# Iot Device Classification in Cloud Organizations Using Deep Belief Networks

**Dr. H. Azath H**
VIT Bhopal University
**Dr. M. NAGESWARA GUPTHA M**
Sri Venkateshwara College of Engineering, Bengaluru
**Dr. L. SHAKKEERA L**
VIT Bhopal University
**Dr. M.R.M. VEERA MANICKAM M.R.M**
Shri Vishnu Engineering College for Women
**B. LANITHA B**
Karpagam Academy of Higher Education
**M. MANIKANDAN M**
Sri Krishna College of Engineering and Technology
**Dr. P. T. ANITHA P.T** ( anithapt74@gmail.com )
Wollega University

**Research Article**

**Keywords:** Deep Belief Network, Device Classification, IoT devices, Cloud based Organization

**Posted Date:** October 15th, 2021

**DOI:** https://doi.org/10.21203/rs.3.rs-964911/v1

# IOT DEVICE CLASSIFICATION IN CLOUD ORGANIZATIONS USING DEEP BELIEF NETWORKS

1. H. AZATH, VIT Bhopal University, INDIA.
2. M. NAGESWARA GUPTHA, VIT Bhopal University, INDIA.
3. L. SHAKKEERA, VIT Bhopal University, INDIA.
4. M.R.M. VEERA MANICKAM, Shri Vishnu Engineering College for Women, INDIA.
5. B. LANITHA, Karpagam Academy of Higher Education, INDIA.
6. M. MANIKANDAN, Sri Krishna College of Engineering and Technology, INDIA.
7. P. T. ANITHA, WOLLEGA UNIVERSITY, ETHIOPIA. anithapt74@gmail.com (Corresponding author)

**Abstract**:

With the rapid increase in the usage of IoT devices, the cyber threats are increasing among the communication between the IoT devices. The challenges related to security surmounts with increasing number of IoT devices due to its functionality and heterogeneity. In recent times, deep learning algorithms are offered to resolve the constraints associated with detection of malicious devices among the networks. In this paper, we utilize deep belief network (DBN) to resolve the problems associated with identification, detection of anomaly IoT devices. Several features are extracted initially to find the malicious devices in the IoT device network that includes storage, computational resources and high dimensional features. These features extracted from the network traffic assists in achieving the classification of devices by DBN. The simulation is performed to test the accuracy and detection rate of the proposed deep learning classifier. The results show that the proposed method is effective in implementing the detection of malicious nodes in the network than existing methods.

**Keywords**: Deep Belief Network, Device Classification, IoT devices, Cloud based Organization.

**Funding and informed consent statement** : No Funding.

**Ethical statement** : Our research article is genuine and not submitted anywhere.

**Author Contribution** : H. Azath: Design of the work, M. Nageswara Guptha: Data Collection, L. Shakkeera: Data Analysis and Interpretation, M.R.M. Veera Manickam: Drafting the article, B.LAnitha Drafting the article, M. Manikandan: Critical revision of the article, P.T.Anitha: Critical revision of the article.

**Data availability statement** : We validated the classification efficiency of the DBM over a dataset called Dstest.

## 1. Introduction

In the internet era, the number of IoT devices is increasing. IoT is a cheap trillion device capable of communicating independently with other devices and remote servers [1-4].

Physical and digital combinations through the conventional web paved the way for the future IoT. IoT is intended to fill the void between the virtual world and the real world as a networking model. The IoT core concept is to connect the all-round artefacts such as RFID tags, mobile devices, sensors and actuators over the Internet through a wired or wireless network. It thus allows objects to communicate and increase the efficiency of the system with each other and their neighbour.

IoT has created new communication opportunities for machines and has extended the applications offered. IoT was traditionally assimilated in data sensing systems reasonably well. The balance between QoS and best effort data is now being increased in the direction of multimedia QoS data.

The network layer, also known as the transmission layer is called the mid-layer in IoT architecture. It is practically divided into the access network and core network in two sub-layers. The network layer's key functionality is to process obtained data from the stuff layer. The best way to transfer data via Internet to IoT servers, devices and applications via one of the communication networks such as Ethernet, WiFi, Long-Term Evolution (LTE) or Satellite is to provide an effective, energy-efficient path. The network access layer is used in various communication protocols to interconnect devices and applications through interfaces or gateways between heterogeneous networks. The core network will decide the optimal path. Currently, all Internet Protocol (IPv4 and IPv6 networks that take into account the interoperability requirement of the heterogeneous IoT network should be accepted as a network layer as well. In addition, devices (smartphones and vehicles), such as Device-to-Device (D2D), are used as sink nodes for end-to-end communication to optimise the IoT uplink transmission as well.

However, the proliferation of the IoT device creates a major problem, which means that operators find it difficult to identify the class of the IoT device and how it functions normally [5,6]. This is attributed to organizational asset management that is distributed in a cloud-based organization across various departments [7].

However, IoT inventory is considered time consuming, which leads to misclassification of appliances operating in a certain network at that moment, by coordinating different departments. It is also considered time consuming

[8,9,10]. The device visibility is also significant for the operator when it is ensured that the device has proper safety segments that may require QoS [11, 12] and can be quarantined when the device breaks.

The above limitations are addressed in the current model which develops a profound learning classification that classifies the IoT device from non-IoT devices accurately. The study uses statistical attributes collected from the home server, including the identity of the IoT device and network traffic on each device, with network traffic [13,14]. Most IoT-environment devices [15]-[17] regularly transmit short data bursts. Our preliminary work studies the state of a cloud-based IoT device with its network traffic vector.

In a cloud based organizational network based on the network identity it is therefore essential to understand IoT traffic. This contributes to remove unnecessary traffic and reduces the burden of IoT device classification. In order to improve classification by understanding the losses, reliability and latency involved, further work is undertaken in the IoT environment [18] [24]-[26]. The most compelling thing, however, is the detection by attack. Therefore, it would result in an appropriate classification of the nature of each device and would require few operations for infiltration [19]. Therefore, analysis of traffic patterns and analysis of network identity [20] [21] are the most important measures in a broad IoT environment [22] [23].

In this paper, the study uses deep belief network (DBN) [27], which is the combination of stacked Restricted Boltzmann Machines (RBM) [28] and labelled backpropagation (BP) [29] classifier for the classification of IoT devices. DBN is hence regarded as a deep learning method that assists the network in classifying the IoT devices based on feature extraction and classification.

The main contribution of the paper involves the following:

- The authors use deep belief network (DBN) to resolve the problems associated with identification, detection of anomaly IoT devices.
- Various extraction of features finds the malicious devices in the IoT device network that includes storage, computational resources and high dimensional features.
- The extracted features from the network traffic assists in achieving the classification of devices by DBN.
- The simulation is performed to test the accuracy and detection rate of the proposed deep learning classifier.

The outline of the paper is given below: Section 2 provides the methods. Section 3 discusses the classifier for the classification of IoT devices. Section 4 evaluates the entire work and section 5 concludes the work.

## 2. Related Works

Meidan, Y. et al. [34] used network traffic data machine learning algorithms to correctly classify network-connected IoT devices. We collect and mark network traffic data from 9 different IoT devices, as well as PCs and smartphones to train and test the classifier. We have trained a multi-stage metastatic classifier with supervised learning, and in the first step, the classifier can differentiate between IoT and non-IoT traffic. Each IoT unit is related to a particular IoT unit class in the second step.

