

Energy efficient resource allocation using deep reinforcement learning with starling murmuration optimization in cloud

Yogeetha Bangalore Rajanna, Anandaraj Shanthi Pichandi

Department of Computer Science and Engineering, Presidency University, Bengaluru, India

Article Info

Article history:

Received May 24, 2025

Revised Jul 19, 2026

Accepted Aug 12, 2026

Keywords:

Cloud computing
Deep reinforcement learning
Energy efficiency
Resource allocation
Starling murmuration
optimization
Task scheduling

ABSTRACT

Cloud computing is one of the widely used technologies due to its advanced features like pay -per-use, scalability, flexibility, and many more. It initially aims at fulfilling users' needs for accessing computing resources and purchasing cloud services as necessary. This is all done within the framework of on-demand resource sharing, which is facilitated by internet-based applications. Efficient resource allocation (RA) in the cloud includes challenges of high-dimensional state spaces and scalability, as the tasks keep increasing. To solve this problem, deep reinforcement learning (DRL) with Taylor-based optimization neighborhood starling murmuration optimization (TBONSMO) algorithm model is used for RA according to the tasks efficiently. The integration of DRL with deep neural networks enhances task scheduling by enabling adaptive and efficient RA. This approach improves the decision-making processes, optimizes energy usage as well as system performance in response to varying workloads and operational demands. The experimental results of the proposed TBONSMO method were calculated as follows: energy consumption of 30.58 kilo watt per hour and resource utilization of 99.54%, which is lesser when compared to the existing deep Q-network (DQN) approach.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Yogeetha Bangalore Rajanna

Department of Computer Science and Engineering, Presidency University
Bengaluru, India

Email: yogeetha.20223CSE0029@presidencyuniversity.in

1. INTRODUCTION

Cloud computing is a widely used cutting-edge method to compute and store a huge amount of data with minimum cost, wherein the data owners utilize their data from the cloud server. Due to the dynamic environment, the efficient placement/allocation of resources in physical servers is a critical endeavor, which directly impacts the resource utilization and overall performance of the cloud services [1]. However, when the user demand for cloud services increases, as well as when the large-scale resources in cloud data centers experience a significant increase in energy consumption, it results in higher operational costs [2], [3]. The various resources requested by the users have diverse input/output (I/O), memory, network, and processing requirements, therefore accordingly the resources are allocated for each task [4]–[6]. As the user demand rapidly changes, the optimal allocation of resources in the cloud becomes increasingly challenging, which leads to inefficient resource allocation (RA) and higher energy consumption [7]. To solve these issues, the resources must be allocated appropriately as per the tasks before they are even processed [8], [9].

The efficiency of cloud computing heavily depends on the task scheduler and how effectively the tasks are managed for allocating the most appropriate virtual resources [10]. The inappropriate RA also affects various parameters, such as energy consumption and makespan, leading to more consumption of energy and higher execution time, thereby impacting the quality of service (QoS) [11], [12]. The traditional

methods used for RA in the cloud failed to consider the distributed and automated nature of the cloud environment, which makes the selection of optimal resources in the model difficult. In existing research, several methods like greedy approaches, approximation algorithms, reinforcement learning, and bio-inspired optimization algorithms are used for allocating resources according to the tasks. However, these existing RA algorithms have several limitations that affect the selection of optimal resources with less energy consumption. Recently, metaheuristic optimization algorithms like particle swarm optimization (PSO), ant colony optimization (ACO), genetic algorithm (GA), and whale optimization algorithm (WOA), have been used for RA in the cloud [13]. To ensure energy-efficient resource allocation (EERA) in this research, the metaheuristic optimization is integrated with a reinforcement learning model to allocate the resources effectively [14], [15]. However, these existing metaheuristic algorithms face challenges in finding the best solution due to limitations such as a lack of search diversity and getting stuck in local optima, which greatly impacts RA. To overcome this problem, a deep reinforcement learning (DRL) with Taylor-based optimization neighborhood starling murmuration optimization (TBONSMO) algorithm model is proposed to efficiently allocate resources for the scheduled tasks.

The advantages and limitations of existing RA methods utilized for allocating resources for tasks in the cloud are discussed as follows: Mangalampalli *et al.* [16] presented a reinforcement learning model known as the deep Q-network (DQN) model, which was utilized to solve task scheduling difficulties. A deep-based task scheduling algorithm (DRLBTSA) was used to find the optimal resources by considering the priority of the task while minimizing makespan, service level agreement (SLA) violation, and energy consumption. However, the DRLBTSA model struggled with scalability difficulties and did not effectively manage the increased complexity of task scheduling when the number of tasks and resources grew. Belgacem *et al.* [17] presented an RA model that involved an intelligent multi-agent system and a reinforcement learning method (IMARM). The agent's performance in the reinforcement learning model in turn improved the performance of cloud RA and reduced the load imbalance. The presented IMARM method achieved less execution time, but there was an increase in energy consumption that affected EERA. However, the presented IMARM-based virtual machine (VM) allocation process had increased overhead due to a rise in the task volume, leading to reduced system reliability and higher dependencies on fault tolerance mechanisms. Sharma *et al.* [18] designed an energy-efficient VM allocation algorithm based on the multi-dimensional virtual machine migration (MDVMM) algorithm. The designed VM allocation algorithm reduced the energy consumption and limited the resource wastage, which facilitated the selection of optimal physical machines (PMs) at the cloud data center. Moreover, the designed MDVMM algorithm reduced the energy consumption, and SLA violations were avoided through selection of appropriate VM migrations. However, this MDVMM approach had a drawback as it was unable to ensure an optimal solution at certain conditions, which was required to achieve multi-objective VM allocation with minimum energy consumption.

