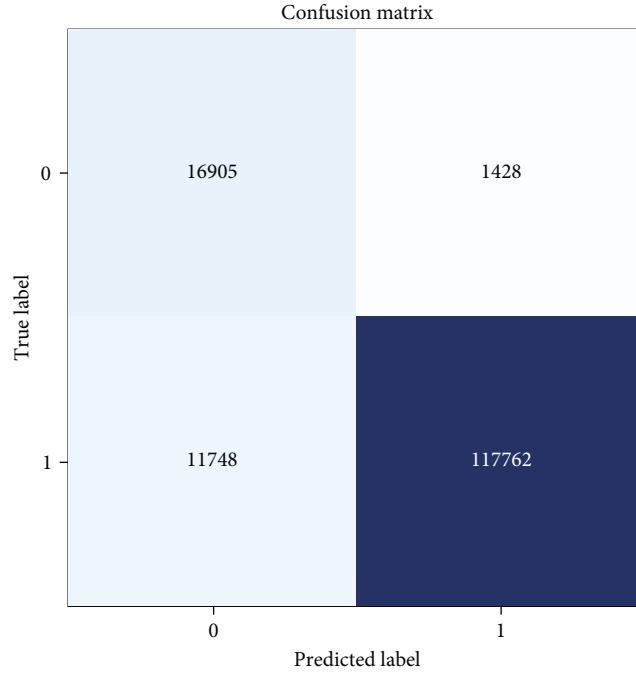


FIGURE 8: Confusion matrix for IoT and non-IoT devices.

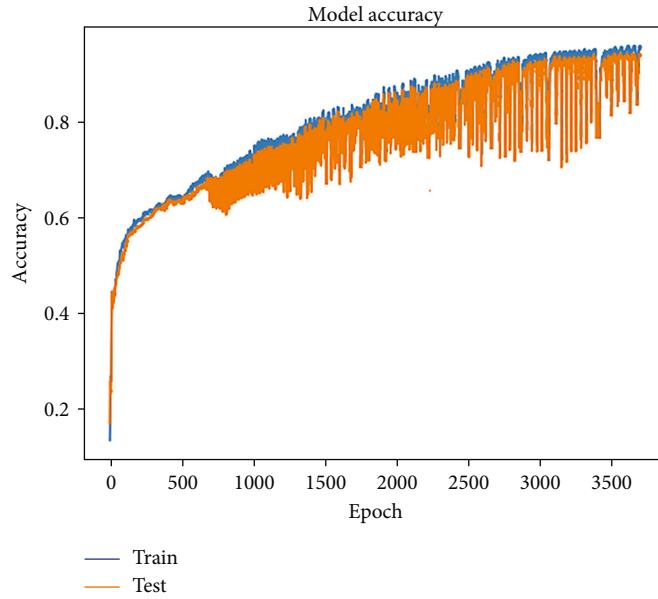


FIGURE 9: Training and testing accuracy of the first-stage DNN.

4. Performance Evaluation

4.1. *Data Construction and Modeling.* Data construction and modeling are as follows:

- (i) A total of 936893 samples were selected from the data pcap files for applying the hierarchical DNN technique
- (ii) 70% of training data for training the models and 30% of data for testing the model

- (iii) We also used cross-validation on the training dataset with 10-folds
- (iv) The hierarchical DNN technique is applied, and the respective accuracies are mentioned in the results

4.2. *Hyperparameter Setting for the Proposed Framework.* The models’ parameters are used to describe a way of converting the input data into the model’s desired output. Hyperparameters instead of model parameters are used to determine the structure of the model in use. The outcome

