Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.038417
Article

Task Offloading and Resource Allocation in IoT Based Mobile Edge Computing Using Deep Learning

Ilyos Abdullaev[1], Natalia Prodanova[2], K. Aruna Bhaskar[3], E. Laxmi Lydia[4], Seifedine Kadry[5,6,7] and Jungeun Kim[8,*]

[1]Dean of the Faculty of Economics, Department of Management and Marketing, Faculty of Economics, Urgench State University, Urganch, 220100, Uzbekistan
[2]Basic Department Financial Control, Analysis and Audit of Moscow Main Control Department, Plekhanov Russian University of Economics, Moscow, 117997, Russia
[3]Department of Computer Science and Engineering, KL Deemed to University, Vaddeswaram, Guntur, Andhra Pradesh, India
[4]Department of Computer Science and Engineering, GMR Institute of Technology, Andhra Pradesh, Rajam, India
[5]Department of Applied Data Science, Noroff University College, Kristiansand, Norway
[6]Artificial Intelligence Research Center (AIRC), Ajman University, Ajman, 346, United Arab Emirates
[7]Department of Electrical and Computer Engineering, Lebanese American University, Byblos, Lebanon
[8]Department of Software, Kongju National University, Cheonan, 31080, Korea

*Corresponding Author: Jungeun Kim. Email: jekim@kongju.ac.kr

Received: 12 December 2022; Accepted: 16 March 2023; Published: 30 August 2023

Abstract: Recently, computation offloading has become an effective method for overcoming the constraint of a mobile device (MD) using computation-intensive mobile and offloading delay-sensitive application tasks to the remote cloud-based data center. Smart city benefitted from offloading to edge point. Consider a mobile edge computing (MEC) network in multiple regions. They comprise N MDs and many access points, in which every MD has M independent real-time tasks. This study designs a new Task Offloading and Resource Allocation in IoT-based MEC using Deep Learning with Seagull Optimization (TORA-DLSGO) algorithm. The proposed TORA-DLSGO technique addresses the resource management issue in the MEC server, which enables an optimum offloading decision to minimize the system cost. In addition, an objective function is derived based on minimizing energy consumption subject to the latency requirements and restricted resources. The TORA-DLSGO technique uses the deep belief network (DBN) model for optimum offloading decision-making. Finally, the SGO algorithm is used for the parameter tuning of the DBN model. The simulation results exemplify that the TORA-DLSGO technique outperformed the existing model in reducing client overhead in the MEC systems with a maximum reward of 0.8967.

Keywords: Mobile edge computing; seagull optimization; deep belief network; resource management; parameter tuning

1 Introduction

The Internet of Things (IoT) has become gradually substantial in daily life as information technology has advanced. The interrelated device collects and exchanges various information among them via a current transmission network interconnected through different IoT nodes [1,2]. A variety of IoT applications might offer users fine-grained and highly precise network facilities. In such cases, the IoT technique is interconnecting a growing number of devices and sensors, which might produce massive quantities of information that may need additional processing, which offer intellect to the service providers and customers [3]. All information should be uploaded to a central server in typical cloud computing (CC), and the outcomes should be transferred back to the devices and sensors after processing [4,5]. Such a technique places various constraints on the network, particularly based on the resource and bandwidth costs for the information broadcast [6]. Fig. 1 depicts the overview of task offloading in the mobile edge computing (MEC) method.

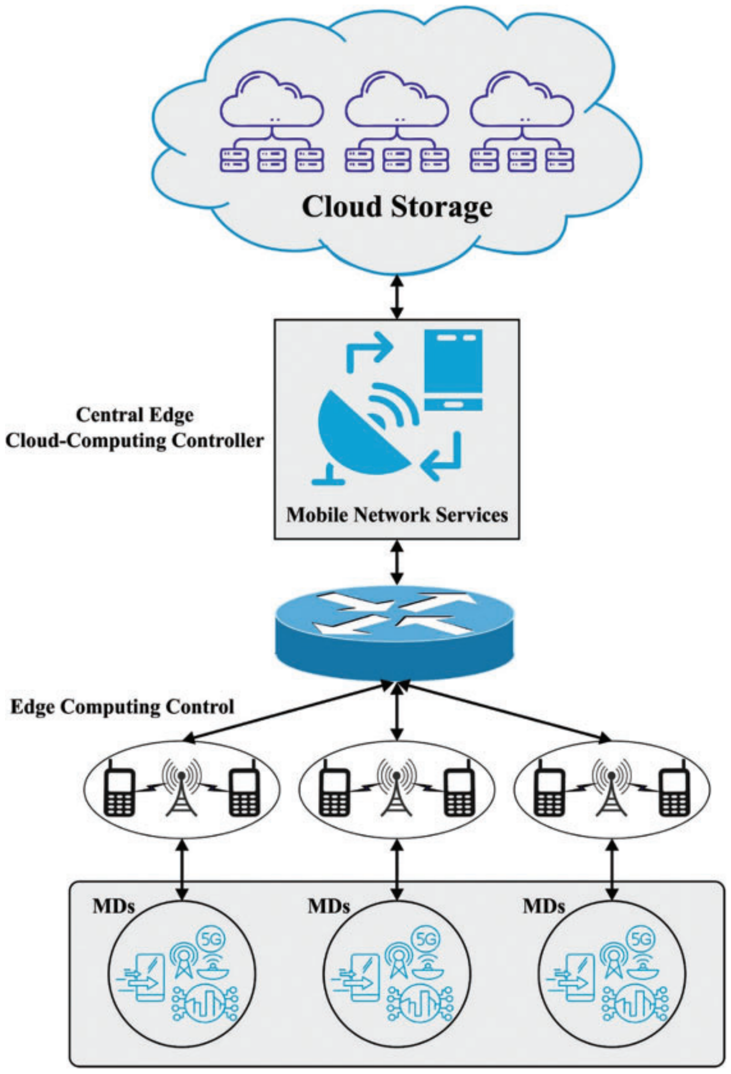

Figure 1: Overview of task offloading in the MEC model

Moreover, once the information quantity is superior, the network's performance will deteriorate [7]. MEC offers storage and computation functions at the edge network closer to the client. Compared to conventional CC, MEC decreases the communication latency of the user; however, it mitigates the load and congestion of the central network [8,9]. As a key technique in MEC, computational offloading could efficiently alleviate the mismatch between the communication and computing abilities of task load and terminal devices [10].