Siddesha *et al.* [19] proposed a RL approach that was utilized to maximize the reward mechanism and design the policy as per task scheduling actions. Task scheduling helps in efficient task allocation and minimizes energy consumption and the reinforcement approach was utilized to balance load and resource efficiency for better performance of the model. However, the RL method struggled to adapt towards the dynamic changes in the cloud environment, such as varying VM capabilities, which potentially affected the efficiency of task allocation and energy consumption. Khan and Santhosh [20] implemented a hybrid model which was a combination of particle swarm and grey wolf optimization (PSGWO) algorithms to identify the most suitable VM for the tasks. A task scheduling model was used to calculate task execution time and monitor VM status during the execution to enhance efficient resource utilization. Task scheduling led to enhanced RA, reduced task execution time, and increased adaptability in dynamic conditions that in turn, improved performance. However, the PSGWO method caused computation overhead due to limited resources, leading to low efficiency of the model. The key contributions of this research are described as follows: the DRL-based RA model optimizes multiple objectives such as cost minimization, energy efficiency, and QoS to allocate the resources for the specific tasks in an effective manner in the cloud to avoid overutilization and underutilization. The proposed starling murmuration optimization (SMO) algorithm is used to fine-tune the DRL model to prevent selection of sub-optimal resources, which ensures efficient RA in the cloud with less energy consumption.

The remaining part of this paper is structured as follows: section 2 explains the proposed TBONSMO approach. The experimental results are illustrated in section 3. Section 4 depicts the discussion of results obtained by proposed method. Section 5 represents the conclusion.

2. METHOD

An energy-efficient model is developed to allocate resources in cloud computing. This utilizes DRL to find optimal resources for processing the tasks efficiently [21]. This DRL model is further enhanced by the

SMO algorithms with two strategies that adapt the user demands dynamically and allocate the resources to manage the tasks with less energy consumption. The proposed RA framework involves four phases: system model, constraints, DRL-based RA, and the proposed SMO algorithm. Figure 1 represents the proposed EERA framework.

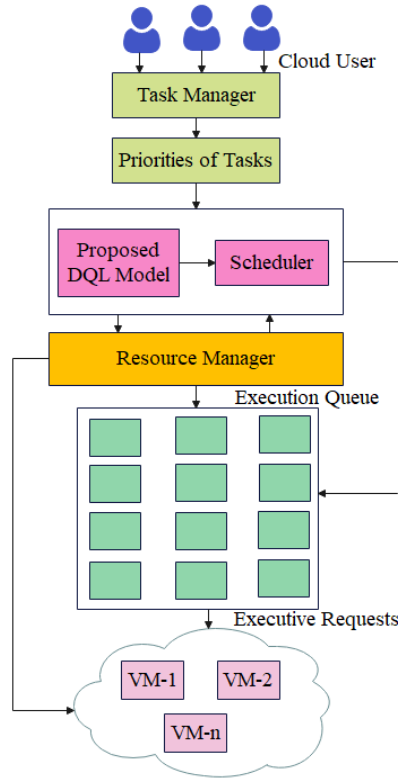


Figure 1. Proposed framework for EERA in cloud environment using DRL-SMO algorithm

2.1. System model

In this research, RA for the tasks in the cloud having industrial equipment (IE) and mobile edge computing (MEC) servers are denoted as d and s . It is assumed that the time interval is split into T time slots. For every time slot t , the IE server d creates $I^d(t)$ tasks by distribution as in (1). Where $a_i^d(t)$, $b_i^d(t)$, and $c_i^d(t)$ represent the size of data, result and number of CPU cycles. After the tasks are generated, constraints are framed to ensure EERA in the cloud.

$$w_i^d(t) = \{a_i^d(t), b_i^d(t), c_i^d(t)\} \quad (1)$$

2.2. Energy consumption model

The main objective of this research is to ensure EERA for the tasks in the cloud. The power consumed by the IE and MEC servers in each moment to execute tasks is mathematically represented in (2) and (3). Here ξ denotes effective switched capacitance. Therefore, the energy estimation for IE and MEC is obtained using (4) and (5). Where $p_i^d(t)$ and $T_i^d(t)$ represent power and delay of IE; $p_i^s(t)$ and $T_i^s(t)$ represent power and delay of MEC server respectively.