Lee, H. et al. [35] offers an architecture of IoT-embedded cloud control. The cloud control server with the suggested frameworks overcomes mixed flow problems. The IoT device built senses multiple system status and sends signals. The knowledge is processed using deep learning analytics for the Fault Detection Classification (FDC) mechanism. The cyber-based simulation is then carried out using the supplied network sleeve model for multi-products.

Sivanathan, A. et al. [36] suggested using network traffic analytics, including standard behaviour mode, for characterising IoT devices. Initially, we gather and summarise traffic trails in an intelligent campus environment using a range of IoT devices including cameras, lighting, facilities, and health monitoring systems; our traces are released to the general public, collected for a period of three weeks. It then analyses traffic tracks to describe statistical characteristics for more than 20 IoT system used in our environment such as data rates and bursting, cycles and signalling patterns. The study establishes a classification system, using these characteristics, which not only distinguishes IoT from non-IoT, but also defines unique IoT devices.

Miettinen, M. et al. [37] implemented IoT Sentinel, a mechanism that automatically determines the types of devices connected to an IoT network, and allows rules to limit communications of vulnerable devices in order to mitigate harm caused by their compromising. They are also lacking in frameworks that help to remove security vulnerabilities in firmware updates or patches. A brownfield approach is needed when networks are protected where these vulnerable devices are present: the introduction of appropriate network protection measures to allow potentially vulnerable devices to coexist without endangering the security of other devices on the same network.

The problem has been addressed by Sivanathan, A., et al. [38] by creating a robust IoT system classification mechanism using network-level traffic characteristics. We have four times our donations. First, we use an intelligent environment that spans cameras, lights, plugs, motion sensors, computers, and medical monitoring systems using 28 different IoT devices. For a period of six months, we capture and synthesise traffic from this infrastructure and publish it for the world as open data. Secondly, we provide details on the underlying traffic networks with statistical features including operation cycles, port numbers, signage trends and cypher suites. Thirdly, we create an algorithm for multiple stage learning machines, demonstrating their ability, based on their network operation, to classify such IoT devices with over 99% accuracy. Finally, we talk about the trade-offs between cost, pace and efficiency in the real time implementation of the classification system.

Bai, L. et al. [39] proposes a new and unheard computer automated devices classification system. The process uses the comprehensive knowledge found in IoT network traffic flows to classify the characteristics of different devices. First, a set of discriminating characteristics is defined from raw network traffic and then an LSTM CNN cascade model is proposed to classify the semanticized system type automatically.

In order to classify IoT system species accurately from the white list, Meidan Y., et al. [40] used Random Forest (a supervised machine learning algorithm), features removed from network traffic results. We have obtained and manually labelled network data on 17 separate IoT devices representing nine types of IoT devices for the training and evaluation of multi-class classifiers. Based on the classification and use of a majority rule in 20 consecutive sessions IoT system models that were not included in the White List were correctly defined in an unknown way in 96% of test cases (on average). Some types of IoT devices are more rapid than others (e.g., sockets and thermostats were successfully detected within five TCP sessions of connecting to the network). The study carried out 110 consecutive sessions required the perfect identification of unauthorised IoT device types; 346 consecutive sessions were required in order to complete the perfect White Listed Classification; of which 110, 99,49% were correct. Further tests have shown that the classifiers trained in one location and tested in another are successful. In addition, there is a debate about the resilience of our IoT-based learning system for listing opponents.

Samie, F., et al. [41] used hierarchical classification method, which divides the problem into three classifiers into two levels of hierarchy. In the first layer a lightweight classifier runs directly on the IoT system and determines whether the computation is to be discharged to the gateway or on board. The second layer consists of a lightweight classifier on the IoT device and a complex classification on the gateway (to distinguish the remaining classes). The experimental results indicate higher accuracy (average 92%), than a nonhierarchical

classifier (using a real world data set for human activity recognition and implemented on a wearable IoT device) (87% on average). The run-time measurements and power measurements on the IoT system show 3 to save energy.

Ammar, N., et al. [42] approached to near-real time grading based on network characteristics derived both from traffic flow characteristics and from packets' payloads in order to differentiate system types. A newly connected computer to a home network is automatically detected by our solution. We also test our method's efficiency by using a representative and heterogeneous collection of real IoT devices. Based on decision tree models using the one-to-all process, our findings are independent recognition with average accuracy of 97%.

Aksoy, A., & Gunes, M. H. [43] introduced a system called System IDentifier (SysID) for automatic classification of IoT devices based on its network traffic. SysID can detect its kind from any individual packet originating from the system. Genetic algorithms (GA) are used to evaluate the related features of various protocol headers, then deploy various ML-algorithms to categorise host-types through analysis of features GA pick, including DecisionTable, J48 Decision Trees, OneR, and Component. GA helps to reduce the difficulty of classifications by removing noisy data features and increases their precision. SysID enables you to identify IoT devices completely automatically using your TCP/IP packets with no expert classification input.

## 3. Methods

The architecture of entire system is given in Figure 1. In areas which require urgent data transmission, cloud-IoT integration uses cloud clusters. It is simple, robust and depends on a single hop and low speed broadcast. The cluster formation protocol is reactively initiated by any cloud Node called CH which snoops on an IOT route data page. The protocol in Figure 1 contains three phases.

The study uses three phased models:

- Sensing plane,
- Control plane and
- Data plane.

The *sensing plane model* is the collection of multiple IoT devices clusters that collects data from different physical environments.

The *control plane model* consists of DBN model that maintains the routing path by matching the data speeds of input IoT devices.

The *data plane model* helps in routing the packets from the collected IoT device nodes for faster transmission of packets to the destination node.

In the first stage, the CH removes from the sniffed packet the gradients of the IoT node that routed them and broadcasts a two-hop limitation to request additional knots by asking them to join the new cluster.

The second stage involves sending a discovery message to IoT to get the best possible gradient between IoT nodes they can communicate; then compare this to the gradient that is declared on a Join request: in the cluster, only CLOUD knots that can communicate better with IoT nodes participate.

The VMs will receive the application for join in the second phase. In the third and last phases of entering the new cluster, cloud nodes communicate to CH. The CH collects the responses from the cluster VM nodes and chooses to exit the node. By collecting replies from all the cluster nodes, CH can estimate the number of messages exchanged by cloud and IoT: appropriate policies can determine the length and guarantee energy demand for each of them.

The proposed DBN model is used for the classification of IoT devices in an organization or industry with different IoT devices. The IoT devices includes: Temperature IoT, lightning IoT, energy usage IoTs, smoke detectors, and gesture sensors. The classifier is responsible of collection of input data from the IoT networks and devices, extraction of IoT device features and classification of devices. The DBN resolves the problems of identification of detection of anomaly IoT devices. Various extraction of features is utilized to find the malicious devices in the network that includes its storage, computational resources and high dimensional features. Such features helps in classification of devices by DBN after the extraction from RBM.