Even though MEC offloading could efficiently increase the efficiency of the wireless network [11], it could not meet the service requirement of each device because of the constraints of computational resources of the MEC server in certain hotspots [12]. The CPU utilization of mobile devices is idle or lower, resulting in inefficient usage and wasted resources to the specific range. Device-to-device (D2D) transmission is a novel technique, which enables the terminal device to directly interact via shared resources controlled by the base station (BS) [13,14]. D2D transmission might decrease BS and transmission delay; however, it saves energy consumption and expands the transmission range. Based on this, researcher workers incorporated D2D with the MEC scheme to improve the efficiency of the user offloading through the suitable computational offloading technique [15].

Tan et al. [16] examined the allocation of communication resources, offloading decisions, relationship decisions, and allocation of computational resource problems in C-MEC. The delay-sensitive task of the user is locally calculated and offloaded to MEC servers or collaborative devices. The objective is to minimize the mobile user's overall power utilization under the delay constraints. The problems are expressed by the mixed-integer nonlinear programming (MINLP) that includes the allocation of computational resources, the joint optimization of task offloading decision, relationship decision, and subcarriers and energy allocation. Deng et al. [17] proposed the autonomous partial offloading scheme for the delay-sensitive computational task in multiple-user industrial IoT (IIoT) MEC systems. The primary objective is to offer offloading facilities with minimal delay. Particularly, the Q-learning mechanism is implemented for providing a discrete partial offloading decision.

Chen et al. [18] proposed a two-stage alternative technique based on sequential quadratic programming (SQP) and deep reinforcement learning (DRL). In the upper level, provided the assigned CPU frequency and broadcast power, the cache and task offloading decision problems are resolved using the deep Q-network (DQN) model. In the lower level, CPU frequency distribution and the optimum communication power with the cache and offloading decisions are attained through the SQP method. Zhu et al. [19] examined task offloading and joint cloudlet deployment problems to diminish the task response delay, user power utilization, and the number of deployed cloudlets.

Dai et al. [20] recommended an effectual offloading structure related to DRL for MEC with edge-CC collaboration. In contrast, the computation-intensive task is locally offloaded or implemented to the cloud server. By conjointly bearing in mind: i) the dynamic edge-CC platform; ii) faster offloading decision, we influence DRL to minimalize the processing delay of tasks by efficiently incorporating the cloud server, the computational resources of vehicles, and edge servers (ESs). Particularly, a DQN is exploited to learn optimum offloading schemes adaptively. In [21], to minimalize the overhead of fog computing networks, involving the energy consumption and task process delay, when preserving the quality of service (QoS) requirement of distinct kinds of ID, the study proposed a QoS-aware resource allocation technique that cooperatively considers the relationship between IDs and fog nodes (FNs), computation resource allocation and communication to enhance the offloading decision when reducing the network overhead.

This study designs a new Task Offloading and Resource Allocation in IoT-based MEC using Deep Learning with Seagull Optimization (TORA-DLSGO) algorithm. The proposed TORA-DLSGO technique addresses the resource management issue in the MEC server, which enables an optimum offloading decision to minimize the system cost. In addition, an objective function is derived based on minimizing energy consumption subject to the latency requirements and restricted resources. The TORA-DLSGO technique uses the deep belief network (DBN) model for optimum offloading decision-making. Finally, the SGO algorithm is used for the parameter tuning of the DBN model. The simulation results exemplify that the TORA-DLSGO technique outperformed the existing model in reducing client overhead in the MEC systems.

The rest of the paper is organized as follows: Section 2 introduces the proposed model and Section 3 offers the experimental validation. Finally, Section 4 concludes the study.

2 The Proposed Model

In this study, we have developed a new TORA-DLSGO algorithm for task offloading and resource allocation in the IoT-based MEC environment. The proposed TORA-DLSGO technique addressed the resource management issue in the MEC server, which enables an optimum offloading decision to minimize the system cost. In addition, an objective function is derived based on minimizing energy consumption subject to the latency requirements and restricted resources.

2.1 Communication Model

MES is based on the telecommunication substructure, namely Long-Term Evolution (LTE) or BS. The mobile drones (MDs) such as drones, smartphones, robots, and tablets are connected to the edge computing controls at the LTE or BS in the nearby region (position) to the computational offloading. Consider that the MEC network (MECN) exists in multiple regions comprising multiple APs, multiple ES, and n MDs represented as $n = \{1, 2, \ldots, n\}$. The MD could interconnect to the MECN via a mobile network or access point. Based on a specific parameter, including response time, energy consumption, ESs' workload, or latency, the MD must find an effective position in the network to carry out offloading. The offloading performance is characterized by $A = \{a_1, a_2, \ldots, a_n\}$, based on offloading decision, whereby X_n indicate the offloading decision $X_n = \{0, 1, 2, 3\}$ (local computing: $X_n = 0$ or nearby the ES: $X_n = 1$, neighbouring to the ES: $X_n = 2$, remote cloud: $X_n = 3$). The computing delay and bandwidth B_n based on processing frequency f_n *can impact the offloading decision*.

The transmission bandwidth amongst MDs and offloading positions are represented as B^e, B^a *and* B^c. Furthermore, the overall transmission delay for the specific MD can be represented as T_n. Furthermore, p_n^t indicates the power utilization for tasks, and p_n^r shows the receiving power utilization. Thus, based on specific variables namely energy consumption, ESs' workload, response time, or latency, the MD need to find d an influential position (nearby the ES or nearby to the remote cloud or ES) for offloading the process. Finally, afterwards, the offloading task to the adjacent ES or nearby ES was finished, and an effective resource allocation technique was mandatory on the ES.

