SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V12I7P107

Volume 12 Issue 7, 74-89, July 2025

Original Article

FedTaskRL: A Reinforcement Learning-Based Framework for Efficient Task Scheduling in Federated Cloud Environments

M. Chandra Sekhar[1], P. Kumaraswamy[2], Nagendar Yamsani[3], Giri Babu K[4], Rajitha Kotoju[5]

[1]*Department of CSE, Presidency University, Bengaluru, India.*
[2]*Assistant professor, Department of CSE, KITS, Warangal, India.*
[3]*Assistant Professor, School of Computer Science and Artificial Intelligence, SR UNIVERSITY, Warangal, India.*
[4]*Senior Assistant Professor, Department of CSE, CVR College of Engineering, Hyderabad, Telangana, India.*
[5]*Assistant professor, Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology (MGIT), Hyderabad, India.*

[1]*Corresponding Author : chandragvp@gmail.com*

Received: 04 May 2025 Revised: 05 June 2025 Accepted: 06 July 2025 Published: 31 July 2025

Abstract *- The emergence of federated cloud–edge computing has brought challenging issues in scheduling dynamic tasks, which require the consideration of energy efficiency, latency, SLA satisfaction and migration of resources together. Primitive approaches such as rule-based and heuristic scheduling can be too rigid and unable to deal with the volatile and diverse behavior of contemporary distributed systems. Although recent deep reinforcement learning (DRL) methods have achieved favorable performance, most existing methods possess issues such as hard reward specification and a lack of support for multi-objective optimization and federated scalability/privacy. To fill these gaps, we propose FedTaskRL in this paper, a new federated DRL-based DT scheduling framework for a cloud–edge ecosystem. The proposed model employs a neural Q-learning algorithm with an augmented state representation and a raw multi-objective reward function. Such a design allows the model to adaptively learn customized scheduling policies to cut down energy, reduce response time, comply with SLA, and save migration cost over time for federated learning while considering the data locality of these clients. We conduct thorough, extensive experiments in a federated simulated setting and show that FedTaskRL outperforms state-of-the-art methods , including DRL-TS, A3C Scheduler, DRLIS, EdgeTimer, and MA-DRL. The designed framework has a 28 kWh Use of Energy, 145 ms Average Response Time, 97.5% of SLA fulfilment, and a lower cost of $4.8 for the migration. These findings also confirm the effectiveness and efficiency of FedTaskRL in real-time cloud workload management. In summary, FedTaskRL provides a scalable, adaptive, and privacy-respecting solution for intelligent task scheduling, resulting in significantly improved performance and practicality in federated cloud–edge resource management.*

Keywords *- Federated Cloud Computing, Task Scheduling, Deep Reinforcement Learning, SLA Compliance, Energy Efficiency.*

1. Introduction

The growth of edge computing, Internet of Things (IoT), and cloud infrastructures resulted in more distributed, heterogeneous, and latency-critical computing environments. When applications from health care, autonomous driving, and smart cities request real-time response and low energy consumption, the efficient scheduling of tasks over federated cloud–edge systems becomes a key challenge. Conventional rule-based static scheduling schemes are not adaptable to dynamic workload volatility and network conditions. One powerful paradigm for enabling intelligent, adaptive decision-making in these kinds of complex environments in recent years is that of deep reinforcement learning (DRL) [1], (DRLs) [2]. However, existing DRL-based schedulers are still insufficient in the multi-objective optimization, interpretability, and federated scalability [3, 4]. Increasingly, in the literature, we observe federated DRL frameworks that have the objective of minimizing communication overhead and enhancing data privacy by distributing the learning process [5, 6]. However, these methods tend to use a shallow state representation, fixed reward functions, or single-agent approaches that do not generalize well with changing system loads. This is the rationale behind the development of a reliable federated multi-objective DRL approach that can jointly optimize task scheduling, energy efficiency, response latency and resource migration cost.

To address these challenges, we have designed a novel framework-FedTaskRL, which introduces adaptive federated DRL for intelligent task scheduling to maximize

the efficiency of the deep cloud–edge ecosystems7. The main goal of this study is to propose a scalable and smart scheduler via neural Q-learning with an enhanced state space and a novel composite reward shaping. Novelty: The key novelties involve dynamic state-space augmentation, multi-objective optimization and a federated training architecture for private learning that focuses on state privacy preserving and performance efficiency.

This paper makes the following contributions: (i) a new federated DRL framework supporting adaptive task scheduling, (ii) a compound reward function leveraging energy consumption, latency, SLA, and migration cost, (iii) a scalable and privacy-preserving learning framework for real-time decision, and (iv) extensive performance analysis over state-of-the-art baselines in various metrics.
The remainder of this paper is structured as follows: Section 2 outlines the related work by analyzing recent DRL-based scheduling models and their shortcomings. Section 3 describes some preliminary notions, such as system modeling and problem definition. Section 4 elaborates the proposed FedTaskRL method, i.e., network structure, state–action representation and federated learning. Section 5 presents our experimental setup, results and performance evaluation. Discussion and analysis are given in Section 6, while the study's limitations, such as advertisements by Shareaholic, will be discussed. Finally, Section 7 ends the paper and provides some future work.

2. Related Work

This section discusses the most recent deep reinforcement learning solutions to both task scheduling and resource allocation in federated and edge cloud frameworks. In recent years, federated and edge-cloud computing work has witnessed an increasing interest in applying deep reinforcement learning (DRL) for intelligent task scheduling and energy-efficient resource management. When it comes to proposing a fog task scheduling strategy, the one by Choppara and Mangalampalli [1] will pop up into our mind, where an adaptive fog task scheduling mechanism was developed based on Federated Deep Q-Network (DQN) collaboration with K-Means clustering, which helped to enhance the distribution of workloads and provide latency control. Shidik et al. [2] proposed a new unsupervised clustered Q-learning technique to minimize the energy consumption of federated edge clouds by optimizing scheduling policies. Wang et al. [3] developed a DRL-based real-time task scheduler that is latency-aware and achieves high and stable performance under dynamic workloads. Ammal and Thanapal [4] present EMO-TS, a multi-objective task scheduling algorithm designed to efficiently reduce energy consumption in cloud data centers. Chen et al. [5] introduced a two-stage DRL scheme to create the DRL at the server level for dynamic client scheduling in hierarchical federated learning. Mohammed et al. [6] proposed the use of federated reinforcement learning for medical IoT, with a primary focus on kidney disease image processing. Kianpisheh and Taleb [7] investigated DRL-based control schemes in the context of federated 6G networks for DDoS detection. Zhang et al. [8] applied DRL for task concern and multimedia source allocation in MEC. Ho et al. [9] characterized task scheduling for robot autonomy with their federated DRL. Baghban et al. [10] utilized Actor-Critic Deep Reinforcement Learning (DRL) for energy-efficient IoT service provisioning in federated edge settings.