$$p_i^d(t) = \xi [f_i^d(t)]^3 \quad (2)$$

$$p_i^s(t) = \xi [f_i^s(t)]^3 \quad (3)$$

$$E_i^d(t) = p_i^d(t) T_i^d(t) \quad (4)$$

$$E_i^s(t) = p_i^s(t) T_i^s(t) \quad (5)$$

2.3. Proposed resource allocation model

For the RA in the cloud, a DRL model is utilized [21], [22] for solving the optimization problem which involves five components known as action space (A), state transition probability (P), state space (S), discount factor, and finally reward function (R) γ . When time slot t is started, the agent in the DRL algorithm decides the action based on the selection of (A) and the condition of the current system (S). After that, the environment in DRL model updates the system's next state s_{t+1} and then receives feedback from the environment to evaluate action effectiveness. Then, the agent develops a policy μ to map the state and actions with respect to the interaction between agent and environment, as represented in (6).

$$\mu: S \rightarrow P(A) \quad (6)$$

The fundamental five elements of DRL model are briefly explained as follows:

- i) State space: once the time slot t begins, the agent starts to acquire information regarding state of the current system environment. State space in time slot t is given in (7).

$$s_t = \{E(t), F(t), D(t), S(t), K(t)\} \quad (7)$$

Where $F(t)$ and $E(t)$ indicates available computing and power resources; $K(t)$ and $D(t)$ indicate position and size of all tasks; and $S(t)$ denotes accessible storage resources.

- ii) Action space: after determination of current state s_t , the agent takes the decision based on the given (8).

$$a_t = \{B(t), P(t), s(t), X(t), f(t)\} \quad (8)$$

Here $P(t)$, $f(t)$, $B(t)$, and $s(t)$ indicate allocated power, computing, bandwidth, and storage resources; and $X(t)$ denotes task offloading decisions.

- iii) State transition probability: in state transition, $p(s_t + 1 | s_t, a_t)$ denotes probability of s_{t+1} for the given s_t and selected a_t . Also, the agent has no previous information about $p(s_{t+1} | s_t, a_t)$, which is only obtained through the environment in DRL algorithm.
- iv) Reward function: the reward is denoted as r_t , which refers to immediate reward established while doing action in the state s_t . To achieve minimum energy-delay cost, the reward r_t in each time slot s_t is formulated as negative for the average cost of IE tasks. The mathematical representation of reward is given in (9).

$$r_t = \sum_{d=1}^D \sum_{i=1}^{I_d} \frac{1}{D} \frac{1}{I_d} C_i^d(t) \quad (9)$$

Since the optimization problem is a high dimensional one, the DRL method is used to address this issue to obtain an optimal policy that increases the long-term reward as depicted in (10).

$$R_t = \sum_{\tau=1}^T \gamma^{t-\tau} r_\tau \quad (10)$$

Wherein R_t denotes the long-term reward.

However, the limitations of this DRL model are that it explores sub-optimal solutions and the occurrence of slow convergence, which lead to inefficient RA. Thus, an optimization algorithm is used to address this issue, which optimizes the policy of DRL for faster convergence [23], [24] and guides the agent to explore diverse regions that prevent the model from getting stuck in local optima. In this research, a SMO algorithm is used for enhancing the DRL model to efficiently allocate resources in the cloud.

2.3.1. Proposed Taylor-based optimization neighborhood starling murmuration optimization algorithm

The SMO algorithm is inspired by the events occurring in nature regarding the murmuration of starlings. During this time, several flocks are grouped to form coordinated flight patterns in the sky. At the time of murmuration, the starlings split and form flocks again in a highly synchronized manner. These flocks frequently change their direction of movement to escape from the predators, but under some conditions, due to uncertainty in the flock movements, the cohesion is affected between them, which lead the predators to hunt these wayward starling birds. Thus, a dynamic multi-flock construction is used to represent the starling behavior in every iteration that moves these birds to another flock population. To search and refine the best resources, the movement strategy of the SMO algorithm is categorized into three behaviors of starling birds which are diving, whirling, and separating. Before that, the starling birds are initialized in the search space for search solutions.

Initialization: at first, a solution candidate (x) matrix is determined by random values within a search space, that denotes the initial positional vectors of a starling bird. The solution matrix is initialized by the positional vectors of the starlings as given in (11).

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} x_1^1 & x_1^1 & \dots & x_1^d \\ x_2^1 & x_2^1 & \dots & x_2^d \\ \vdots & \vdots & \dots & \vdots \\ x_N^1 & x_N^1 & \dots & x_N^d \end{bmatrix} \quad (11)$$

Where x denotes the candidate solution of the prey, d indicates dimensions, and N refers to the number of candidates. If the solution is moved using an exploration and exploitation strategy, then it sometimes extends beyond the scope of the search space to arrive at an optimal solution. Therefore, it is necessary to establish boundary conditions to ensure that the search space remains within a surrounded area. The boundary conditions are expressed mathematically in (12). Where X_{min} and X_{max} denote the minimum value of the current solution and the maximum attainable value.

$$x_j^i(t) = Range \dots [X_{min} - X_{max}]; \text{ where } \begin{cases} i = 1, 2, \dots, d \\ j = 1, 2, \dots, N \end{cases} \quad (12)$$

