Telematics and Informatics Reports 7 (2022) 100009

Contents lists available at ScienceDirect

Telematics and Informatics Reports

journal homepage: www.elsevier.com/locate/teler

Towards an effective deep learning-based intrusion detection system in the internet of things

Pampapathi B M[a,*], Nageswara Guptha M[a], M S Hema[b]

[a] Department of Computer Science & Engineering, Sri Venkateshwara College of Engineering, Bengaluru, Karnataka, India
[b] Department of Information Technology, Anurag University, Hyderabad, Telangana, India

ARTICLE INFO

Keywords:
Filtered deep learning neural network
Modified K-means algorithm
Advanced message queuing protocol
Levy flight based Elephant herding optimization algorithm
Message queuing telemetry transport

ABSTRACT

Distributed Sensor Networks play a vital role in the day-to-day world of computing applications, from the cloud to the Internet of Things (IoT). These computing applications devices are normally attached with the microcontrollers like Sensors, actuators, and Adriano network connectivity. Defensive network with an Intrusion Detection System thus serves as the need of modern networks. Despite decades of inevitable development, the Intrusion Detection System is still a challenging research area as the existing Intrusion Detection System operates using signature-based techniques rather than anomaly detection. The existing Intrusion Detection System are thus facing challenges for improvement in Intrusion Detection, Handling heterogeneous data sources is hard for discovering zero-day attacks in IoT networks. This paper presents Filtered Deep Learning Model for Intrusion Detection with a Data Communication approach. The proposed model is composed of five phases: Initialization of Sensor Networks, Cluster Formation in addition to Cluster Head Selection, Connectivity, Attack Detection, and Data Broker. The proposed Model for Intrusion Detection was found to outperform the existing Deep Learning Neural Network and Artificial Neural Network. Experimental results showed a better result of **96.12% accuracy** than the dominant algorithms.

1. Introduction

In current years, cloud computing (CC) is getting increasingly well-liked in educational along with commercial places [1], in addition, its function is also largely accepted for data storage along with retrieval on the cloud environment [2]. For the basis of increasing the number of data on the Cloud servers(CS), the cloud storage consent to store data in numerous inaccessible sites typically hold by "top" business and running their owner solutions, e.g., Google Drive, Dropbox, Amazon Simple Cloud Storages Service (S3) [3] that means the data is amassed distributive. Even if numerous attractive advances to data gathering were developed in the past decade, it still stays as a research focus in full sway with several significant challenges. Certainly, the incessant decrease in sensor size along with cost, the diversity of sensors present on the marketplace together with the incredible progress on Sensor Node communication technologies have expanded potentially the effect of IoT [4].

IoT comprises SN that is able to compute, sense together with communicate the data by particular communication [5]. For sensing, the SN are utilized, in addition, for computation, numerous algorithms are utilized, at last, for the communication, disparate communication protocols are utilized, for instance, Building Automations and Control Network (BACnet), Distributed Network Protocols version 3 (DNP3) [6], Constrained Application Protocol (CoAP) [7], Message Queue Telemetry Transports (MQTT) [8], together with XMPP (Extensible Messaging and Presences Protocol), AQMP, et cetera, [9]. By utilizing these protocols, the data is amassed. That gathered data might have malicious activities; therefore, the discovery of malevolent activities is a significant procedure. Malevolent activities mean Intrusion Detections (ID) [10]. An Intrusion is a compilation of activities that violate security policies, encompassing the integrity along with confidentiality of data together with the accessibility of services, by means of exploiting the susceptibilities in the security process along with the scheme's execution observed via IDS [11].With the increasing deployment of IoT systems and the truth that their safety impacts IoT systems themselves in addition to different structures related to the internet, we focus our research on IoT security.

The working architecture of any Intrusion Detection Model includes Normal data, Attacked data, Events collector and IoT devices as shown in Fig. 1.

The existing Intrusion Detection System are thus facing challenges for improvement in Intrusion Detection, Handling heterogeneous data sources is hard for discovering zero-day attacks in IoT networks. This paper presents Filtered Deep Learning Model for Intrusion Detection with a Data Communication approach. Within the proposed Filtered Deep gain-

* Corresponding author.
E-mail addresses: bm.pampapathi77@gmail.com (P. B M), mnguptha@yahoo.com (N.G. M), ghema_shri@yahoo.co.in (M.S. Hema).

https://doi.org/10.1016/j.teler.2022.100009
Received 2 May 2022; Accepted 27 May 2022

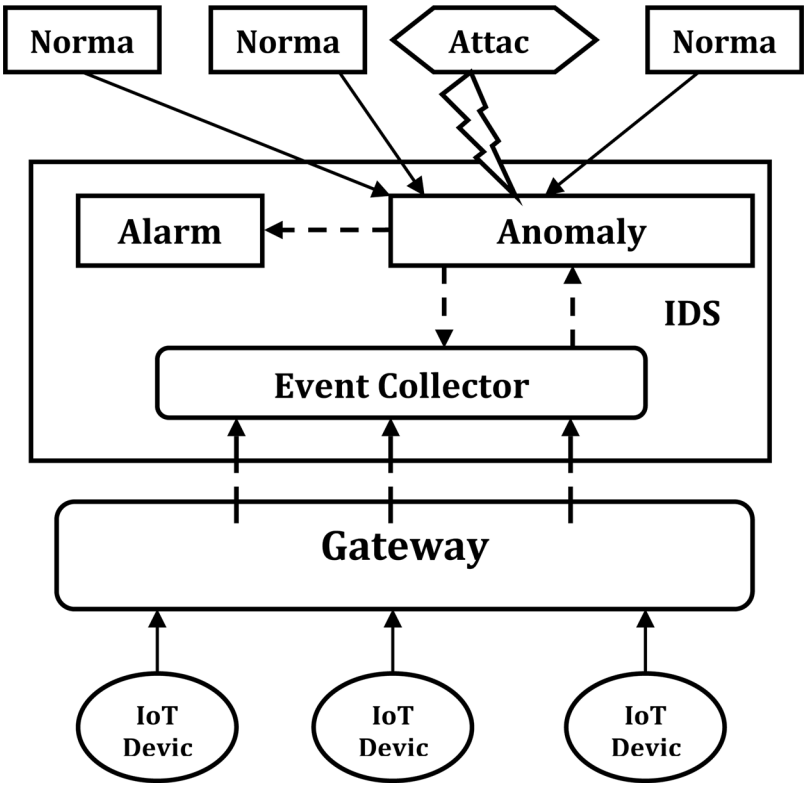


Fig. 1. Architecture of intrusion detection system.

ing knowledge of Model for Intrusion Detection, first the sensor nodes are initialized, and then the sensor nodes are grouped, and the cluster head selection is accomplished by way of the use of the Modified K-Means (MKM) algorithm. After that, the sensed values are amassed by using MQTT and AQMP protocol connectivity, which support each light and heavyweight device. Subsequent, the Intrusion is detected within the sensed values, which includes pre-processing and class steps. In pre-processing, the statistics is dependent and attack is classified the usage of the FDLNN set of rules. The attacked records is saved into the log database and ordinary records is distributively allotted to servers which can be denoted because the facts broking, which includes two levels, specifically, characteristic extraction and allocation. The information is allotted using the Levy Flight based Elephant Herding Optimization (LFEHO) algorithm.

In the proposed approach MQTT protocol is used. MQTT support large-scale network which is a many-to-many communication protocol for passing messages between multiple clients through a central broker. It decouples producer and consumer by letting clients publish and having the broker decide where to route and copy messages. While MQTT has some support for persistence, it does best as a communications bus for live data.

The paper is organized as: Section 2 surveys the associated works concerning the proposed technique. In sections 3, a concise discussion about the phases of proposed Model; Section 4 examines the Investigational results, and Section 5 will convey this paper's conclusion.

2. Related work

Intrusion Detection Systems (IDs) in IoT search for assault signatures that are particular styles that commonly indicate malicious or suspicious cause. This IDs fashions the normal utilization of the network as a noise characterization. Something notable from the noise is believed to be an Intrusion hobby. Remarkable procedures of classifying IDs primarily based mostly on Anomaly detection, Signature based totally misuse, host based totally, network based totally and stack based absolutely.

