Peer-to-Peer Networking and Applications (2021) 14:2554–2564
https://doi.org/10.1007/s12083-020-01014-0

A machine learning-assisted data aggregation and offloading system for cloud–IoT communication

Osama Alfarraj[1]

Received: 22 August 2020 / Accepted: 7 October 2020 / Published online: 25 October 2020

Abstract

Data aggregation and dissemination in cloud-based internet of things (IoT) are main issues because of interoperability problems in communication. In an IoT environment, data handling and offloading are constant processes that avoid communication failure and increase service utilization levels. This paper introduces a machine learning (ML)-assisted data aggregation and offloading (ML-DAO) system to improve the reliability of cloud–IoT communication. The method introduced helps reduce the response time and routing cost errors in data aggregation and improve the data service rate. The data handling rate is also enhanced using the IoT assisted by fog elements that maximize edge-level communication. Cloud–IoT communication quality is measured on the basis of time and service attributes; ML techniques are designed to enhance the level's precision while aggregating the data. To achieve optimum communication quality, the proposed ML-DAO operates on certain measurable functional metrics. The performance of the system is assessed using the following metrics: route cost error, processing time, aggregation delay, service utilization rate, failure probability, and response time. Experimental results prove the consistency of the proposed scheme, as the metrics are optimized with lesser unallocated data chunks.

Keywords Data aggregation · Data offloading · IoT · Fog computing · Machine learning

1 Introduction

The internet of things (IoT) is a recent development in communication technology that facilitates smart communication between users and devices/machines. The IoT encompasses real-world devices and communication technologies with a virtual representation to meet end users' application needs. The devices, called *things*, act in a smart manner through human–machine interactions. In IoT network user equipment is the building block, as it possesses sensing, communicating, and processing abilities [1, 2]. The IoT handles heterogeneous information by adopting a wide range of applications and services from different platforms to provide effective user-level communication. The virtual representation feature of the network improves flexibility and scalability in application and resource sharing. Information from multiple sources is accumulated, stored, and processed as digital information to enable machine–machine and machine–human interactions. Because of the smart communication and decision-making features of things or devices, the IoT is used in environment, habitat monitoring, and transport-assisting services, social networks, the healthcare industry, and commercial and residential applications [3, 4]. The IoT connects multiple heterogeneous sources and storage in a distributed manner with the support of internet technology. The network handles volumes of information through distributed access for which multi-level optimization is necessary in order to provide reliable end user communication. Flexibility and ease of access features are achieved by integrating distributed cloud services to improve IoT reliability and cope with increasing user demands. Storage, connectivity, information access, and retrieval operations are shared between the IoT and cloud architectures. The advantages of the IoT and cloud are harmonized to aid a wide range of the above-mentioned applications [5, 6].

With the growing demand for user applications, communication technologies play a vital role in computing services. Fog computing is a communication-supporting paradigm that extends cloud applications and services to the edge of the network. Users beneath the network edge utilize the

This article is part of the Topical Collection: *Special Issue on Network In Box, Architecture, Networking and Applications*
Guest Editor: Ching-Hsien Hsu

✉ Osama Alfarraj
oalfarraj@ksu.edu.sa

[1] Computer Science Department, Community College, King Saud University, 11437 Riyadh, Saudi Arabia

applications, services, and platform of cloud services in an interoperable manner [7, 8]. A fog layer is commonly placed between the cloud and the user plane to ease computing, access, and service sharing prospects [9, 10]. Fog is a decentralized computing paradigm designed to avoid latency issues in cloud resource allocation and sharing. Similar to the cloud layer, the fog layer consists of dedicated servers and gateways to accept user requests and process them. Devices such as routers, switches, servers, and access points with smart computing ability are the elements of the fog layer. Fog elements perform the same tasks as the cloud from the network edge [11, 12]. The integration of multi-level computing and service paradigms improves the reliability of the communication system. Nevertheless, processing information and addressing complex problems remain key challenges in achieving relevant optimization. Processing and decision-making algorithms and techniques are necessary to improve the rate of optimization [13].

Machine learning (ML) is an efficient approach to analyze and extract information from complex processing systems. ML is applied in the IoT to process sensor data accumulated from the deployed environment [14, 15]. The learning process is incorporated into the IoT–fog–cloud architecture because of its multi-layer perspective in solving complex problems [16]. The operational constraints of fog elements require some external processing support, such as optimization algorithms, learning, and decision-making systems, to achieve high reliability. Latency reduction, traffic classification, and raw data processing are some advantages of incorporating ML into the IoT architecture. The smart computing ability of IoT devices is improved through external optimization and decision-making systems. ML is applied in the medical, industrial, automation, and environment assessment fields [17, 18]. The aim of integrating heterogeneous communication technologies is to provide seamless communication and achieve improved user satisfactory levels. The process of data handling and distribution in heterogeneous networks is a complex task. More specifically, data aggregation and offloading computed tasks for the equivalent amount of data in the network are complex. The volume of data circulated in the communication network cannot be predicted because of varying devices and resource densities [19, 20]. Improper data handling minimizes the quality of communication by increasing congestion and pending tasks, thus increasing service response lags. Therefore, the methods designed to optimize the data handling process must ensure congruency with offloading and the processing capacity of heterogeneous network integration. The main contributions of this paper are as follows:

(i) Designing a learning-based aggregation scheme to achieve the optimal route cost in order to meet the increasing user demands in the fog layer. Addressing the optimal route cost constraint improves the reliability of the communication system by achieving errorless service utilization. Based on the output of the learning process, the aggregation scheme minimizes the error in the route cost to achieve a high rate of service utility.

(ii) Designing a learning-assisted data offloading scheme to minimize the response time to IoT user requests. This scheme provides two benefits: minimizing route cost errors and minimizing failure in service requests.

(iii) Conducting a comparative analysis of the proposed ML-DAO method with existing research approaches to verify the introduced method consistency using different evaluation metrics.

The rest of the paper is organized as follows. Section 2 discusses different studies on the data aggregation process. Section 3 analyzes the ML-assisted data aggregation and offloading (ML-DAO) scheme. Section 4 examines the efficiency of the system and concludes the paper.

2 Related works

Kayes et al. [21] presented a formal context-aware role-based access for IoT users to handle functionalities in critical situations. The functions for users are derived from the contextual roles of the IoT. The dynamicity in handling IoT roles is modeled through an ontology approach to gain control over access policies. Control policies and user roles are induced to handle information and control the offloading process in the IoT. This method is more feasible in handling heterogeneous data and models of its own to adapt to derived user-level operations. Kim et al. [22] designed an IoT broker architecture to facilitate different protocols' interoperability. The authors extended their contribution by designing an interoperable service platform to support IoT applications and services without constraints. The designed architecture prototype supports multiple intelligent services with basic IoT operations. However, security requirements cannot be met by the proposed architecture.