Flock construction: the starling birds form the murmuration M wherein a group of starlings is divided to improve the population diversity by using a separate search technique. After getting divided from the murmuration, each flock flies and displays behaviors of diving and whirling search strategy. The flock construction of the starlings is done by using (13).

$$P_{sep} = \frac{\log(t+d)}{2 \log(1t_{max})} \quad (13)$$

Here, P_{sep} represents a new population. During the separating strategy, the position update of starlings between various t flocks is mathematically expressed in (14).

$$x_i^{t+1} = x_i^t + \mathcal{D} \times (X_r^t - x_r^t) \quad (14)$$

Where x_i^t and x_i^{t+1} denote the current position and next position of the starlings: X_r^t and X_r^t indicate the starling's random position in the selected flock and entire population; Q indicates the separation operator, which maintains the population diversity by using a quantum harmonic oscillator. The position of every starling in each flock is updated by using a diving or whirling strategy. The starlings' selection strategy is done using (15).

$$x_i^{t+1} = \begin{cases} \text{Diving Strategy} & Q_q < \mu_q \\ \text{Whirling Strategy} & Q_q \geq \mu_q \end{cases} \quad (15)$$

Where Q_q represents the quality of the flock q ; μ_q indicates the average quality of all flocks in the population. The separation operator in SMO algorithm is mathematically formulated in (16).

$$Q_q(t) = \frac{\sum_{i=1}^k \frac{1}{N} \sum_{j=1}^N f v_{ij}(t)}{\frac{1}{n} \sum_{i=1}^n f v_{qi}(t)} \quad (16)$$

Where $f v_{ij}$ indicates fitness values of i^{th} starling in the flock population and k denotes the count of a flock in a murmuration M . In this SMO algorithm, the diving behavior is utilized in the exploration phase, and the whirling behavior is included in the exploitation phase, respectively. The brief explanation about the diving and whirling strategies for starling position update is mentioned in the actual SMO algorithm [25]. The mathematical expression of the whirling behavior in the exploitation phase used to refine the optimal resources for tasks is given in (17).

$$x_i(t+1) = x_i(t) + \cos(r_a) \times (X_R(t) - X_N(t)) \quad (17)$$

Here, r_a denotes random values between $[0, 1]$, X_R indicates a member from the flocks' population and X_N represents different starlings that are not selected from the previous iteration among the flocks.

However, the proposed SMO algorithm faces challenges in selecting an optimal solution due to its drawbacks such as falling in local optima and lack of global exploration, which affect the implementation of

EERA in cloud. Therefore, in this research two improvisation strategies are utilized to enhance the RA process by the Taylor-based optimal neighborhood strategy (TBON) and the dynamic opposition strategy (DOS), which are explained as follows: the proposed TBON strategy is an extension of the basic optimal neighborhood strategy. Unlike the conventional technique, this proposed Taylor strategy carefully develops the search ability of the best agent, which also prevents SMO from falling into the local optimum. This proposed strategy enhances the behavior of the respective member that helps to explore more potential solutions located near the search area. This diverse exploration assists the proposed SMO algorithm to increase learning ability of agent and enhance the exploitation process. An optimal neighborhood is mathematically represented by (18).

$$x_i(t+1) = (1 \pm \sigma \times (0,1))^\alpha \times x_{best}(t) \oplus f(x_i(t), x_j(t), x_{best}(t), C) \quad (18)$$

Where $\oplus$ denotes mathematical operators among multiplication, addition, division, and subtraction; x_{best} indicates global best solution amongst the obtained solutions and also denotes the representative member $|\psi(R_D)\rangle$; The operator which does not involve the candidate solution is denoted by C ; σ and α are the control parameters; U stands for any random number. Upward and downward quantum dives are the two diving behaviors of starlings, which are updated by the proposed TBON strategy as mathematically expressed in (19) and (20). Wherein ψ^{down} and ψ^{up} denote downward and upward quantum dives.

$$|\psi(t+1, x_i)\rangle = 1 + \alpha(\sigma \times U(0,1)) + \frac{\alpha(\alpha-1)}{2!} (\sigma \times U(0,1))^2 + \dots + \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!}, |\psi(R_D)\rangle - |\psi^{down}(x_i)\rangle \times (|\psi(x_i)\rangle - |\psi(x_r)\rangle) \quad (19)$$

$$|\psi(t+1, x_i)\rangle = 1 + \alpha(\sigma \times U(0,1)) + \frac{\alpha(\alpha-1)}{2!} (\sigma \times U(0,1))^2 + \dots + \frac{\alpha(\alpha-1)\dots(\alpha-n+1)}{n!}, |\psi(R_D)\rangle - |\psi^{up}(x_i)\rangle \times (|\psi(x_i)\rangle - |\psi(x_j)\rangle + |\psi(\delta_1)\rangle) \quad (20)$$

