

Enhancing resource management in fog-cloud internet of things systems with deep learning-based task allocation

Vijayalakshmi Venkatesan, Saravanan Murugan

Department of Networking and Communications, Faculty of Engineering and Technology, SRM Institute of Science and Technology, Chennai, India

Article Info

Article history:

Received May 17, 2024

Revised Aug 9, 2024

Accepted Aug 20, 2024

Keywords:

Energy consumption

Fog computing

Quality of service demand

Resource utilization

Task scheduling

ABSTRACT

A fog-cloud internet of things (IoT) system integrates fog computing with cloud infrastructure to efficiently manage processing data closer to the source, reducing latency and bandwidth usage. Efficient task scheduling in fog-cloud system is crucial for optimizing resource utilization and minimizing energy consumption. Even though many authors proposed energy efficient algorithms, failed to provide efficient method to decide the task placement between fog nodes and cloud nodes. The proposed hybrid approach is used to distinguish the task placement between fog and cloud nodes. The hybrid approach comprises the parametric task categorization algorithm (PTCA) for task categorization and the multi metric forecasting model (MMFM) based on deep deterministic policy gradient (DDPG) recurrent neural networks for scheduling decisions. PTCA classifies tasks based on priority, quality of service (QoS) demands, and computational needs, facilitating informed decisions on task execution locations. MMFM enhances scheduling by optimizing energy efficiency and task completion time. The experimental evaluation outperforms the existing models, including random forest (RF), support vector machine (SVM), and k-nearest neighbors (KNN). The proposed result shows an accuracy rate of 89%, and energy is consumed 50% lesser than the existing models. The proposed research advances energy-efficient task scheduling, enabling intelligent resource management in fog-cloud IoT environments.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Vijayalakshmi Venkatesan

Department of Networking and Communications, Faculty of Engineering and Technology, SRM Institute of Science and Technology

Kattankulathur-603203, Chennai, Tamil Nadu, India

Email: vijayalv@srmist.edu.in

1. INTRODUCTION

The internet of things (IoT) and fog computing environments are rapidly developing technologies that facilitate the seamless communication and exchange of data between devices [1]. However, as connected devices grow, efficient task scheduling becomes crucial to optimize resource utilization and energy consumption. Energy-efficient task scheduling addresses the challenge of allocating tasks to appropriate resources within an IoT-fog environment that minimizes energy consumption while maximizing the overall system performance [2]. This is especially important as many devices in IoT-fog environments are battery-powered and have limited energy resources. One of the key motivations for energy-efficient task scheduling in IoT-fog environments is the need to extend the lifetime of devices [3]. By efficient scheduling of tasks and leveraging the computational capabilities of fog nodes, energy consumption can be minimized, ultimately prolonging the life of devices and reducing the need for frequent replacement.

The general utilization of the resources can be optimized by planning and scheduling the task based on the power they consume and the ability of the available resources [4], [5]. This helps in accomplishing tasks without much latency and improves the experience of users or is beneficial to user interactions. The dynamic nature of these environments, with devices joining and leaving the network frequently, makes it difficult to predict resource availability accurately. This uncertainty causes a challenge when scheduling tasks and may lead to suboptimal resource allocation. Furthermore, deciding where to execute tasks, either at the fog nodes or in the cloud, adds complexity to the task scheduling process. While executing tasks at the fog node can reduce latency and energy consumption, it might not always be feasible due to limited resources or specific application requirements.

The proposed methodology uses the parametric task categorization module (PTCM) algorithm which serves as the cornerstone of our task scheduling platform, providing a mechanism for classifying tasks based on multiple parameters such as priority, quality of service (QoS) requirements, and processing characteristics. Once tasks are classified by the parametric task categorization algorithm (PTCA) algorithm, the deep deterministic policy gradient (DDPG)-based multi-metric forecasting model (MMFM) model is used to optimize task scheduling decisions. DDPG is a Q-learning based reinforcement learning algorithm that takes policy gradient logic into account, built from experience replay and requires no prior model. It is composed of actor and a critic which make up the DDPG method. Whereas actor inputs states and outputs actions, critic takes state and two acts to output the Q-value [6]. For energy-efficient scheduling of tasks DDPG is utilized in fog-cloud systems and offers adaptive real-time adjustments, ensuring optimal resource allocation to meet QoS demands efficiently [7]. Its integration enables enhanced decision-making, surpassing conventional techniques by dynamically optimizing energy consumption and task completion time.