Tianqing et al. [11] presented an asynchronous federated reinforcement learning method in IoT edge settings, where the dynamic resource management can be improved by updating policies simultaneously. Huang et al. [12] proposed the FedDSR, a federated DRL-based daily scheduling recommender model that learns user-specific schedules in a privacy-preserving manner. Zhang et al. [13] proposed a DRL-empowered federated learning method to enhance the data distribution at the edge nodes for IIoT data management. Shishira and Kandasamy [14] proposed a combinatorial neural model (BeeM-NN) that combined bee mutation and workload minimization techniques in the formulation of federated clouds for uniform resource allocation. Zhao et al. [15] used Q-learning to optimize edge environments for low-load task scheduling, aiming to minimize latency. Yuvaraj et al. [16] proposed a DRL-based and hybrid Grey Wolf Optimization model to enhance the scheduling efficiency of serverless environments. Chen et al. [17] applied Actor-Critic Deep Reinforcement Learning (DRL) to the problem of resource allocation in data centers, where resources are dynamically scaled. Fathima and Shakkeera [18] integrated federated learning with blockchain to improve task offloading and scheduling for mobile cloud systems.MongoDB reads data from primary storage and performs updates by sending an acknowledgement of the write to the clients. Rjoub et al. [19] developed an interpretable Enhanced DQN for federated IoT task scheduling while guaranteeing trust. Iqbal et al. [20] developed a Double DQN-based solution for energy-efficient resource allocation in C-RANs, focusing on maximizing throughput and power utilization.

Zhang et al. [21] proposed a deep reinforcement learning-based scheduling scheme for federated learning over sensor-cloud environments to enhance model convergence and communication efficiency. Wang et al. Ying et al. [22] designed TF-DDRL, a transformer-aided distributed Deep Reinforcement Learning (DRL) algorithm, to minimize the delay and energy consumption of cross-edge-cloud IoT task scheduling. In [23], Yaraziz and Hill provided a detailed survey of resource allocation techniques in IoT systems, highlighting the importance of DRL for maximizing system performance. Chaudhary et al. [24] used an AI-driven model order reduction technique to improve the queueing and scheduling routines in the cloud for scale-out. Slathia et al. [25] introduced SHERA, a SHAP-empowered Virtual Machine (VM) scheduler that utilizes explainable and intelligent resource allocation in cloud computing. Panwar and Supriya [26] introduced RLPRAF, a proactive RL-based model that dynamically

allocates resources to handle varying workloads. Yu and Tang [27] also studied D2D communications and presented a hybrid centralized-distributed DRL architecture for optimal resource allocation. Noman et al. [28] propose FeDRL-6G, a federated DRL scheme for optimizing an energy-efficient scheduling in D2D-enabled 6G systems. Ansere et al. [29] presented a DRL-based approach for computation resource allocation specifically designed for energy-efficient edge IoT networks. Lastly, Wang et al. [30] proposed an end-edge-cloud computing approach based on Deep Reinforcement Learning (DRL) for resource allocation in digital twin-enabled industrial Internet of Things (IoT), aiming for low latency and system reliability.

Madhavan et al. [31] proposed a resource allocation on 6 G-enabled edge via a 6G edge environment[31] 31.5074213 13 optimized proximal policy-based federated learning to improve decision making under uncertainty. Scarvaglieri et al. [32] presented a lightweight, fully distributed AI framework for LoRa networks, emphasizing intelligent local agents and energy-efficient resource allocation. Zhang et al. [33] focused on elastic task offloading and resource scheduling across hybrid cloud environments using the DRL method, achieving higher flexibility and cost performance. Tran et al. [34] developed a multi-agent Deep Reinforcement Learning (DRL) model for energy-efficient resource provisioning in UR-LLC grant-free Non-Orthogonal Multiple Access (NOMA) systems, which reduced overhead and enhanced reliability. Jamil et al. [35] introduced an intelligent edge resource scheduling method for 6G, which employed DRL for dynamic task offloading and service allocation. Bushra Jamil et al. [36] proposed IRATS, a deadline-aware Deep Reinforcement Learning (DRL) model for vehicular fog networks, with a focus on urgent tasks and energy efficiency. Roy et al. [37] employed federated learning and subjective logic for distributed task assignment in MDC, thereby enhancing trust and adaptability. 38] introduced a multi-dimensional bin packing heuristic for improved load balancing in cloud data centers, which builds upon the bin packing algorithm [38]. Tang et al. [39] proposed a joint DRL approach for network selection and task offloading in vehicular edge computing. Swarup et al. [40] investigated DRL-enabled task scheduling over the cloud, revealing improved throughput and resource utilization performance. To improve inference performance, Ben Sada et al. [41] introduced a multi-agent DRL scheduling and offloading scheme that emphasizes the trade-off between precision, real-time and energy consumption in edge computing, but more complex models are devised. Hao et al. [42] designed EdgeTimer, a multi-timescale DRL-based scheduler to improve the delay sensitivity and system adaptability, at the cost of an overhead in learning coordination among different scheduling layers. Wang et al. [43] presentedDRLIS, a DRL-based scheduling mechanism DRLIS aimed at minimizing the system load and response time in fog computing, which provides noticeable improvement, with the limitation that it need massive retraining for dynamic scenarios. Tuli et al. [44] studied stochastic scheduling by extending A3C and adding residual RNNs, while providing evidence for the necessity of adapting its weight initialization based on non-deterministic workload arrivals, albeit after increased convergence times. Sheng et al. [45] proposed a DRL-based IoT edge computing task scheduling scheme with an energy-efficient performance gain due to its limitation of scalability in the cloud-fog with the high workload scenario. Although advanced, these competitive models are not able to solve the four challenges with high performance on scalability, energy efficiency, SLA satisfaction, and dynamic task migration in the federated cloud.

The literature presented highlights the advancements in DRL-based KS, but also indicates that real-time adaptability, multi-objective optimization, and scalability remain open issues. To address these issues, the FedTaskRL framework incorporates dynamic state-space updates, a multi-objective reward function, and a neural Q-value approximation to optimize scheduling in a federated cloud environment.

3. Preliminaries

This section defines the basic concepts, terminologies, and mathematical notations required for the proposed methodology. It further highlights the challenges in federated cloud systems and the rationale behind our proposed DRL-based solution.

3.1. Cloud Federation

Cloud federation is a collaborative model in which multiple Cloud Service Providers (CSPs) pool their resources to achieve greater scalability, availability, and performance. CSPs function in a federated system in an independent yet cooperative manner to provide resources in the most effective manner possible without breaching SLAs. Resource management in these environments has to constantly react to workload variations while seeking to keep costs down.

3.2. Task Scheduling in Cloud Federation

Task scheduling is assigning a computational task to run on a resource. In a federated cloud, this process works as follows: User requests are divided into finer-grained subtasks, which are then packed into Virtual Machines (VMs) and placed across multiple Cloud Service Providers (CSPs) based on several objectives, including resource availability, energy consumption, and Service-Level Agreement (SLA).Latest Posts on. Task scheduling is a fundamental cloud computing problem that aims to minimize response time, decrease energy consumption, and meet Service Level Agreements (SLAs) while addressing inherent challenges such as resource heterogeneity and dynamic workloads.