The proposed SMO algorithm has another drawback, as it often fails to traverse the entire search space while dealing with complex problems. This makes it difficult for the model to perform global exploration. Thus, a DOS is used to maximize the search space dynamically, which helps to extend the search space that assists in attaining any global solution probability. The process of the DOS is described as follows: To create an opposition point $\bar{x}$ according to x position as expressed mathematically in (21).

$$\bar{x}_i, d = x_d^{min} + x_d^{max} - x_i, d = 1, 2, \dots, Dim \quad (21)$$

Then, the position of the candidate solution $\bar{x}$ depends on a random number that provides a new position x_r for the candidate solution as is shown mathematically in (22).

$$x_r = rand(\bar{x}), rand \in [0, 1] \quad (22)$$

Then, DOS is employed for updating the position of individuals by randomly when the present location x is near x_r , and move from the current position into a different direction, which is mathematically formulated in (23).

$$\bar{x}_{d0} = x + rand(x_r) = x \quad (23)$$

The proposed RA model based on the DRL approach enables dynamic RA in cloud computing by learning policies that minimize energy utilization also maintains the service quality. Moreover, an optimization algorithm, SMO is combined with DRL to enhance learning speed, avoid local optima, and fine-tune decisions. The integration of DRL with SMO algorithm and the dual strategies used to overcome the limitations of the SMO algorithm have improved resource utilization, which significantly minimized the energy consumption in cloud environments.

3. RESULTS AND DISCUSSION

Performance analysis of the TBONSMO model-based EERA for implementation in cloud computing is illustrated in this section. The simulation of the TBONSMO-based VM allocation is performed in cloud sim tool with a system configuration of Windows 10, Intel i5 processor, and 16 GB RAM. Quantitative and qualitative analyses of the proposed method as well as the existing state-of-the-art

approaches is represented in subsection 3.1, while the comparative analysis of the same is given in subsection 3.2. To estimate the effectiveness of the TBONSMO for implementing EERA, the different performance measures utilized are energy consumption (kilo watt/hour ((KW/h)), makespan (millisecond (ms)), throughput (number of jobs per second), execution time (seconds (s)), and resource utilization (%).

3.1. Quantitative and qualitative analysis

Performance evaluation of the TBONSMO-based RA model is analyzed in this section. The state-of-the-art methods that were previously utilized for VM allocation in the cloud are the bat optimization algorithm (BOA), manta ray foraging optimization (MRFO), graylag goose optimization (GGO), and fennec fox optimization (FFO) algorithms. These algorithms are utilized for performance evaluation along with that of the proposed enhanced opposition-based (EOB)–MRFO algorithm. Figures 2 to 6 illustrate the performance results of EOB–MRFO as per a few diverse performance metrics.

Performance analysis of TBONSMO based EERA model in cloud computing is done in terms of makespan. Figure 3 shows the performance of proposed TBONSMO model that involves reinforcement learning. The performance is analyzed alongside those of the existing state-of-the-art methods. From Figure 3, it is clear that the proposed TBONSMO algorithm efficiently achieved lesser makespan in the RA of the respective tasks in the cloud environment.

Performance analysis of TBONSMO method is evaluated with respect to execution time (seconds). Figure 4 shows the performance of TBONSMO model that involves reinforcement learning. The performance is analyzed alongside those of the related recent-year optimization algorithms. Figure 4 clearly shows that the proposed TBONSMO algorithm efficiently allocates resources in lesser execution time when compared to existing methods used for RA in the cloud.

The performance analysis of TBONSMO based EERA model in the cloud is carried out in terms of throughput. Figure 5 shows the performance of the proposed TBONSMO model with reinforcement learning. The performance is analyzed alongside those of the existing recent-year optimization algorithms. It is clear from Figure 5 that TBONSMO algorithm efficiently allocates resources for the tasks in the cloud environment, resulting in high throughput while compared to other existing optimization algorithms.

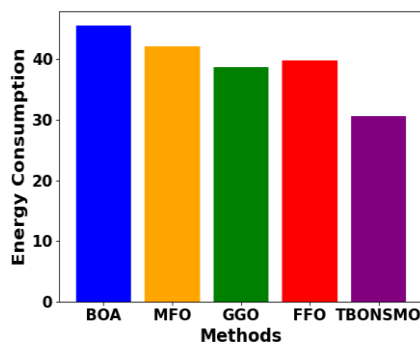


Figure 2. Performance of TBONSMO based on energy consumption (KW/h)

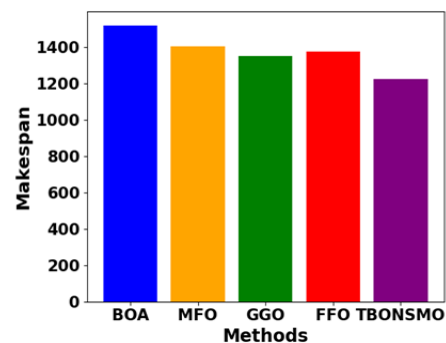


Figure 3. Performance of proposed TBONSMO in terms of makespan (ms)