Ullah et al. [23] proposed a student interaction model for a smart city environment. The interaction model is supported by software-defined networking (SDN)–IoT to achieve better interoperability and scalability features. A latent semantic analysis model is incorporated into the SDN to identify the similarity of the interaction text between teachers and students. The model improves the rate of information analysis and retrieval. Puschmann et al. [24] introduced a clustering scheme to balance IoT information dissemination. This scheme protects against bandwidth exploitation by minimizing transmission loss for unplanned user traffic with real-time traffic correlation. Resources, service providers, and IoT controllers are synchronized to achieve better traffic handling. Because of allocation and resource awareness problems, however, this integration becomes complex.

Xiao et al. [25] proposed an optimization technique to minimize management cost and delays in data center communication. In this optimization technique, bandwidth exploitation, storage utilization, and service migration are accounted for to improve communication reliability. The controlled cloud, which interacts with the data center, uses scheduling and admission control policies to support service migration. Lu et al. [26] proposed an IoT–cloud architecture with the support of edge computing to resolve the issues in big data analysis. In this process, the IoT with data-oriented map (IoT DeM) reduction is used in the managed clouds. This architecture model is designed to predict the performance of edge clouds distributed across the environment. The performance of these clouds is assessed by evaluating the job execution rate of the nodes. The locally weighted linear regression technique used predicts the job execution time of each edge node to analyze big data. This model achieves lesser relative errors with controlled time delay.

The fuzzy c-means approach has been introduced by Bu [27] to handle and analyze big data in the IoT environment. The data are classified into clusters using the canonical decomposition method, in which limited attributes of the data are evaluated to improve the efficiency of analysis. The attributes of the data are compressed using a bijection function to meet the computing requirements of IoT devices. The method introduced maximizes clustering efficiency and minimizes execution time. Cheng et al. [28] proposed a fogflow framework to improve the openness and interoperability of fog in a smart city environment. The programming model of fogflow allows a flexible design of services that are easily available in cloud and edge architectures. Function and data reusability are the other advantages of fogflow, minimizing storage and computational complexity in a smart city environment.

Yacchirema et al. [29] extended the concept of big data analytics in the IoT to treat obstructive sleep apnea (OSA) disorder in the field of medicine. A novel smart city application-oriented system is designed to treat OSA by monitoring and reporting patient information. In this IoT system, fog-assisted notification and behavior-based predictive analysis are incorporated to detect and report the emergency conditions of the end user. This system operates in a technical, syntactic, and semantic manner to provide interoperability. Edge node-assisted data transmission in a cloud-centric IoT architecture has been designed by Zhao et al. [30] to address cloud bandwidth exploitation. Initially, the architecture gains knowledge of the bandwidth requirement of the edge nodes and then assigns IoT data. IoT data are replicated for the optimal number of bandwidths, in which the maximum number of requests is processed. Excess user requests are met by the central cloud by extending the support of edge nodes.

He et al. [31] proposed a big data analytics model based on fog for smart city environments. Data analytics is facilitated through a multi-tier model comprising on-demand and dedicated fogs (D-fog). Both fog models are opportunistic; D-fog differs from on-demand fog by mitigating delayed cloud response. It supports a wide range of computing engines with a distributed cloud–fog environment where a large scale of IoT users is available. The introduced method improves overall performance, maximizes service utilities and data analytics services, and enhances the quality of services.

From the above survey, the process of data handling in the IoT is retarded when some quality metrics are compromised, such as service utilization [26], response time [27], and communication errors [26, 31]. Some conventional drawbacks in [23, 25, 29], such as request density and rate of handling, have been optimized in [27, 31]. The notified features lag because of complex processing and centralized decision making which leads to creating the communication errors across the networks. Integrating heterogeneous networks in such cases results in congestion at the service level. To address these issues, this paper proposed a data aggregation and offloading scheme based on learning.

3 Machine learning-assisted data aggregation and offloading scheme

3.1 IoT sensor–fog–cloud data processing architecture

Figure 1 illustrates the architecture of the IoT sensor–cloud. The data processing architecture operates between the sensor fog and fog end user layers.

The architecture and its components are briefly discussed as follows. The architecture consists of three layers: the consumer layer, the fog layer, and the cloud layer.

Consumer layer The consumer layer comprises IoT sensors and end user applications distributed across the globe. IoT sensors are deployed in the agricultural, environmental, residential, and healthcare industries. These sensors are equipped with radio and sensing units to relay sensed information. The sensed information includes temperature, humidity, location, alerts, and health information that are depending on the application sensors. A region aggregator (RA) is responsible for integrating the information received from the sensor. The RA interacts with the fog layer to access cloud layers' storage and processing. The RA present here is different from a border routing device. This device is responsible for handling data traffic to reduce congestion alongside route discovery. It does not solve the routing problems between the devices; rather, it takes the dual responsibilities of low-cost route discovery and congestion prevention.

End user applications are either commercial or non-commercial, requiring sensed information. Sensed information is provided to the end user for further processing, on request. Generally, end user applications raise a request to the cloud to access and retrieve stored information. With the introduction of the fog layer, delayed data retrieval and access are minimized.

Fog layer The fog layer comprises nodes, temporal storage, and gateways. This layer holds a set of instances and service categories for the request received. The fog nodes process the requests of the end user application to offload data from the cloud. The storage pre-fetches information from the cloud to improve the rate of service.

Cloud layer The architecture is modeled with conventional cloud components. These components include a dedicated server, storage, third-party applications, and cloud service. The cloud interconnects multiple networks with a wide A The roles and responsibilities of the elements in the architecture were designed based on existing network models. The above architecture integrates different heterogeneous networks to improve the flexibility of user communication. User communication adopts different technologies for information exchange; the presence of different scales of networks provides ease of information exchange depending on the region of communication.

3.2 ML-DAO methodology

This method performs two types of operations: data aggregation and offloading. Data aggregation improves reliability in communication between the IoT and the fog layer at the time of sensor information accumulation. The RA binds the collected information as a single entity and forwards it to the fog layer. The independent fog elements are responsible for offloading the received information to the end users.