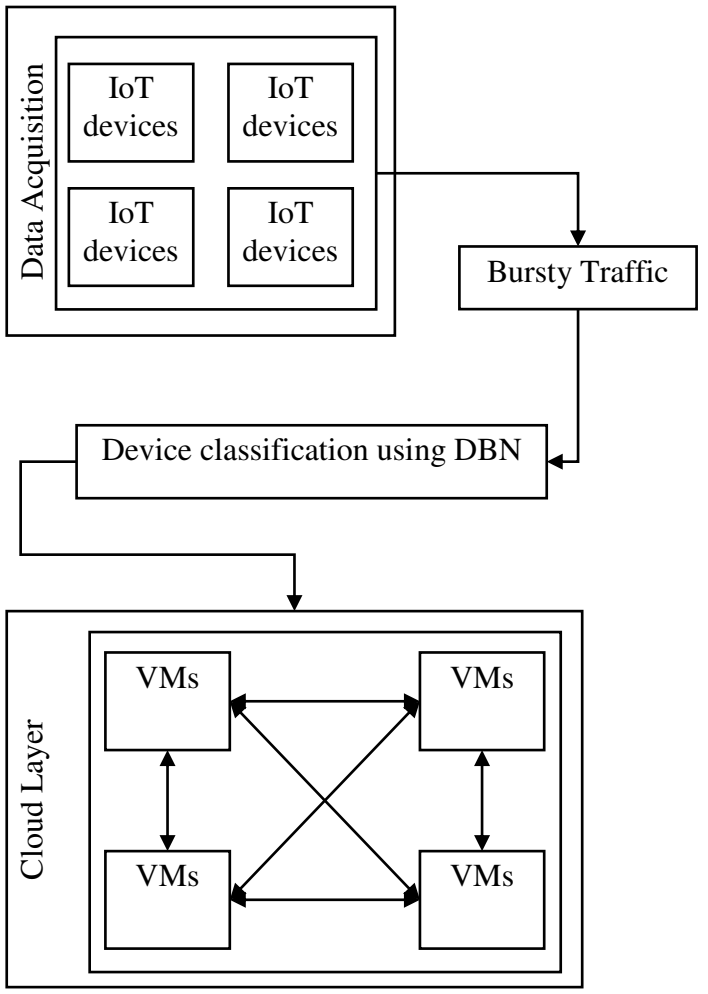


Figure 1: IoT Classification Architecture using DBN model

## 4. Deep Belief Network

DBN is regarded as a deep learning method, which is used mainly for the process of feature extraction and classification. The Figure 2 shows the structure of DBN network, which has stacked Restrict Boltzmann Machines (RBM) and further it has a classifier for IoT device classification in its deep network.

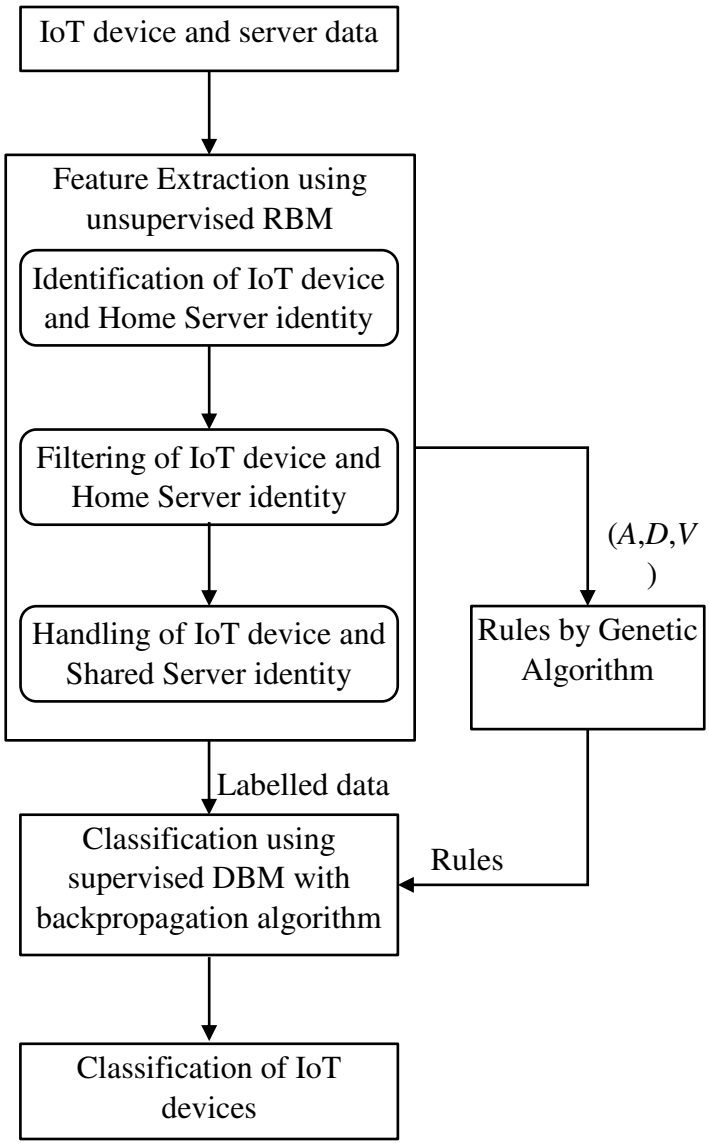


Figure 2: Architecture of RCNN for IoT device classification

The DBN is formed with the nesting of stacked RBM with a classification layer and the unsupervised RBM training with greedy algorithm improves the pattern of training. The output of the entire network is improved is by getting feedback from the past RBM output. In the final part, the supervised training performs the process of classification with its classification layer. The probabilistic graphical model in RBM uses stochastic neural network to interpret the results based on two states (output) of neurons that includes both active states and inactive states. The feature extraction model uses an unsupervised Boltzmann Machine with its two layered RBM.

### 4.1. Data Representation

The observation IoT data is represented using the visible layers $v = (v_1, v_2..., v_n)$ and feature extraction layer is the approximation of hidden layer $h = (h_1, h_2..., h_m)$. The top two layers (associative memory module) acts as a combined distribution between the output and the last hidden layer ($h$) layer.

The DBN has two phases of learning process that includes pre-training or unsupervised learning and fine-tuning or supervised learning. A contrastive divergence is an unsupervised learning that helps in training the stacked RBMs in a hierarchical manner. Backpropagation (BP) algorithm uses its supervised learning pattern to fine-tune the bias and weights in the RBM network. The unsupervised training at DBN optimizes the entire RBM to optimally extract the features from the input IoT device data.

The inputs of the classifier includes the following. The study only takes into account the device communicating in the network. The purpose of the study is to find the server domain names connecting IoT devices regularly. Furthermore, servers are shared between different types of IoT devices, to prevent replications.

*Identification of Home Server and device Identity*: The identities of home servers are booted using IoT devices and the communication made by the server is then learned and monitored. In addition, the domestic server traffic is recorded in IoT networks. The domain name will finally be removed from the home server.

*Filtering of Home Server and device Identity*: The domain identity of two different server and IoT devices are excluded, i.e. 3rd party and human server. Exclusion may prevent false positive events during the classification process. The third-party server is not a domestic server which accesses domestic and international IoT devices. The system operated on a mobile phone, or laptop, can then be misclassified as IoT devices by this server. Identifying a human-facing server is based on the content of individuals by answering the web request. By including third-party servers, various other clients interact with each other in a false positive way.