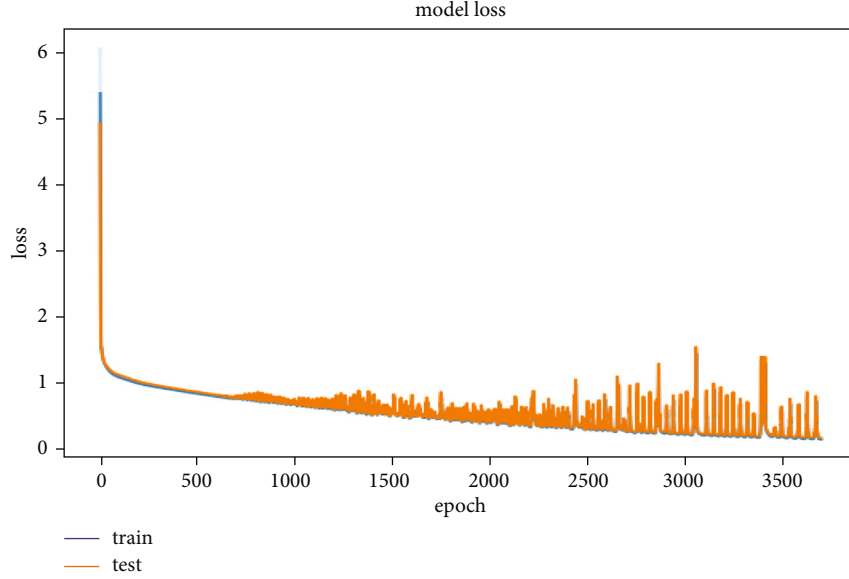


FIGURE 10: Loss of model with 0.2 of the first-stage DNN.

of the model can be changed by changing the values of hyperparameters. Therefore, the choice of hyperparameters is a crucial task and plays an important role. Keeping hyperparameters in mind is half part of the solution. The second part of the solution is knowing what kind of hyperparameter will be best for the model. In the proposed framework, we have set different hyperparameters according to requirements. Table 3 shows the values of nonoptimal hyperparameters that we have used in hierarchical deep neural networks but not getting high performance. Figures 5 and 6 represent the confusion matrix of the first- and second-stage hierarchical deep neural networks with nonoptimal hyperparameters. Figure 7 represents the accuracy and loss of the proposed framework for 99 epochs for nonoptimal hyperparameters.

Table 4 shows the optimal hyperparameters and achieved the required performance of the system.

In the proposed framework, we have applied the hierarchical DNN technique, and the respective performance measures are mentioned in the results.

Figure 8 describes the confusion matrix of the first-stage deep neural network with 4 hidden layers with a different number of neurons for optimal hyperparameters. In this figure, along the y -axis, the actual values are presented, and along the x -axis, predicted values are presented in which 0 shows non-IoT devices and 1 shows IoT devices.

The accuracy of the model can be measured as follows:

$$\begin{aligned}
 \text{Accuracy} &= \frac{\text{All right Diagnols}}{(\text{Total Number of samples})} \\
 &= \frac{117762 + 16903}{147,841} \\
 &= 0.9089(\text{with unseen data}),
 \end{aligned}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{Total predicted true}},$$

	0	1	2	3	4	5
0	1666	207	14	13	39	16
1	109	3014	42	61	159	21
2	20	58	2701	1	12	228
3	2	34	3	201	0	81
4	14	47	5	0	245	1
5	6	8	16	3	1	596

FIGURE 11: Confusion matrix of the second-stage DNN for IoT device classification.

TABLE 5: Classification report.

Label	Support	Precision	Recall	F1-score
0	1817	0.85	0.92	0.88
1	3368	0.88	0.89	0.89
2	2781	0.89	0.97	0.93
3	279	0.63	0.72	0.67
4	456	0.79	0.54	0.64
5	943	0.95	0.63	0.76

$$\text{Recall} = \frac{\text{True Positive}}{\text{Total Actual true}}. \tag{10}$$