2.2 Task Modeling

Consider that every MD has M independent huge real-time task that is remotely implemented in the MEC network or locally implemented in the MD by the computational offloading [22]. Consequently, a task could not be separated into subtasks to be processed in several devices. Task size is represented as D_n (transmitted data size), and R_n represents the computational resource needed to serve these tasks (CPU cycle number). Thus, D_n and R_n are positively related as $R_n = \theta D_n$, θ constant. The task is implemented on the MEC network or locally implemented on the MD, *and* D_n remains unchanged.

2.3 DBN-Based Decision Making

In the presented TORA-DLSGO technique, the DBN model is used for optimum offloading decision-making purposes. The DBN is encompassed of restricted Boltzmann machine (RBM) and backpropagation neural network (BPNN) [23]. The training model was detached into pre-training and finetuning. The former train the RBM in unsupervised means, whereas the later fine-tunes the bias and weight parameters of the pretrained method via executing the BP technique. The DBN is mainly encompassed by RBM and is derived from the energy function of thermodynamics. Fig. 2 illustrates the infrastructure of DBN. Furthermore, RBM is composed of the upper layer as a hidden layer and the lower layer as a visible layer. The energy model of RBM can be formulated using the following expression:

$$E(v, h|\theta) = -\sum_{i=1}^{I} a_i v_j - \sum_{j=1}^{J} b_j h_j - \sum_{i=1j}^{I} \sum_{=1}^{J} w_{ij} v_i h_j \tag{1}$$

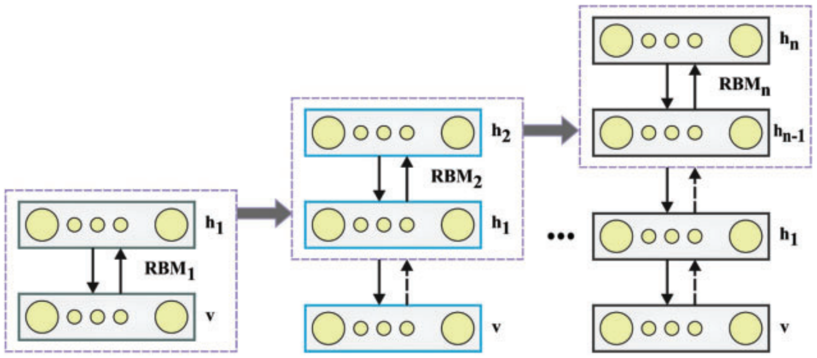

Figure 2: Architecture of DBN

From the expression, *I* and *J* correspondingly indicate the number of nodes in the visible and hidden neurons, $v = (v_1, v_2 \ldots, v_i)$ and $h = \left(h_1, h_2 \ldots, h_j\right)$ correspondingly refer to the state of visible and hidden neurons, $\theta = \left\{w_{ij}, a_i, b_j\right\}$ specifies the module parameter of RBM, w_{ij} denotes the weight of i-th and j-th nodes of the visible and hidden neurons, a_i specifies the bias of i-th nodes of visible state. $_{Bj}$ indicates the bias of j-th neurons of the hidden layer. The joint possibility distribution can be obtained according to the energy function of RBM:

$$p(v, h|\theta) = \frac{e^{-E(v,h|\theta)}}{Z(\theta)} \tag{2}$$

From the expression: $Z(\theta) = \sum_{v,h} e^{-E(v,h|\theta)}$ shows the mean normalization factor. The likelihood of h_j hidden neurons and v_i the visible neurons are activated by:

$$P\left(h_j|v\right) = \sigma\left(b_j + \sum_{i=1}^{I} W_{ij} v_i\right) \tag{3}$$

$$P(v_i|h) = \sigma\left(a_j + \sum_{j=1}^{J} W_{ij} h_j\right) \tag{4}$$

Now, $\sigma(\cdot)$ represent the activation function. To enable the neural network (NN) for resolving complex problems, it is important to add a nonlinear factor to the NN using the activation function. When training the RBM, the highest probability obtains the parameter set $\theta = \left\{w_{ij}, a_i, b_j\right\}$, and the trained dataset is exploited for RBM training to accomplish the highest possible function is formulated below:

$$\mathrm{L}(\theta) = \prod_{l=1}^{N} \mathrm{L}(\mathrm{v}|\theta) = \prod_{l=1}^{N} \mathrm{p}(\mathrm{v}) \tag{5}$$

Now, N shows the number of instances. Furthermore, the random gradient rise method is exploited to calculate the derivation of the parameter of the highest probability function, including the bias and weight of every node in the following:

$$\frac{\partial \ln \mathrm{L}}{\partial \mathrm{w}_{ij}} = \sum_{l=1}^{N} \left[\mathrm{P}\left(\mathrm{h}_j = 1|\mathrm{v}^l\right) \mathrm{v}_i^l - \sum_{\mathrm{v}} \mathrm{P}(\mathrm{v})\, \mathrm{P}\left(\mathrm{h}_j = 1|\mathrm{v}\right) \mathrm{v}_i\right] \tag{6}$$

$$\frac{\partial \ln \mathrm{L}}{\partial \mathrm{a}_i} = \sum_{l=1}^{N} \left[\mathrm{v}_i^l - \sum_{\mathrm{v}} \mathrm{P}(\mathrm{v})\, \mathrm{v}_i\right] \tag{7}$$

$$\frac{\partial \ln \mathrm{L}}{\partial \mathrm{b}_j} = \sum_{l=1}^{N} \left[\mathrm{P}\left(\mathrm{h}_j = 1|\mathrm{v}^l\right) - \sum_{\mathrm{v}} \mathrm{P}(\mathrm{v})\, \mathrm{P}(\mathrm{h}_i = 1|\mathrm{v})\right] \tag{8}$$

2.4 Parameter Optimization

Finally, the SGO algorithm is used for the parameter tuning of the DBN model. SGO was based on the migration and attacking performance of seagulls [24]. The mathematical process of attacking and migrating the prey was determined in the following. The migration system stimulates the set of seagulls that move in every place. At this point, the seagulls can accomplish 3 conditions: To avoid collision betwixt neighbors (that is, other seagulls), an added parameter A was utilized to assess a novel search position.

$$\vec{C}_s = A \times \vec{P}_s(x) \tag{9}$$

In which x denotes the existing iteration in the subsequent, $\vec{C}_s$ implies the search agent position which could not collide with another searching agent, $\vec{P}_s$ denotes the existing search agent position.