*Handling of the Shared Server and device Identity*: The server and device identity between multiple IoT devices is similar, so that the classification of devices is uncertain. Hence, classification becomes uncertain if several IoT devices share similar server identities. The similar types of devices are combined to avoid such constraints. On the other hand, the classification of IoT devices cannot be guaranteed when the name of a server shared by multiple IoT devices is partially overlapping. Therefore, IoT devices with common server identity are merged to prevent such a false classification.

### 4.2. Feature Extraction

Consider a set of $(v,h)$, where the energy function $E(v,h|\theta)$ is used to train the RBM in an unsupervised way,

$$E\left(v,h|\theta\right)=-\sum_{i=1}^{n}a_i v_i-\sum_{j=1}^{m}b_i h_i-\sum_{i=1}^{n}\sum_{j=1}^{n}v_i w_{ij} h_j \quad (1)$$

where

$\theta$ is defined as the RBM parameter $\theta=\{W_{ij}, a_i, b_j\}$ with

$w$ as the weight connection of each layer, and

$a$ is the bias of the visible hidden layer neuron and

$b$ is the bias of the hidden layer neuron.

Hence the joint probability distribution between the output and the last hidden layer ($h$) layer:

$$P\left(v,h|\theta\right)=\frac{e^{-E\left(v,h|\theta\right)}}{Z\left(\theta\right)} \quad (2)$$

$$Z\left(\theta\right)=\sum_{v,h}e^{-E\left(v,h|\theta\right)} \quad (3)$$

Using Gibbs sampling process, the conditional probability distribution of $a$ and $b$ is given by:

$$P\left(h_j=1|v,\theta\right)=sigmoid\left(b_j+\sum_i v_i w_{ij}\right) \quad (4)$$

$$P\left(v_j=1|v,\theta\right)=sigmoid\left(a_j+\sum_i h_i w_{ji}\right) \quad (5)$$

Using Eq.(3), the probability of hidden layer being in an active state is hence can be derived. The RBM tends to provide a symmetric feature over probability of the activation state of a visible layer neuron and hidden layer state $h$. The activation state probability is hence determined using the Eq.(4).

### 4.3. Classification

Decide a class label with a set of training data points, each linked to a class label, for one or more previously unknown test instances. Most classification algorithms normally consist of two phases:

*Training phase*: This phase constructs a training model from the training sessions. The group marks in the training data set can intuitively be understood as a mathematical model in summary form.

*Testing phase*: The training model in this phase will be used to determine a class label (or group identifier) of one or more uneven test instances.

It is supposed to be D with n data points and d characteristics or dimensions. Furthermore, every data point of D has a label drawn from {1 ... k}. In certain models, binary labels (k = 2) are supposed to be easy. In the latter case, it is assumed that labeling is drawn from {−1, +1}. However, the assumption that labels are drawn from {0, 1} is sometimes notations. This chapter uses one of these conventions according to the classification. A training model from D is designed to predict the invisible test cases label. A classification algorithm may be produced one of two types:

The neural network's most basic form is a perceptron model with just one layer and one output layer. As the input layers transmit attribute values only without using a math function on the inputs, the perception model can only understand this function as a straightforward linear model based on one output node.

In practice, more complex models may need to be learned from multilayer neural networks. Multi-layer neural networks have a hidden layer in addition to input and output layers. The hidden layer nodes can be connected to various topology types as a matter of principle. For example, the hidden layer may consist of several layers and nodes in one layer can be added to the next layer nodes. The feed-forward multi-layer network is known as this.

The nodes in one layer should also be completely connected to the nodes in the next layer. The topology of a multi-layer feed forward network is therefore automatically determined when the number of layers and nodes in each layer have been specified by the analyst. The fundamental viewpoint can be seen as a single layer feed forward network.

One common model is one with only one hidden layer of a multi-layer feed network. This can be considered as a two-layer feed-forward network. A network of two layers of feedback. It is also the case that the feed forward network does not limit itself to the use of linearly signed input functions. Arbitrary functions, like logistical, sigmoid or hyperbolic tangents may be applied to the various nodes in the hidden layer and output layer.

If the $X_i$ tuple is applied to give a $Z_i$ output value, this function is as follows:

$$Z_i = \sum_{j=1}^{d} w_j \frac{1}{1+e^{-x_i^j}} + b$$

In the case of a feature calculated at the hidden level nodes, a $Z_i$ value is not the predicted output of the final class label in {−1, +1} anymore. The result will then be forwarded to the next level.

The training in the single layer neural network was relatively simple because the expected output of the output node knew the label value of the training was the same. The known fact of the soil was used in order to create a minor problems of optimization by a gradient reduction method and to update weights. As the output node is the only weight neuron within a single network, it is possible to update the process easily. The problem in multi-layer networks is that the output of hidden layer nodes is unknown as they do not link outputs to training labels. Therefore, when a training example is incorrectly classified there is the way to update the weights of these nodes.

When a classification error occurs, feedback from the nodes in the forward layers to the earlier layers is clearly required. This can be achieved by the back-propagation algorithms. The back-up algorithm includes two main phases used during the weight update process for each training instance:

Forward phase: In this step, the inputs for a training instance are entered in the neural network. This leads to a further cascade of calculations across the entire layer by using the existing weight set. To verify if the forecast label is an error, the final forecast output can be compared to the class label for the training facility.

Backward phase: The main aim of the backward phase is to learn weight from subsequent error levels and thus evaluate the node output in the previous layers. The estimate of the hidden layer node error shall be calculated on the basis of error assessments and layer node weights. This is then used by an error gradient to calculate and update the weights of this node. At a conceptual level, the actual update equation is not very different from the fundamental perception. The only differences arise because the non-linear function is common in cached layer nodes and because failure is not directly calculated with the comparison of the output and the exercise label via back propagation. This whole process is reversed to update the network node weight.

The algorithm for multilayer updates is identical to the one-layer algorithm. The main difference is that it is not possible to use the weight of the hidden layer nodes. Instead, the update procedure is replaced by the aforesaid forward approach. The update process of nodes to convergence is repeated through cycling via training data in periods as in the case of the single-layer network. It may take millennia for a neural network to learn the weight by means of training data of different nodes.

A network with neural multi-layer capturing arbitrary features is stronger than the SVM kernel. A multi-layer neural network can capture non-contiguous class distributions with different decision boundaries in different data regions, in addition to identifying the decision bounds for arbitrary shapes. The different nodes within the hidden layer can logically trace the different decision boundaries within different data regions, and between the nodes in the output layer the results can be combined.

Further nodes and layers can approximate almost any function. This is broader than what can be captured by a kernel-based nonlinear decision-making limit. In this sense, neural network approaches to the universal function are considered. The cost is that several implementation problems occur in the design of neural networks:

There are generally more nodes and covert layers available, but the risk of overrun is equivalent. There is little guidance on the design of neural network topology as the process of classification of neural networks is not very well understood. Although certain hill climbing methods may be used to provide a limited extent the correct topology of the neural network, the issue of good neural red design remains open.