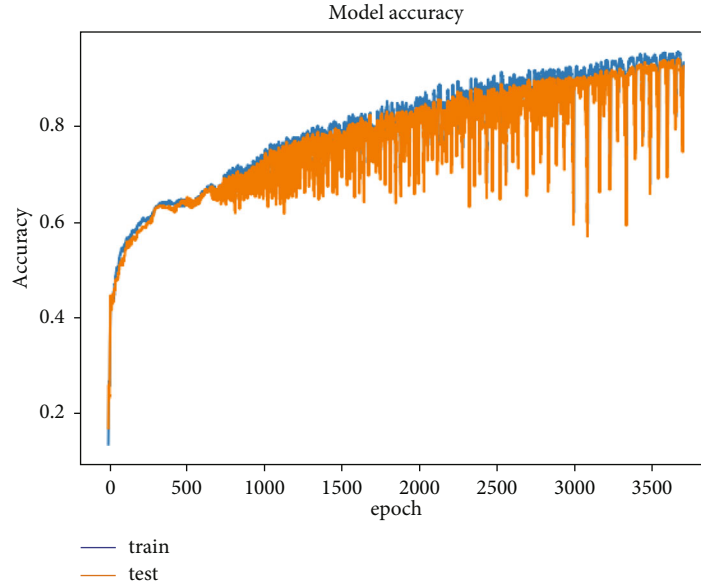


FIGURE 12: Training and testing accuracy of the second-stage DNN for IoT device classification.

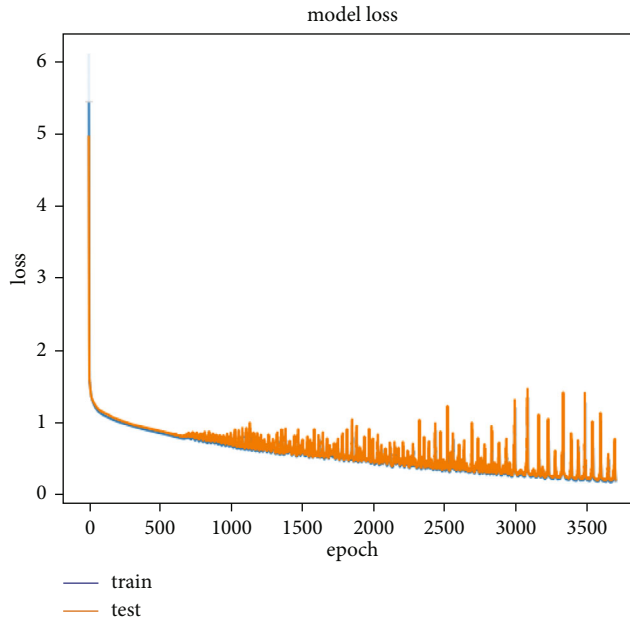


FIGURE 13: Training and testing loss of the second-stage DNN.

Figure 9 shows the accuracy (0.9120) curve for training as well as for testing the accuracy of the first-stage deep neural network (DNN) used to discernate IoT devices and non-IoT devices. Figure 10 shows the loss curve for training as well as for testing the loss of the first-stage deep neural network (DNN) used to distinguish between IoT devices from non-IoT devices.

Figure 11 shows the confusion matrix of the second-stage deep neural network used for IoT device classification using the heterogeneous network dataset.

Figure 11 describes the confusion matrix of the second-stage deep neural network with 4 hidden layers with a different number of neurons for optimal hyperparameters. In this

graph, along with the y-axis, the actual values are presented, and along the x-axis, the predicted values are presented within the range 0-5. Every value from 0 to 5 represents a particular class. The accuracy of the model can be measured as follows:

$$\begin{aligned}
 \text{Accuracy} &= \frac{(\text{All Right Dignals})}{(\text{Total Number of samples})} \\
 &= \frac{(1666 + 3014 + 2701 + 201 + 245 + 696)}{9288} \\
 &= 0.917958 \text{ (On unseen data),}
 \end{aligned}$$

TABLE 6: Comparison matrix.