$$A = f_c - (x \times (f_c/\mathrm{M}ax_{iteration})) \tag{10}$$

where: $x = 0,\ 1,\ 2,\ \ldots Max_{iteration}$

Wherein f_c represents the control of the frequency of A using slowly minimizing in f_c to 0. At this point, the value of f_c is set to 2. Then avoiding the collision betwixt neighbors, the searching agent moves to a better neighbor direction.

$$\vec{M}_s = B \times \left(\vec{P}_{bs}(x) - \vec{P}_s(x)\right) \tag{11}$$

Assume $\vec{M}_s$ be the search agent and place $\vec{P}_s$ to better adequate search agent $\vec{P}_{bs}$ (viz., appropriate seagulls). The performance of B was assigned arbitrarily using responsible to appropriate balancing betwixt exploitation as well as exploration. B can be measured as:

$$B = 2 \times A^2 \times rd \tag{12}$$

During this formula, rd exemplifies an arbitrary integer that lies between zero and one. Eventually, the search agent upgrades their place nearby to a better search agent.

$$\vec{D}_s = \left|\vec{C}_s + \vec{M}_s\right| \tag{13}$$

Consider $\vec{D}_s$ to be the distance betwixt the better appropriate searching agent and searching agent (viz., better seagull which fitness value is least). The exploitation step concentrates on utilizing the experience and history of the searching method. In prey attacking, the spiral performance occurs in the air. Seagulls can change the angle and attack speed from the migration. It can be remembered as altitude with wing and weight. The x, y, and z planes can be provided in the subsequent.

$$x' = r \times cos\,(k) \tag{14}$$

$$y' = r \times sin\,(k) \tag{15}$$

$$z' = r \times k \tag{16}$$

$$r = u \times e^{kv} \tag{17}$$

wherein r defines the radius of all the turns of spirals, k distinguishes the arbitrary value in $\left[0 \le k \le 2A\right]$. u and v imply the constant to define the spiral shape, and e symbolizes the base of the natural logarithm. It could be measured in the subsequent formula:

$$\vec{P}_s\,(x) = \left(\vec{D}_s \times x' \times y' \times z'\right) + \vec{P}_{bs}\,(x) \tag{18}$$

At this point, $\vec{P}_s\,(x)$ supplies the optimum result and upgrades another searching agent's place. Based on the abovementioned communication and computation techniques, the energy consumption and the overall time consumption are evaluated as follows:

$$t_n = \alpha_n \cdot \left(t_n^{up} + t_n^{MEC}\right) + (1 - \alpha_n) \cdot t_n^{IoT} \tag{19}$$

$$e_n = \alpha_n \cdot e_n^{up} + (1 - \alpha_n) \cdot e_n^{IoT} \tag{20}$$

The mathematical modelling is used to minimize the energy utilization of every IoT device subjected towards the limited resources and the latency requirement:

$$\begin{aligned} &\min_{B_n, f_n, \alpha n} \Sigma_{n=1}^{N} e_n \; s.t. \\ &(c1)\; t_n \le T \\ &(c2) \sum\nolimits_{n=1}^{N} B_n \le B \\ &(c3) \sum_{n=1}^{N} f_n \le F^{MEC} \\ &(c4)\; \alpha_n \in \{0, 1\} \end{aligned} \tag{21}$$

Now F^{MEC} shows the overall computation resources of the MEC server [25]. For these constraints, constraint $(c1)$ shows that the implementation time of IoT devices n could not exceed the delay threshold to guarantee the QoE. Considering that the implementation time of IoT tasks is within the delay threshold, a satisfactory user experience could be attained. For instance, in the community access control technique, when the delay threshold of face detection techniques is 0.1 s, the user experience is fulfilled if the face detection is finished within 0.1 s. Users have similar QoE to complete face detection within 0.1 and 0.01 s; it is optional to pursue a low processing time that is worthless in the real-time scene. Constraint $(c2)$ shows that the number of assigned sub-bandwidth could not surpass the overall bandwidth of BS. Constraint $(c3)$ represents that the computational resource assigned to every IoT device through the MEC server could not surpass the overall computational resource of the MEC server. Constraint ($c4$) represents that IoT device tasks are processed on the MEC server or IoT device n. When $\alpha_n = 0$, the task of IoT devices would be processed on IoT devices. When $\alpha_n = 1$, the task of IoT devices would be offloaded to the MEC server.

3 Results and Discussion

The experimental study of the TORA-DLSGO model is investigated in detail. Table 1 and Fig. 3 report an overall convergence rate (CR) examination of the TORA-DLSGO model with other models such as DQN, Deep Deterministic Policy Gradient (DDPG), and fuzzy logic based DDPG (FL-DDPG) on distinct episodes. The resultant values indicate that the TORA-DLSGO model reaches optimal CR values under each episode.

Table 1: CR analysis of TORA-DLSGO approach with other techniques under distinct episodes

		Convergence rate		
Episode	DQN	DDPG	FL-DDPG	TORA-DLSGO
2000	0.4384	−2.4226	−2.7219	−0.7592
4000	0.4384	−2.0234	−1.0919	−0.2935
6000	0.7710	−0.5264	−0.1604	0.1057
8000	0.3718	−0.4266	0.2388	0.3718
10000	0.4716	−0.2935	0.4051	0.5049
12000	0.4051	−0.1604	0.3386	0.4051
14000	0.2720	0.0392	0.4384	0.5382
16000	0.5714	−0.0274	0.2055	0.5049
18000	0.5382	−0.0606	0.2388	0.4051
20000	0.2720	0.1390	0.2720	0.8708
22000	0.6380	−0.2935	0.0392	0.6380
24000	0.7710	0.1057	0.1390	0.4384

Fig. 4 represents the reward analysis of the TORA-DLSGO model under different aggregative intervals. The results highlighted that the TORA-DLSGO model had obtained improved performance with effective reward values. In addition, the TORA-DLSGO model has accomplished increasing reward at 30000 aggregation intervals.