Qiu *et al.* [8] proposed a novel energy management algorithm using the DDPG method for wireless sensor networks with renewable energy sources demonstrates superior long-term performance and effective energy management in real-world scenarios. The proposed work by Li *et al.* [9] addresses energy consumption in Edge of things computing, a novel scaling mechanism monitors IoT application workloads, categorizes them by intensity, and dynamically adjusts processor speed to balance QoS and energy efficiency, though continuous monitoring may sometimes increase power consumption and degrade performance. Babar and Khan [10] examines a scalable energy-efficient paradigm that uses recursive clustering to prioritize high-priority tasks on fog servers, employing Euclidean distance for clustering and round-robin scheduling for cluster leaders, but faces challenges like overfitting and increased computational costs. Yu *et al.* [11] explores the integration of an unmanned aerial vehicle into a wireless IoT network, employing a fly-hover-communicate protocol for data collection and device charging. It formulates a multi-objective optimization problem which maximizes data rate and harvested energy. In a mobile edge computing system, a dynamic offloading mechanism based on deep Q network (DQN) is suggested in [12]. Long short-term memory (LSTM) and Q-learning for offloading destination selection were integrated [13] to forecast the channel gain in the network. The deep reinforcement learning (DRL)-based algorithm can produce higher accuracy and better self-correcting performance as compared to existing techniques [14]. As IoT systems continue to expand, managing the scalability and complexity of task scheduling becomes increasingly challenging [15]. DDPG offers a promising solution by adapting to evolving system dynamics and optimizing scheduling decisions in real-time.

The main contributions of the proposed work are as follows:

- Efficient task scheduling in fog-cloud IoT systems is crucial for optimizing resource utilization and minimizing energy consumption. Existing energy-efficient algorithms struggle to effectively determine task placement between fog and cloud nodes, complicating the decision of local versus cloud processing.
- The proposed hybrid approach classifies tasks based on priority, QoS demands, and computational needs, and optimizes energy efficiency and task completion time. This facilitates informed task placement and enhances overall system performance.
- The proposed hybrid approach outperforms existing models (random forest (RF), support vector machine (SVM), and k-nearest neighbors (KNN)) by achieving 89% accuracy in task scheduling, reducing response time to 80 ms, and energy consumption by 50%. This advances energy-efficient task scheduling and enables more intelligent resource management in fog-cloud IoT environments.

The article is organized as the proposed energy-efficient MMFM-DDPG scheduling framework in section 2. Section 3 shows the mathematical model for categorization and energy-efficient scheduling. Section 4 represents the simulation and result. Finally, section 5 describes the conclusion.

2. PROPOSED METHOD

Integrating fog and cloud computing in IoT presents unprecedented opportunities for advancing services and improving outcomes. Efficient task scheduling, facilitated by DDPG, emerges as a critical enabler in realizing these benefits. However, several challenges related to scalability, complexity, and QoS requirements

persist, warranting further research and innovation in this domain. As shown in Figure 1, the proposed system is designed to be scalable, adaptive, and modular to meet the varied requirements and constraints.

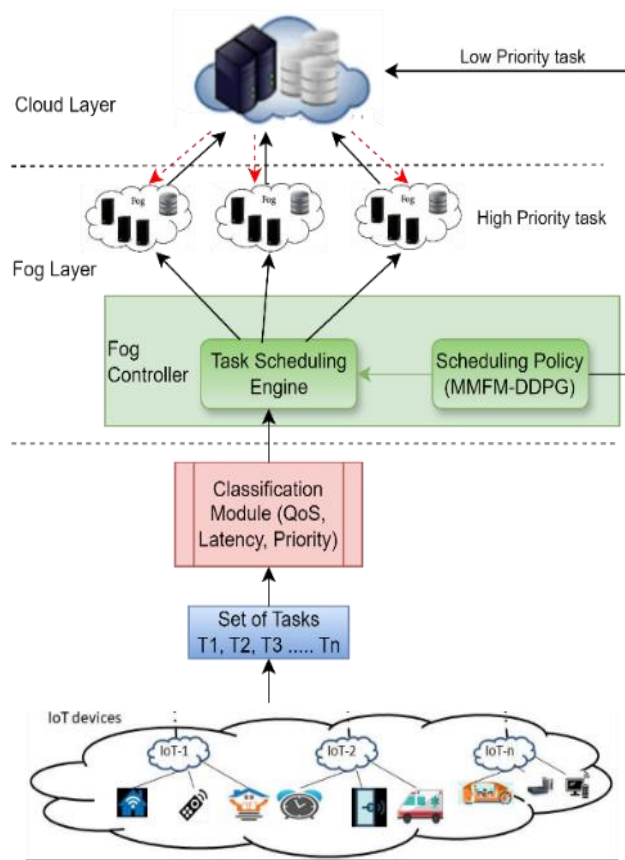


Figure 1. System architecture of task offloading approach

Task scheduling in distributed computing environments necessitates a systematic approach to allocate computational resources efficiently. The PTCM utilizes a decision tree-based approach to classify tasks into distinct categories based on various parameters such as priority, QoS requirements, and processing characteristics. It constructs a decision tree iteratively by selecting the most informative features and splitting the dataset based on those features until the entire dataset is correctly classified. Each decision tree node represents a feature, and each branch represents a possible feature value. The leaf nodes contain the class labels assigned to the tasks based on the selected features. This categorization ensures that tasks are prioritized and managed effectively, enabling the system to allocate resources that maximizes performance and meet user expectations.