Pampapathi B M et al. [12] introduced 'two' methods for IoT data distribution to the cloud utilizing IANFIS in addition to secure transmission of those distributed data utilizing MECC. Initially, the IoT device is authenticated by means of performing registration, login, as well as verification for preventing the device as of unauthorized utilization. Subsequently, the device is accessed to sense the information as of associated objects by utilizing smart sensors; in addition, the communication of disparate IoT sensors is performed by employing the MQTT protocol. By means of the IANFIS, this sensed information as of disparate IoT devices is distributed to the specific cloud server (CS). Then by employing the MECC algorithm, the data is sent securely to the user. Lastly, the local calculation (LC) is carried out by means of HFPGA. The experimentation is executed for analyzing the proposed work's performance. The outcomes exhibited that the proposed techniques work better when weighed against other existing algorithms.

Farhan Siddiqui et al. [13] attained a safe information switch device betwixt IoT gadgets with the applied open-source execution of the restrained packages Protocol (CoAP) and Datagram shipping Layers protection (DTLS). The analysis tested that the usage of a CoAP-DTLS execution with a symmetric key ciphers suite brought about major overall performance costs. A safe hyperlink with DTLS over CoAP utilized about 10 percentage greater electricity than an unsecured connection. additionally, the latency take a look at exposed over a a hundred percentage augmentation in general latency time for safe messages contrasted to while no encryption changed into applied. The approach additionally emphasized numerous of the execution challenges whilst raising a actual IoT testbed for safe experimentation.

Amjad Mehmood et al. [14] superior a information-concentrated context-conscious method for dealing with the Intrusions produced by means of malevolent nodes. The know-how-focused Intrusion Detection Machine (KB-IDS) split the networks into clusters in this kind of method that each cluster turned into rendered a CH. The CH located the overall performance of anyone its associate nodes inside its cluster. The behavioural statistics become recorded inside the shape of super activities. focused at the cluster along with CH, the facts become accumu-

lated. The scheme assisted the swapping of CH with different companion nodes each time important. The KB-IDS illustrated sensibly appropriate simulation outcomes contrasted to different protection alongside non-protection structures.

Sivakami Raja and Saravanan Ramaiah [15] added a method for modeling IDS in a cloud. The id schooling dataset become separated into a few clusters with alike patterns healthy in to the same cluster concentrated on the modified ok-means algorithm with Minkowski distance. each one of the ensuing clusters was said with positive membership features with the aid of statistical imply along with deviation. After that, the related parameters have been superior with the aid of dynamical superior getting to know. The scheme final results changed into attained thru the empirical exam. in a while, the final results changed into contrasted with the desired outcome focused on which detection quotes have been meant.

Rajendra Patil et al. [16] recommended hypervisor levels dispensed network security (HLDNS) structure, which turned into organized on every processing servers of CC. At each server, it found the essential digital machines (VM) related community visitors to/from the virtual networks, internal community, in conjunction with external network for identity. The shape increased a binary bats set of rules (BBA) haing ’2′ fitness capabilities for bringing approximately the possible individual as of cloud network site visitors. The derived characters were carried out to the Random Forests classifiers for noticing the Intrusions on cloud networks site visitors in conjunction with Intrusion signals had been produced. The gadget’s performance changed into evaluated concerning community security.

Jiye Park et al. [17] delivered a secure communique scheme alongside focused on distinct challenges in placed Datagram Transports Layers safety (DTLS) into useful resource- restrained surroundings. The scheme applied an IoT–Cloud collaboration, in which DTLS handshake allocation became the principal constituent. The scheme also employed in conjunction with envisioned the scheme within the actual IoT testbed, wherein limited gadgets were interconnected with each other in a multi-hop style. Evaluation results illustrated that the scheme considerably lessened DTLS handshake latency, implementation code length, together with power intake.

The intrusion detection system (IDS) [18–20] discusses about the method used to detect the intruders in the cloud computing networks. Recurrent neural networks, auto encoders in deep learning neural network, rectified liner system with neural networks are used to detect the attack in the data over the network. The various machines learning strategy [21] like SVM with extreme learning, Naive Bayes with Net, ripple rule learner, decision tree, Adaboost gives high accuracy for data selection process. The ecludien distance based KNN algorithm is used in [22–24] for effective data classification. The artificial neuron using deep neural network as of in human brain functions is implemented in [[25], [26]] for more non linearity problems based real life situations. The feature extraction with unit of information gain and theory [27–29] used to rank the features based on the gain performance. The correlations which are linear between the features are extracted using Pearson linear correlation coefficient algorithm [30] used to identify the correlation coefficient between the features. This result helps evaluates the feature with high efficiency and relevant.

Nadim Rana et al. [31] delivered a work on recent developments regarding amalgamation Nature-inspired algorithms in deep learning architectures. A brief view from Recent developments, strengths, challenges, and opportunities for future research Regarding the synergies between natured-inspired algorithms and deep learning Foot. He found that synergies between nature-inspired algorithms and the communities of deep learning research are limited given the little interest it has attracted in literature. Work has the potential to bridge the communication gap between nature-inspired algorithms and deep learning research communities.

Johnson et al.[32]the Survey on deep learning with class imbalance provides the most comprehensive analysis of deep learning methods for addressing the class imbalance data problem. Fifteen studies published between 2015 and 2018 are summarized and discussed, exploring a number of advanced techniques for learning from imbalanced data with DNNs. It has been shown that traditional machine learning techniques for handling class imbalance can be extended to deep learning models with success. Te survey also finds that nearly all research in this area has been focused on computer vision tasks with CNNs. Despite a growing demand for big data analytics solutions, there is very little research that properly evaluates deep learning in the context of class imbalance and big data. Deep learning from class imbalanced data is still largely understudied, and statistical evidence which compares newly published methods across a variety of data sets and imbalance levels does not exist.

Summarizing the recent typical studies, this paper analyzes and refines the challenge of ID’s in the IoT. IoT encompasses lots of Sensor Node that is utilized to monitor the data. But there is a chance for the monitored data to be affected by means of Intrusion. Though the prevailing research works detect the intrusion, their data storage performance is not good and the accuracy is also low Lacking of available datasets may be the biggest challenge. Thus, effectual Intrusion Detective System with distribution of data to cloud storage system centred becomes an important research direction in the IoT.

3. Phases of filtered deep learning model for intrusion detection

In current years, IoT has established as an effectual means of interconnecting sensor devices with concurrent communication in addition to data processing. During that DC, the intruder might affect the data; thus, prior to storing these data into the CS, the malicious activities must be detected. This paper efficiently amasses the data into the CS centred on the IDS utilizing the FDLNN. The system comprises ’5′ phases: i) initialization of sensor networks, ii) Cluster Formation (CF) in addition to CH selection, iii) connectivity, iv) Attack Detection (AD), and v) data broker. At the preliminary stage, the SN is initialized, then they are clustered and the CH is chosen utilizing the MKM. Next, the API gathers the sensed data as of the SN that utilizes the AQMP along with the MQTT protocol. By utilizing this protocol, this system renders a higher outcome for lightweight and heavy devices. Subsequent to gathering the data, with the help of the connectivity, the Intrusion is detected in that gathered sensor data, which comprises ’2′ steps: a) pre-processing and b) classification. It is processed in the training as well as testing phase. Initially, the data is trained with the TON_IoT data sources. At the time of testing, just the sensed data is taken. During classification, the proposed technique utilizes the new FDLNN which could classify the normal from the attacked data. Subsequent to the identification of normal and attacked data, the attacked data are amassed in the log database and the normal data is distributively amassed into the CS. The storing phase comprises ’2′ steps, i) Feature Extraction(FE) and ii) data allocation. The LFEHO distributively allocates the data to the CS. For this allocation, the fixed threshold value of the extracted features is regarded as the fitness function. Centred on this step, the proposed work communicated and efficiently amassed in the CS. The proposed system’s block diagram is exhibited as Fig. 2.