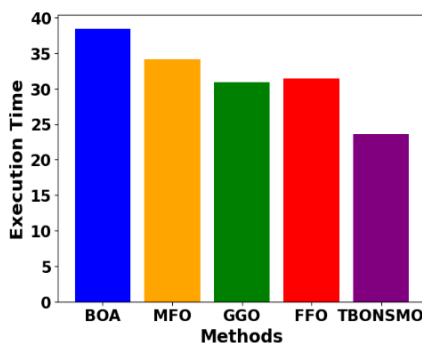


Figure 4. Performance of proposed TBONSMO in terms of execution time (s)

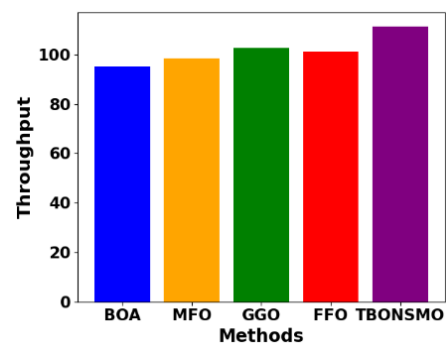


Figure 5. Performance of proposed TBONSMO in terms of throughput (number of jobs per second)

Performance analysis of TBONSMO method is evaluated in light of resource utilization. Figure 6 depicts the performance of the proposed TBONSMO model with reinforcement learning alongside the existing state-of-the-art methods, to show how efficiently the model utilizes the resources. From Figure 6, it is clear that the proposed TBONSMO algorithm effectively utilized the resources for allocation to the respective tasks in a better manner than other state-of-the-art methods.

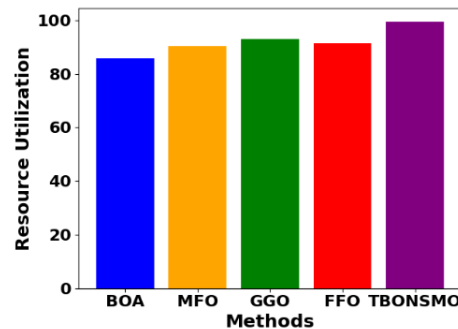


Figure 6. Performance of proposed TBONSMO in terms of resource utilization (%)

3.2. Comparative analysis

The comparison analysis of the proposed TBONSMO algorithm for EERA in the cloud is analyzed in this section. Existing RA techniques that are utilized in the cloud, such as DRLBTSA [16], IMARM [17], hybrid whale optimization algorithm (HWOA)-differential evolution operator (DEO) [19], and particle swarm-grey wolf optimization (PSGWO) [20] are utilized for comparison alongside the proposed EOB-MRFO algorithm. Table 1 illustrates the comparison results of the proposed method for various scenarios using varied performance metrics.

Table 1. Comparative analysis of proposed TBONSMO algorithm

Methods	Energy consumption (KW/h)	Makespan (ms)	Execution Time (s)	Throughput (number of jobs/ second)	Resource utilization (%)
DRLBTSA [16]	37.92	432.56	N/A	N/A	N/A
IMARM [17]	± 0.065	N/A	± 0.737	N/A	N/A
RL [19]	N/A	1105.80	N/A	96.80	99.50
Proposed TBONSMO	30.58	1223.20	23.56	111.50	99.54

4. DISCUSSION

The proposed model of the TBONSMO algorithm with DRL that is utilized for RA in the cloud attains greater performance than the existing RA methods. The proposed RA model achieves energy consumption of 30.58 J, makespan of 1,223.20 s, and resource utilization of 99.54, which are better outcomes than those of the existing RA works in the cloud environment. The proposed method utilizes Taylor series-based dynamic opposition learning strategies to address the limitations of the SMO algorithm that helps to enhance the RA process for the tasks. The existing RA models such as the DRLBTSA model [16] have limitations as these models struggle with scalability difficulties and are also not effective in task scheduling when number of tasks and resources increases. The IMARM model [17], based on a VM allocation process, has increased overhead due to rising task volume that reduces system reliability, leading to higher dependencies in the fault tolerance mechanism. The traditional RL methods based on RA face challenges in adapting to dynamic changes in the cloud environment, such as varying VM capabilities, that potentially affect the efficiency of RA and increase energy consumption.

5. CONCLUSION

The proposed DRL and TBONSMO algorithm model is employed to allocate resources by achieving energy efficiency depending on the tasks. The combination of DRL with an optimization allocation model is used to enhance the energy efficiency of the RA in the cloud environment to reduce energy consumption for allocating tasks. The combination of DRL with an SMO algorithm enhances task scheduling by enabling adaptive and efficient RA. This approach improves the decision-making processes, optimizes the

energy usage as well as the system performance, in response to varying workloads and operational demands. The combination of DRL with deep neural networks enhances task scheduling by enabling adaptive and efficient RA. The experimental results of the proposed TBONSMO method are computed as follows: 30.58 KW/h and resource utilization of 99.54%, which is optimally lesser when compared to the existing DRLBSTA approach. In the future, an advanced metaheuristic optimization method may be used to improve the security of RA in the cloud so that the confidentiality of the sensitive data is ensured.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Yogeetha Bangalore Rajanna	✓	✓		✓		✓	✓		✓		✓			✓
Anandaraj Shanthi Pichandi		✓	✓		✓	✓		✓		✓		✓	✓	