3.2.1 Data aggregation

Data aggregation in the IoT layer is facilitated by the RA. The sensors are distributed in a random manner. Data aggregation is the process of accumulating information from multiple sources at the same time or based on time slots. The accumulated information is stored and transmitted for the users demanding them through requests. The information is collected

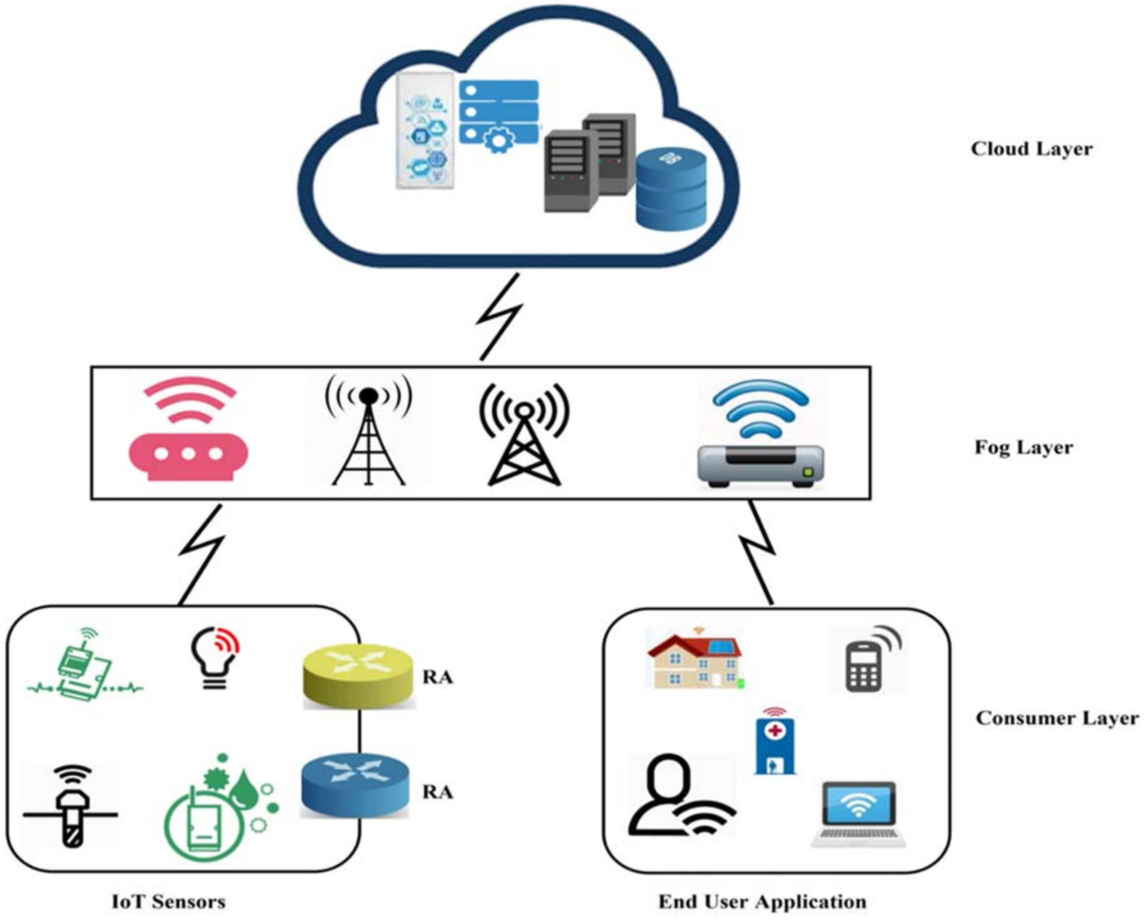

Fig. 1 IoT sensor–fog–cloud architecture

from heterogeneous devices from different environments, such as sensors and cloud resources. The intermediate communicating devices are responsible for handling the accumulated information. Depending on the position of the IoT sensors, RAs accumulate information in either a single hop or multi-hop. Constructing an aggregation tree increases the cost of operation, as it cannot be changed frequently for mobile IoT sensors. Similarly, the number of sensing devices that contribute to the aggregation process varies at different time intervals.

The data aggregation routes between IoT sensors and the fog layer are not permanent. The number of routes for relaying sensed information must be low to achieve lesser route costs. The aggregation process also covers the maximum number of achieved sensors in this interval. Let ρs_i represent the probability of the i^{th} sensor to transmit sensed information in the aggregation time interval t_a; the route cost (r_c) is estimated as

$$r_c = 1 - \prod_{i \in s}^{t_a} (1 - \rho s_i), \tag{1}$$

where s represents the set of IoT sensors participating in t_a.

The possibility of $min\{r_c\}$ achieving profitable aggregation is low because of different levels of aggregation (a_l). The aggregation level relies on indirect sensors communicating with the RA.

In Fig. 2, represented that the aggregation level which has several levels and sensor nodes. j is the total number of levels in t_a for s_a active sensor nodes' optimal route cost, which is estimated using Eq. (2),

$$\mathrm{r}_\mathrm{c}^* = \frac{\mathrm{N_r}}{\mathrm{R}} * \alpha, \tag{2}$$

where N_r is the network radius, and R is the communication range of RA. α is a variable computed as $\frac{s_a}{s_i}, s_i \in a_l$.

The route cost metric is defined using connectivity and the RA range. The minimum route cost is achieved by selecting neighbors with the maximum connectivity, and it ensures covering a reliable number of neighbors. The optimal route cost is made to vary with the number of s_a in eacht_a. With the help of an adaptive neural network, the optimality of the route cost is assessed for the available s_a. Let X be the set of inputs for the neural learning process; here, r_c^* is the output estimated for the s_i participating in aggregation. The hidden layer output determines the number of s_a for which r_c^* is estimated. Unlike conventional neural learning, this process is evaluated as an iterative procedure. The process of neural learning is represented in Fig. 3.

The hidden layer output $\{h_1, h_2 \ldots \ldots h_\mathrm{n}\} \in \mathrm{H}$ generates the number of s_a from s_i. Therefore, the error e in each layer is estimated as the difference in route cost (i.e., the route cost estimated for s_i during t_a), and the actual route cost for s_a is considered to identify the error. The partial derivation of neural learning is represented as

$$\left.\begin{array}{r} h_1 = \left[\left(r_{c1}^*\right)^{h_0 x_0} + \left(r_{c1}^*\right)^{{h_0}^2} h_0 (a_l - 1) + e_1 \right] \\ h_2 = \left[\left(r_{c2}^*\right)^{h_1 x_1} + \left(r_{c2}^*\right)^{{h_1}^2} h_1 (a_l - 2) + e_2 \right] \\ \vdots \\ h_n = \left[\left(r_{cn}^*\right)^{h_{n-1} x_{n-1}} + \left(r_{cn}^*\right)^{{h_{n-1}}^2} h_{n-1} \left(a_j\right) + e_n \right] \end{array}\right\}, \tag{3}$$

where $\{e_1, e_2 \ldots .. e_\mathrm{n}\}$ and

$$e = r_c \sim r_c^*. \tag{4}$$

The process of the hidden layer is induced to find the errors in processingr_c^* for each iterate. The error is regarded as the difference between the computed and the observed route costs in order to find the actual error that occurs after communication. Based on this error, the next set of hidden layer processing is optimized.