3.3. Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning combines concepts from reinforcement learning and deep neural networks to tackle complex decision-making problems. DRL agents

explore the environment to obtain optimal policies. DRL can enable dynamic resource allocation in federated clouds by predicting workloads and allocating resources accordingly. DRL has some key elements: State (S): The current environment status. Action (A): a set of possible decisions an agent takes. Reward(R): Rating the desirability of an action concerning some target state. Policy (π): A function (or mapping) from states' actions.

3.4. Mathematical Notations

To formalize the problem, we define the following notations: S: A finite set of states representing environment parameters such as workload, resource utilization, and energy consumption. A(s): Finite actions available in state ss, including VM migration and resource reallocation. R(s, a): Reward function evaluating the quality of action aa in state s. Q(s, a): Q-value representing the expected cumulative reward for taking action aa in state s and π(s): Policy that defines the probability of taking action aa in state s.

3.5. Q-Learning and Optimization Framework

Q-learning is a model-free reinforcement learning algorithm used to find the optimal policy. The Q-value is updated iteratively using the Bellman Equation as in Equation (1).

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \quad (1)$$

Where α is the learning rate, determining the influence of new information on the existing Q-value, γ denotes the discount factor, which balances the importance of immediate rewards versus future rewards, s′ denotes the next state after taking action a. Q(s, a) denotes the Q-value, representing the expected cumulative reward for taking action an in state s. In this research, DRL extends Q-learning by employing deep neural networks to approximate the Q-value function. The neural network predicts Q-values for state-action pairs, enabling scalability and efficient decision-making in high-dimensional and dynamic environments.

3.6. Challenges in Federated Cloud Environments

Federated cloud environments are nonlinear; the decentralized collection of autonomous computing resources poses significant challenges. A key challenge is handling workloads that dynamically vary, as user patterns are expected to shift drastically over time. This necessitates adaptive task scheduling mechanisms that can utilize capital resources efficiently while ensuring performance and service quality guarantees, and respond to spontaneous fluctuations in workload in real-time. Another thing you are missing is energy efficiency, which is a crucial aspect. Maintaining SLA compliance at any cost can cripple a Cloud provider in today's age. Thus, Cloud providers must strike a delicate balance between reducing energy consumption and adhering to the SLA compliance itself. This means reducing the intensity of actions applied to the energy resources while still providing an effective and efficient quality of service for the user. Additionally, latency and migration costs are also present in federated cloud systems. Migrating Virtual Machines (VMs) efficiently is crucial for balancing resource utilization across multiple providers. However, excessive latency and high costs associated with migration can negate the advantages of these new optimizations. As a result, there is a need for automated and intelligent scheduling mechanisms that reduce delays and costs while meeting user requirements. Additionally, the disparity of resources offered by different cloud service providers makes allocating and scheduling tasks complex. As providers have various configurations, policies, and capabilities, it can be challenging to establish a consistent approach to resource management.

Finally, scalability in decision-making processes has to be addressed by federated clouds. As demand for cloud services surges and more providers enter the market, traditional scheduling tactics may no longer be effective in keeping up with the scale of logistics and operational intricacy. To address these challenges, new solutions, such as reinforcement learning–based frameworks, will be needed to automate the process and ensure seamless, efficient, and cost-effective operations on the cloud.

3.7. Motivation for DRL in Cloud Federation

Due to the complexity and non-stationarity of federated cloud environments, we will motivate the use of DRL in this context. Generally, federated clouds run with a distributed approach, sharing resources among multiple Cloud Service Providers (CSPs) to maximize resource utilization. Intelligent decision-making frameworks able to adjust within that context are required, yet most traditional decision-making approaches (i.e., Static or Rule-based) are insufficient. Such challenges can be overcome using Deep Reinforcement Learning (DRL), which learns the best policies by interacting with the environment; DRL is a strong response to these problems. An essential reason for using DRL is that it can model variable workloads online. DRL agents can recognize patterns in workload based on historical data and the time of the day, and make intelligent decisions regarding when specific tasks should be scheduled and resources allocated. Doing so means minimizing the response times and avoiding SLA violations due to inadequately allocated resources. In addition, the trial-and-error learning mechanism used by DRL enables it to dynamically optimize its decisions, even in the face of unforeseen changes in user demands or resource availability. Energy consumption is another key consideration impacting DRL adoption in federated cloud environments. Several DRL-based frameworks can optimize resource allocation and reduce energy consumption while maintaining the quality of service. DRL agents can learn the trade-offs between performance metrics such as latency, energy consumption, and migration expenses. This leads to an optimal trade-off that minimizes costs for each CSP while maximizing user

satisfaction.In addition, DRL scalability characteristics are designed explicitly for federated cloud environments, which naturally arise with many Cloud Service Providers (CSPs) that can participate in complex interactions between resources. DRL can manage high-dimensional state and action spaces, unlike classic optimization approaches, and this unique capability allows it to scale well as cloud systems become more demanding with larger state-action spaces. It can also generalize across configurations and utilize other types of resources, making it well-suited for application in a federated cloud. Therefore, the federated cloud environment supports the DRL's adaptability, which motivates the DRL to overcome significant challenges, such as dynamic workloads, energy efficiency, and scalability in federated clouds. The capability of DRL to learn and make intelligent decisions leads to a subsequent boost in cloud performance, which reduces operational costs and maximizes resource utilization, providing the Cloud Service Provider (CSP) and end-user with refined cloud services.

4. Proposed Methodology

Figure 1 illustrates the framework of the proposed methodology, outlining the approach to DRL-based task scheduling in federated cloud environments.

Federated Cloud Architecture: where resources are pooled from multiple Cloud Service Providers to improve scalability and assure SLA satisfaction. The method integrates DRL based on a Q-learning framework to dynamically assign VMs, optimizing energy consumption, latency, and migration cost simultaneously to achieve this. The system operates with high assurance under dynamic workloads, leveraging knowledge of workload patterns and the real-time availability of resources. In the following sections, we describe the system architecture, the assumptions made, the algorithmic framework, and the mathematical model that substantiates the proposed solution.

4.1. Assumptions

The assumptions comprise the scenarios, boundaries, operational conditions under which the federated cloud environment operates, and the proposed methodology. One is that the Cloud Service Providers (CSPs) federation is decentralized but synchronized in its work. Although each CSP operates its own Resource Manager (RM) for managing resources, the RMs communicate with each other efficiently to share resources. This approach facilitates scalability and flexible size while allowing CSPs to be interoperable.