Effective task scheduling also requires an ability to predict future workloads and adapt resource allocation accordingly. The MMFM integrates advanced machine learning techniques to provide this predictive capability. Utilizing a reinforcement learning framework, MMFM analyzes historical task data to forecast trends and patterns in workload demands. This foresight enables the system to proactively adjust its scheduling strategy, ensuring that computational resources are optimally utilized, response times are minimized, and QoS criteria are consistently met. DDPG is one type of reinforcement learning where the policy is learned directly without the use of a parametric maximum strategy through retesting the memory and constantly learning through valued ideas of Q-learning and policy gradient logics.

The DDPG algorithm is implemented by the actor and the critic. The critic takes a state and acts to give the Q-value, while the actor takes the states and gives actions. By iteratively adjusting its parameters in response to observed rewards and system variables, the DDPG-based MMFM model learns this policy. This enables it to adjust to changing environmental conditions and gradually optimize task scheduling decisions [16]. Once the tasks have been categorized and scheduled, the system's overall energy consumption may be computed by adding up the energy used by each task on each device. DDPG is used in fog-cloud systems for

task scheduling that is energy-efficient and provides adaptive real-time modifications to ensure optimal resource allocation to fulfil QoS demands effectively [17]. Through its integration, decision-making is improved, and it outperforms traditional methods by dynamically optimizing energy consumption and task completion time. By incorporating the MMFM, our system is equipped to forecast fluctuations in task workloads and adapt the scheduling strategy accordingly. This predictive functionality facilitates proactive resource allocation, ensuring optimal use of computational resources, minimizing response times, and meeting QoS criteria.

The proposed MMFM-DDPG algorithm shown in Algorithm 1 learns from historical task data, which includes previous workload patterns, resource utilization, and environmental conditions. Through iterative training and optimization, the model gains a deep understanding of the underlying dynamics that influence task scheduling within the system. The DDPG algorithm used in the forecasting model for efficient scheduling in fog-cloud based systems shown in Figure 2 aims to optimize the allocation of tasks between fog nodes and cloud servers to minimize energy consumption while maintaining the QoS.

Algorithm 1. Proposed multi-metric forecasting model with deep deterministic policy gradient (MMFM-DDPG)

```

Input: Classified tasks, Critic and Actor Networks
Output: Optimized task scheduling for energy efficiency and QoS fulfillment
Initialize MMFM Lists:
    Create lists for scheduled tasks ( $L_{st}$ ) and high-priority tasks ( $L_{ht}$ ).
Initialize DDPG:
    Initialize critic and actor networks for task scheduling decisions.
    Initialize a replay buffer to store experience tuples ( $s_t, a_t, r_t, s_{t+1}$ ).
MMFM Task Evaluation and Scheduling:
For each task T in the classified task list:
    Initialize  $T_{best} = \text{None}$ 
    For each task T in the list:
        If  $T \geq \text{MMFM}(T)$ :
             $T_{best} = T$ 
        Else if Priority(T) = high and usage(T) < MMFM(T):
            Append the task  $L_{ht} = L_{ht} \cup \{T\}$ 
            If  $\sum_{t \in L_{ht}} \text{usage}(t) \geq \text{MMFM}(T)$ :
                 $T_{best} = \arg \max_{t \in L_{ht}} \text{priority}(t)$ 
            End inner loop.
    If  $T_{best} \neq \text{None}$ :
         $L_{ht} = L_{ht} \cup \{T_{best}\}$ 
    Else:
        Task list = Task list  $\cup \{T\}$ 
    If  $T \neq \text{Null}$  and  $T_{best} = \text{None}$ :
         $L_{st} = L_{st} \cup \{T\}$ 
        If  $T_{best} \neq \text{None}$ :
             $L_{st} = L_{st} \cup \{T\}$ 
DDPG for Energy Efficiency and QoS:
For each episode from 1 to M:
    Initialize the starting state  $s_0$ .
    While the state is not a terminal state:
        Select an action  $a_t$ :
            
$$a_t = \begin{cases} \text{random action} & \text{with probability } \epsilon \\ \pi(s_t | \theta^\pi) & \text{otherwise} \end{cases}$$

        Execute action  $a_t$  and observe reward  $r_t$  and next state  $s_{t+1}$ .
        Store the experience tuple ( $s_t, a_t, r_t, s_{t+1}$ ) in the replay buffer.
        Update the current state to  $s_{t+1}$ .
        Sample a random mini-batch from the replay buffer.
        Set target value
            
$$y_j = r_j + \alpha Q(s_{j+1} | \theta^Q) | \theta^Q$$

        Update the critic network by minimizing the loss  $L_c$ :
            
$$L_c = N^{-1} \sum_{j=1}^N (y_j - Q(s_j, a_j | \theta^Q))^2$$

        Update the actor network using the gradient:
            
$$\nabla_{\theta^\mu} \cong N^{-1} \sum_{j=1}^N \nabla_{a_t} q(s_j, a_j | \theta^Q) \nabla_{\theta^\mu} \mu(s_j | \theta^\mu)$$

        Perform soft updates on the target networks:
            
$$\theta^Q = \tau \theta^Q + (1 - \tau) \theta^Q, \quad \theta^\mu = \tau \theta^\mu + (1 - \tau) \theta^\mu$$