There are two possibilities for the hidden layer output based on Eq. (4).

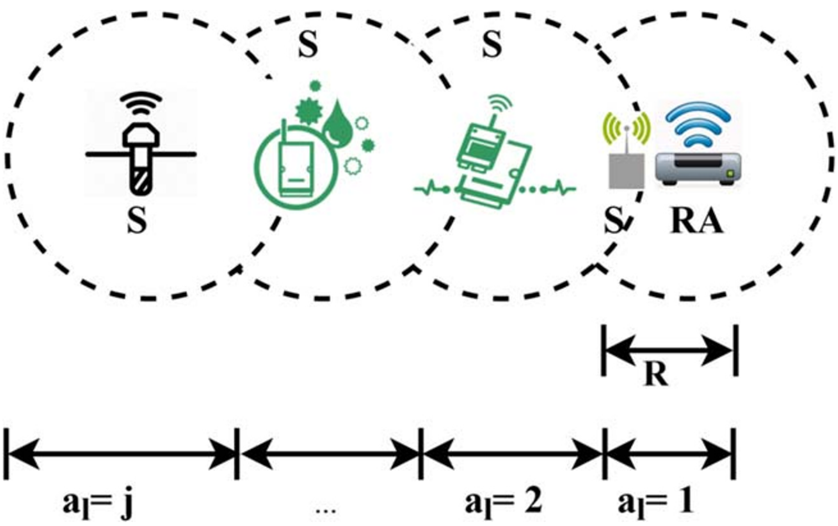

Fig. 2 Aggregation level representation

Possibility 1 e is positive.

Analysis 1 If$\mathrm{r_c} > \mathrm{r_c^*}$, then the error is positive. Therefore, in the next iterate, the aggregation routes are constructed to satisfy. This ensures maximum sensor coverage and data collection in single and multi-hop. From (4), as the same $\mathrm{r_c^*}$ is maintained,

$$h_2 = \left.\begin{matrix}\left[\left(r_{c1}^*\right)^{h_1 x_1} + \left(r_{c1}^*\right)^{{h_1}^2}.h_1\Big((a_l{-}2) + e_2\Big]\\ \vdots\end{matrix}\right\}. \quad (5)$$

Equation (5) is the hidden layer output until $\mathrm{r_c} < \mathrm{r_c^*}$.

Possibility 2 e is negative.

Analysis 2 e is negative if $\mathrm{r_c} < \mathrm{r_c^*}$. In this case, the estimated route for increases the actual route cost. This is due to sensor unavailability or route failure, as the IoT sensor is mobile. Therefore, the number of observed in the first iterate is minimized in the next iterate until $\mathrm{r_c} > \mathrm{r_c^*}$ or $\mathrm{r_c} = \mathrm{r_c^*}$.

The route cost function operates in a cooperative manner with conventional routing protocols to improve route selection. The conventional routing protocols rely on the distance metric for path selection. The outputs of the learning process are utilized by the routing protocols to select better neighbors for routing.

3.2.2 Offloading process

In the offloading process, the relevant information is delivered to the requested users. Similarly, the processes exceeding the capacity of the devices are shared across the neighbors for parallel computations. This minimizes overloading of the device by reducing data and task congestion rate with less delay. The advantage of ML is further extended to the interactions between the fog layer and end users in the consumer layer. For the observed s_a accumulated data, timely offloading is essential to prevent data unallocation and minimize retrieval time. A congested data request increases the probability of failure because of the varying densities of end users and timed-out service replies.

Fog elements have a limit of accepting and processing end user requests. Fog elements assign the sensed information to the requests from the consumer layer. The sensed information is fetched from the cloud by the fog elements to minimize the request waiting time. Let P_{ij} be the probability for an end user device's request being mapped to the j^{th} fog element such that