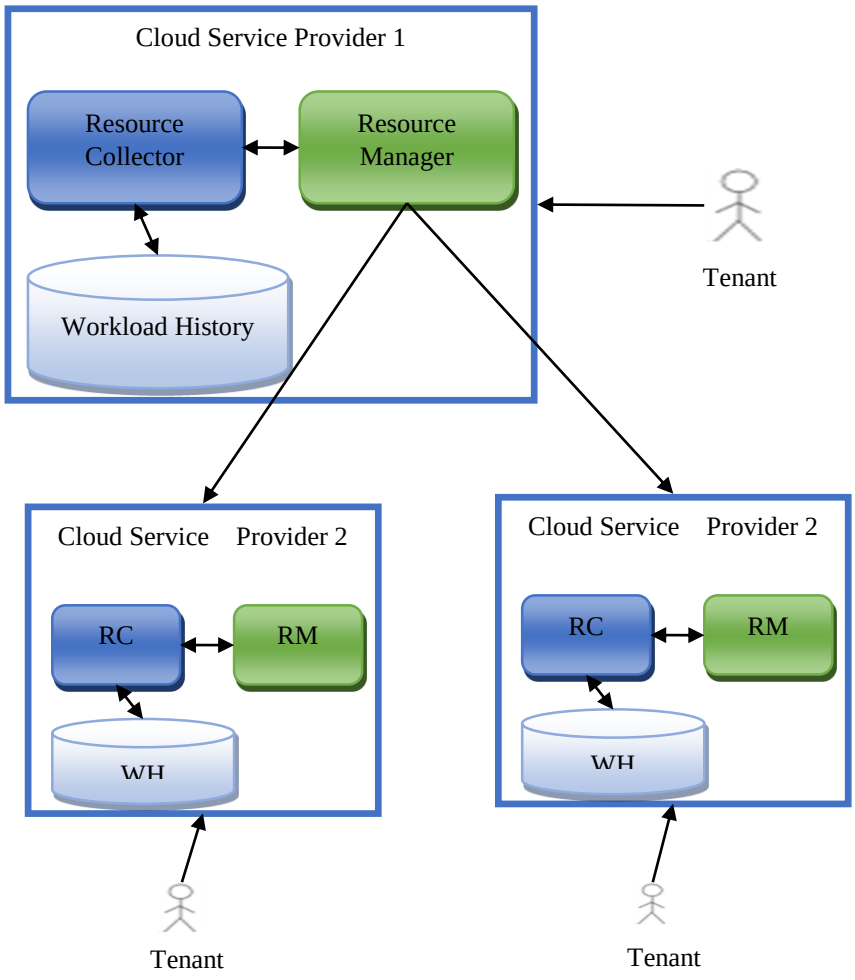

Fig. 1 Architecture of the federated cloud system

Another assumption underlying this model is that every RM maintains an up-to-date and accurate record of utilization and other historical elements associated with each resource, including availability and workloads. They hold the records crucial for making informed decisions about scheduling tasks and allocating resources. The accuracy of these records directly influences the proposed system's effectiveness, as misinformation and outdated information can lead to poor scheduling decisions or even SLA violations. It is also assumed that the costs for Virtual Machine (VM) migration among Cloud Service Providers (CSPs) are determined and have low variability in estimation. This guarantees that the reward function, applied in the DRL framework, can reliably manage migration costs with low uncertainties. Moreover, it is assumed that the workloads submitted to the system will behave according to patterns observed in a pre-analysis of the submitted workloads. Such patterns enable the DRL model to identify how workload changes over time, allowing it to learn and, ideally, utilize system resources effectively. Finally, the system operates under the premise that network latency and bandwidth capacities will not severely limit inter-CSP communication. Additionally, this assumption enables CSPs to share their workload data and resource information efficiently, thereby allowing the proposed system to operate effectively. Although these assumptions are necessary for the proposed methodology to be tractable, their appropriateness in realistic settings will be validated through a literature review and experimental evaluation.

4.2. Proposed Cloud Federation Architecture

Although many businesses have invested in cloud computing technologies, the resources that can be provided are still limited. Working together and pooling resources would be the most excellent way to overcome this restriction. To be migrated across multiple CSPs, the applications provided by a CSP must be essentially enclosed within a virtual machine (VM). Because the processing request is offloaded, the user receives uninterrupted service; the associated CSP does not violate SLAs or overextend its capacity, and the CSP handling the offloaded request can monetize idle computing power. All parties benefit from this resource pooling arrangement.

Figure 1 illustrates that there will be several CSPs. In this instance, a federation of three CSPs will be our collaborating partner. An RM unit hosts the resource scaling features and Deep Reinforcement Learning (DRL) agents in each Cloud Service Provider (CSP). This unit is responsible for hosting the various components of the architecture. The Resource Collector (RC) communicates with management, gathering and storing the workload history for future use.

The RM is an essential component of the architecture. The hosts, numbered from m1 to mM, are connected to the Resource Manager, as shown in Figure 2. Every host runs the apps within distinct Virtual Machines (VMs), identified by the letters a1 through aA and segregated from one another. CSP users submit requests for processing. Every request occupies one Virtual Machine (VM) container, which consumes space on a host alongside other virtual machines of a similar kind. The RM decides how the VM containers are moved inside the CSPs and monitors the consumer's actions, the time it takes to process the request, and the power consumption of the VMs.