Technique	Dataset	Result
Random forest (identification of devices)	UNSW	82.34%
Decision tree (identification of devices)	UNSW	79.88%
CNN-LSTM (categories [13])	UNSW	74.5%
Proposed framework (HDNN)	UNSW	91.30% (average)

$$Precision = \frac{(True\ Positive)}{(Total\ predicted\ true)},$$

$$Recall = \frac{(True\ Positive)}{(Total\ Actual\ true)}. \quad (11)$$

Table 5 shows a classification report of the model, which has different types of performance parameters. Precision can be calculated by dividing the true positive by the total number of predicted true as given in the equation. Similarly, recall can be calculated by dividing the true positives by the total number of actual trues as shown in the equation.

Figure 12 shows the accuracy curve for training and testing accuracy of the second-stage deep neural network (DNN) used to classify IoT devices with a validation accuracy of 0.9179. The y -axis represents the accuracy of the proposed model, and the x -axis represents the number of epochs. We trained our model for 3800 epochs. The blue curve shows the training accuracy, and the orange line shows the accuracy of the test dataset, which is clear that there is no overfitting, and the performance of the model is better. Figure 13 shows the loss curve for training as well as for testing of the second-stage deep neural network (DNN) used to classify IoT devices into their respective class using traffic stream.

Table 6 shows a performance metric on existing techniques that we have applied and our proposed framework.

Previous systems were able to identify and classify IoT devices with a very small range of the dataset, and these were not end-to-end systems for a smart or intelligent environment. But the proposed HDNN is an end-to-end system that is used to discriminate IoT devices from non-IoT devices and, at the same time, classify IoT devices into their respective categories with reasonable accuracy, as shown in the proposed framework diagram. According to the comparison matrix, it is clear that the proposed system gave better performance than the previous ones.

5. Conclusion

With the emergence of the Internet of Things (IoT), a significant number of IoT devices are built in various areas, such as businesses, households, warehouses, and highways. The appropriate security of IoT devices is crucial since the state of different IoT devices has other properties. The literature survey demonstrated an excellent number of cited works on IoT device identification and classification. Still, most of the work applied to small enterprise networks is based on static information such as port information, MAC addresses, and model train test for devices.

The proposed framework based on hierarchical deep neural networks (DNNs) is used to discriminate IoT and non-IoT devices and classify IoT devices to their specific category. It has been shown that the proposed method is capable of an end-to-end system to distinguish IoT and non-IoT with 91% accuracy, besides classifying IoT devices to the respective classes with an accuracy of 91.33% in heterogeneous networks. According to the comparison matrix as shown in Table 6, it has been clear that already-proposed models were machine learning algorithms like random forest and decision tree for the classification of IoT devices in the environment in which only IoT devices were present.

However, proposed framework in this research can identify IoT devices and non-IoT devices and classify legitimate IoT devices into their specific classes with approximately 91% accuracy as an end-to-end system in a smart environment. This helps formulate administrative policies and regulate/police traffic in the network for better QoS management. In a future study, we aim to examine a broader range of IoT device types and non-IoT devices for building an intelligent environment, explore new communication technologies and as well as new deep learning techniques, and experiment with data from IoT devices compromised with spyware and cyber espionage and detection of unauthorized devices for security purpose.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors are grateful to the Taif University Researchers Supporting Project (number TURSP-2020/36), Taif University, Taif, Saudi Arabia. This research work was also partially supported by the Faculty of Computer Science and Information Technology, University of Malaya, under Postgraduate Research Grant PG035-2016A.

References

- [1] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in IoT networks: a survey," *Journal of Network and Computer Applications*, vol. 154, article 102538, 2020.