3.1. Initialization of sensor network

Firstly, the nodes are initialized on the IoT Sensors. A IoT is basically a system of a network comparing of spatially distributed independent devices, like Zigbee, IPv6 over Low-Powers Wireless Personal Areas Network (6LoWPAN), Bluetooth, et cetera, which utilizes sensors to jointly monitor environmental or physical conditions. The initialized SN are mathematically indicated as,

$$W_N = \{s_1, s_2, s_3, \ldots\ldots, s_n\}\ (or)\ s_i,\ \ i = 1, 2, 3, \ldots\ldots, n \tag{1}$$

where, W_N signifies the IoT Sensor set and s_n implies the n-number of SN. (Fig. 3, Table 1)

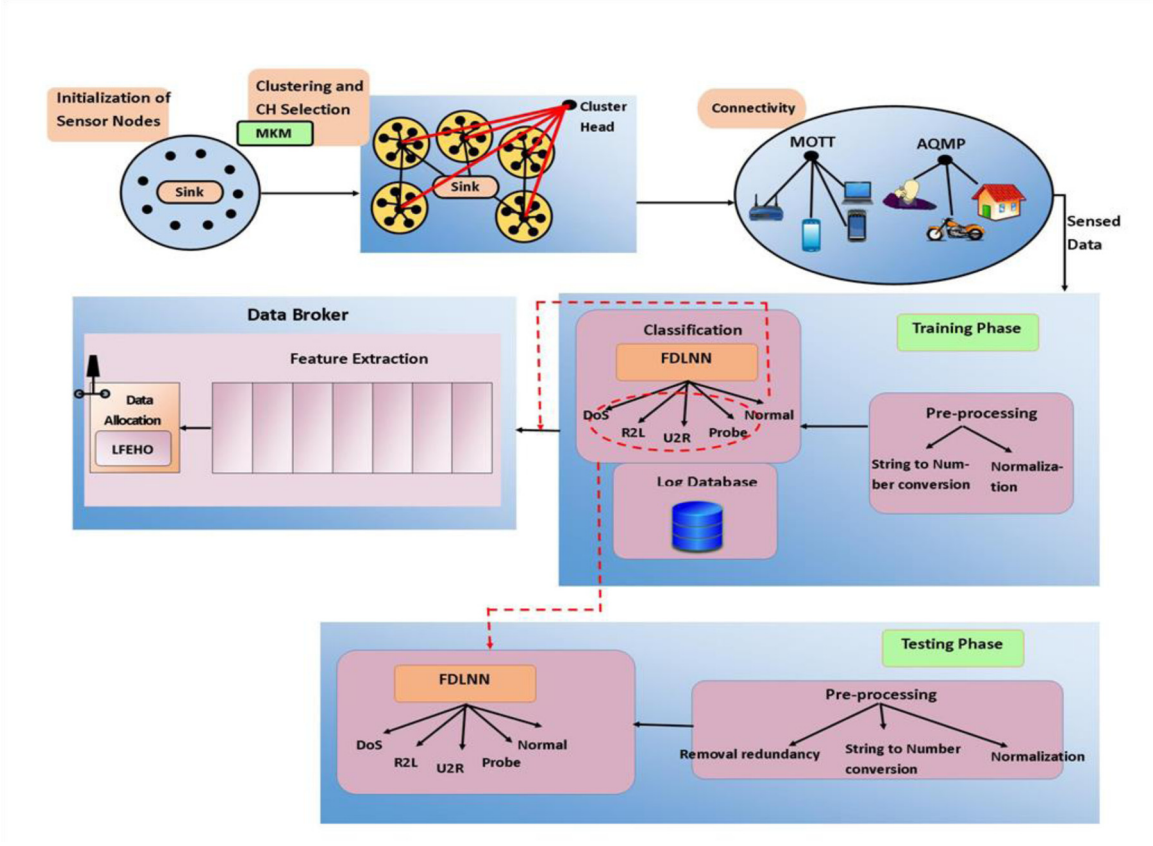


Fig. 2. Flow diagram of proposed methodology.

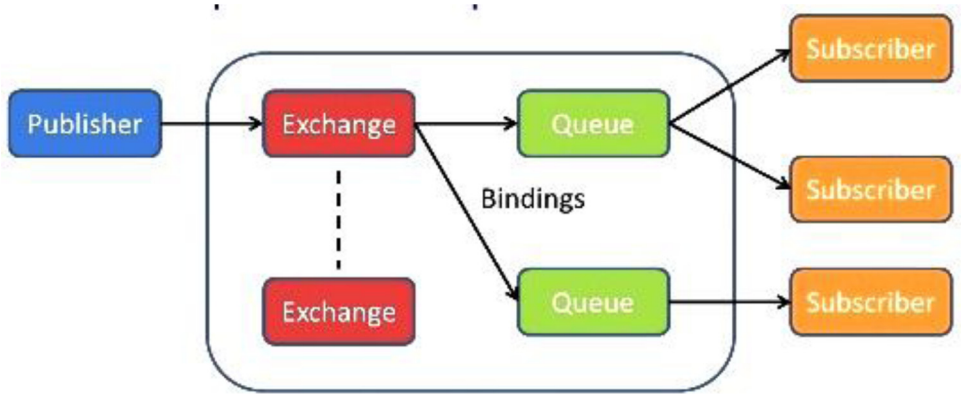


Fig. 3. Data communication by MQTT and AMQP.

3.2. Cluster formation and cluster head selection

Here, the initialized SN is formed into clusters, and then Modified K-Means (MKM) algorithm chooses the CH. A famous partitioning approach termed K-means clustering is an idea, within which there is a requirement to classify the provided SN as K clusters; the value of K (Number of clusters) is stated by means of the user, which is fixed. This algorithm is more helpful in building the clusters aimed at WSN's different real-world applications. In addition, the algorithm's feature is simple, greatly reliable, and speedy convergence of iterations; it performs re-clustering at failure states. In this 1st centroid, every cluster is chosen for clustering, and then as per the chosen centroid, data points encompassing a minimal distance as of a provided cluster, is assigned to that specific cluster. Generally, Euclidean Distance is utilized for computing the distance of data points as of the specific centroid. However, the fundamental Euclidean is not appropriate for a large quantity of data since the SN sense a large volume of data environmentally, in order that, here, Mahalanobiss distance is regarded for rendering better cluster outcomes than the Euclidean distance computation and attains better clustering accuracy. The CF along with CH selection comprises several steps and those steps are elucidated as,

Step 1: K-means algorithm is employed for CF with the target IoT Sensor Network. Presume that the IoT Sensor Network of n nodes is bifurcated into k clusters. Initially, k out of n nodes are arbitrarily chosen as the cluster centroids $\{c_1, c_2, c_3, \ldots\ldots\ldots, c_n\}$ (or) c_i, $\quad i = 1, 2, 3, \ldots\ldots n$.

Step 2: Here, compute the distance betwixt the SN and the chosen cluster centroids utilizing the Mahalanobiss distance computation. The pros of this distance computation are that it considers co-variance, which aids in gauging the strength/similarity betwixt the cluster centroids and the SN. The distance computation is mathematically written as,

$$\delta = \sqrt{\left(c_i - s_i\right)^T \mu \left(c_i - s_i\right)} \tag{2}$$

Table 1
Pseudo code for the proposed LFEHO algorithm.

Input: Extracted features, $U_i = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7\}$
Output: Data allocation

begin
initialization Set generation counter t = 1,
Set Maximum Generation M_G
  compute fitness function
  while t < M_G **do**
  Sort all the elephants according to their fitness
  for all clans cl_v, in the population **do**
    for all elephants q in clan cl_v **do**
      Update $z_{cl_v,q}$and generate $z_{new,cl_v,q}$ by using,
      $z_{new,cl_v,q} = z_{cl_v,q} + \psi + (z_{best,cl_v} - z_{cl_v,q}) \times \xi$
      if $z_{cl_v,q} = z_{new,cl_v,q}$ **then**
        Update $z_{cl_v,q}$ and generate $z_{new,cl_v,q}$ by using
        $z_{new,cl_v,q} = \psi \times (z_{center,cl_v})$
      end if
    end for
  end for
  for all clans cl_v, in the population **do**
    replace the worst elephant in clan by using
    $z_{worst,cl_v} = z_{\min} + (z_{\max} - z_{\min} + 1) \times \xi$
  end for
  evaluate population by the newly updated positions (i.e check fitness function)
  t = t+1
  end while
  return the best solution (i.e., allocate the data)
end

where,

δ - Distance calculation

μ - Co-variance matrix

Step 3: allocated the cosine similarity point to the cluster.

Step 4: Set every cluster position to the mean of the entire data points allocated to the cluster.

Step 5: Step 3 and step 4 are repeatedly executed till no more changes found.

Step 6: Subsequent to CF, the CH is picked as of the generated clusters during CH selection. Here, every node of a cluster is allocated with ID number as per the distance as of the centroid, allotting a smaller number to the nearer one. The ID number of a node implies the order to be selected as the CH. Thus, the ID number plays a significant part in the assortment of a node as CH. In addition, residual energy plays a significant part in the CH selection procedure. The residual energy (Rd_e) is mathematically indicated as,

$$Rd_e = l_e - t_e \tag{3}$$

where, Rd_e defines the difference betwixt the initial energy (l_e) and current energy (t_s) of the SN. Here, the CH's residual energy is checked each round for retaining the network's connectivity. If the CH's energy is lesser than the preset threshold, the node in the subsequent order is chosen as a new CH. The freshly voted CH informs other nodes of the change of the CH.