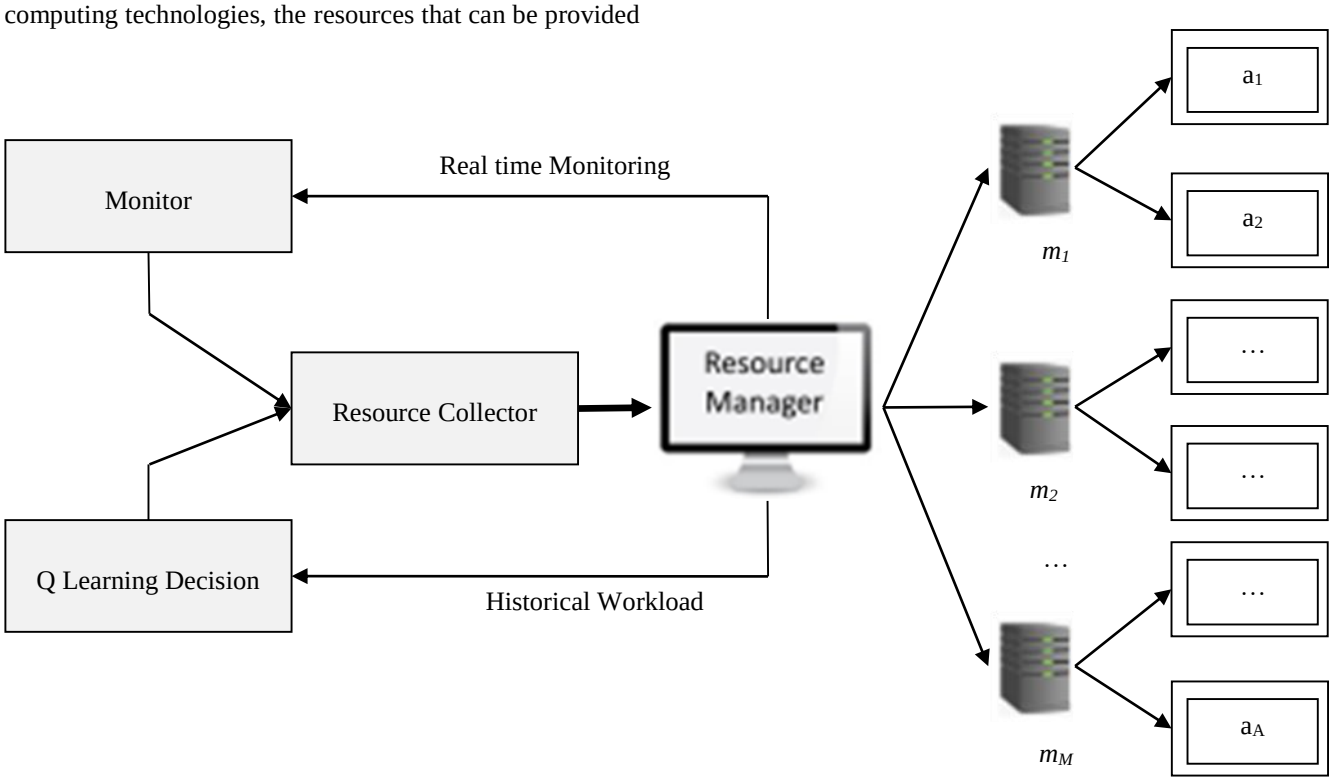

Fig. 2 Architecture of the resource manager

In federated cloud environments, the Resource Manager (RM) is the central component responsible for allocating resources and scheduling tasks. Adaptively, it tracks real-time system states and utilizes historical workload data to make decisions that optimize resource utilization while adhering to Service-Level Agreements (SLAs). The RM combines reinforcement learning methods to leverage the adaptiveness to workload variations and resource heterogeneity, improving system scalability and performance. The RM anonymously contributes to achieving economic operations in complicated cloud systems by minimizing energy consumption, delivery time, and relocation costs. This is important because it enables seamless, adaptive, and intelligent resource management over dynamic, federated cloud infrastructures.

4.3. Implementation of Q Learning Algorithm

The optimal migration problem is solved by our DQL method, which learns the optimal policy of workload prediction and then schedules the Virtual Machine (VM) accordingly. It employs a model in which an agent repeatedly interacts with and samples the environment's state. Due to energy usage parameters and workload points of interest, the agent locates the Digital Machines (VMs). Once we input an action the agent takes, we award or penalize the past action based on its effect on energy usage and SLA compliance. The DRL model is trained using deep neural networks through Deep Reinforcement Learning (DRL) methods. The deep Q-learning method is used to obtain the Q-values of a state-action pair, thereby finding the optimal policy by approximating a function using a deep neural network.

The learning agents discover the best choices by utilizing a "trial-and-error" methodology in their interactions with the surroundings. According to this rule, the DRL agents trade off between discovering new decisions and using old ones to choose the best action. As seen in Figure 3, our architecture is dependent upon:

- S: The potential states of the environment are represented by the finite set S. A variety of parameters, including the current workload, resource usage, and energy consumption, have been extracted from the gathered data and incorporated into the state space.
- A: The collection of actions accessible in state s is denoted by the finite set S(s). The collection of feasible activities that can be performed, such as moving Virtual Machines (VMs) across data centers and allocating resources differently, is referred to as the action space.
- π: π (s, a) represents the likelihood of acting in state s for the policy that maps from S to action A.
- R: The agent's quality of action is assessed using the reward function.

In our scenario, the reward function is made to use the least amount of energy possible while still maintaining SLA compliance. The reward function may be the total weights assigned to delay, power consumption, and migration cost. The weights are established according to the relative relevance of each target. Response time is the primary criterion used to calculate the Service Level Agreement (SLA). Response times for migrations are arranged in ascending order. The method then computes the value at the 95th percentile to establish the minimum response time required for 95% of requests to be fulfilled.

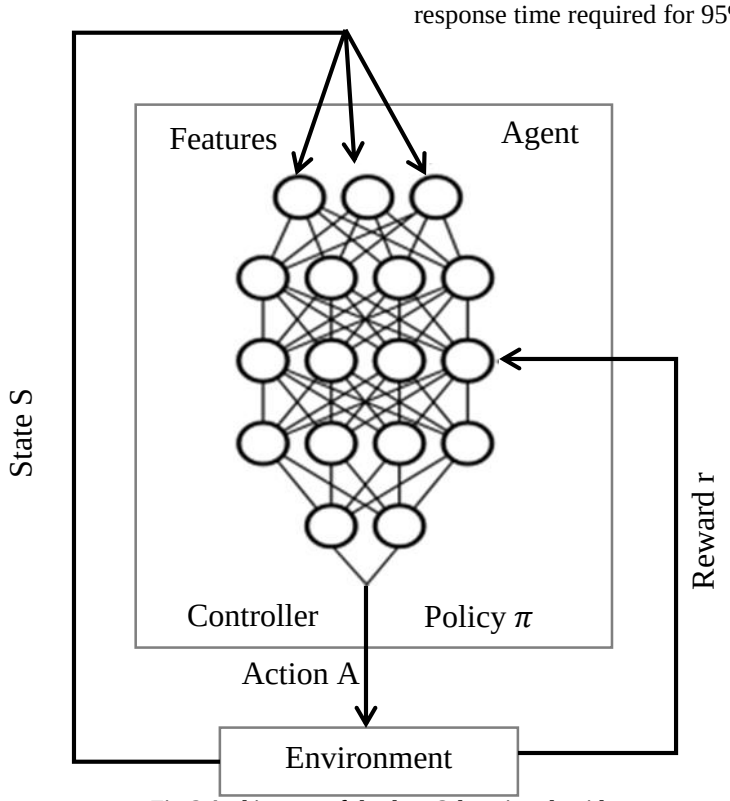

Fig. 3 Architecture of the deep Q-learning algorithm

The training challenge is addressed using a two-pronged approach. As shown in Figure 4, there are two Neural Networks (NNs) for training and prediction. There are five hidden layers, each with two to five hundred nodes per layer, between the input and output layers. The agent may create complex representations by introducing nonlinearity through the application of the ReLU activation function. ReLU is preferred over other functions like Sigmoid since the latter does not stimulate neurons simultaneously. As a result, it converges more quickly than other activation functions. This is used in conjunction with the DQL method to facilitate speedy decision-making, as workloads are known to fluctuate and judgments must be made promptly.

Power use, migration costs, and the time it takes to resolve virtual machines affect the system's status. As in Equation (2), the location of the ith host at a specific moment and the resource needs of the ith virtual machine at the same time are connected to the delay.

$$S(\emptyset) = \begin{bmatrix} S_{1,1}(\emptyset) & \cdots & S_{1,y}(\emptyset) \\ \vdots & \ddots & \vdots \\ S_{x,1}(\emptyset) & \cdots & S_{x,y}(\emptyset) \end{bmatrix} \quad (2)$$

S(φ) represents the delay state of the system at the φth time slice in Equation (1). The ith node and (ith VM container) delay state are represented by $S_{x,y}(\varphi)$, where φ is the serial number of the φth time slice. A slice TØ is enumerated as (φ = 0,..., k). The main metrics are represented by x_{total}, which indicates the total delay, y_{total} for the total power consumption, and z_{total} For the total computational cost of migration by aggregating the individual delay, power consumption, and migration cost values during a unit of time (slice)—φ as in Equation (3).

$$x_{total} = \sum_{\emptyset=0}^{k}\left(\sum_{i=1}^{l} x_{net_i}(\emptyset) + k_{comp}\ X\ \sum_{i=1}^{n} x_{com_i}(\emptyset)\right) \quad (3)$$