Table 2 and Fig. 5 exhibits a comparative reward study of the TORA-DLSGO model with other existing models [25]. The results portrayed the improvements of the TORA-DLSGO model with other models under all bandwidth (BW). For instance, with BW of 5, the TORA-DLSGO model reaches an improving reward of 0.8196, whereas the random, greedy, DQN, DDPG, and FL-DDPG models attain degrading reward values of 0.5604, 0.5616, 0.7001, 0.7515, and 0.7623, respectively. At the same time, with BW of 10, the TORA-DLSGO technique reaches an improving reward of 0.9832 while the random, greedy, DQN, DDPG, and FL-DDPG models attain degrading reward values of 0.5807, 0.9247, 0.9438, 0.9605, and 0.9677 correspondingly. At the same time, with BW of 5, the TORA-DLSGO approach reaches an improving reward of 0.9952, whereas the random, greedy, DQN, DDPG, and FL-DDPG methods attain degrading reward values of 0.5819, 0.9856, 0.9880, 0.9904, and 0.9952 correspondingly.

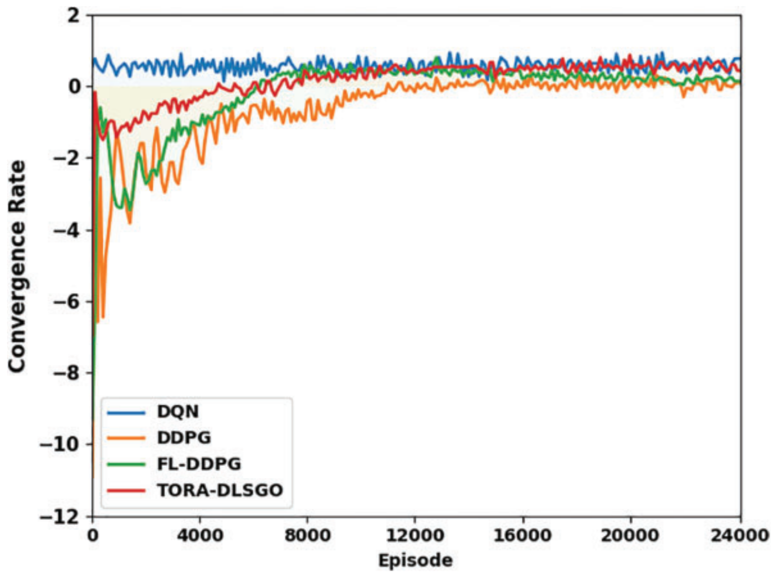

Figure 3: CR analysis of TORA-DLSGO approach under distinct episodes

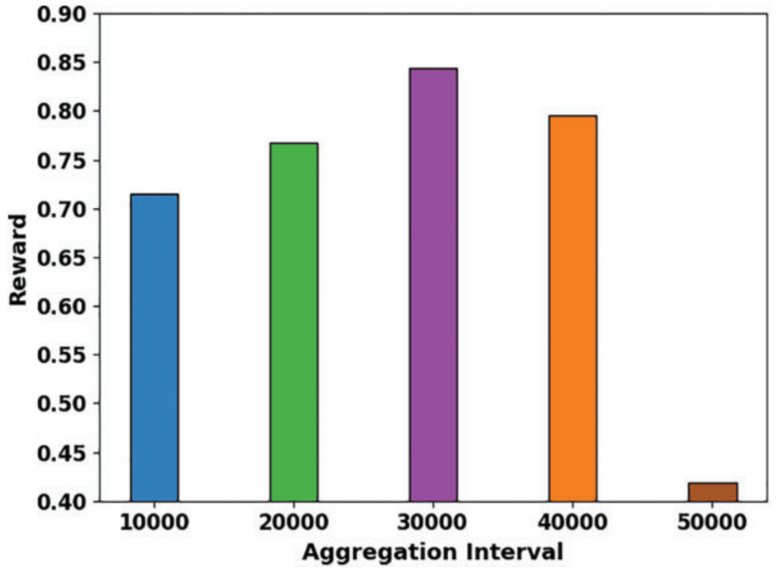

Figure 4: Reward analysis of TORA-DLSGO approach under different aggregative interval

Table 2: Reward analysis of TORA-DLSGO approach with other systems under distinct bandwidth

	Reward					
Bandwidth	Random	Greedy	DQN	DDPG	FL-DDPG	TORA-DLSGO
5	0.5604	0.5616	0.7001	0.7515	0.7623	0.8196
6	0.5700	0.6058	0.7312	0.7718	0.8542	0.8948
7	0.5676	0.6858	0.7682	0.7945	0.8889	0.9331

(Continued)

Table 2 (continued)

	Reward					
Bandwidth	Random	Greedy	DQN	DDPG	FL-DDPG	TORA-DLSGO
8	0.5783	0.7670	0.8530	0.9068	0.9295	0.9582
9	0.5831	0.8530	0.9008	0.9426	0.9534	0.9773
10	0.5807	0.9247	0.9438	0.9605	0.9677	0.9832
11	0.5807	0.9785	0.9809	0.9785	0.9844	0.9904
12	0.5819	0.9856	0.9880	0.9904	0.9904	0.9952