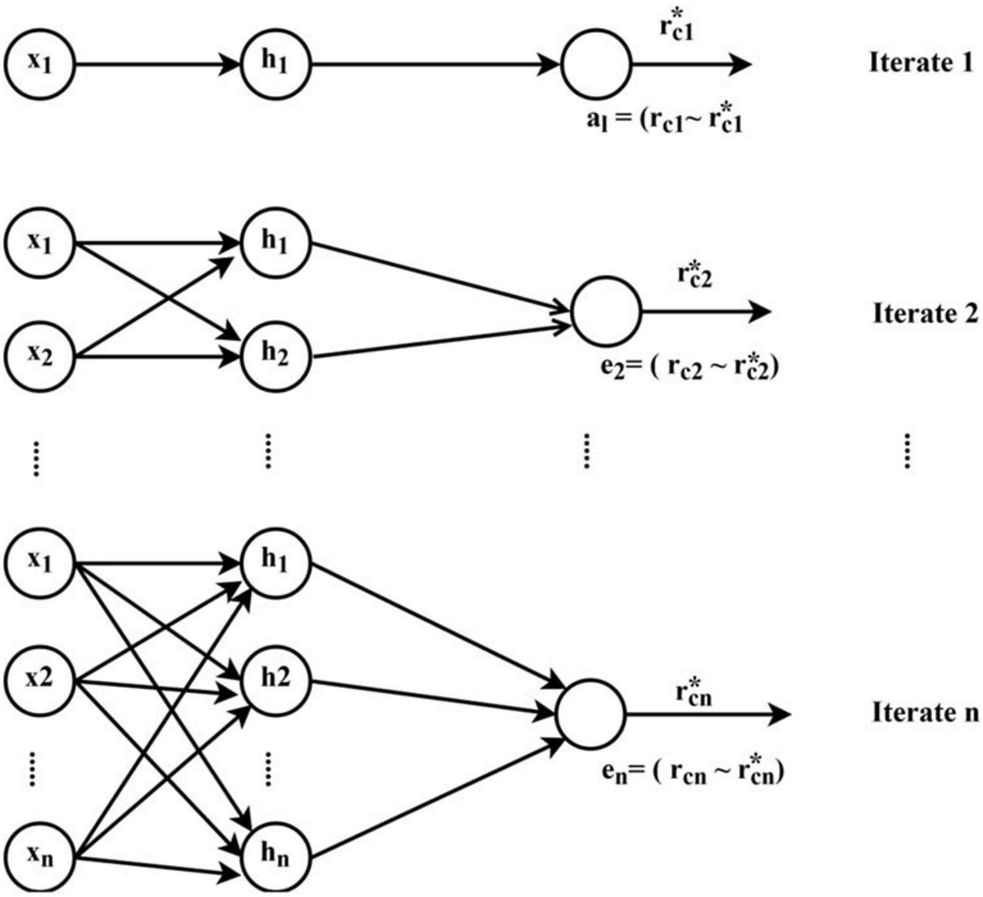

Fig. 3 Learning process

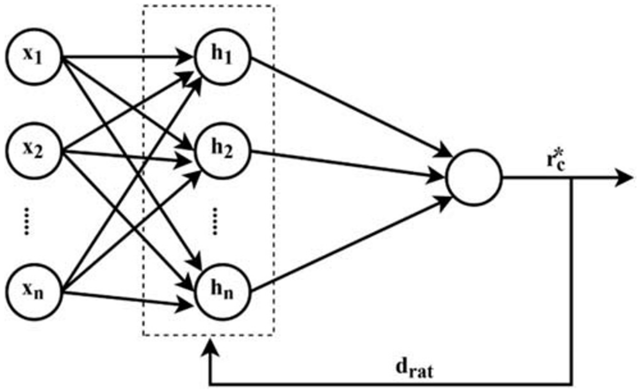

Fig. 4 Learning refined by considering $\mathbf{d_{rat}}$

$$\left.\begin{array}{r} P_{ij} = 1 \ if \ request \ i \ is \ mapped \ j \\ P_{ij} = 0 \ \ otherwise \end{array}\right\} \tag{6}$$

As the request space of the fog element is limited, the optimality of ensuring $P_{ij} = 1$ is verified with the request arrival rate (ra_r). The request arrival rate is computed using Eq. (7),

$$ra_r = r_n * t_r \tag{7}$$

where r_n is the number of requests transmitted at time t_r.

Let c_j represent the capacity of the j^{th} fog element; then, if $ra_r > c_j$, the fog element offloads requests to its neighbor. The offloading of user requests increases the waiting time rather than dropping. The rate of aggregation is irrespective of the number of end users and requests. To improve the consistency of response and the service utility rate, downloading or data access success is estimated. Let d_p denote the download probability of the end user, which is estimated using Eq. (8):

$$d_p = 1 - \prod_{i \in t_a} e^{-ra_{ri}} \tag{8}$$

Table 1 Configuration parameters and values

Parameter	Value		
	IoT Node	Fog Node	Server/ Storage
Number of Devices	520	100	4/1
RA	30	-	-
Clock Speed	666 MHz	1,300 MHz	2.4 GHz
Physical Memory	512 Mb	2 Gb	4 Gb
Request/Sec	5	300	1500
Capacity	2 Gb	8 Gb	1 TB
Bandwidth	1 Mbps	10 Mbps	100 Mbps

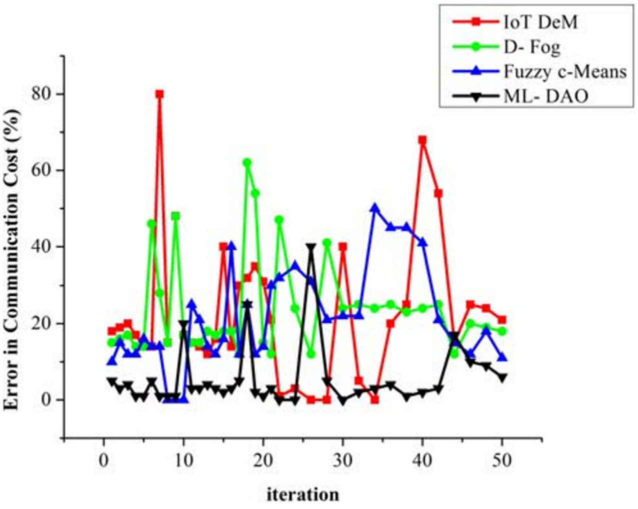

Fig. 5 Communication error analysis

The ratio of download d_{rat} is then computed using

$$d_{rat} = \frac{r_n}{d_a/t_a}, \tag{9}$$

where d_a is the data accumulated at t_a interval, and r_n is the number of requests from the consumer layer. Now, the ML process instigated in the previous interaction is modeled to work in a recursive manner. The d_{rat} achieved in the previous iterate is ensured at each iterate with a minimum route cost. Figure 4 illustrates the process of recursive learning with respect to the consideration of d_{rat}. The learning process is designed to achieve a higher d_{rat} for d_a and t_a by collecting data from s_a out of s_i sensors. More precisely, for s_a sensors, the route cost is r_c^* with lesser errors.

This process of learning is different from conventional neural learning, as it is designed to derive the specific r_c^* by considering d_{rat} as the learning constraint. Similarly, the learning constraints are used for training the inputs at each

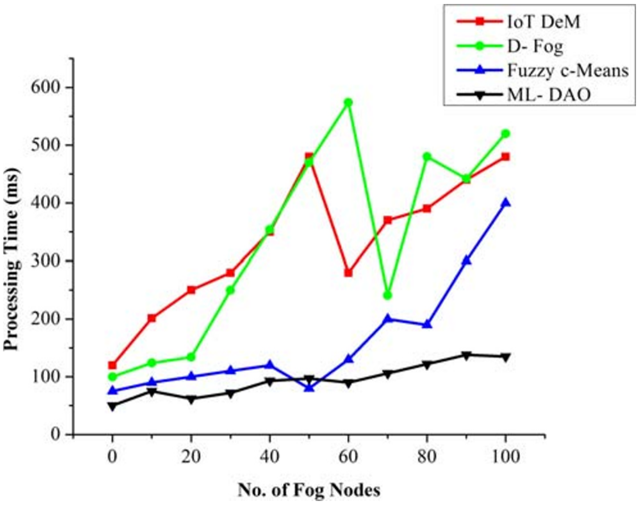

Fig. 6 Processing time analysis

transition; in this case, the hidden layer is optimized based on the learning constraint. The response time t_{res} for offloading d_a for requests r_n is computed using Eq. (10).

$$t_{res} = 2 * (t_a + t_w + t_{d_{rat}}), \tag{10}$$

where t_w is the wait time of the request, and $t_{d_{rat}}$ is the time for downloading.

From probability cases 1 and 2, the time delay and d_p for a particular t_a are explained as follows:

Case 1 e is positive (i.e., $r_c > r_c^*$).

Solution 1 As discussed earlier, the data aggregated in this case are high with the optimal route cost. If $(d_{rat} * r_n) \leq c_j$, then the request wait time is zero. Therefore, $t_{res} = 2 * (t_{d_{rat}})$ here, $t_a = 0$ as the data are already accumulated, and they need to be assigned to IoT requests.