```

In this algorithm, actor and critic neural networks are initialized to make task assignment decisions based on the current system state, which includes information about tasks, fog nodes, and cloud servers. During each episode, the system state is reset, and for each task, an action (task assignment) is selected using the current policy and exploration strategy. The action's impact is simulated in the fog-cloud IoT environment to compute a reward based on energy efficiency and QoS fulfillment. The experience tuples comprising the state, action, reward, next state, and termination flag are stored in a replay buffer. The actor and critic networks are then updated using the sampling policy gradient and loss function, while the target networks are updated using soft updates. This process iterates over multiple episodes to train the DDPG model to make efficient task scheduling decisions tailored for fog-cloud IoT systems.

Once trained, the MMFM continually analyzes incoming data to produce real-time forecasts of future workload trends. These forecasts are then incorporated into the task-scheduling process, enabling the system to allocate resources proactively based on anticipated demand. The integration of MMFM into the task-scheduling framework provides several key advantages. Firstly, it boosts the system's responsiveness by facilitating proactive resource allocation in anticipation of potential workload changes. This proactive approach guarantees smooth operation even in an environment of varying workloads and helps to reduce potential performance issues.

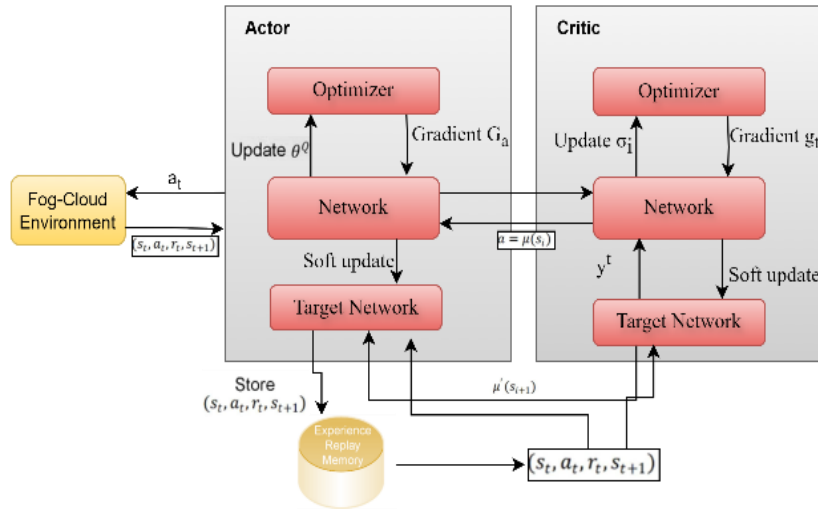


Figure 2. DDPG used in MMFM

3. MATHEMATICAL MODELLING

The parametric task categorization algorithm (PTCA) employs a decision tree-based approach to classify tasks into distinct categories based on predefined criteria. This algorithm uses entropy and information gain which is used to build a decision tree for task categorization. The feature space, represented by X , includes task characteristics, while Y represents the set of possible task categories. The decision tree learns a mapping $f: X \rightarrow Y$ that partitions the feature space into disjoint regions, each corresponding to a specific task category. This mapping is derived through an iterative process that maximizes information gain at each node, resulting in an effective classification model for tasks in IoT environments. Entropy is used to measure the uncertainty in a dataset based on the task categories such as priority, QoS demands and computational needs and is calculated as:

$$Entropy(D) = -\sum_{i=1}^n p_i \log_2(p_i) \quad (1)$$

where p_i represents the proportion of tasks belonging to class i , in the dataset D , and the number of different classes is represented as n .

3.1. Information gain

Information gain is the quantity of entropy that may be saved when a specific criterion is used as a splitting feature to optimize a dataset's categorization. Based on the "feature F " the provided data set, designated as D , is further separated into sub-sets S_1, S_2, \dots , and S_n . The weighted sum of the entropies of these subsets is the entropy following the split. The information gain is the difference between the pre-split and post-split entropies. Information gain is calculated as (2).

$$InfoGain(S, F) = Entropy(D) - \sum_j \frac{|S_j|}{|S|} * Entropy(S_j) \quad (2)$$

The feature with the highest information gain is selected as the node for the current split.