Centred on this CH and CF selection, the sensed information is attained by utilizing the optimal routing.

3.3. Connectivity

Here, utilizing the AQMP along with MQTT protocol, the sensed data is communicated with the Application programming interfaces, which is useful for data brokers. The proposed work considers '2' connectivity since MQTT is only utilized for lightweight devices, like Mobile devices, and embedded systems wherein minimal overhead is needed and bandwidth is expensive. In MQTT, these features are just appropriate for lightweight devices and it not helpful for big devices. Thus, the proposed paper also regards the Advanced Messages Queuing Protocol (AMQP), which is modelled with more advanced features. The AMQP encompasses a dependable message queue, and also it renders secure communication. It proffers an extensive gamut of features associated with messaging, like a reliable queuing, topic-centred publish-and-subscribe messaging, flexible routing, as well as transactions. AMQP exchanges messages in an assortment of ways: directly, in fan-out form, by topic, or centred upon headers. AMQP stands as a binary protocol and usually needs a fixed header of 8-bytes. The heavy devices' connectivity utilizing AMQP is implied as,

$$A_p = h_d \tag{4}$$

where, A_p implies the AMQP and h_d signifies the heavy devices. A binary protocol termed MQTT usually needs a fixed header of 2bytes with smaller message payloads. MQTT is most appropriate for larger networks of smaller devices that require be monitoring or controlling as of a back-end server on the Internet. The lightweight devices' connectivity utilizing the MQTT is implies as,

$$M_t = t_d \tag{5}$$

where, M_t - MQTT protocol, t_d- Lightweight devices

3.4. Attack detection

Here, the attack is effectively detected as of the communicated data, which comprises steps say: i) pre-processing and ii) classification phase. For this AD, the data is previously trained by means of utilizing the TON IoT datasets. The training as well as testing procedure of AD is elucidated in this section.

3.4.1. Pre-processing

Originally, the data is in an unclear format; therefore, the pre-processing is required for structuring the data. During training, the dataset comprises the redundant data in order that the redundant is eradicated. However, in the testing, the sensed data is arbitrarily attained by means of the SN, in order that, in the testing phase, the redundancy removal is not regarded. Next, in training along with testing phases, if any string data is present, then the string is transmuted into the numerical format. Subsequent to string conversion, the values are normalized by utilizing the linear scaling technique that means the values as of their natural gamut to a standard gamut typically regard zero and one. The subsequent formula is regarded for the scale to a range:

$$S_r = \left(d - d_{\min}\right) / \left(d_{\max} - d_{\min}\right) \tag{6}$$

where, S_r implies the scale to range, d signifies the communicated data, $d_{\min}$ and $d_{\max}$ implies the minimum as well as maximum values of the communicated data. Lastly, the normalized values are regarded as the matrix format, which comprises more attributes pertinent values, and it is mathematically denoted as,

$$S_r(set) = \left\{o_1, o_2, o_3, \ldots\ldots o_n\right\} \ (or) \ o_i, \ \ i = 1, 2, \ldots\ldots n \tag{7}$$

where, $S_r(set)$ signifies the total normalized value set and o_n implies the n-number of matrix values of the normalized values.

3.4.2. Classification using FDLNN

The prevailing research works detect the Intrusion using existing Deep Learning Neural Network (DLNN) and, the existing Artificial Neural Network (ANN), and their data storage performance is not good and the accuracy is also low. Subsequent to normalization, the matrix form of the data is inputted to the FDLNN. The normal Neural Network encompasses '3' disparate layers like the input, hidden, along with output layer. Here, '3′ hidden layer are taken as it examines the data deeply and renders a more accurate outcome. And the inputted matrix format values comprise more attribute pertinent values; however, some values might comprise unnecessary values. Thus, here, originally the Filtered function is utilized, which means the convolution together with pooling functions is implied as the Filtered function. In order that the proposed FDLNN comprises the '4' layers: Filtered, input, hidden, and also output layer. Among these four FDLNN layers, the input layer is implied as the

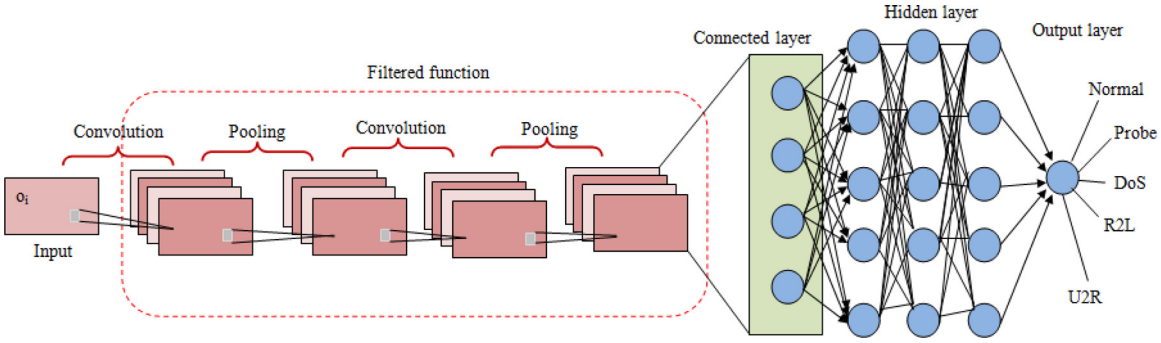


Fig. 4. Structure of the FDLNN.

fully connected layer (FCL). The FDLNN's structure is exhibited using Fig. 4,

Initially, the normalized values' matrix form is inputted to the convolution function. The convolution function is computed as,

$$B_a^{lr} = B_a(o_i) \tag{8}$$

where,

B_a^{lr} - Convolution function output

$B_a(\cdot)$ - Non-linear activation functions of convolution

After that $B_a^{lr}$is inputted to the pooling function. It is typically placed betwixt '2′ convolutional layers. Every feature map (explicitly, the matrix form of normalized values) of a pooling layer and its equivalent feature map of the prior convolutional layer are united. The pooling function is implied as,

$$P_a^{lr} = P_a\left(B_a^{lr}\right) \tag{9}$$

where,

P_a^{lr} - Pooling layer output

$P_a(\cdot)$ - Pooling function's activation function

Subsequent to numerous convolutional layers and pooling layer, one or multiple FCLs that intend to execute higher-level reasoning might exist. The FCL function is implied asC_i. The FCL's output is inputted to the hidden layers "τ^{lr}"which then give the output of,

$$\tau^{lr} = b + \sum_{i=1}^{n} C_i \cdot \chi_i \tag{10}$$

where,

χ_i- Specific layer's weight value

b - Bias value

C_i - Input data

After that, gauge the output unit via adding up the entire weights of the inputted values. This is the computation to achieve the neurons' value in the output layer.

$$\phi^{lr} = b_1 + \sum_{i=1}^{n} \left(\tau^{lr}\right)_i \cdot \chi_i \tag{11}$$

where, ϕ^{lr}signifies the output unit. Then, the algorithm for training the network is grounded on the minimization of an energy function signifying the instantaneous error. That is to say, assess the error function in the output by employing the succeeding equation.

$$e_{rr} = (Tar_t - \phi^{lr}) \tag{12}$$

where, e_{rr} implies the error signal, ϕ^{lr}implies the output signal together with Tar_t signifies the NN's targeted output. The system's error is ought to be lessened. The FDLNN is categorised as the normal or the attacked data. The attacked data are DoS, R2L, U2R, together with probe. The aforementioned pre-processing in addition to the classification is processed for training as well as testing phases. The attained normal data is signified as,

$$N_d = f_i, \qquad i = 1, 2, 3, \ldots\ldots, n \tag{13}$$

where, N_d implies the normal data as of the sensed values and f_i signifies the n-number of normal data rendered in the sensed values.

3.5. Data broker

Subsequent to AD, the attack data are amassed into the log database. After that, the normal data collected by means of the data broker is amassed into the distributed CS, which implies that the data is evenly distributed to the accessible resources with the intention of data distribution. The features are well-extracted as of the data and the available CS, and after that, the data is distributed to the CS utilizing LFEHO. Those '2' steps are elucidated as in the sub-section.