Neural Network is a computational model based on neural network structure and biological functions. Typically, it is best suited for non-linear data. ANN's ability to learn through training data sets is one of its most recognised advantages. These are interconnected to the first neurons, which sends data to the second layer and sends data to the third layer's output neurons, by three layers as shown in Figure 1. The flow of information is one way here.

All these links are related to a weight, which a neural network learns during training and controls the cost of the overall model. The cost will be low if the forecast has a minimum error. The test cases are separate and used in various areas such as image processing, object recognition and data classification using feed forward neural network (FFNNs).

Another type of ANN is a retrieval of neural networks that are structurally identical to FFNNs but allow connections within the same hidden layer between the neuronal networks. It is able to map all historical data to the final output by allowing historical data to be stored in the inner state of the network.

Even though one or several units fail to react to the network, the ANN works. Vast and efficient neural networks must be implemented with numerous processing and storage resources. Although the brain has hardware designed to process signaling using a neuron graph, a neural network designer can be forced to complete millions of database rows of connections using Von Neumann Technologies, which can take huge amounts of computer memory or hard disc space, by simulating even a most simplified form.

Deep learning, which is an end to the end process, is a subdivision of master learning. Machine learning algorithms require structured data and further outputs are produced by means of more data sets, and if the desired output is not obtained then human intervention is necessary. Deep Learning has one of its advantages: the capacity to learn and capture high-level characteristics in an incremental order using supplied data, eliminating the need for know-how in the field.

#### 4.3.1. Rule Set for Classification

The DBN classification layer is not a similar variant of RBM layer, since it uses a feed-forward neural network. The rules for classifying the IoT devices are fed into the DBN classification layer. The rules are specified as the IF-THEN rules that enables network being described with decentralized information. The classification rule over each class label is extracted using the following manner,

$$\begin{aligned} R_i : IF & \left( x_{s1} > V_1 \; or \; x_{s1} < V_1^{'} \right) \\ & \wedge \left( x_{s2} > V_2 \; or \; x_{s2} < V_2^{'} \right) \wedge \ldots \\ & \wedge \left( x_{sn} > V_n \; or \; x_{sn} < V_n^{'} \right) then \; y = c \end{aligned} \tag{6}$$

where

$R_j$ is rule label, $1< j<N$, $N$ is the total hidden units in the supervised BP classification layer,

($x_{s1}$, $x_{s2}$,…,$x_{sn}$) is the rule elements for the selected features,

$V$ is regarded as the learned real value using an algorithm,

$y$ is regarded as the class label $c$, which is assigned to a specific class

For effective extraction of IoT data, the genetic algorithm [33] is used to extract the rules for classification that defines the relationship between the classification rules and its respective class labels. It improves the classifier efficiency and reduces the loss in information. The extraction of classification rule using genetic algorithm using $N$ genes with its three components that includes device type ($A$), device responsiveness ($D$) and device participation in network ($V$).

The evaluation of quality rule is considered as the fitness function:

$$Fitness = \frac{\left( \frac{TP}{\left( TP + FN \right)} \right)}{\left( \frac{TN}{\left( FP + TN \right)} \right)} \tag{7}$$

Where,

TP is regarded as the True Positive;

FN is regarded as the False Negative;

TN is regarded as the True Negative;

FP is regarded as the False Positive;

$\frac{TP}{TP+FN}$ is regarded as the sensitivity of each device classification rule and

$\frac{TN}{FP+TN}$ is regarded as the specificity of each device classification rule.

## 3. Results and Discussions

In this section, we validate the classification efficiency of the DBM over a dataset called Dstest. The features including device identity, server identity and network traffic rate are considered for optimal feature extraction and classification. The classification of IoT devices takes place with a set of 300 IoT sensor nodes, where it is divided into three sets in an organizations for the purpose of simulation. The DBN is compared in terms of with and without Genetic Algorithm for rule extraction, and with other existing methods like: ANN [29], FFNN [30], BPNN [31], DNN [32], DBM and DBM-GA. The proposed method is tested against various performance metrics like accuracy, specificity, sensitivity, F-measure, execution time and mean error rate.

***Accuracy****:* Accuracy is defined as the measure of the results of classification of benign and malignant IoT device is defined as follows

$$accuracy = \frac{TN + TP}{TP + TN + FN + FP}$$

***Sensitivity****:* Sensitivity offers the testing capability on finding if the instances are correct or not, which is expressed as

$$sensitivity = \frac{No.\ of\ \ TP}{No.\ of\ \ TP + No.\ of\ \ FN}$$

**Specificity**: Specificity offers the testing capability on finding if the reliable instances without network conditions.

$$specificity = \frac{No.\ of\ \ TN}{No.\ of\ \ TN + No.\ of\ \ FP}$$

***F-Measure***: The F-measure is the measure of accuracy based on the precision and recall metrics.

$F1$ strategy = $(2PR)/(P + R)$

$P$ is the precision and $R$ is the recall

***Execution Time****:* The execution time measures the classifier run time to complete the entire process of classification.

***MAE***: The mean absolute error is the measures of the evaluation on true predicted value and final prediction results, which is expressed as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|$$

Table.1. Accuracy

| **Devices** | **ANN** | **FFNN** | **BPNN** | **DNN** | **DBM** | **DBM-GA** |
|---|---|---|---|---|---|---|
| 300 | 41.98 | 45.27 | 49.29 | 52.54 | 58.86 | 79.22 |
| 250 | 42.55 | 48.08 | 52.36 | 56.68 | 64.81 | 81.10 |
| 200 | 45.27 | 54.82 | 59.74 | 67.79 | 76.50 | 83.30 |
| 150 | 78.99 | 82.14 | 84.21 | 87.26 | 92.44 | 92.77 |
| 100 | 80.87 | 83.89 | 85.90 | 88.98 | 94.00 | 94.38 |

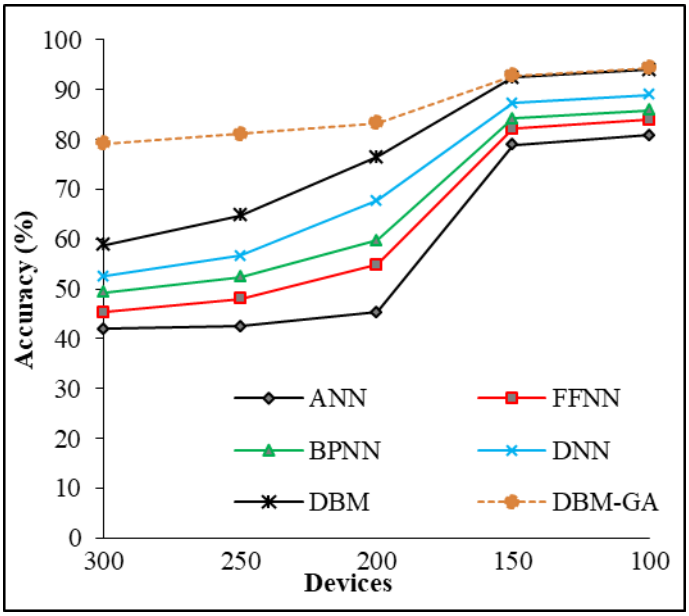


Fig.3. Accuracy

The Fig. 3 (Table 1) provides the results of classification accuracy between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has higher accuracy in classifying the instances based on the rules obtained from the genetic algorithm. The accurate classification from the ruleset provides optimal classification of devices than other existing methods. With increasing number of devices, the accuracy tends to degrade due to complexity in the task of computing the instances of more devices.