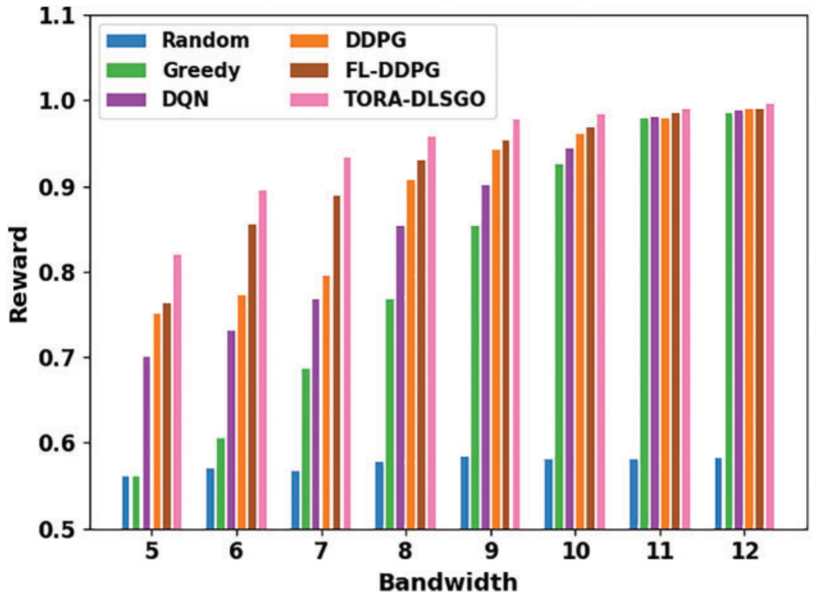

Figure 5: Reward analysis of TORA-DLSGO methodology under distinct bandwidth

A brief set of energy consumption (ECM) inspection of the TORA-DLSGO model is examined under distinct BW values in Table 3 and Fig. 6. The obtained results pointed out the enhanced efficacy of the TORA-DLSGO model under each BW value. For instance, with BW of 5, the TORA-DLSGO model results in minimalized ECM of 2.0389, whereas the random, greedy, DQN, DDPG, and FL-DDPG models reach maximized ECM of 5.8015, 5.8015, 3.6821, 3.0678, and 2.7914, respectively. With BW of 10, the TORA-DLSGO technique results in minimalized ECM of 0.1653 while the random, greedy, DQN, DDPG, and FL-DDPG models reach maximized ECM of 5.3561, 0.9331, 0.7488, 0.5492, and 0.3342 correspondingly. Similarly, with BW of 12, the TORA-DLSGO technique results in decreased ECM of 0.0578, whereas the random, greedy, DQN, DDPG, and FL-DDPG models reach maximized ECM of 5.4636, 0.3956, 0.3035, 0.2267, and 0.1653 correspondingly.

Table 3: ECM analysis of TORA-DLSGO methodology with other systems under distinct bandwidth

	Energy consumption					
Bandwidth	Random	Greedy	DQN	DDPG	FL-DDPG	TORA-DLSGO
5	5.8015	5.8015	3.6821	3.0678	2.7914	2.0389
6	5.6018	5.0029	3.0832	2.5917	2.1617	1.2403
7	5.5865	3.7436	2.6992	2.2846	1.5935	0.9178
8	5.3561	2.6992	1.8546	1.3631	1.0406	0.3956
9	5.4022	1.5935	1.0713	0.9178	0.6567	0.2574
10	5.3561	0.9331	0.7488	0.5492	0.3342	0.1653
11	5.4022	0.5799	0.4110	0.3495	0.2267	0.1038
12	5.4636	0.3956	0.3035	0.2267	0.1653	0.0578

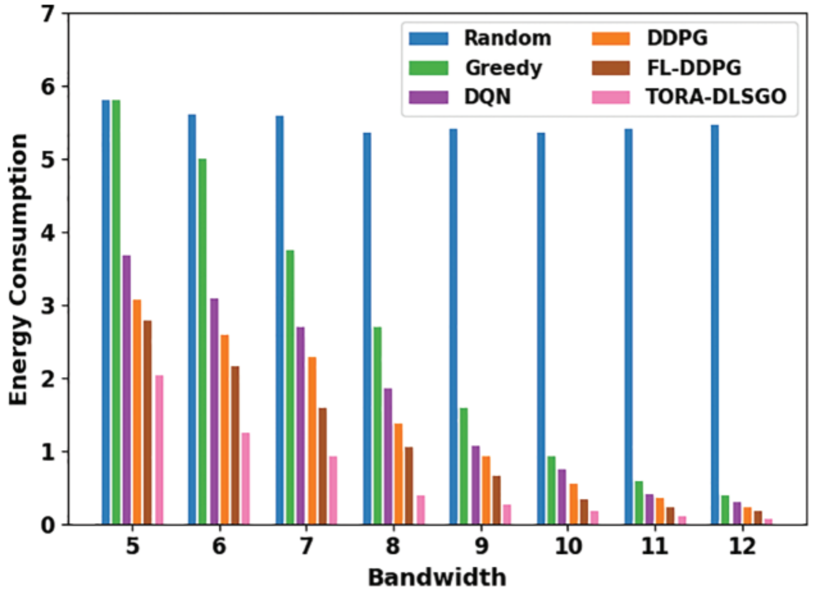

Figure 6: ECM analysis of TORA-DLSGO method under distinct bandwidth

Table 4 and Fig. 7 show a comparative analysis of the TORA-DLSGO technique with other existing models. The results portrayed the improvements of the TORA-DLSGO approach with other models under all delay thresholds (DET). For instance, with a DET of 80, the TORA-DLSGO technique achieves an improving reward of 0.5862. At the same time, the random, greedy, DQN, DDPG, and FL-DDPG models attain degrading reward values of 0.4178, 0.4913, 0.4887, 0.4991, and 0.5519 correspondingly. Simultaneously, with DET of 90, the TORA-DLSGO approach attains an improving reward of 0.8352 while the random, greedy, DQN, DDPG, and FL-DDPG systems accomplish degrading reward values of 0.4900, 0.5842, 0.6190, 0.6667, and 0.8111 correspondingly. Concurrently, with DET of 100, the TORA-DLSGO model attains an improving reward of 0.8967, whereas the random, greedy, DQN, DDPG, and FL-DDPG methods attain degrading reward values of 0.5661, 0.6976, 0.7389, 0.8034, and 0.8743 correspondingly.