3.5.1. Feature extraction

Here, the Speed, Task Cost, Weight, data size, memory, bandwidth, and disk space, which are identified as the suitable parameters for sensed normal data, is stored into the distributed CS. Parameters are learned or estimated purely from the data during training as the algorithm used tries to learn the mapping between the input features and the labels or targets. These parameters are determined based on Usability, Connectivity, Security, Testing in the real world, Coverage of all types of testing those mentioned parameters are extracted. This is mathematically represented as,

$$U_i = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7\} \tag{14}$$

where, the terms $u_1, u_2, u_3, u_4, u_5, u_6$ and u_7 denotes the memory, data size, speed, task cost, weight, bandwidth, and disk space respectively. Thus, these features are explained as follows,

(a) Memory: The cloud storage system has the memory size of "u_1" . $u_1$is extracted utilizing the below expression

$$u_1 = C_{sl} \times S_{sl} \tag{15}$$

where, C_{sl}signifies the count of the storage locations, and S_{sl} indicate the size of each storage location.

(b) Data size: Here, the obtained normal data is regarded and its size is extracted as expressed below,

$$u_2 = \frac{Tot_s}{1 + Tot_s p_e^2} \tag{16}$$

where, u_2 denotes the data size, Tot_s represents the total size of the sensed values, and p_e denotes the allowed probability for committing an error in a sensed data.

(e) Speed: The obtained sensor data speed "u_3"is calculated as follows,

$$u_3 = tt/hold_f \tag{17}$$

where, tt is the turnaround time, and $hold_f$ is the holding up time of gathering the data.

(f) Task cost: The task cost is extracted, which means the sensed normal data is distributively stored into the CS. The demand cost is determined using the equation:

$$u_4 = d_r * hold_t \tag{18}$$

where, u_4 is the task cost, d_r is the data rate of task demands, and $hold_t$ is the holding up time of task demands.

(g) Weight: After speed and task cost extraction, the weight of the data "u_5"is extracted by the combination of the u_3 and $u_4$extracted values. It has the denotation of:

$$u_5 = \lambda * \left(u_3 + u_4\right) \tag{19}$$

Here,

λ - Constant value$\in$ [0, 1].

(h) Bandwidth: Here, the bandwidth of the data "u_6"is extracted by multiplying the total number of tasks (i.e., how many data are stored into the CS) (Num_t) and the usage of data weight (u_5).

$$u_6 = Num_t \times u_5 \tag{20}$$

(i) Disk space: Disk space of the CS "u_7" is the combination of the free space "$free_s$"and used space "$used_s$"of the CS and is expressed as,

$$u_7 = free + used_s \tag{21}$$

3.5.2. Data allocation to the distributed cloud server

After FE, the data are stored into the distributed CS using the LFEHO algorithm. For this storing process, fix the successful threshold value for the extracted features. Based on this threshold fixing, the data storage onto the CS is done. The normal elephant herding optimization algorithm (EHOA) uses the random calculation which might bring about an overlapping problem. This paper employs the LFEHO algorithm to avert this overlapping problem. The EHOA is a novel population-centric optimization approach, which imitates elephant's herding behaviours, and could be modelled into operators say i) clan updating operators and ii) separating operators. Here, first, the clan operator is explained, after that the separating operators are explained.

In clan operators, the matriarch is the leader who guides all the elephants living together in each clan. The complete elephant populace is split into$q$ clans. Every elephant $q$of the clan υmoves according to the matriarch. For the elephant q in clancl_υ, it can be updated as in the following Eq. (22):

$$z_{new,cl_\upsilon,q} = z_{cl_\upsilon,q} + \psi + \left(z_{best,cl_\upsilon} - z_{cl_\upsilon,q}\right) \times \xi \tag{22}$$

where, $z_{cl_\upsilon,q}$ and $z_{new,cl_\upsilon,q}$ are the elephant q's updated old and new positions in clancl_υ, respectively, $\psi \in [0, 1]$ signifies the scale factor and it is the fittest elephant individual in clan $cl_\upsilon$and ξ denotes the levy flight (LF) distribution.

ξ could be evaluated as,

$$\xi = Levy \sim y^{-\varepsilon}(1 < \varepsilon \leq 3) \tag{23}$$

where, $y^{-\varepsilon}$ indicates the levy distribution function. The fittest elephant of each clan could be updated using Eq. (24):

$$z_{new,cl_\upsilon,q} = \psi \times \left(z_{center,\,cl_\upsilon}\right) \tag{24}$$

where, the term new individual $z_{new,cl_\upsilon,q}$defined in Eq. (24) is generated by the information obtained by all the elephants in clancl_υ. Also, $z_{center,\,cl_\upsilon}$ is the center of clan cls_υ, and for the υ^{th} dimension it could be evaluated as follows:

$$z_{center,cl_\upsilon,\upsilon} = \frac{1}{num_{cl_\upsilon}} \times \sum_{q=1}^{num_{cl_\upsilon}} z_{cl_\upsilon,q,\upsilon} \tag{25}$$

where, $1 \leq \upsilon \leq D$ indicates the υ^{th} dimension, and D signifies its total dimension, num_{cl_υ} denotes the number of members in clancl_υ, $z_{cl_\upsilon,q,\upsilon}$ is the $\upsilon^{th}$of the elephant individual$z_{cl_\upsilon,q}$. The clancl_υ's center "$z_{center,\,cl_\upsilon}$" could be evaluated through Dcalculations by Eq. (25).

The secondary phase is the separating operator which is done at each generation, which is evaluated below.

$$z_{worst,cl_\upsilon} = z_{\min} + \left(z_{\max} - z_{\min} + 1\right) \times \xi \tag{26}$$

where, $z_{\max}$ and $z_{\min}$ are the respective upper and lower bounds of the elephant individual position, z_{worst,cl_υ} symbolizes the worst individual in $cl_\upsilon$and ξ represents the LF function. LFEHO could be detailed using Pseudo code shown below,

Above steps explains the data broker using the LFEHO algorithm. In this, the LF function is deployed for the random value selection for providing a better result. The afore-said two operators are also demonstrated in the pseudo-code. By using this LFEHO algorithm, the data is successfully distributed to the CS.

4. Result and discussion

Here, the proposed DC and effective distributed cloud storage system's performance centred on IDS utilizing FDLNN is analyzed. The proposed work implemented in the working platform of JAVA.

4.1. Database description

The TON_IoT datasets are new generations of Industry 4.0/Internet of Things (IoT) and Industrial IoT (IIoT) datasets for evaluating the fidelity and efficiency of different cybersecurity applications based on Artificial Intelligence and Deep Learning algorithms. The datasets have been called 'ToN_IoT' as they include heterogeneous data sources collected from Telemetry datasets of IoT and IIoT sensors Aimed at the IDS's performance examination, the proposed protocol employs the ToN-IoT dataset that encompasses the normal and attacked data. The attacked data types are Distributed Denial of Services (DDoS), ransomware, Denial of Services (DoS), password cracking attack, data injection, etc. The ToN-IoT datasets comprise collected various data sources as of the IoT services' Telemetry data, and also the Operating Systems logs, along with the IoT Network's traffic of that had been gathered as of the medium-scale network's realistic representation modelled in the IoT and Cyber Range Labs present at the University of New South Wales (UNSW), Canberra.

4.2. Performance analysis

In this section, the performance examination is performed in '2' ways, i.e., performance analysis for the proposed IDS and the performance analysis for the presented data broker. The analysis is elucidated in the subsection.

(a) Performance analysis for IDS

Here, the proposed FDLNN algorithm's performance is contrasted with the existing Deep Learning Neural Network (DLNN) and the existing Artificial Neural Network (ANN) concerning the recall, precision, F-Measure, and accuracy, which are elucidated in Table 2,

Discussion: Table 2 demonstrates the performance examination of the proposed IDS scheme utilizing FDLNN with the existent DLNN, ANN, CNN, EOCSL, along with SVM-centred on (a) precision, recall, along with (b) F-Measure, accuracy. The proposed scheme's performance is diverse centred on the numbers of SN that mean the data are collected as of the number of SN. The examination is made by collecting data as of 100 to 500 SN. When the SN count is 100, the proposed FDLNN encompasses 78.91% F-Measure, 78.34% precision, along with 79.50% recall, in addition, the FDLNN's accuracy is the 79.12%. For the same node count, the prevailing DLNN, ANN, CNN, EOCSL along with SVM offer low performance compared to the proposed method, in this, the DLNN and EOCSL is much superior to the other prevailing algorithms. Likewise, for the remaining number of node counts, the proposed FDLNN centred IDS scheme attains high outcomes than the prevailing algorithm. Therefore, it deduces that the FDLNN fulfils the intention of this research technique. The graphical assessment of the IDS is exhibited in Fig. 5.