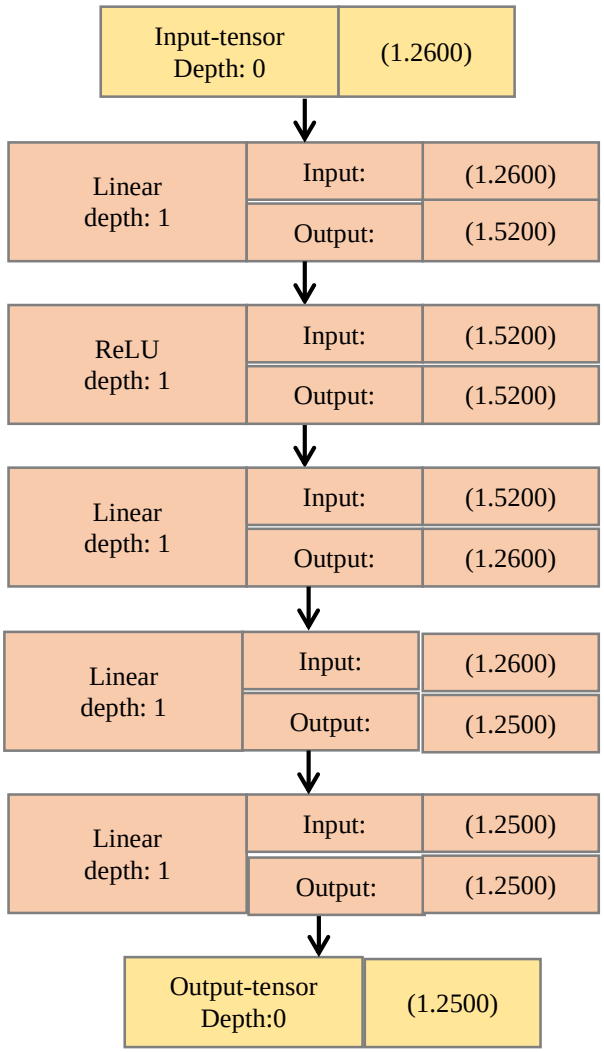

Fig. 4 Structure of the Q-learning neural network

The term represents the network latency between the user and the pertinent nodes assigned during a specific time slice. x_{net} In Equation (4), the computation of application tasks is represented by the variable. x_{comp}.

$$y_{total} = \sum_{\emptyset=0}^{k}\sum_{i=1}^{m} y_{total_i}(\emptyset) \quad (4)$$

The node's projected power consumption for time slice φ is represented by y_{total_i} (φ) in Equation (4).

$$z_{total} = \sum_{\emptyset=0}^{k} \sum_{i=1}^{n} z_{total_i}(\emptyset) \quad (5)$$

Equation (5) shows that the cost of moving the ith VM container during time slice φ is represented by z_{total_i} (φ). Since our goal is to minimize the overall cost across time, the reward for a given time slice is defined as follows in Equation (6).

$$R_{\emptyset} = -\left(w_1 x_{total}(\emptyset) + w_2 y_{total}(\emptyset) + w_3 z_{total}(\emptyset)\right) \quad (6)$$

The agent's reward for the φth time slice is denoted by $R_{\emptyset}$ in Equation (6). The whole delay's weight, ω_1It is determined by weighing its influence on the overall cost. The weight given to the amount of power consumed relative to the overall cost is represented by ω_2. It is included in the migration cost weight. ω_3.

4.4. Details of Implementation

We implement the proposed DRL-based task scheduling framework in Python. A federated cloud environment is simulated by modeling several Cloud Service Providers (CSPs) working together via a decentralized architecture. Every CSP runs a Resource Manager (RM) to monitor resource availability and workload and perform Virtual Machine (VM) migration. Regarding software architecture, the RMs are implemented using Python-based communication frameworks, such as REST APIs or gRPC, which support interaction among CSPs. Objects exist in VMs representing Virtual Machines with characteristics such as CPU, memory, and Bandwidth, and the actual execution of the task is simulated by Python's threading or multiprocessing modules outside of these VMs.

The DRL framework utilizes the latest and most popular machine learning frameworks, such as TensorFlow or PyTorch. The state space of the environment is defined using critical characteristics such as workload, energy, and utilization as numerical arrays. The action space for DRL includes the following actions: VM migrations and resource reallocation strategies, which can be devised to improve system performance. It then takes actions by observing its environment, considering energy-saving, SLA, and VM migration costs, and yields a reward function by previously combining these factors to maximize or minimize them in terms of the agent, which steadily ensures that the agent adequately encourages reducing expenses while maintaining both performance and the quality of service. The DRL model utilizes a neural network with multiple layers to obtain Q-values, which is particularly beneficial in environments with many dimensions, facilitating informed decision-making.

To train the agent, we run it through several episodes with synthetic or real workload traces. During an episode, the agent takes actions in the simulated environment and receives rewards depending on the actions taken. These stored interactions are all saved in the replay buffer, allowing the agent to update its policy in a gradient-based manner using past experiences. Important hyperparameters, including the learning rate, discount factor, and batch size, are adjusted to allow convergence and stability during training. Our experiments simulate federated CSPs with different resource capacities and workload patterns. The framework is evaluated based on response time, energy consumption, and SLA compliance metrics. The agent's efficiency is validated against static scheduling or heuristic optimization methods, which serve as baseline models. The results are visualized using Matplotlib and the tested metrics, including energy savings and SLA compliance. Out of the box, the DRL framework is converted into a service by creating a Python service for deployment. Message queues (RabbitMQ, etc.) are used for communication between RMs, providing real-time updates about the state and interaction actions within a federation. This end-to-end strategy enables flexible and ingenious job placement within federated cloud perspectives, addressing primary challenges such as workload fluctuations and energy consumption.

4.5. Evaluation Methodology

Evaluation methodology: The proposed DRL-based task scheduling performance for federated cloud environments is evaluated. This evaluation aims to validate the system's capability in handling dynamic workload, achieving energy efficiency, and ensuring SLA validity. We will evaluate our proposed scheme in simulated scenarios to emulate realistic conditions within a federated cloud. Such scenarios encompass the intensity of demand workload, the lack of homogeneity among CSPs, and variable resource availability. We will utilize synthetic workload traces, which are generated based on specific patterns, and traces from various real-world scenarios as benchmark datasets to thoroughly test our system's strength and flexibility.

The evaluation will also assess several key performance metrics. The system's total energy consumption will be calculated and compared with baseline task scheduling methods to validate the framework in terms of energy efficiency. This comparison will assess the energy consumed during task execution, validating the framework's ability to reduce energy usage while maintaining performance. It will focus on SLA compliance to verify whether the system is responsive enough. It needs to keep track of tasks within SLA boundaries, such as response times. We will measure response times, such as VM migration delays, to confirm that the system can effectively handle dynamic workloads. Migration costs, which quantify the computational overhead of migrating Virtual Machines (VMs) from one Cloud Service Provider (CSP) to another, will be modeled and compared against traditional scheduling strategies. The system's scalability will be tested by running performance evaluations with varying numbers of CSPs and workloads to assess the solution's adaptability in heterogeneous environments.