Table 4: Reward analysis of TORA-DLSGO methodology with other systems under distinct delay threshold

	Reward					
Delay threshold	Random	Greedy	DQN	DDPG	FL-DDPG	TORA-DLSGO
80	0.4178	0.4913	0.4887	0.4991	0.5519	0.5862
85	0.4733	0.5184	0.5481	0.6061	0.6332	0.6712
90	0.4900	0.5842	0.6190	0.6667	0.8111	0.8352
95	0.5249	0.6125	0.7067	0.7144	0.8318	0.8654
100	0.5661	0.6976	0.7389	0.8034	0.8743	0.8967

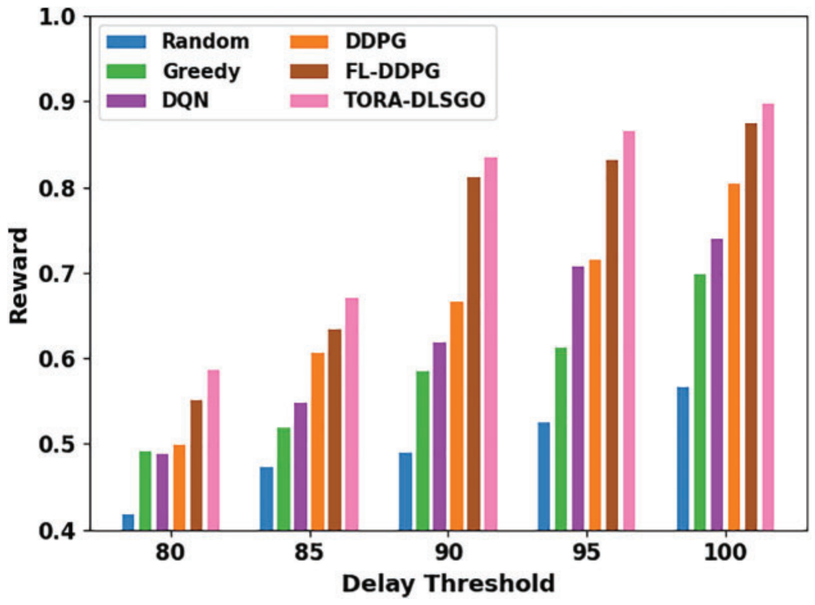

Figure 7: Reward analysis of TORA-DLSGO approach under distinct delay threshold

A brief set of delay (DEL) inspections of the TORA-DLSGO approach is inspected under distinct IoT index values in Table 5 and Fig. 8. The attained outcomes pointed out the superior efficiency of the TORA-DLSGO technique under each IoT index value. For example, with an IoT index of 1, the TORA-DLSGO approach results in minimalized DEL of 0.0434. At the same time, the random, greedy, DQN, DDPG, and FL-DDPG methods attain the highest DEL of 0.0992, 0.0677, 0.0759, 0.0552, and 0.0853 correspondingly. Along with that, with an IoT index of 5, the TORA-DLSGO method results in decreased DEL of 0.0227, whereas the random, greedy, DQN, DDPG, and FL-DDPG models reach the highest DEL of 0.0316, 0.0989, 0.0874, 0.0722, and 0.0675 correspondingly. Similarly, with an IoT index of 9, the TORA-DLSGO model results in a reduced DEL of 0.0230. In contrast, the random, greedy, DQN, DDPG, and FL-DDPG approaches obtain maximized DEL of 0.0992, 0.0997, 0.0798, 0.0811, and 0.0709 correspondingly.

Table 5: Delay analysis of TORA-DLSGO approach with other systems under distinct IoT index

	Delay					
IoT index	Random	Greedy	DQN	DDPG	FL-DDPG	TORA-DLSGO
0	0.0311	0.0533	0.0300	0.0363	0.0287	0.0196
1	0.0992	0.0677	0.0759	0.0552	0.0853	0.0434
2	0.0992	0.0314	0.0675	0.0808	0.0772	0.0224
3	0.0287	0.0651	0.0539	0.0992	0.0921	0.0214
4	0.0997	0.0952	0.0780	0.0850	0.0806	0.0227
5	0.0316	0.0989	0.0874	0.0722	0.0675	0.0227
6	0.0306	0.0992	0.1000	0.0989	0.0523	0.0243
7	0.0329	0.0989	0.0850	0.0575	0.0541	0.0332
8	0.0984	0.0994	0.0657	0.0568	0.0774	0.0269
9	0.0992	0.0997	0.0798	0.0811	0.0709	0.0230

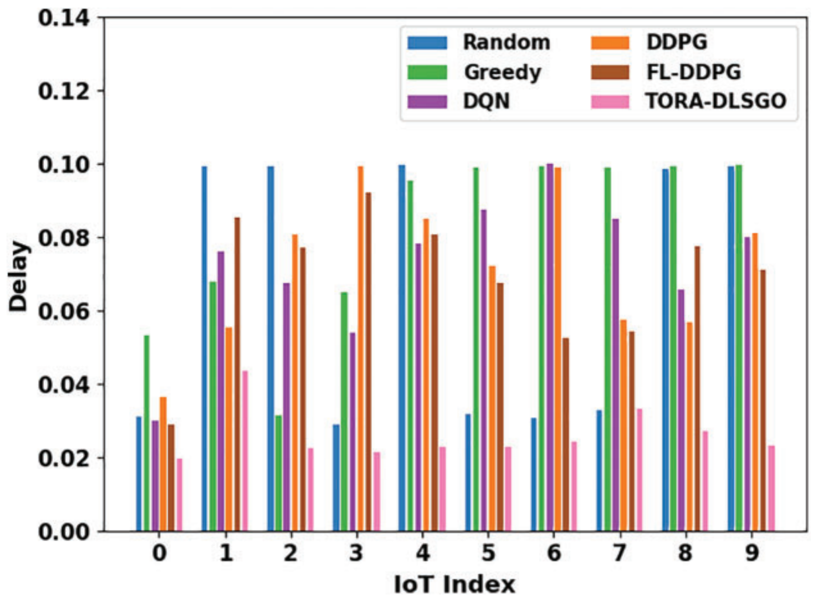

Figure 8: Delay analysis of TORA-DLSGO approach under distinct IoT index

4 Conclusion

In this study, we have developed a new TORA-DLSGO algorithm for task offloading and resource allocation in the IoT-based MEC environment. The proposed TORA-DLSGO technique addressed the resource management issue in the MEC server which enables an optimum offloading decision to minimize the system cost. In addition, an objective function is derived based on the minimization of energy consumption subject to the latency requirements and restricted resources. In the presented TORA-DLSGO technique, the DBN model is used for optimum offloading decision-making purposes. Finally, the SGO algorithm is used for the parameter tuning of the DBN model. The simulation results exemplify that the TORA-DLSGO technique outperformed the existing model in the reduction of client overhead in the MEC systems. In future, metaheuristic-based task scheduling schemes can be designed to optimize makespan in the IoT-based MEC environment.