(b) Performance analysis for data broker

Here, the data allocation's performance utilizing LFEHO is contrasted with the existing IANFIS and ANFIS algorithms centred upon response time, process time, Average Turnaround Time (ATT), throughput, together with Average Waiting Time (AWT), which are graphically signified in Figs. 6–10.

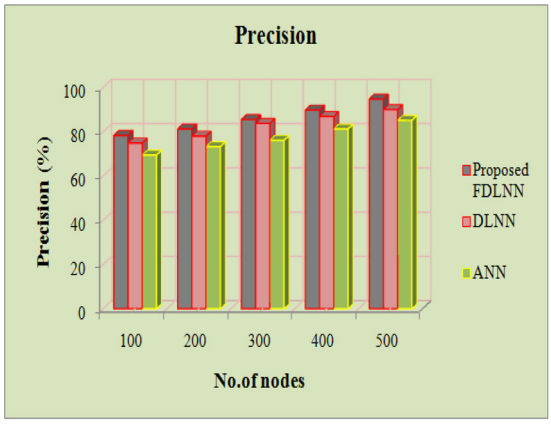
Table 2
Demonstrate the performance of the proposed FDLNN with the existing DLNN and ANN in terms of (a) precision, recall, and (b) F-Measure, Accuracy.

(a)

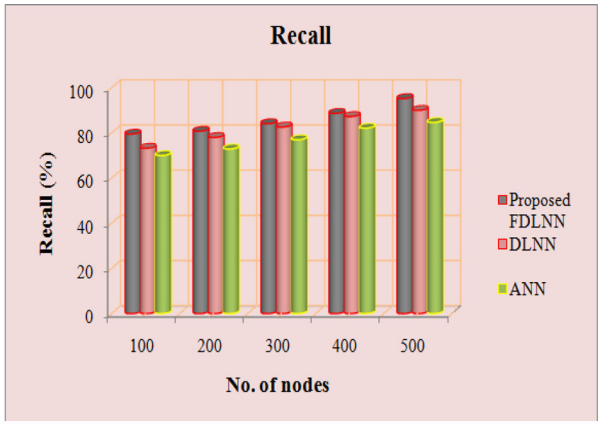
Number of nodes	Proposed FDLNN		DLNN		ANN	
	Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
100	78.34	79.50	74.89	73.23	69.45	70.00
200	81.23	80.78	78.01	78.01	73.24	72.89
300	85.53	84.12	83.78	82.65	76.12	76.98
400	89.78	88.67	86.90	87.29	81.23	82.12
500	94.75	95.23	89.99	90.03	85.14	84.67

(b)

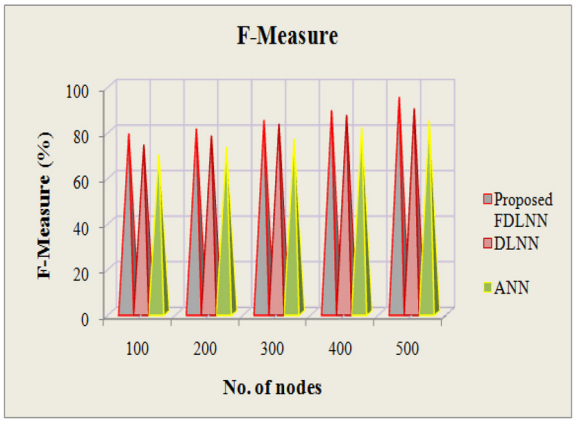
Number of nodes	Proposed FDLNN		DLNN		ANN	
	F-Measure	*Accuracy*	*F-Measure*	*Accuracy*	*F-Measure*	*Accuracy*
100	78.91	79.12	74.05	72.90	69.72	70.05
200	81.00	81.67	78.01	76.83	73.06	73.87
300	84.81	86.78	83.21	81.23	76.54	77.23
400	89.22	89.90	87.09	85.75	81.67	82.18
500	94.98	96.12	90.00	88.23	84.90	84.87



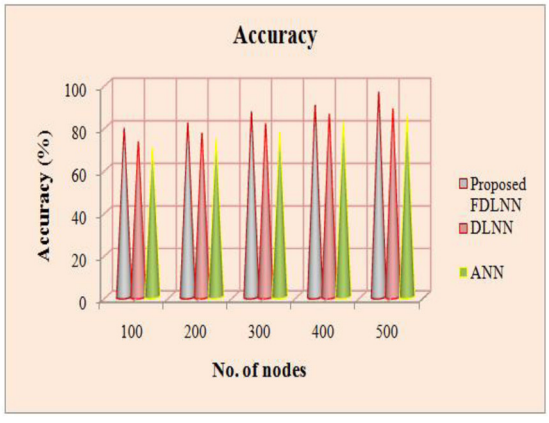
(a) precision



(b) recall



(c) F-measure



(d) accuracy

Fig. 5. Comparative analysis of IDS using different algorithms based on (a) precision, (b) recall, (c) F-measure, and (d) accuracy.

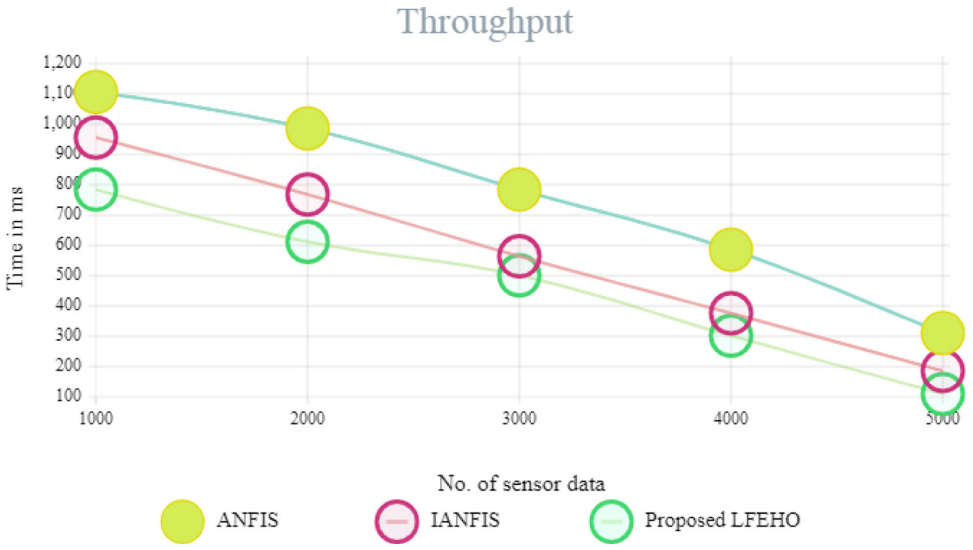


Fig. 6. Throughput analysis.

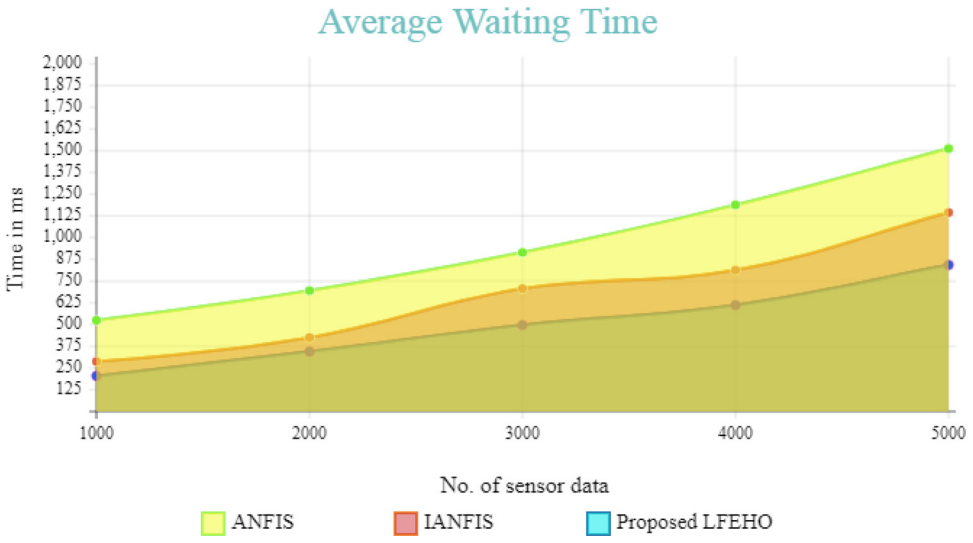


Fig. 7. AWT analysis of data broker algorithms.