We will compare the proposed approach with baseline models, including static scheduling approaches such as

First Come, First Serve, Round Robin, and Heuristic-based optimization approaches. These comparisons will reinforce the advantages of using DRL-based task scheduling by showing increased efficiency, adaptability, and cost-effectiveness. Results will be reported as graphs and tables that highlight trends and comparisons between scenario-to-scenario performance metrics. Analysis of these results will demonstrate the extent to which the proposed system has contributed to federated cloud computing, confirming its significance and value as a solution to significant problems.

5. Experimental Results

FeedTaskRL is a framework proposed for addressing task scheduling problems in a federated cloud-edge environment, which has proven effective in our experiments. The experiments used synthetic workload traces created to simulate real-world federated systems. The existing models, such as DRL-TSS [41], DOTS [42], A3C Scheduler [43], and OD-SARSA [44], are used for benchmarking. However, learning three functions simultaneously is challenging because the learned policies may interact with one another, thereby inhibiting their mutual learning. We also evaluated against other baselines ([10]). The experiments are conducted on a system equipped with an Intel Core i7 processor, an NVIDIA RTX 3060 GPU, and TensorFlow-based implementations. They evaluated performance using parameters such as energy, SLA violations, response time, and migration cost.

5.1. Experimental Setup

We then introduce the experimental setup details that serve as the basis for evaluating the DRL-based task scheduling framework in federated cloud environments. A cloud federation simulator was implemented using Python and libraries for machine learning and simulation, where the experiments described were performed. The federated environment featured various CSPs with specific resource and workload configurations to closely approximate reality. The simulations were conducted on a machine equipped with an Intel Core i7 CPU, 16 GB RAM, and an NVIDIA RTX 3060 GPU to train the DRL model. We performed the experiments using Ubuntu 20.04, with Python 3.9, TensorFlow, and PyTorch used to build and train the DRL framework, respectively. Other libraries include NumPy and Pandas for preprocessing and analysis, and Matplotlib was used for visualization.

The federated cloud environment consisted of three Cloud Service Providers (CSPs) with varying resource capacities, specifically in terms of CPU cores, memory, and Bandwidth. We used synthetic traces based upon classic workload patterns, creating variability and complexity to challenge the system as it would encounter in the real world. These Virtual Machines (VMs) were powered with CPU, memory, and Bandwidth to resemble real-world applications. The DRL model used the following hyperparameters: learning rate = 0.001, gammas = 0.99, and batch size = 64. The neural network, with three fully connected layers and ReLU activation functions, was empirically selected to approximate the Q-value. The nodes in each layer were selected based on the dimensionality of the state and action spaces.

The model efficiency was verified under real-world constraints by distributing workloads on the CSPs according to predefined patterns such as peak-load scenarios and random variations. The corresponding evaluation metrics, including energy consumption, SLA compliance, total response time, and migration costs, were measured for each simulation run. All performance metrics report the average of several repetitions of each experiment to guarantee reproducibility. This configuration enables us to create a well-defined and controlled environment for experimenting with the proposed framework for managing task scheduling and resource utilization in federated cloud systems.

5.2. Performance Comparison with Baselines

In this section, we first compare the performance of the proposed FedTaskRL framework with baseline models, including First Come First Serve (FCFS), Round Robin (RR), and Genetic Algorithm (GA). FedTaskRL is comparatively evaluated against state-of-the-art methods in terms of energy consumption, SLA violation, average response time, migration cost, scalability, and adaptability, and shows promising results in optimizing the task-scheduling problem in federated cloud environments.

Table 1. Performance comparison of FedTaskRL with baseline models

Metric	FedTaskRL	First Come First Serve (FCFS)	Round Robin (RR)	Genetic Algorithm (GA)
Energy Consumption (kWh)	28	60	48	42
SLA Compliance (%)	97.5	78	85	91
Response Time (ms)	145	310	260	195
Migration Cost ($)	4.8	19.2	14.5	9.6
Scalability	Excellent	Poor	Moderate	Good
Adaptability	Excellent	Poor	Poor	Good

In Table 1, we provide an exhaustive comparison of the proposed FedTaskRL framework with three baselines: First Come, First Serve (FCFS), Round Robin (RR), and Genetic Algorithm (GA). For the evaluation, the framework was deployed to handle multiple critical performance metrics. A practical framework should address the key challenges in federated cloud environments, including energy consumption, SLA violations, response times, migration costs, scalability, and adaptability.

FedTaskRL reduces energy usage by 28 kWh compared to 60 kWh by FCFS, 48 kWh by RR, and 42 kWh by GA. This shows that the DRL framework can dynamically allocate resources and reduce idle energy waste. Additionally, the proposed framework demonstrates high SLA compliance, completing 97.5% of tasks while adhering to defined SLA constraints. This indicates a significant increase compared to the static strategies implemented by FCFS and RR, which provided 78% and 85% compliance, respectively; it is even higher than GA's 91% compliance.

FedTaskRL takes into consideration access time, which is one of the critical metrics in the case of real-time cloud services, and it has been found to minimize the average response time to 145 ms, compared to 310 ms with use of FCFS, 260 ms with the use of RR, and 195 ms with the use of GA. Due to its ability to predict workload patterns and dynamically allocate resources, FedTaskRL processes tasks efficiently. In addition, the migration costs per migration in FedTaskRL are only $ 4.80, which is significantly lower than those in the FCFS system ($ 19.20), RR system ($ 14.50), and GA system ($ 9.60). These decreases are due to less cost-efficient assignments of VMs learned by the DRL agent. Its most significant strength is its scalability and adaptability, as FedTaskRL excels even as the number of CSPs increases from two to five to handle multiple workloads. FCFS and RR demonstrate inadequate scalability and flexibility, whereas GA scores decently. It indicates that FedTaskRL outperforms state-of-the-art schemes in managing resource heterogeneity and workload variability for federated task management in cloud environments. The metrics employed for evaluation implement the core attributes of Federated cloud task scheduling: Efficiency, Quality of Service, and System Resilience. These experiments also reinforce the practicality of FedTaskRL as an energy impact optimization application widely used in real-world situations to reduce costs while guaranteeing the Service Level Agreement.

5.3. Ablation Study

We conduct an ablation study to assess the contribution of each component in the FedTaskRL framework to its overall performance. The study methodically ablates or alters various features, including the tuning of the reward function, the continuously updated state-space, and the incorporation of neural networks, to evaluate the effect of these features on key metrics such as energy consumption, SLA satisfaction, response time, and relocation cost. This process emphasizes the importance of each component in enabling high performance and pinpoints where the framework might be overly dependent on specific aspects of design. Ablation studies help in understanding how robust and effective a model is, thereby ensuring the model is efficient and scalable as well.