Funding Statement: This research was supported by the Technology Development Program of MSS (No. S3033853).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Dong, Y. Xia and J. Kamruzzaman, "Quantum particle swarm optimization for task offloading in mobile edge computing," *IEEE Transactions on Industrial Informatics*, 2022. https://doi.org/10.1109/TII.2022.3225313
[2] L. Tan, Z. Kuang, J. Gao and L. Zhao, "Energy-efficient collaborative multi-access edge computing via deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, 2022. https://doi.org/10.1109/TII.2022.3213603
[3] L. Sun, L. Wan and X. Wang, "Learning-based resource allocation strategy for industrial IoT in UAV-enabled MEC systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5031–5040, 2020.
[4] S. Talwani, J. Singla, G. Mathur, N. Malik, N. Z. Jhanjhi *et al.,* "Machine-learning-based approach for virtual machine allocation and migration," *Electronics*, vol. 11, no. 19, pp. 3249, 2022.
[5] S. K. Mishra, S. Mishra, A. Alsayat, N. Z. Jhanjhi, M. Humayun *et al.,* "Energy-aware task allocation for multi-cloud networks," *IEEE Access*, vol. 8, pp. 178825–178834, 2020.
[6] G. G. Wang, M. Lu, Y. Q. Dong and X. J. Zhao, "Self-adaptive extreme learning machine," *Neural Computing and Applications*, vol. 27, no. 2, pp. 291–303, 2016.
[7] Y. Wang, X. Qiao and G. G. Wang, "Architecture evolution of convolutional neural network using monarch butterfly optimization," *Journal of Ambient Intelligence and Humanized Computing*, 2022. https://doi.org/10.1007/s12652-022-03766-4
[8] W. Fan, Z. Chen, Z. Hao, Y. Su, F. Wu *et al.,* "DNN deployment, task offloading, and resource allocation for joint task inference in IIoT," *IEEE Transactions on Industrial Informatics*, 2022. https://doi.org/10.1109/TII.2022.3192882
[9] H. Lu, C. Gu, F. Luo, W. Ding, S. Zheng *et al.,* "Optimization of task offloading strategy for mobile edge computing based on multi-agent deep reinforcement learning," *IEEE Access*, vol. 8, pp. 202573–202584, 2020.
[10] Y. Wang, J. Yang, X. Guo and Z. Qu, "Satellite edge computing for the internet of things in aerospace," *Sensors*, vol. 19, no. 20, pp. 4375, 2019.
[11] Z. Wang, G. Xue, S. Qian and M. Li, "CampEdge: Distributed computation offloading strategy under large-scale AP-based edge computing system for IoT applications," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6733–6745, 2020.
[12] S. Mao, S. He and J. Wu, "Joint UAV position optimization and resource scheduling in space-air-ground integrated networks with mixed cloud-edge computing," *IEEE Systems Journal*, vol. 15, no. 3, pp. 3992–4002, 2020.
[13] H. Guo and J. Liu, "UAV-enhanced intelligent offloading for internet of things at the edge," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2737–2746, 2019.
[14] Y. Zhang, X. Zhou, Y. Teng, J. Fang and W. Zheng, "Resource allocation for multi-user MEC system: Machine learning approaches," in *Int. Conf. on Computational Science and Computational Intelligence (CSCI)*, Las Vegas, NV, USA, pp. 794–799, 2018.

[15] J. Gao, Z. Kuang, J. Gao and L. Zhao, "Joint offloading scheduling and resource allocation in vehicular edge computing: A two layer solution," *IEEE Transactions on Vehicular Technology*, 2022. https://doi.org/10.1109/TVT.2022.3220571

[16] L. Tan, Z. Kuang, L. Zhao and A. Liu, "Energy-efficient joint task offloading and resource allocation in OFDMA-based collaborative edge computing," *IEEE Transactions on Wireless Communications*, vol. 21, no. 3, pp. 1960–1972, 2021.

[17] X. Deng, J. Yin, P. Guan, N. N. Xiong, L. Zhang *et al.,* "Intelligent delay-aware partial computing task offloading for multi-user Industrial Internet of Things through edge computing," *IEEE Internet of Things Journal*, 2021. https://doi.org/10.1109/JIOT.2021.3123406

[18] Q. Chen, Z. Kuang and L. Zhao, "Multiuser computation offloading and resource allocation for cloud–edge heterogeneous network," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3799–3811, 2021.

[19] X. Zhu and M. Zhou, "Multiobjective optimized cloudlet deployment and task offloading for mobile-edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 20, pp. 15582–15595, 2021.

[20] F. Dai, G. Liu, Q. Mo, W. Xu and B. Huang, "Task offloading for vehicular edge computing with edge-cloud cooperation," *World Wide Web*, pp. 1–19, 2022. https://doi.org/10.1007/s11280-022-01011-8

[21] X. Huang, Y. Cui, Q. Chen and J. Zhang, "Joint task offloading and QoS-aware resource allocation in fog-enabled Internet-of-Things networks," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7194–7206, 2020.

[22] T. Alfakih, M. M. Hassan, A. Gumaei, C. Savaglio and G. Fortino, "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA," *IEEE Access*, vol. 8, pp. 54074–54084, 2020.

[23] H. Zhao, J. Liu, H. Chen, J. Chen, Y. Li *et al.,* "Intelligent diagnosis using continuous wavelet transform and gauss convolutional deep belief network," *IEEE Transactions on Reliability*, 2022. https://doi.org/10.1109/TR.2022.3180273

[24] N. M. Alfaer, H. M. Aljohani, S. Abdel-Khalek, A. S. Alghamdi and R. F. Mansour, "Fusion-based deep learning with nature-inspired algorithm for intracerebral haemorrhage diagnosis," *Journal of Healthcare Engineering*, vol. 2022, pp. 1–12, 2022.

[25] X. Chen and G. Liu, "Federated deep reinforcement learning-based task offloading and resource allocation for smart cities in a mobile edge network," *Sensors*, vol. 22, no. 13, pp. 4738, 2022.