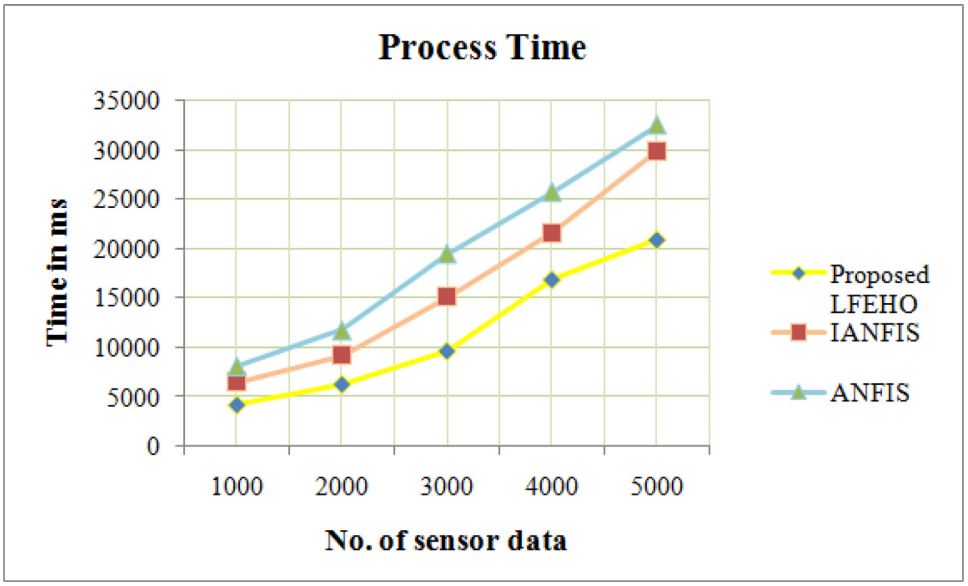


Fig. 8. Performance analysis of data broker algorithms based on the process time.

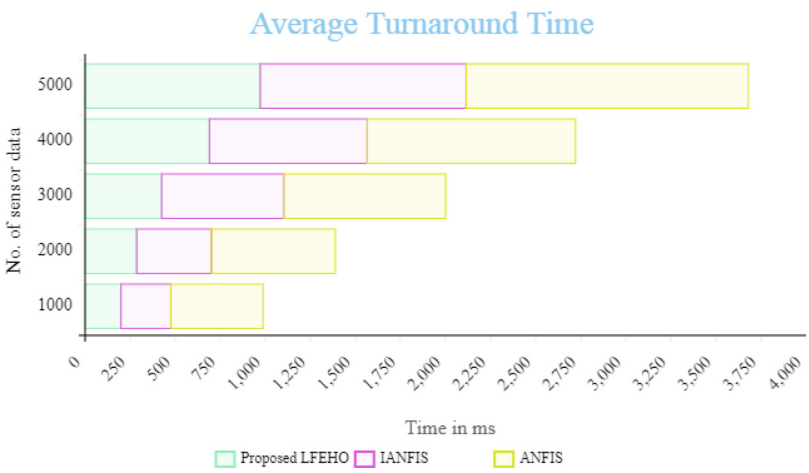


Fig. 9. ATT analysis.

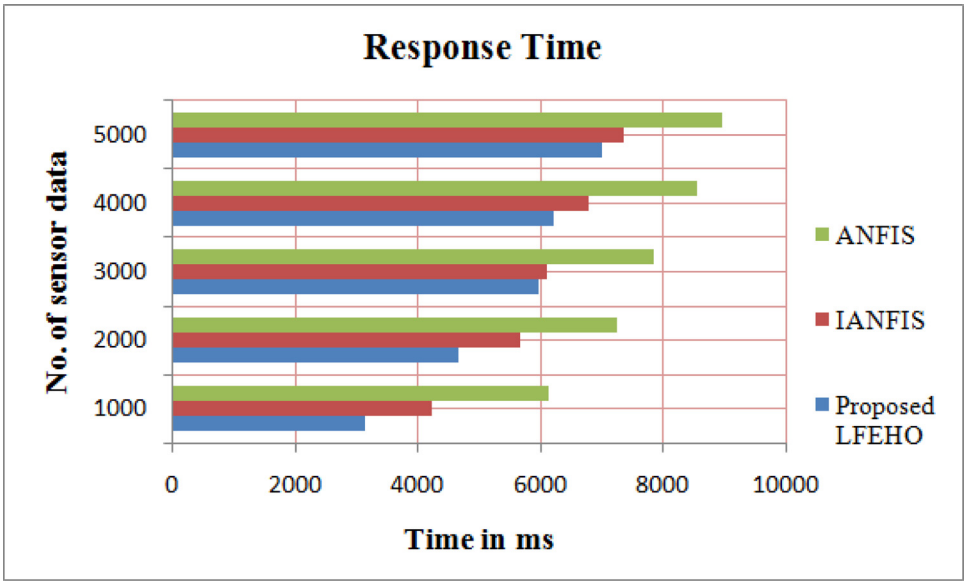


Fig. 10. Compared the performance of the LFEHO with the IANFIS and ANFIS based on the response time metric.

Discussion: Fig. 6 demonstrates the relative examination of the proposed data allocation technique (LFEHO) with the prevailing IANFIS along with ANFIS centred upon the throughput metrics. For the performance examination, 1000 to 5000 sensed normal data is engaged. Throughput is a crucial gauge for the examination of system reliability. When the data count is 4000, the proposed LFEHO has 302ms along with the IANFIS along with ANFIS encompass 376ms along with 586ms throughput time. For the remaining data count, say 1000, 2000, 3000, along with 5000, the proposed data broker algorithm have 784ms, 612ms, 501ms, along with 110ms respectively. The prevailing data broker algorithms have high throughput time than the proposed data broker algorithm. Therefore, the proposed LFEHO centred data allocation is found to render improved performance than the prevailing algorithms.

Discussion: Fig. 7 contrasted the proposed LFEHO's performance with the prevailing IANFIS along with ANFIS centred on the AWT. AWT is the time that the system waits to apportion the resources into the distributed CS. The best system takes lesser waiting time to assign the resources into the CS. Likewise, the LFEHO obtains lesser waiting time that is 205ms, 345ms, 498ms, 613ms, along with 842ms for 1000, 2000, 3000, 4000, along with 5000 counts of data correspondingly. The proposed LFEHO is weighed against the prevailing algorithms. The IANFIS is improved than the ANFIS, but, the IANFIS as well takes an extended time than the proposed LFEHO for every data count. Therefore, it concludes that the proposed LFEHO offers improved performance weighted against the prevailing algorithms.

Discussion: Fig. 8 illustrates the proposed LFEHO's performance with that of the existing IANFIS along with ANFIS centred on the process time metric. The processing time indicates the total time engaged to finish the entire process. Now, for the 1000 data count, the LFEHO obtains 4213ms time to finish the entire process, which is low when contrasted with the IANFIS along with ANFIS. For the remaining data counts, the proposed obtains lesser time to finish the entire process. The description of the procedure time identifies that the proposed LFEHO renders superior performance to the IANFIS along with ANFIS.

Discussion: Fig. 9 illustrates the ATT examination of the proposed along with the prevailing data broker. Turnaround time identifies the grouping of the waiting time along with the execution time. In the preceding examination, the waiting time along with process time is improved for the LFEHO, so, this turnaround time will be low for the LFEHO. For instance, when the data count was 5000, the LFEHO obtains 972ms turnaround time but the prevailing IANFIS along with ANFIS contains 1142ms along with 1566ms turnaround time. The IANFIS is improved compared to the ANFIS but it also obtains an extended turnaround time than the proposed LFEHO. At last, the description shows that the LFEHO achieves improved performance than the prevailing algorithms.

Discussion: Fig. 10 contrasted the response time examination of the LFEHO with the prevailing IANFIS along with ANFIS. The response time illustrates how much time engaged to react to the sensor data. At this time, the performance is diverse centred upon the data count. The prevailing IANFIS obtains 5655ms response time for 2000 data. For the same data count, the proposed obtains 4670ms response time as well as the prevailing ANFIS obtains 7255ms response time. As of this, the LFEHO obtains lesser time, which takes the 1st place, along with the IANFIS takes the 2nd place, and finally, ANFIS holds 3rd place. Likewise, for the balance data counts, the LFEHO obtains lesser time. Therefore, it deduces that LFEHO centred data allocation offers an improved outcome than the prevailing algorithms.