Table.2. Sensitivity

| Devices | ANN | FFNN | BPNN | DNN | DBM | DBM-GA |
|---|---|---|---|---|---|---|
| 300 | 47.85 | 53.82 | 56.32 | 60.88 | 66.36 | 66.90 |
| 250 | 50.47 | 54.24 | 58.23 | 61.58 | 68.34 | 69.38 |
| 200 | 51.67 | 55.37 | 62.64 | 67.35 | 72.54 | 79.44 |
| 150 | 54.57 | 58.92 | 63.60 | 69.19 | 83.32 | 84.91 |
| 100 | 61.85 | 59.94 | 63.89 | 77.21 | 83.51 | 87.10 |

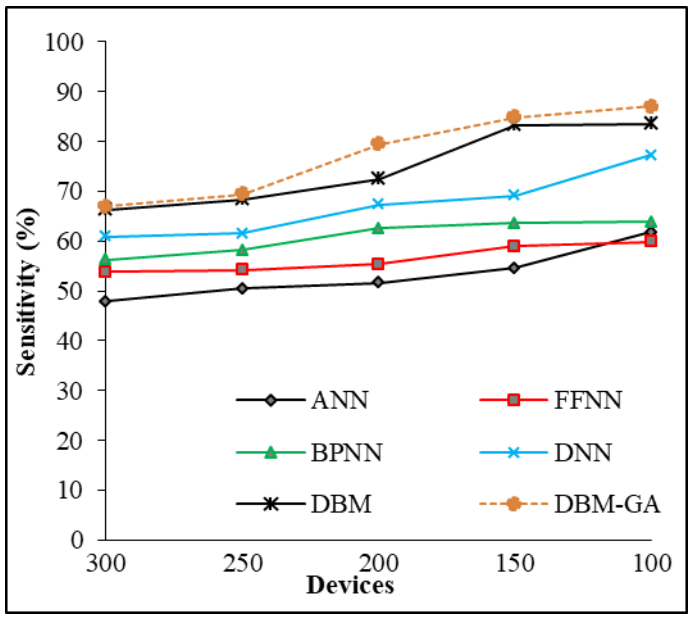


Fig 4. Sensitivity

The Fig. 4 (Table 2) provides the results of sensitivity between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has higher sensitivity in classifying the instances based on the rules obtained from the genetic algorithm. The accurate classification from the ruleset provides optimal classification of devices than other existing methods. Furthermore, the sensitivity rate increases in both the proposed methods as well as existing methods with the increasing number of features. The proposed method largely deals with the removal of insignificant characteristics that improve the sensitivity rate.

Table.3. Specificity

| Devices | ANN | FFNN | BPNN | DNN | DBM | DBM-GA |
|---|---|---|---|---|---|---|
| 300 | 56.22 | 61.04 | 66.00 | 71.47 | 77.79 | 78.87 |
| 250 | 59.11 | 63.06 | 67.80 | 73.86 | 80.33 | 80.99 |
| 200 | 59.87 | 64.89 | 68.45 | 74.06 | 80.79 | 83.09 |
| 150 | 79.72 | 82.77 | 84.81 | 88.44 | 93.48 | 93.85 |
| 100 | 80.23 | 83.38 | 85.39 | 89.00 | 94.4 | 95.07 |

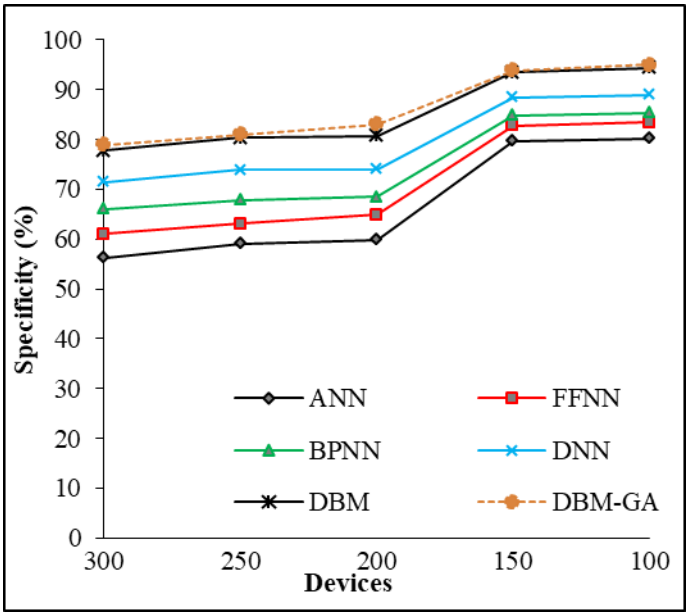


Fig 5. Specificity

The Fig. 5 (Table 3) provides the results of specificity between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has higher specificity in classifying the instances based on the rules obtained from the genetic algorithm. The accurate classification from the ruleset provides optimal classification of devices than other existing methods. Furthermore, the rate of specificity increases both in the methods proposed and in current methods with increasing numbers of features. The method proposed is mainly concerned with the elimination of insignificant characteristics which improve the specificity rate.

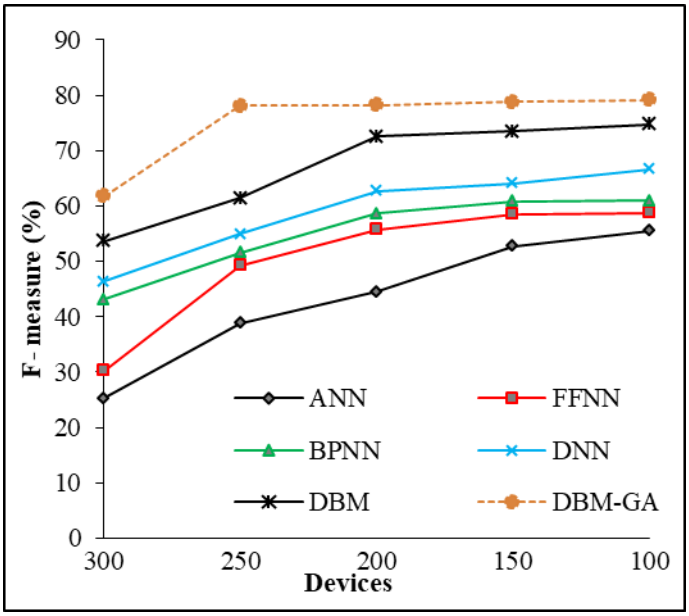


Fig 6. F-Measure

Table.4. F-measure

| Devices | ANN | FFNN | BPNN | DNN | DBM | DBM-GA |
|---|---|---|---|---|---|---|
| 300 | 25.28 | 30.31 | 43.17 | 46.33 | 53.62 | 61.81 |
| 250 | 38.79 | 49.21 | 51.53 | 54.98 | 61.46 | 78.14 |
| 200 | 44.48 | 55.73 | 58.59 | 62.67 | 72.54 | 78.18 |
| 150 | 52.68 | 58.56 | 60.88 | 64.11 | 73.52 | 78.84 |
| 100 | 55.52 | 58.78 | 60.94 | 66.67 | 74.79 | 79.15 |

The Fig. 6 (Table 4) provides the results of F-measure between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has higher F-measure in classifying the instances based on the rules obtained from the genetic algorithm. The accurate classification from the ruleset provides optimal classification of devices than other existing methods. With increasing number of devices, the accuracy tends to degrade due to complexity in the task of computing the instances of more devices.