C : Conceptualization	I : Investigation	Vi : Visualization
M : Methodology	R : Resources	Su : Supervision
So : Software	D : Data Curation	P : Project administration
Va : Validation	O : Writing - Original Draft	Fu : Funding acquisition
Fo : Formal analysis	E : Writing - Review & Editing	

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] S. Nagarajan, P. S. Rani, M. S. Vinmathi, V. S. Reddy, A. L. M. Saleth, and D. A. Subhahan, "Multi agent deep reinforcement learning for resource allocation in container-based cloud environments," *Expert Systems*, vol. 42, no. 1, Jun. 2025, doi: 10.1111/exsy.13362.

[2] A. K. Samha, "Strategies for efficient resource management in federated cloud environments supporting infrastructure as a service (IaaS)," *Journal of Engineering Research*, vol. 12, no. 2, pp. 101–114, Jun. 2024, doi: 10.1016/j.jer.2023.10.031.

[3] S. D. A. Javaheri, R. Ghaemi, and H. M. Naeen, "An autonomous architecture based on reinforcement deep neural network for resource allocation in cloud computing," *Computing*, vol. 106, no. 2, pp. 371–403, Oct. 2024, doi: 10.1007/s00607-023-01220-7.

[4] C. Fang *et al.*, "DRL-driven joint task offloading and resource allocation for energy-efficient content delivery in cloud-edge cooperation networks," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 12, pp. 16195–16207, Jul. 2023, doi: 10.1109/TVT.2023.3297362.

[5] H. Hu, D. Wu, F. Zhou, X. Zhu, R. Q. Hu, and H. Zhu, "Intelligent resource allocation for edge-cloud collaborative networks: a hybrid DDPG-D3QN approach," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 8, pp. 10696–10709, Mar. 2023, doi: 10.1109/TVT.2023.3253905.

[6] H. Liu *et al.*, "Robustness challenges in reinforcement learning based time-critical cloud resource scheduling: a meta-learning based solution," *Future Generation Computer Systems*, vol. 146, pp. 18–33, Sep. 2023, doi: 10.1016/j.future.2023.03.029.

[7] F. Shi, "A genetic algorithm-based virtual machine scheduling algorithm for energy-efficient resource management in cloud computing," *Concurrency and Computation: Practice and Experience*, vol. 36, no. 22, Jul. 2024, doi: 10.1002/cpe.8207.

[8] S. Gurusamy and R. Selvaraj, "Resource allocation with efficient task scheduling in cloud computing using hierarchical auto-associative polynomial convolutional neural network," *Expert Systems with Applications*, vol. 249, Sep. 2024, doi: 10.1016/j.eswa.2024.123554.

[9] S. Prathiba and S. Sankar, "Energy-efficient resource allocation in cloud infrastructure using L3F-MGA and E-ANFIS," *Measurement: Sensors*, vol. 31, Feb. 2024, doi: 10.1016/j.measen.2023.100965.

[10] A. K. Sangaiah, A. Javadpour, P. Pinto, S. Rezaei, and W. Zhang, "Enhanced resource allocation in distributed cloud using fuzzy meta-heuristics optimization," *Computer Communications*, vol. 209, pp. 14–25, 2023, doi: 10.1016/j.comcom.2023.06.018.

[11] A. Moazeni, R. Khorsand, and M. Ramezanpour, "Dynamic resource allocation using an adaptive multi-objective teaching-learning based optimization algorithm in Cloud," *IEEE Access*, vol. 11, pp. 23407–23419, 2023, doi: 10.1109/ACCESS.2023.3247639.