5. Conclusion

DC and effective cloud storage is a challenging one. Many existing works are putting their effort to give a better system. This paper presented a DC and effective distributed cloud storage system centered on the IDS using the FDLNN algorithm. For IDS, the proposed system uses the FDLNN algorithm, and for the data broker, uses the LFEHO algorithm, and for DC, the MQTT and AQMP connectivity are used. The MQTT is best for lightweight devices, and the AQMP is best for the heavyweight devices. For the IDS analysis, the TON_IoT datasets is taken and the performance of the FDLNN is compared with the existing DLNN and ANN algorithms based on the precision, recall, F-Measure, and accuracy. Then, the performance of the data broker is analyzed, for which the proposed LFEHO is compared with the existing IANFIS and ANFIS algorithm based on average turnaround time, throughput, process time, response time as well as AWT. The IDS performance is analyzed based on SNs count and the data broker is analyzed grounded on the number of data. ***For 500 nodes, the FDLNN has 96.12% accuracy***. In data broker analysis for all data count, the proposed LFEHO attained better performance. Hence, the proposed system is found to attain excellent performance on considering the existing algorithms.

In the future, the proposed system could be enhanced by using advanced algorithms and the advanced connectivity protocol, which supports all the devices.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This study was not funded by any grant.

Human and animal rights

This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent

Informed consent was not required as no humans or animals were involved.

References

[1] E. Daniel, N.A. Vasanthi, LDAP: a lightweight deduplication and auditing protocol for secure data storage in cloud environment, Cluster Comput. 22 (1) (2019) 1247–1258.
[2] B. Ramachandran, K. Subramaniam, Secure and efficient data forwarding in untrusted cloud environment, Cluster Comput. 22 (2) (2019) 3727–3735.
[3] R. Di Pietro, M. Scarpa, M. Giacobbe, A. Puliafito, Secure Storage as a Service in Multi-Cloud Environment, Lect. Notes Comput. Sci. (2017) 328–341.
[4] A. Ari, A. Adamou, Asside Christian Djedouboum, Abdelhak mourad gueroui, alidou mohamadou, and zibouda aliouat, "big data collection in large-scale wireless sensor networks, Sensors 18 (12) (2018) 4474.
[5] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, A OneM2M Intrusion detection and prevention system based on edge machine learning, IEEE/IFIP Network Operations and Management Symposium, NOMS, April 2020.
[6] M. Zolanvari, M.A. Teixeira, L. Gupta, K.M. Khan, R. Jain, Machine Learning-Based Network vulnerability analysis of industrial internet of things, IEEE Internet Things J. 6 (4) (2019) 6822–6834.
[7] M. Martí, C. Garcia-Rubio, C. Campo, Performance evaluation of CoAP and MQTT_SN in an IoT environment, Multidiscip. Digit. Publishing Instit. Proc. 31 (1) (2019) 49.
[8] V. Alaparthy, S.D. Morgera, Modeling an intrusion detection system based on adaptive immunology, Int. J.Interdiscip. Telecommun. Netw. 11 (2019) 42–55.
[9] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, P. Faruki, Network intrusion detection for IoT security based on learning techniques, IEEE Commun. Surv. Tut. (2019).
[10] S.A. Alabady, F. Al-Turjman, Low complexity parity check code for futuristic wireless networks applications, IEEE Access 6 (April 2018) 18398–18407.
[11] M. Sarnovsky, J. Paralic, Hierarchical intrusion detection using machine learning and knowledge model, Symmetry 12 (2) (2020) 203.
[12] B.M. Pampapathi, M. Nageswara Guptha, M.S. Hema, Data distribution and secure data transmission using IANFIS and MECC in IoT, J. Ambient Intell. Human Comput (2021), doi:10.1007/s12652-020-02792-4.
[13] F. Siddiqui, J. Beley, S. Zeadally, G Braught, Secure and lightweight communication in heterogeneous IoT environments, Internet Things (2019).
[14] A.A. Khanan Mehmood, M.M. Umar, S. Abdullah, K.A.Z. Ariffin, H Song, Secure knowledge and cluster-based Intrusion Detection Mechanism for smart wireless sensor networks, IEEE Access 6 (2018) 5688–5694.
[15] S. Raja, S. Ramaiah, An efficient fuzzy-based hybrid system to cloud Intrusion Detection, Int. J. Fuzzy Syst. 19 (1) (2017) 62–77.
[16] R. Patil, H. Dudeja, C Modi, Designing an efficient security framework for detecting Intrusions in virtual network of cloud computing, Comput. Secur. (2019).
[17] J. Park, H. Kwon, N. Kang, IoT–Cloud collaboration to establish a secure connection for lightweight devices, Wireless Netw. 23 (3) (2017) 681–692.
[18] V.L.L. Thing, IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach, in: 2017 IEEE Wireless Communications and Networking Conference, WCNC, San Francisco, CA, 2017, pp. 1–6, doi:10.1109/WCNC.2017.7925567.
[19] S. Ding, G. Wang, Research on intrusion detection technology based on deep learning, in: 2017 3rd IEEE International Conference on Computer and Communications, ICCC, Chengdu, 2017, pp. 1474–1478, doi:10.1109/CompComm.2017.8322786.
[20] Mayank Agarwal, Dileep Pasumarthi, Santosh Biswas & Sukumar Nandi(2016). "Machine learning approach for detection of flooding DoS attacks in 802.11 networks and attacker localization". Int. J. Mach. Learn. Cyber. 7, 1035–1051. doi:10.1007/s13042-014-0309-2.
[21] I. Ahmad, M. Basheri, M.J. Iqbal, A. Rahim, Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection, IEEE Access 6 (2018) 33789–33795, doi:10.1109/ACCESS.2018.2841987.
[22] B. Trstenjak, D. SasaMikac, KNN with TF-IDF based framework for text categorization, Procedia Eng. Volume 69 (2014) 1356–1364, doi:10.1016/j.proeng.2014.03.129.
[23] S. Tan, An effective refinement strategy for KNN text classifier, Expert Syst. Appl. Volume 30 (Issue 2) (2006) 290–298 PagesISSN 0957-4174, doi:10.1016/j.eswa.2005.07.019.
[24] M.O. Mughal, S. Kim, Signal Classification and Jamming Detection in Wide-Band Radios Using Naïve Bayes Classifier, IEEE Commun. Lett. 22 (7) (2018) 1398–1401, doi:10.1109/LCOMM.2018.2830769.
[25] M. Weiyang, C.L. Gutterman, Y. Li, Z. Shengxiang, G. Zussman, D.C. Kilper, Deep-neural-network-based wavelength selection and switching in ROADM systems, J. Opt. Commun. Netw. Vol. 10 (10) (2018) D1–D11, doi:10.1364/JOCN.10.0000D1.
[26] F. Farahnakian, J. Heikkonen, A deep auto-encoder based approach for intrusion detection system, in: 2018 20th International Conference on Advanced Communication Technology, ICACT, Korea (South), 2018, pp. 178–183, doi:10.23919/ICACT.2018.8323688. Chuncheon-si Gangwon-do.
[27] R. Garreta, G. Moncecchi, Learning Scikit-Learn: Machine Learningin Python, Packt Publishing Ltd, Birmingham, U.K., 2013.
[28] Z. Gao, Y. Xu, F. Meng, F. Qi and Z. Lin(2014), "Improved information gain-based feature selection for text categorization," *2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace &Electronic Systems (VITAE)*, Aalborg, pp. 1-5, doi:10.1109/VITAE.2014.6934421.
[29] C.E. Shannon, A mathematical theory of communication, SIGMOBILE Mob. Comput. Commun. Rev. 5 (January 2001) (2001) 3–55 1, doi:10.1145/584091.584093.
[30] Z. Haomiao, D. Zhihong, X. Yuanqing, F. Mengyin, A new sampling method in particle filter based on Pearson correlation coefficient, Neurocomputing Volume 216 (2016) 208–215 2016, doi:10.1016/j.neucom.2016.07.036.
[31] H. Chiroma, A.Y. Gital, N. Rana, M. Abdulhamid Shafi'i, A.N. Muhammad, A.Y. Umar, A.I. Abubakar, Nature inspired meta-heuristic algorithms for deep learning: Recent progress and novel perspective, in: Science and Information Conference, Springer, Cham, 2019, pp. 59–70.
[32] J.M. Johnson, T.M. Khoshgoftaar, Survey on deep learning with class imbalance, J. Big Data 6 (2019) 27, doi:10.1186/s40537-019-0192-5.