3.2. Energy consumption

The total energy consumption E_{total} can be expressed as the sum of the energy consumed by each task i across all steps t .

$$E_{total} = \sum_{i=1}^N \sum_{t=1}^T E_{i,t} \quad (3)$$

where $E_{i,t}$ is the amount of energy used by task i at time step t , and N is the total number of tasks. This computation provides an accurate assessment of the system's total energy efficiency by accounting for variables including task execution time, device power consumption rates, and any additional overhead related to task execution [18].

Within the realm of fog computing, the task scheduling dilemma revolves around assigning IoT tasks to appropriate fog nodes from a pool of potential candidates to enhance QoS. This research specifically considers QoS parameters such as latency, energy consumption, and network utilization. The total number of tasks is denoted by $T = \{t_1, t_2, \dots, t_n\}$ and fog nodes as $F = \{f_1, f_2, \dots, f_m\}$.

The overall transmission duration of task considering both fog and cloud nodes is given by (4).

$$T_{trans,total} = P_{ij} * T_{trans}(X_{ij}) + P_{icloud} * T_{trans}(X_{icloud}) \quad (4)$$

where $T_{trans}(X_{ij})$ is the time taken to transmit a task to a fog node and $T_{trans}(X_{icloud})$ is the time taken to transmit a task to a cloud node. P_{ij} and P_{icloud} are the probability of tasks being transmitted to fog and cloud. The time taken to execute task t_i on fog node f_j and to the cloud is determined as (5), (6).

$$T_{exec}(X_{ij}) = T_{trans}(X_{ij}) + T_{proc}(X_{ij}) \quad (5)$$

$$T_{exec}(X_{icloud}) = T_{trans}(X_{icloud}) + T_{proc}(X_{icloud}) \quad (6)$$

The task's response time t_i of the task is further processed across several layers, including the execution time of the receiver node and the time taken by the task to be transmitted from the sender node to the receiver node. The formula (7) and (8) is employed to calculate the response time of task t_i handled at fog node f_j .

$$T_{response}(X_{ij}) = T_{trans}(X_{ij}) + T_{proc}(X_{ij}) \quad (7)$$

$$T_{response}(X_{icloud}) = T_{trans}(X_{icloud}) + T_{proc}(X_{icloud}) \quad (8)$$

To calculate the energy needed to transmit task t_i to fog node f_j , the transmission time is multiplied by a constant coefficient as (9) and (10):

$$E_{total}(X_{ij}) = \lambda * T_{rtotal}(X_{ij}) * (1 + \mu) \quad (9)$$

$$E_{total}(X_{icloud}) = \lambda * T_{rtotal}(X_{icloud}) * (1 + \mu) \quad (10)$$

where $E_{total}(X_{ij})$ is the total energy consumption for executing task X_{ij} on a fog node. $E_{total}(X_{icloud})$ total energy consumption for executing task X_{icloud} on a cloud node. λ is the transmission energy coefficient, representing the energy required per unit of data transmission. $T_{rtotal}(X_{ij})$ total transmission time for task X_{ij} on a fog node $T_{rtotal}(X_{icloud})$ total transmission time for executing task X_{icloud} on a cloud node. μ is the processing energy coefficient, representing the energy required for processing tasks relative to the transmission energy.

4. SIMULATION AND RESULT

Simulations are carried out in this section to ensure that the suggested algorithm works properly. For implementing the MMFM using DDPG in a fog computing environment, several simulation parameters and setup configurations are considered. The simulation environment was designed to emulate a fog-cloud IoT system [19], incorporating multiple fog nodes and cloud servers. Key parameters included the number of fog nodes, which were varied between 5 to 20, and cloud servers set to 3 for managing offloaded tasks. Task generation was simulated using a Poisson process to mimic real-time task arrivals, with task attributes such as priority, QoS demands, and computational requirements being randomly assigned based on predefined distributions [20], [21].

Network bandwidth and latency were configured to reflect realistic conditions, with average latency ranging between 5 to 20 ms for fog nodes and higher latencies for cloud interactions. The DDPG agent was trained over 10,000 episodes, with each episode consisting of 50 task scheduling iterations. The reward function was designed to balance energy efficiency and task completion time, encouraging optimal task allocation between fog nodes and cloud servers. Hyperparameters for the DDPG model included a learning rate of 0.001, a discount factor of 0.99, and a batch size of 64. The simulation was executed using a combination of Python libraries, such as TensorFlow for the DDPG implementation and SimPy for the discrete-event simulation of the fog computing environment [22]. This setup ensured a comprehensive evaluation of the MMFM's performance in dynamically adjusting task allocations to meet QoS requirements and enhance energy efficiency.