- [12] S. Singhal, N. Gupta, P. Berwal, Q. N. Naveed, A. Lasisi, and A. W. Wodajo, "Energy efficient resource allocation in cloud environment using metaheuristic algorithm," *IEEE Access*, vol. 11, pp. 126135–126146, 2023, doi: 10.1109/ACCESS.2023.3330434.
- [13] M. Kumar, K. Dubey, S. Singh, J. K. Samriya, and S. S. Gill, "Experimental performance analysis of cloud resource allocation framework using spider monkey optimization algorithm," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 2, Nov. 2023, doi: 10.1002/cpe.7469.
- [14] S. Singh, P. Singh, and S. Tanwar, "Energy-aware resource allocation via MS-SLNO in cloud data center," *Multimedia Tools and Applications*, vol. 82, no. 29, pp. 45541–45563, May 2023, doi: 10.1007/s11042-023-15521-8.
- [15] A. Tarafdar, S. Sarkar, R. K. Das, and S. Khatua, "Power modeling for energy-efficient resource management in a cloud data center," *Journal of Grid Computing*, vol. 21, no. 1, Feb. 2023, doi: 10.1007/s10723-023-09642-5.
- [16] S. Mangalampalli, G. R. Karri, M. Kumar, O. I. Khalaf, C. A. T. Romero, and G. M. A. Sahib, "DRLBTSA: deep reinforcement learning based task-scheduling algorithm in cloud computing," *Multimedia Tools and Applications*, vol. 83, no. 3, pp. 8359–8387, Jun. 2024, doi: 10.1007/s11042-023-16008-2.
- [17] A. Belgacem, S. Mahmoudi, and M. Kihl, "Intelligent multi-agent reinforcement learning model for resources allocation in cloud computing," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 2391–2404, Jun. 2022, doi: 10.1016/j.jksuci.2022.03.016.
- [18] N. K. Sharma, S. Bojjagani, Y. C. A. P. Reddy, M. Vivekanandan, J. Srinivasan, and A. K. Maurya, "A novel energy efficient multi-dimensional virtual machines allocation and migration at the cloud data center," *IEEE Access*, vol. 11, pp. 107480–107495, Sep. 2023, doi: 10.1109/ACCESS.2023.3320729.
- [19] K. Siddesha, G. V. Jayaramaiah, and C. Singh, "A novel deep reinforcement learning scheme for task scheduling in cloud computing," *Cluster Computing*, vol. 25, no. 6, pp. 4171–4188, Jun. 2022, doi: 10.1007/s10586-022-03630-2.
- [20] M. S. A. Khan and R. Santhosh, "Task scheduling in cloud computing using hybrid optimization algorithm," *Soft Computing*, vol. 26, no. 23, pp. 13069–13079, Nov. 2022, doi: 10.1007/s00500-021-06488-5.
- [21] H. M. Do, T. P. Tran, and M. Yoo, "Deep reinforcement learning-based task offloading and resource allocation for industrial IoT in MEC federation system," *IEEE Access*, vol. 11, pp. 83150–83170, 2023, doi: 10.1109/ACCESS.2023.3302518.
- [22] L. Cheng, Y. Wang, F. Cheng, C. Liu, Z. Zhao, and Y. Wang, "A deep reinforcement learning-based preemptive approach for cost-aware cloud job scheduling," *IEEE Transactions on Sustainable Computing*, vol. 9, no. 3, pp. 422–432, May 2024, doi: 10.1109/TSUSC.2023.3303898.
- [23] P. V. Lahande, P. R. Kaveri, J. R. Saini, K. Kotecha, and S. Alfarhood, "Reinforcement learning approach for optimizing cloud resource utilization with load balancing," *IEEE Access*, vol. 11, pp. 127567–127577, 2023, doi: 10.1109/ACCESS.2023.3329557.
- [24] R. F. Mansour, H. Alhumyani, S. A. Khalek, R. A. Saeed, and D. Gupta, "Design of cultural emperor penguin optimizer for energy-efficient resource scheduling in green cloud computing environment," *Cluster Computing*, vol. 26, no. 1, pp. 575–586, Feb. 2023, doi: 10.1007/s10586-022-03608-0.
- [25] H. Zamani, M. H. N.-Shahraki, and A. H. Gandomi, "Starling murmuration optimizer: a novel bio-inspired algorithm for global and engineering optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 392, Mar. 2022, doi: 10.1016/j.cma.2022.114616.

BIOGRAPHIES OF AUTHORS



Yogeetha Bangalore Rajanna     is currently pursuing her Ph.D. at Presidency University, where she is dedicated to advancing research in the fields of green cloud computing and deep reinforcement learning. She holds an M.Tech. in Computer Science and Engineering from VTU University (2019) and a Bachelor of Engineering from M. S. Ramaiah Institute of Technology, also under VTU University (2014). Additionally, she earned a Diploma in Computer Science and Engineering from the Technical Board of Karnataka in 2011. With over four years of teaching experience, she currently serves as an assistant professor at Presidency University. Her commitment to academic excellence was recognized when she received the Best Project Award during the Open Day project exhibition at M. S. Ramaiah Institute of Technology. She has made contributions to the field through her research, with one of her published papers titled "Generative adversarial networks for self-supervised transfer learning in medical image classification". Based in Bengaluru, she continues to inspire students and colleagues through her passion for teaching and research. She can be contacted at email: yogeetha.2023CSE0029@presidencyuniversity.in.



Anandaraj Shanthi Pichandi     received B.E. (Computer Science and Engineering) degree from Madras University, Chennai in the year 2004, M.Tech. (Computer Science and Engineering) with Gold Medal from Dr. MGR Educational and Research Institute, University in the year 2007 (distinction with honor) and Ph.D. in August 2014. He is currently working as a professor and HoD-cyber security and internet of things under Department of Computer Science and Engineering at Presidency University, Bengaluru. He has 20+ years of teaching experience. His areas of interest include information security, data science, and networks. He wrote five book chapters in IGI global publications, Springer, and Elsevier, and authored five books. He filed 7 patents for the growth of research. He has published 50+ papers in various national and international journals, national and international conferences. He is serving as an editorial member for reputed journals. He also attended many national workshops, FDP, and seminars. He is a member of ISTE, CSI, a member of IACSIT, and a member of IAENG. He can be contacted at email: anandaraj@presidencyuniversity.in.