- [2] Z. Guan, J. Li, L. Wu, Y. Zhang, J. Wu, and X. Du, "Achieving efficient and secure data acquisition for cloud-supported internet of things in smart grid," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1934–1944, 2017.
- [3] D. Yu, L. Zhang, Y. Chen, Y. Ma, and J. Chen, "Large-scale IoT devices firmware identification based on weak password," *IEEE Access*, vol. 8, pp. 7981–7992, 2020.
- [4] M. Jindal, J. Gupta, and B. Bhushan, "Machine learning methods for IoT and their Future Applications," in *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pp. 430–434, Greater Noida, India, 2019.
- [5] A. Sivanathan, D. Sherratt, H. H. Gharakheili et al., "Characterizing and classifying IoT traffic in smart cities and campuses," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPs)*, pp. 559–564, Atlanta, GA, USA, 2017.
- [6] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.
- [7] B. Lam and C. Larose, *How did the internet of things allow the latest attack on the internet?*, Ed, 2016.
- [8] A. Alkhalil and R. A. Ramadan, "IoT data provenance implementation challenges," *Procedia Computer Science*, vol. 109, pp. 1134–1139, 2017.
- [9] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.
- [10] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, and M. Guizani, "A survey of machine and deep learning methods for Internet of Things (IoT) security," 2018, <http://arxiv.org/abs/1807.11023>.
- [11] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices," *IEEE Internet of Things Journal*, vol. 6, pp. 8182–8201, 2019.
- [12] M. Chiang and T. Zhang, "Fog and IoT: an overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.
- [13] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang, "Automatic device classification from network traffic streams of internet of things," in *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*, Chicago, IL, USA, 2018.
- [14] P. Bajpai, A. K. Sood, and R. J. Enbody, "The art of mapping IoT devices in networks," *Network Security*, vol. 2018, no. 4, pp. 8–15, 2018.
- [15] A. Sivanathan, H. H. Gharakheili, F. Loi et al., "Classifying IoT devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2019.
- [16] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Managing IoT cyber-security using programmable telemetry and machine learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 60–74, 2020.
- [17] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18042–18050, 2017.
- [18] R. Falk and S. Fries, "Using managed certificate whitelisting as a basis for internet of things security in industrial automation applications," *International Journal on Advances in Security*, vol. 8, no. 1 & 2, p. 2015, 2015.
- [19] Y. Meidan, M. Bohadana, A. Shabtai et al., "Detection of unauthorized IoT devices using machine learning techniques," 2017, <http://arxiv.org/abs/1709.04647>.
- [20] P. R. J. Pêgo and L. Nunes, "Automatic discovery and classifications of IoT devices," in *2017 12th Iberian Conference on Information Systems and Technologies (CISTI)*, Lisbon, Portugal, 2017.
- [21] R. Ferrando and P. Stacey, "Classification of device behaviour in internet of things infrastructures," in *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, New York, 2017.
- [22] J. Shen, Y. Li, B. Li, H. Chen, and J. Li, "IoT eye an efficient system for dynamic IoT devices auto-discovery on organization level," in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 294–299, New York, NY, USA, 2017.
- [23] J. N. Suárez and A. Salcedo, "ID3 and k-means Based methodology for Internet of Things device classification," in *2017 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, pp. 129–133, Cuernavaca, Mexico, 2017.
- [24] M. Miettinen, S. Marchal, I. Hafeez et al., "IoT sentinel demo: automated device-type identification for security enforcement in IoT," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 2511–2514, Atlanta, GA, USA, 2017.
- [25] I. Cvitić, D. Peraković, M. Periša, and M. Botica, "Novel approach for detection of IoT generated DDoS traffic," *Wireless Networks*, vol. 27, no. 3, pp. 1573–1586, 2021.
- [26] A. S. Hsu, *Automatic Internet of Things device category identification using traffic rates*, Virginia Tech, 2019.
- [27] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, <http://arxiv.org/1511.06434>.
- [28] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "IoT type-of-traffic forecasting method based on gradient boosting neural networks," *Future Generation Computer Systems*, vol. 105, pp. 331–345, 2020.
- [29] O. Salman, I. H. Elhaji, A. Chehab, and A. Kayssi, "A machine learning based framework for IoT device identification and abnormal traffic detection," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, article e3743, 2022.
- [30] I. Cvitić, D. Peraković, B. Gupta, and K.-K. R. Choo, "Boosting-based DDoS detection in Internet of Things systems," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2109–2123, 2022.