The performance analysis of algorithms and models involves evaluating their accuracy, and computational efficiency. Accuracy measures the correctness of predictions, while computational efficiency assesses resource usage. Generalization examines how well models apply to unseen data [23], [24]. Comparative analysis against baselines and state-of-the-art methods benchmarks performance. Evaluating these aspects ensures models meet the desired criteria for real-world applications, balancing predictive power, resource consumption, and interpretability. This analysis guides iterative refinement, enhancing model reliability, efficiency, and effectiveness in solving target problems [25].

MMFM has established itself as a top performer among predictive algorithms, offering close to the best solution for the problem with an output close to 0.89. The difference in the achieved score rises to a notably more impressive level compared to results obtained by other prominent reinforcement learning algorithms, namely random forest (RF), support vector machine (SVM), and k-nearest neighbors (KNN). The evaluation process was conducted in a detailed manner, by collecting several metrics of performance which included F1 scores, RMSE, and MSE. Through this, it was demonstrated that MMFM not only had very good accuracy and prediction but also showed a high variable of consistency and reliability in case the varied scenarios and datasets were considered.

The horizontal bar graph illustrated in Figure 3 shows the accuracy comparison among four models: RF, SVM, KNN, and MMFM. The x-axis of the graph represents the accuracy level, proposed MMFM performs better compared to other existing models. The area chart as illustrated in Figure 4 signifies the energy consumption of four models which includes the RF, SVM, KNN and the proposed MMFM in the x-axis. In the y-axis representing energy consumption The SVM is the most energetic model consuming a quantity of about 80 kWh, whereas the KNN model is the second most energetic consuming a quantity of about 75 kWh.

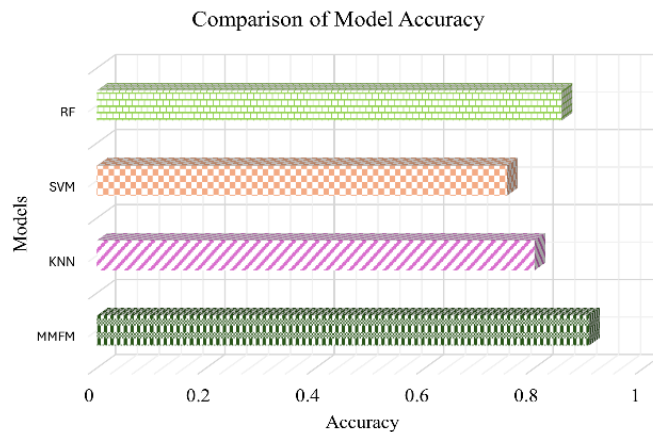


Figure 3. Shows the accuracy comparison

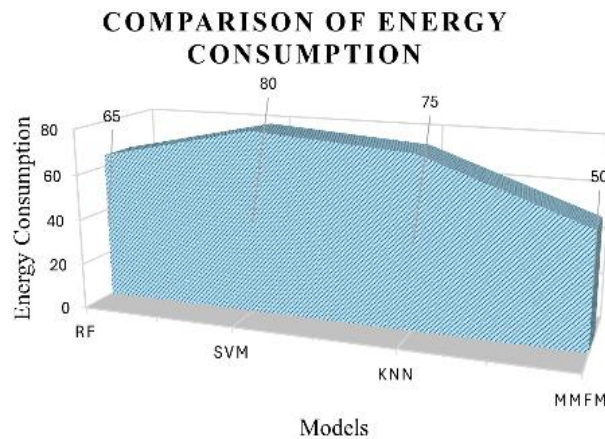


Figure 4. Energy consumption for different models

MMFM, on the other hand, uses less than 50 units of energy resulting in monumental efficiency. To assess the changes in the total energy consumption for various sizes of tasks under various circumstances, as well as to compare the expenditures with the previous methods, Figure 5 is provided. Our proposed algorithm’s energy consumption is always less than that in traditional local computing as well as the full offloading. In Figure 6, the adventure consists of an accumulation of rewards by showing the four climbing passages. The x-axis stands for the episodes having 0-3 values, while the y-axis demonstrates the desired reward from 0 to 15. The line in graph pattern is aimed at demonstrating a positive correlation, which is an increase in total rewards as the episodes get forward. Firstly, the reward may start from about 3 and quickly increase to approximately 15 as the episodes move to four. Therefore, this kind of growth shows the system performance of the model to be upgraded over time which should bring higher rewards over episodes.