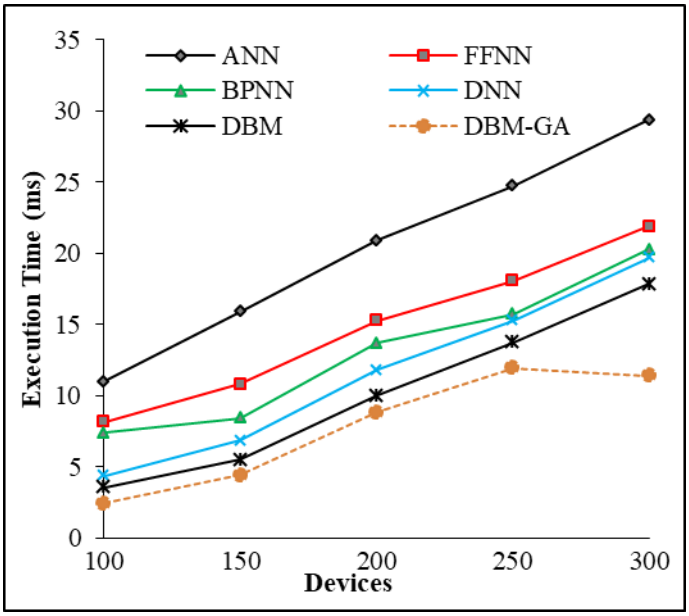


Fig 7. Execution Time (ms)

Table.5. Execution Time (ms)

| Devices | ANN | FFNN | BPNN | DNN | DBM | DBM-GA |
|---|---|---|---|---|---|---|
| 100 | 11.00 | 8.18 | 7.38 | 4.37 | 3.57 | 2.42 |
| 150 | 15.93 | 10.83 | 8.42 | 6.86 | 5.50 | 4.43 |
| 200 | 20.92 | 15.30 | 13.72 | 11.82 | 9.99 | 8.85 |
| 250 | 24.75 | 18.06 | 15.73 | 15.28 | 13.77 | 11.94 |
| 300 | 29.40 | 21.89 | 20.26 | 19.70 | 17.84 | 11.40 |

The Fig. 7 (Table 5) provides the results of Execution Time between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has reduced Execution Time in classifying the instances based on the rules obtained from the genetic algorithm. With increasing number of devices, the execution time tends to increase due to complexity in the task of computing the instances of more devices.

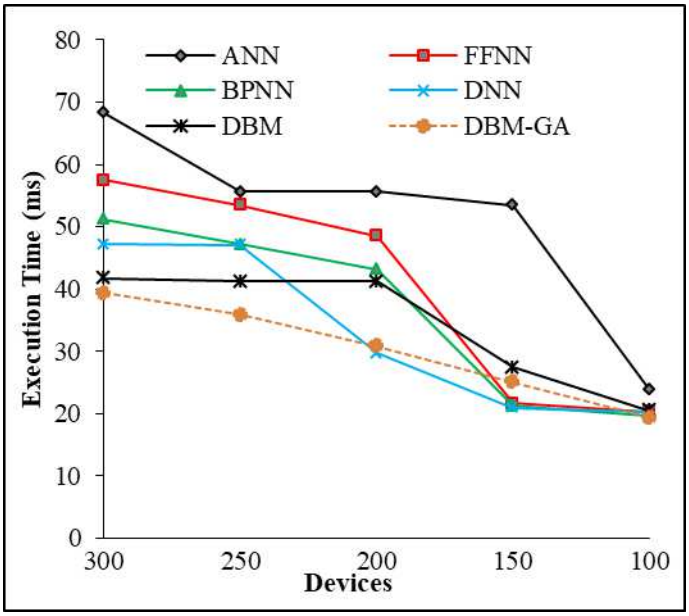


Fig 8. Mean Absolute error

Table 6. Mean Absolute error

| Devices | ANN | FFNN | BPNN | DNN | DBM | DBM-GA |
|---|---|---|---|---|---|---|
| 300 | 68.29 | 57.51 | 51.19 | 47.13 | 41.68 | 39.31 |
| 250 | 55.66 | 53.47 | 47.15 | 47.09 | 41.24 | 35.84 |
| 200 | 55.62 | 48.55 | 43.18 | 41.21 | 30.81 | 29.68 |
| 150 | 53.46 | 27.39 | 25.01 | 21.61 | 21.19 | 20.90 |
| 100 | 23.90 | 20.50 | 20.33 | 20.32 | 19.61 | 19.33 |

The Fig. 8 (Table 6) provides the results of MAE between the proposed DBM and DBM-GA and existing classifiers: ANN, FFNN, BPNN and DNN. The result shows that the proposed method has reduced MAE in classifying the instances based on the rules obtained from the genetic algorithm. With increasing number of devices, the MAE tends to increase due to complexity in the task of computing the instances of more devices. The proposed method is said to deal largely with the elimination of insignificant features that improves the MAE rate.

## 4. Conclusions

In the present study, we use DBN to extract and classify the benign and malicious IoT devices based on the features of IoT devices and the server identity in the network of an industry or organisations. DBN operated successfully as a combined approach of feature extraction and classification with unsupervised and supervised learning, respectively. The unsupervised learning uses RBM and supervised learning uses BP for optimal classification of IoT device features. Extraction of multiple features from the network efficiently classifies the devices. The simulation results confirms the efficacy of the proposed classification model and the results show that the DBM has better malicious device detection in the network than existing deep learning classifiers methods.

**References**


[1] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, *17*(4), 2347-2376.

[2] Sethi, P., & Sarangi, S. R. (2017). Internet of things: architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, *2017*.

[3] Lee, H. J., & Kim, M. (2018). The Internet of Things in a smart connected world. *Internet of Things-Technology, Applications and Standardization*, 91.

[4] Dang, L. M., Piran, M., Han, D., Min, K., & Moon, H. (2019). A survey on internet of things and cloud computing for healthcare. *Electronics*, *8*(7), 768.

[5] Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems*, *29*(7), 1645-1660.

[6] Minerva, R., Biru, A., & Rotondi, D. (2015). Towards a definition of the Internet of Things (IoT). *IEEE Internet Initiative*, *1*(1), 1-86.

[7] Indu, I., Anand, P. R., & Bhaskar, V. (2018). Identity and access management in cloud environment: Mechanisms and challenges. *Engineering science and technology, an international journal*, *21*(4), 574-588.

[8] Lomotey, R. K., Pry, J. C., & Chai, C. (2018). Traceability and visual analytics for the Internet-of-Things (IoT) architecture. *World Wide Web*, *21*(1), 7-32.