Case 2 If e is negative, $r_c < r_c^*$.

Solution 2 In this case, the rate for aggregation is deficient because of a higher r_n or s_a lesser. Therefore, the aggregation is still processed at the time of request mapping.

$$t_{res} = 2\left[t_{d_{rat}} + (t_{a1} + t_{w1}) + (t_{a2} + t_{w2}) + \cdots . + (t_{aj} + t_{wj})\right], \tag{11}$$

$$t_{res} = 2\left[t_{d_{rat}} + \sum_{j=1}^{j}((t_{ai} + t_{wi})\right]. \tag{12}$$

In case 1, $(d_{rat} * r_n)$ is confined to the capacity of the fog element with no offloading requirement.

In case 2, the capacity of the fog element is exceeded, and $(c_j - r_n)$ is offloaded via its neighbor to the end user. In this case, propagation time t_p is considered between the two neighboring fog elements, so $t_{res} = t_{res} + t_p$ is the actual response time for a request r_n from the consumer layer.

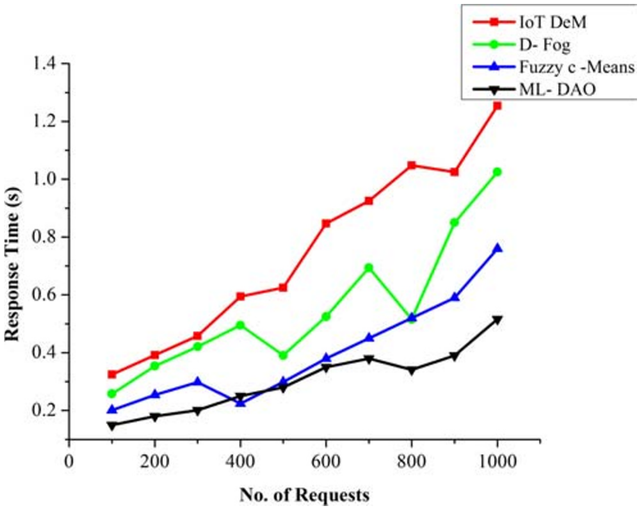

Fig. 7 Response time analysis

4 Experimental analysis

The performance of the proposed ML-DAO is assessed through iFogSim simulations [32]. The configurations for setting up the architecture as in Fig. 1 are described in Table 1, which shows the configuration considered in the evaluation model with the minimum requirements of the devices. The configuration parameters are discussed for low-configuration and low-cost systems with appreciable processing limits. The following metrics are considered for a comparative performance analysis of the proposed ML-DAO with the existing IoT DeM, D-Fog, and Fuzzy C-Means, as discussed in the Related works section: error ratio, processing and response time, failure probability, data service utility, aggregation time, and unallocated data chunks.

4.1 Error analysis

Figure 5 illustrates the comparison of errors between the existing methods and the proposed method. The route cost error is low, as the next iterate of the learning process constructs routes based on the previous e value. Based on probabilities 1 and 2, ther_c for the next aggregation interval is designed. The offloading process recommends a more precise aggregation path construction. This minimizes the route cost error for increasing r_n.

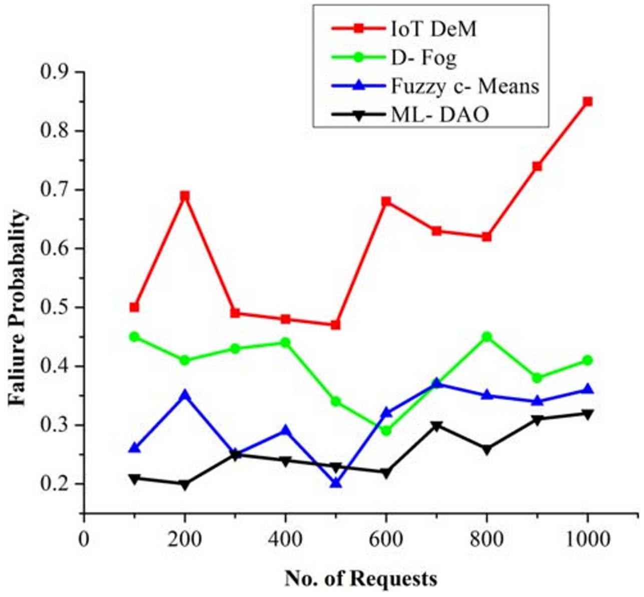

Fig. 8 Failure probability analysis

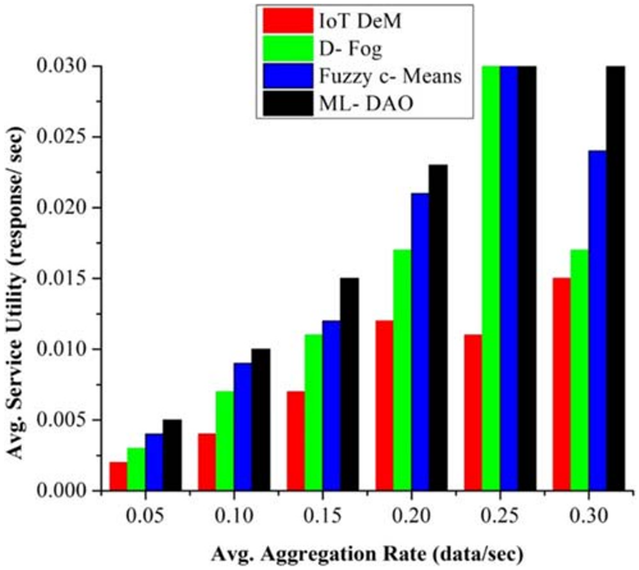

Fig. 9 Service utility analysis

In some cases, $r_c = r_c^*$ or $r_c > r_c^*$; the previous route cost is maintained for d_a in two successive t_a intervals. Therefore, the error is the same at some iterates (iterates 4–5, 7–9, 11–12, and 22–24 in Fig. 5). If $r_c < r_c^*$ is true, the route cost error increases (iterates 10, 18, 26, 44, and 46 in the Fig. 5). The proposed ML-DAO minimizes the communication error by 17.87%, 16.67%, and 11.36% compared with the existing IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.2 Processing time analysis

The comparison of the existing and proposed methods' processing time is illustrated in Fig. 6. The processing time is estimated in the fog layer for all$P_{ij} = 1$. The time interval between d_a and r_n response is estimated as the processing time. After eacht_a, if $ra_r < c_j$, then $t_w \neq 0$; the processing time therefore increases. This demands more promising route cost error minimization in the next learning process and scrutinizes the aggregation route in the successivet_a intervals. This also reduces the processing time in the fog layer despite varyingr_n. The proposed ML-DAO requires 17.97%, 18.51%, and 16.56% less processing time compared with IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.3 Response time analysis