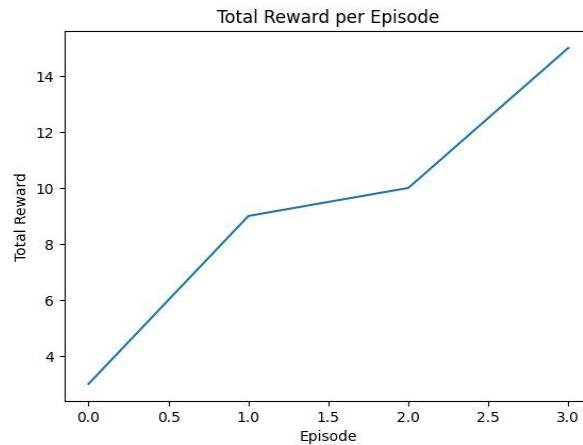


Figure 6. Total rewards per episode

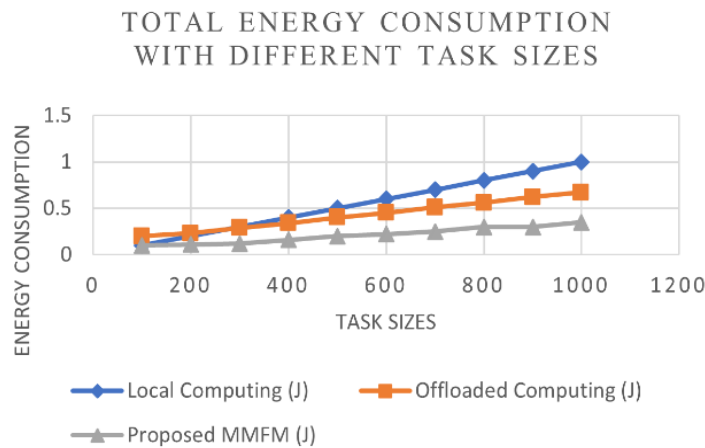


Figure 5. Total energy consumption with different task sizes

5. CONCLUSION





The main goal is to preserve local energy resources by prioritizing critical tasks for migration to the cloud. The implementation of the PTCA enabled efficient task transfer to the Amazon Web Services (AWS) cloud infrastructure, leading to effective resource distribution and optimization. Furthermore, the task scheduler, empowered by the predictive features of the MMFM, adeptly orchestrated task assignments based on forthcoming predictions. This holistic strategy ensured that vital tasks were prioritized, maximizing local energy conservation while sustaining system efficiency. Throughout the study, the effective task migration to the AWS cloud via the PTCM algorithm highlighted the viability and efficiency of fog-cloud-based approaches for resource-demanding applications. By tapping into the scalability and computational capabilities of the cloud, the system exhibited improved efficiency and adaptability in managing varied

workloads. Additionally, incorporating the MMFM model into the task-scheduling framework was pivotal in refining resource management and boosting system efficiency. Through predictive analytics, the task scheduler was able to foresee future workload requirements, facilitating proactive decision-making and optimal resource distribution. This methodology not only enhanced task completion rates but also promoted energy conservation by migrating non-essential tasks during peak demand periods.