[9] Yazici, M. T., Basurra, S., & Gaber, M. M. (2018). Edge machine learning: Enabling smart internet of things applications. *Big data and cognitive computing*, *2*(3), 26.

[10] Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., & Sivaraman, V. (2018). Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, *18*(8), 1745-1759.

[11] Nieto, A., & Lopez, J. (2014). Analysis and taxonomy of security/QoS tradeoff solutions for the future internet. *Security and Communication Networks*, *7*(12), 2778-2803.

[12] McNeil, P. (2017). Secure Internet of Things Deployment in the Cement Industry: Guidance for Plant Managers. *IEEE Industry Applications Magazine*, *24*(1), 14-23.

[13] Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., & Sivaraman, V. (2018). Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, *18*(8), 1745-1759.

[14] Bai, L., Yao, L., Kanhere, S. S., Wang, X., & Yang, Z. (2018, October). Automatic device classification from network traffic streams of internet of things. In *2018 IEEE 43rd Conference on Local Computer Networks (LCN)* (pp. 1-9). IEEE.

[15] Kousalya, A., Sakthidasan, K., & Latha, A. (2019). Reliable service availability and access control method for cloud assisted IOT communications. *Wireless Networks*, 1-12.

[16] Zhou, L., Niu, X., Zhao, S., Zhao, X., Cao, N., Ding, J., & Wang, R. (2020). Research on congestion rate of classified storage narrow channel picking system for IoT security. *Wireless Networks*, 1-17.

[17] Tsai, C. W., Lai, C. F., & Vasilakos, A. V. (2014). Future Internet of Things: open issues and challenges. *Wireless Networks*, *20*(8), 2201-2217.

[18] Awoyemi, B. S., Alfa, A. S., & Maharaj, B. T. (2020). Resource Optimisation in 5G and Internet-of-Things Networking. *Wireless Personal Communications*, 1-32.

[19] Boubiche, D. E., Athmani, S., Boubiche, S., & Toral-Cruz, H. (2020). Cybersecurity issues in wireless sensor networks: current challenges and solutions. *Wireless Personal Communications*, 1-37.

[20] Yerima, S. Y., & Al-Begain, K. (2011). Novel radio link buffer management schemes for end-user multi-class traffic in high speed packet access networks. *Wireless Personal Communications*, *61*(2), 349-382.

[21] Sobin, C. C. (2020). A Survey on Architecture, Protocols and Challenges in IoT. *Wireless Personal Communications*, 1-47.

[22] Zheng, D., Qin, B., Li, Y., & Tian, A. (2020). Cloud-Assisted Attribute-Based Data Sharing with Efficient User Revocation in the Internet of Things. *IEEE Wireless Communications*, *27*(3), 18-23.

[23] Kwak, J. Y., Cho, C., Shin, Y., & Yang, S. (2019). IntelliTC: intelligent inter-DC traffic controller for the Internet of everything service based on fog computing. *IET Communications*, *14*(2), 193-205.

[24] Kousik, N., Natarajan, Y., Raja, R. A., Kallam, S., Patan, R., & Gandomi, A. H. (2020). Improved Salient Object Detection Using Hybrid Convolution Recurrent Neural Network, Expert Systems with Applications, Vol 166, 114064, 2021.

[25] M. Sivaram, A. S. Mohammed, V. Manikandan, V. Porkodi et al., "Improved enhanced dbtma with contention-aware admission control to improve the network performance in manets," Computers, Materials & Continua, vol. 60, no.2, pp. 435–454, 2019.

[26] M. Sivaram, V. Porkodi, A. S. Mohammed, V. Manikandan, "Retransmission DBTMA Protocol with Fast Retransmission Strategy to Improve the Performance of MANETs," in IEEE Access, vol. 7, pp. 85098-85109, 2019, doi: 10.1109/ACCESS.2019.2918723.

[27] Xu, F., Fang, Z., Tang, R., Li, X., & Tsui, K. L. (2020). An unsupervised and enhanced deep belief network for bearing performance degradation assessment. *Measurement*, 107902.

[28] Chen, Z., Ma, W., Dai, W., Pan, W., & Ming, Z. (2020). Conditional restricted Boltzmann machine for item recommendation. *Neurocomputing*, *385*, 269-277.

[29] Patel, M. S., & Mazumdar, H. S. (2014). Knowledge base and neural network approach for protein secondary structure prediction. *Journal of theoretical biology*, *361*, 182-189.

[30] Jiang, H., Xi, Z., Rahman, A. A., & Zhang, X. (2020). Prediction of output power with artificial neural network using extended datasets for Stirling engines. *Applied Energy*, *271*, 115123.

[31] Sarraf, A. (2020). A tight upper bound on the generalization error of feedforward neural networks. *Neural Networks*, 127, 1-6.

[32] Das, L., Sivaram, A., & Venkatasubramanian, V. (2020). Hidden Representations in Deep Neural Networks: Part 2. Regression Problems. *Computers & Chemical Engineering*, 139, 106895.

[33] Hussain, A., Manikanthan, S. V., Padmapriya, T., & Nagalingam, M. (2019). Genetic algorithm based adaptive offloading for improving IoT device communication efficiency. *Wireless Networks*, 1-10.

[34] Meidan, Y., Bohadana, M., Shabtai, A., Guarnizo, J. D., Ochoa, M., Tippenhauer, N. O., & Elovici, Y. (2017, April). ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis. In *Proceedings of the symposium on applied computing* (pp. 506-509).

[35] Lee, H. (2017). Framework and development of fault detection classification using IoT device and cloud environment. *Journal of Manufacturing Systems*, *43*, 257-270.

[36] Sivanathan, A., Sherratt, D., Gharakheili, H. H., Radford, A., Wijenayake, C., Vishwanath, A., & Sivaraman, V. (2017, May). Characterizing and classifying IoT traffic in smart cities and campuses. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 559-564). IEEE.

[37] Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A. R., & Tarkoma, S. (2017, June). Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)* (pp. 2177-2184). IEEE.

[38] Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., & Sivaraman, V. (2018). Classifying IoT devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, *18*(8), 1745-1759.

[39] Bai, L., Yao, L., Kanhere, S. S., Wang, X., & Yang, Z. (2018, October). Automatic device classification from network traffic streams of internet of things. In *2018 IEEE 43rd Conference on Local Computer Networks (LCN)* (pp. 1-9). IEEE.

[40] Meidan, Y., Bohadana, M., Shabtai, A., Ochoa, M., Tippenhauer, N. O., Guarnizo, J. D., & Elovici, Y. (2017). Detection of unauthorized iot devices using machine learning techniques. *arXiv preprint arXiv:1709.04647*.

[41] Samie, F., Bauer, L., & Henkel, J. (2020). Hierarchical Classification for Constrained IoT Devices: A Case Study on Human Activity Recognition. *IEEE Internet of Things Journal*.

[42] Ammar, N., Noirie, L., & Tixeuil, S. (2020, June). Autonomous Identification of IoT Device Types based on a Supervised Classification. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)* (pp. 1-6). IEEE.

[43] Aksoy, A., & Gunes, M. H. (2019, May). Automated iot device identification using network traffic. In ICC 2019-2019 IEEE International Conference on Communications (ICC) (pp. 1-7). IEEE.