An increase in r_n increases the data response time from the fog layer (Fig. 7). In the proposed ML-DAO, the response time t_{res} is analyzed for two probabilities of the route cost error. From Fig. 5, as the route cost error is less, the response time is estimated using Eq. (10) for each iterate of case 1. As the session of t_a is completed and $(d_{rat}*r_n) \leq c_j$, the requests are instantly mapped with the availabled_a. If $r_c < r_c^*$, the data allocation process exceeds its t_a, so r_n observes a t_w. Similarly, if $(d_{rat}*r_n) > c_j$, the remaining r_n is offloaded to the next fog neighbor. In this scenario, t_p is considered. In the subsequent learning process, $r_c > r_c^*$ or $r_c = r_c^*$, which is ensured to minimize the response time of d_a in the $(t_a + 1)$ interval. The results prove that the proposed ML-DAO service requests have controlled response times that are 14.71%, 12.41%, and 8.03% less than those of IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.4 Failure probability

Figure 8 shows the comparisons between the existing methods and the proposed ML-DAO for failure probability. The number r_n that is left unserviced or lost because of a longer t_w is less in the proposed ML-DAO. ML-DAO achives a lower failure probability by retaining a lesser e for each t_a. The learning

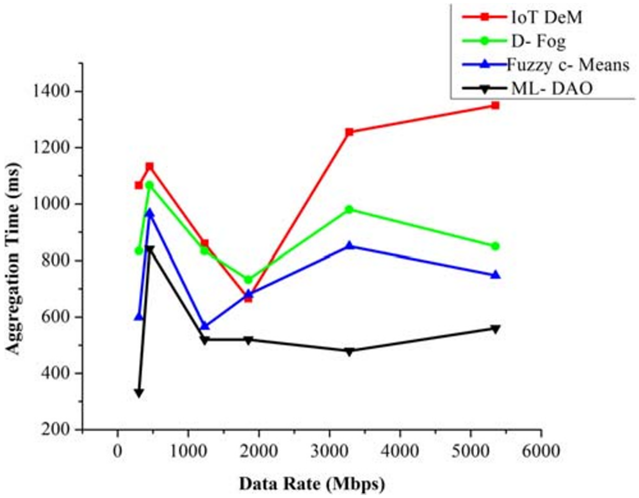

Fig. 10 Aggregation time analysis

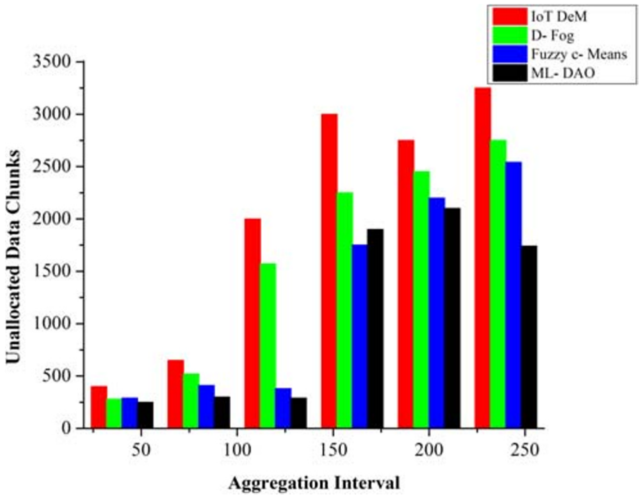

Fig. 11 Unallocated data chunks analysis

process minimizes the t_w by ensuring that the maximum r_n services exceed the capacity of the fog node, and t_w increases. This condition is refined by selecting the appropriates_a among s_i to achieve a lesser e and the maximum coverage. The failure probability of the proposed ML-DAO is 15.59%, 5.49%, and 2.78% less than that of IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.5 Service utility comparison

The rate of service utility increases with an increase in r_n satisfaction. The failure probability of the proposed ML-DAO is low, improving the data dissemination rate. The d_a in time t_a is estimated for d_{rat} to improve the rate of data assignment. Different from d_p, d_{rat} varies with user interest and resource availability for which the maximum service gain is achieved. The experienced d_{rat} from the output layer is fed to the hidden layer to improve the rate of d_a as per user interest. The interest in each response session is considered to gather d_a from the IoT sensors. The wait time because of data deficiency is minimized, which, in turn, improves service utility (Fig. 9). ML-DAO achieves 12.25%, 10.83%, and 8.17% higher service utility than IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.6 Aggregation time analysis

Figure 10 illustrates the comparison in aggregation time between the existing and proposed ML-DAO methods. In a particular t_a, the aggregation time is decided based on the routes constructed. The routes are constructed as per the recommendations of the previous learning iterate. The optimality inr_c, d_{rat} leads to the routes are dynamically adjusted for every new t_a interval. Therefore, the aggregation time relies on the number of s_a and the amount of data accumulated at each t_a. In the proposed ML-DAO, routes are formed to meet d_{rat}, achieving a lower failure probability. The optimal aggregation route formation with lesser e and maximum r_c achieves lesser time in data collection. The proposed ML-DAO requires 14.63%, 8.53%, and 6.28% less aggregation time compared with IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

4.7 Unallocated data chunks

The number of unserviced r_n in the proposed ML-DAO is less because of lesser failures. The rate of data service utilized is also high. The optimality in aggregation and recommendations of the learning process achieves the maximum r_n service. Therefore, the rate of unallocated d_a of each t_a is less in the proposed ML-DAO. Pre-estimation of d_p and d_{rat} improves the data assignment rate, leaving out a few chunks are unallocated. The response time and $t_w = 0$ (most cases) are the extended reasons for the lesser unallocation of d_a (Fig. 11). From the results, ML-DAO minimizes unallocation by 11.62%, 9.18%, and 7.87% compared with IoT DeM, D-Fog, and Fuzzy c-Means, respectively.

5 Conclusion

This paper proposes an ML-assisted data aggregation and offloading scheme to improve cloud-assisted IoT communication. The data aggregation phase is designed to minimize delay in data gathering, whereas the offloading process improves the rate of service utilization by minimizing failure probability. Both distinct phases are monitored by an ML process that controls aggregation route cost errors through recommendations. Minimizing route cost errors reduces the processing time of fog layer elements and improves the request processing rate. The extended ML process in the offloading phase minimizes the request response time, failure probability, and unallocated aggregated data. Interoperability between cloud and the IoT is improved by introducing fog elements. Experimental results verify the reliability of the proposed scheme, as it is shown to improve service utilization and minimize failure, cost errors, and unallocated data chunks. In the future, optimization techniques are included with ML techniques to improve data aggregation efficiency.