REFERENCES

- [1] R. Buyya, "Modelling and simulation of fog and edge computing environments using iFogSim toolkit," in *Fog and edge computing: Principles and paradigms*, no. April, 2018, pp. 1–35.
- [2] A. G. Jakwa, A. Y. Gital, S. Boukari, and F. U. Zambuk, "Performance evaluation of hybrid meta-heuristics-based task scheduling algorithm for energy efficiency in fog computing," *International Journal of Cloud Applications and Computing*, vol. 13, no. 1, 2023, doi: 10.4018/IJCAC.324758.
- [3] V. Sindhu, M. Prakash, and P. Mohan Kumar, "Energy-efficient task scheduling and resource allocation for improving the performance of a cloud–fog environment," *Symmetry*, vol. 14, no. 11, 2022, doi: 10.3390/sym14112340.
- [4] M. Hussain, L. F. Wei, A. Lakhan, S. Wali, S. Ali, and A. Hussain, "Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing," *Sustainable Computing: Informatics and Systems*, 2021.
- [5] P. Singh, M. Dutta, and N. Aggarwal, "A review of task scheduling based on meta-heuristics approach in cloud computing," *Knowledge and Information Systems*, vol. 52, no. 1, pp. 1–51, 2017, doi: 10.1007/s10115-017-1044-2.
- [6] Y. Hou, L. Liu, Q. Wei, X. Xu, and C. Chen, "A novel DDPG method with prioritized experience replay," in *2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, 2017, vol. 2017-January, pp. 316–321, doi: 10.1109/SMC.2017.8122622.
- [7] Y. Dong and X. Zou, "Mobile robot path planning based on improved DDPG reinforcement learning algorithm," in *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, Oct. 2020, pp. 52–56, doi: 10.1109/ICSESS49938.2020.9237641.
- [8] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8577–8588, 2019, doi: 10.1109/JIOT.2019.2921159.
- [9] G. Li *et al.*, "Energy efficient data collection in large-scale internet of things via computation offloading," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4176–4187, Jun. 2019, doi: 10.1109/JIOT.2018.2875244.
- [10] M. Babar and M. Sohail Khan, "ScalEdge: a framework for scalable edge computing in Internet of things–based smart systems," *International Journal of Distributed Sensor Networks*, vol. 17, no. 7, 2021, doi: 10.1177/15501477211035332.
- [11] Y. Yu, J. Tang, J. Huang, X. Zhang, D. K. C. So, and K.-K. Wong, "Multi-objective optimization for UAV-Assisted wireless powered IoT networks based on extended DDPG algorithm," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6361–6374, Sep. 2021, doi: 10.1109/TCOMM.2021.3089476.
- [12] J. Baek and G. Kaddoum, "Heterogeneous task offloading and resource allocations via deep recurrent reinforcement learning in partial observable multifog networks," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 1041–1056, Jan. 2021, doi: 10.1109/JIOT.2020.3009540.
- [13] M. Tang and V. W. S. Wong, "Deep reinforcement learning for task offloading in mobile edge computing systems," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 1985–1997, 2022, doi: 10.1109/TMC.2020.3036871.
- [14] Q. Fan, J. Bai, H. Zhang, Y. Yi, and L. Liu, "Delay-aware resource allocation in fog-assisted IoT networks through reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 7, pp. 5189–5199, Apr. 2022, doi: 10.1109/JIOT.2021.3111079.
- [15] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiqzaman, and D. O. Wu, "Edge computing in industrial internet of things: architecture, advances and challenges," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 4, pp. 2462–2488, 2020, doi: 10.1109/COMST.2020.3009103.
- [16] P. Zang, "Application of ID3 decision tree classification algorithm in mathematical data analysis," in *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*, Nov. 2023, pp. 1–6, doi: 10.1109/ICIICS59993.2023.10421137.
- [17] X. He, Y. Kuang, N. Song, and F. Liu, "Intelligent navigation of indoor robot based on improved DDPG algorithm," *Mathematical Problems in Engineering*, vol. 2023, pp. 1–11, 2023, doi: 10.1155/2023/6544029.
- [18] K. Alatoun, K. Matrouk, M. A. Mohammed, J. Nedoma, R. Martinek, and P. Zmij, "A novel low-latency and energy-efficient task scheduling framework for internet of medical things in an edge fog cloud system," *Sensors*, vol. 22, no. 14, 2022, doi: 10.3390/s22145327.
- [19] B. Krishnamurthy and S. G. Shiva, "Energy efficient double critic deep deterministic policy gradient framework for fog computing," in *2022 IEEE World AI IoT Congress (AIoT)*, Jun. 2022, pp. 521–527, doi: 10.1109/AIoT54504.2022.9817157.
- [20] X. Cai, S. Geng, D. I. Wu, J. Cai, and J. Chen, "A multicloud-model-based many-objective intelligent algorithm for efficient task scheduling in internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9645–9653, 2021, doi: 10.1109/JIOT.2020.3040019.
- [21] B. Jamil, H. Ijaz, M. Shojafar, K. Munir, and R. Buyya, "Resource allocation and task scheduling in fog computing and internet of everything environments: a taxonomy, review, and future directions," *ACM Computing Surveys*, vol. 54, no. 11s, 2022, doi: 10.1145/3513002.
- [22] S. Chen, X. Ge, Q. Wang, Y. Miao, and X. Ruan, "DDPG-based intelligent rechargeable fog computation offloading for IoT," *Wireless Networks*, vol. 28, no. 7, pp. 3293–3304, 2022, doi: 10.1007/s11276-022-03054-1.
- [23] S. Tuli *et al.*, "AI augmented edge and fog computing: trends and challenges," *Journal of Network and Computer Applications*, vol. 216, 2023, doi: 10.1016/j.jnca.2023.103648.
- [24] M. Abd Elaziz, L. Abualgah, and I. Attiya, "Advanced optimization technique for scheduling IoT tasks in cloud-fog computing environments," *Future Generation Computer Systems*, vol. 124, pp. 142–154, 2021, doi: 10.1016/j.future.2021.05.026.
- [25] J. Baek, G. Kaddoum, S. Garg, K. Kaur, and V. Gravel, "Managing fog networks using reinforcement learning based load balancing algorithm," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2019, pp. 1–7, doi: 10.1109/WCNC.2019.8885745.

BIOGRAPHIES OF AUTHORS

Vijayalakshmi Venkatesan     working as assistant professor in the Department of Networking and Communications, School of Computing, College of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur. She has received a bachelor's degree in computer science and engineering from Anna University, master's in information technology from Bharath University, Chennai. Now she is pursuing Ph.D degree in computer science and engineering at SRM Institute of Science and Technology. She can be contacted at email: vijayalv@srmist.edu.in.



Saravanan Murugan     working as a professor in the Department of Networking and Communications, School of Computing, College of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India. He received the M.E. degree in computer science and engineering from the Government College of Technology, Coimbatore, Tamil Nadu, India, and the Ph.D. degree in computer science and engineering from Periyar Maniammai University, Thanjavur, Tamil Nadu, India. His research interests include cloud computing, fog computing, computer vision in edge computing and cloud security. He can be contacted at email: saravanm7@srmist.edu.in.