Acknowledgements This work is funded by the Researchers Supporting Project No. (RSP-2020/102) King Saud University, Riyadh, Saudi Arabia.

References

1. Alavi AH, Jiao P, Buttlar WG, Lajnef N (2018) Internet of Things-enabled smart cities: State-of-the-art and future trends. Measurement 129:589–606
2. AlFarraj O, Tolba AAlZubi,A (2019) Optimized feature selection algorithm based on fireflies with gravitational ant colony algorithm for big data predictive analytics. Neural Comput Appl 31(5):1391–1403
3. Sicari S, Rizzardi A, Miorandi D, Coen-Porisini A (2018) A risk assessment methodology for the internet of things. Comput Commun 129:67–79
4. Fouad H, Mahmoud NM, El Issawi MS, Al-Feel H (2020) Distributed and scalable computing framework for improving request processing of wearable IoT assisted medical sensors on pervasive computing system. Comput Commun 151:257–265
5. Haw R, Alarm M, Hong C (2014) A context-aware content delivery framework for QoS in mobile cloud. Proc. IEEE NOMS, pp 1–6
6. Al-Makhadmeh Z, Tolba A (2020) SRAF: Scalable Resource Allocation Framework using machine learning in user-centric internet of things. Peer-to-peer networking and applications. https://doi.org/10.1007/s12083-020-00924-3

7. Sheron PF, Sridhar KP, Baskar S, Shakeel PM (2019) A decentralized scalable security framework for end-to-end authentication of future IoT communication. Transactions on Emerging Telecommunications Technologies, e3815. https://doi.org/10.1002/ett.3815
8. Mubarakali A, Durai AD, Alshehri M, AlFarraj O, Ramakrishnan J, Mavaluru D (2020) Fog-based delay-sensitive data transmission algorithm for data forwarding and storage in cloud environment for multimedia applications. Big Data. https://doi.org/10.1089/big.2020.0090
9. Baskar S, Periyanayagi S, Shakeel PM, Dhulipala VS (2019) An energy persistent range-dependent regulated transmission communication model for vehicular network applications. Comput Netw 152:144–153. https://doi.org/10.1016/j.comnet.2019.01.027
10. Said O, Al-Makhadmeh Z, Tolba A (2020) EMS: An energy management scheme for green IoT environments. IEEE Access 8: 44983–44998
11. Oteafy SMA, Hassanein HS (2018) IoT in the fog: A roadmap for data-centric IoT development. IEEE Commun Mag 56(3):157–163
12. Wang J, Tang Y, He S, Zhao C, Sharma PK, Alfarraj O, Tolba A (2020) LogEvent2vec: logEvent-to-vector based anomaly detection for large-scale logs in internet of things. Sensors 20(9):2451
13. Naha RK, Garg S, Georgakopoulos D, Jayaraman PP, Gao L, Xiang Y, Ranjan R (2018) Fog computing: Survey of trends, architectures, requirements, and research directions. IEEE Access 6: 47980–48009
14. Alsiddiky A, Awwad W, Fouad H, Hassanein AS, Soliman AM (2020) Priority-based data transmission using selective decision modes in wearable sensor based healthcare applications. Comput Commun 160:43–51
15. Li H, Ota K, Dong M (2018) Learning IoT in edge: Deep learning for the internet of things with edge computing. IEEE Network 32(1):96–101
16. Rahim A, Ma K, Zhao W, Tolba A, Al-Makhadmeh Z, Xia F (2018) Cooperative data forwarding based on crowdsourcing in vehicular social networks. Pervasive Mob Comput 51:43–55
17. Ji H, Alfarraj O, Tolba A (2020) Artificial intelligence-empowered edge of vehicles: architecture, enabling technologies, and applications. IEEE Access 8:61020–61034
18. Tolba A, Al-Makhadmeh Z (2020) A recursive learning technique for improving information processing through message classification in IoT–cloud storage. Comput Commun 150:719–728
19. Kato N et al (2017) The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective. IEEE Wirel Commun 24(3):146–53. https://doi.org/10.1109/MWC.2016.1600317WC
20. AlFarraj O, Tolba A, Alkhalaf S, AlZubi A (2019) Neighbor predictive adaptive handoff algorithm for improving mobility management in VANETs. Comput Netw 151:224–231
21. Kayes ASM, Rahayu W, Dillon T (2018) Critical situation management utilizing IoT-based data resources through dynamic contextual role modeling and activation. Computing
22. Kim J, Jeon Y, Kim H (2016) The intelligent IoT common service platform architecture and service implementation. J Supercomput 74(9):4242–4260
23. Ullah F, Wang J, Farhan M, Jabbar S, Naseer MK, Asif M (2018) LSA based smart assessment methodology for SDN infrastructure in IoT environment. Int J Parallel Program
24. Puschmann D, Barnaghi P, Tafazolli R (2016) Adaptive clustering for dynamic IoT data streams. IEEE Internet Things J :1–1
25. Xiao W, Bao W, Zhu X, Liu L (2017) Cost-aware big data processing across geo-distributed datacenters. IEEE Trans Parallel Distrib Syst 28(11):3114–3127
26. Lu Z, Wang N, Wu J, Qiu M (2018) IoTDeM: An IoT Big Data-oriented MapReduce performance prediction extended model in multiple edge clouds. J Parallel Distrib Comput 118:316–327
27. Bu F (2018) An efficient fuzzy c-means approach based on canonical polyadic decomposition for clustering big data in IoT. Future Gener Comput Syst 88:675–682
28. Cheng B, Solmaz G, Cirillo F, Kovacs E, Terasawa K, Kitazawa A (2018) FogFlow: Easy programming of IoT services over cloud and edges for smart cities. IEEE Internet Things J 5(2):696–707
29. Yacchirema DC, Sarabia-Jacome D, Palau CE, Esteve M (2018) A smart system for sleep monitoring by integrating IoT with big data analytics. IEEE Access 6:35988–36001
30. Zhao W, Liu J, Guo H, Hara T (2018) Edge-node-assisted transmitting for the cloud-centric internet of things. IEEE Netw 32(3):101–107
31. He J, Wei J, Chen K, Tang Z, Zhou Y, Zhang Y (2018) Multitier fog computing with large-scale IoT data analytics for smart cities. IEEE Internet Things J 5(2):677–686
32. Gupta H, Dastjerdi AV, Ghosh SK, Buyya R (2017) iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things. Edge and Fog computing environments. Softw Pract Experience 47(9):1275–1296

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.