Table 2. Ablation study of FedTaskRL components

Component	Energy Consumption (kWh)	SLA Compliance (%)	Response Time (ms)	Migration Cost ($)
Full FedTaskRL Model	28	97.5	145	4.8
Without Reward Function Tuning	35	88	190	6.7
Without Dynamic State-Space Updates	42	82	220	9.4
Without Neural Network (Baseline Q-Learning)	48	75	260	14.5

In Table 2, we conduct an ablation study to analyze the contribution of each component to the FedTaskRL framework. It identifies critical elements of reward tuning for functions, dynamic state space, and neural networks. It benchmarks their contributions to essential performance measurements, including energy, SLA, response time, and migration cost. The full FedTaskRL model achieves the best performance across all metrics: 28 kWh of energy consumption, 97.5% SLA compliance, an average response time of 145 ms, and a migration cost of $4.80. This demonstrates the ability of the integrated design to optimize tasks and scheduling adaptively.

Performance degrades if the reward function is not tuned to balance energy efficiency, SLA compliance, and migration cost. The increase in energy consumption to 35 kWh, the SLA compliance of only 88%, and the response time of 190 ms demonstrate the deleterious effects of a poorly crafted reward function and underscore the need for a calibrated reward function to guide learning. Depriving dynamic state-space updates, which log workload and resource variations in real time, leads to further deterioration. The energy consumption increases to 42 kWh, the SLA compliance decreases to 82%, and the response time rises to 220 ms, indicating that real-time state-space adaptation is a vital function, as it enables the DRL agent to adapt to an ever-changing environment.

Lastly, replacing the neural network with a Q-learning model is an explicit limitation as it loses the capacity to generalize within high-dimensional state spaces. The results are somewhat worse: energy consumption increases to 48 kWh, SLA capacity drops to 75%, response time rises to 260 ms, and migration costs skyrocket, highlighting that classical Q-learning is inadequate to cope with the complexity of the federated cloud system. An ablation study visually represents how each component contributes to the overall performance of FedTaskRL. It confirms the design rationale behind the framework, highlighting how reward function tuning, dynamic state-space updates, and deep neural networks eliminate the

need for task scheduling and target resources, making the system efficient and robust.

5.4. Scalability Testing

Scalability testing was conducted to assess whether FedTaskRL can continue to perform well with the increasing number of Cloud Service Providers (CSPs) and the increased workload intensity. In this section, we evaluate metrics of energy consumption, SLA compliance, response time, and migration costs for different system scalabilities, demonstrating that the framework is robust and adaptive in a dynamically changing, large-scale, and wide-scale federated cloud environment.

Table 3. Scalability testing results

Number of CSPs	Workload Intensity	Energy Consumption (kWh)	SLA Compliance (%)	Response Time (ms)	Migration Cost ($)
3	Low	28	97.5	145	4.8
5	Medium	32	94.0	160	6.2
10	High	38	89.0	190	9.4

The results of the scalability testing of the FedTaskRL framework, in terms of different system scales defined by the number of CSPs and workload intensity, are summarized in Table 3. These results demonstrate that FedTaskRL can respond to increasing complexity effortlessly without compromising competitive performance metrics. The optimal performance condition, characterized by an energy consumption of 28 kWh, SLA compliance of 97.5%, a response time of 145 ms, and a migration cost of $4.80, is achieved for three CSPs with low workload intensity. This validates that the framework is efficient in smaller environments with less resource contention.

When the system scales to five CSPs but with medium workload intensity, the energy consumption reaches 32 kWh, the SLA compliance drops slightly to 94%, the response time rises to 160 ms, and the migration costs increase to $6.2. This is expected since the changes occur due to the need for more resource allocation and task scheduling decisions in a more complex environment. At high workload intensity with ten CSPs, energy consumption is 38 kWh, SLA compliance is 89%, and response time is 190 ms, while migration costs are $ 9.40 due to the increased computational overhead for managing a data-intensive federated system. This framework demonstrates its scalability, as the performance levels remain acceptable despite these increases. Scalability testing demonstrates that FedTaskRL handles varying scales of operation with competitive performance metrics. This makes it a potential candidate with high adaptability to growing system complexity since it provides a highly effective and robust solution for dynamic and large-scale federated cloud environments.

5.5. Robustness Testing

Robustness testing assesses how well FedTaskRL can handle unexpected scenarios, including sudden workload increases, resource failures, or environmental fluctuations. This section demonstrates the framework's ability to support SLA compliance and load-balanced energy control and resource usage by analyzing performance metrics during these extreme scenarios.

Table 4. Robustness testing results

Scenario	Energy Consumption (kWh)	SLA Compliance (%)	Response Time (ms)	Migration Cost ($)
Normal Conditions	28	97.5	145	4.8
Workload Spike (+50%)	35	91.0	190	7.2
Resource Failure (-30% CPU)	40	87.0	220	9.5

FedTaskRL is resilient to customary conditions, and the two challenging scenarios simulated are a 50% workload increase (increasing worker load from 1 to 3) and a 30% capacity shrink (resulting in 50% more CPU consumed), as shown in Table 4. These results highlight the framework's robustness in adjusting to and maintaining acceptable performance levels under unhealthy conditions. FedTaskRL demonstrates superior performance, achieving an energy consumption of 28 kWh, a Service-Level Agreement (SLA) of 97.5%, a response time of 145 ms, and migration costs of $4.80 when tested under normal operating conditions. This sets a standard for testing performance against more challenging situations.For an all-scenario workload spike where the task volume increases by 50%, the energy consumption is 35 kWh, SLA compliance is 91.0%, the FS response time is 190 ms, and migration costs rise to $ 7.20 due to resource contention. Even under these changes, FedTaskRL is robust enough to reallocate resources for increased tasks within the new environment while keeping its performance indicators at reasonable levels. In the case of a simulated resource failure that causes a 30% decrease in CPU capacity, the energy consumption rises further to 40 kWh, SLA compliance decreases to 87.0%, and the response time increases to 220 ms; additionally, migration costs are equal to $9.5, as the system attempts to migrate tasks to achieve the provided performance using the limited resources available.

Nevertheless, FedTaskRL dampens the impact in this extreme scenario and provides business continuity with considerable SLA compliance.

This robustness testing thus confirms that FedTaskRL can dynamically adapt to challenging conditions, reinforcing its robustness and potential for deployment in real-world federated cloud environments.

5.6. Convergence Analysis

Convergence analysis examines the learning stability and efficiency of FedTaskRL. This section demonstrates the DRL agent's adaptation by optimizing task scheduling in the federated cloud environment, as evidenced by the analysis of cumulative rewards across episodes. Analysis of convergence patterns confirms that the learning process is robust and effective.

Table 5. Convergence analysis results

Training Episodes	Average Cumulative Reward	Energy Consumption (kWh)	SLA Compliance (%)	Response Time (ms)
100	-150	45	82	230
500	50	35	91	190
1000	150	28	97.5	145

Table 6. Comparison with state-of-the-art approaches

Approach	Energy Consumption (kWh)	SLA Compliance (%)	Response Time (ms)	Migration Cost ($)
FedTaskRL (Proposed)	28	97.5	145	4.8
DRL-TS [41]	36	89.3	200	7.9
A3C Scheduler [42]	40	87	220	9.6
DRLIS [43]	33	91.5	170	6.2
EdgeTimer [44]	31	93	160	5.5
MA-DRL [45]	29	95	150	5

The convergence analysis of the FedTaskRL framework, as shown in Table 5, indicates the evolution of various reward metrics as the agent learns over increasing training episodes. This analysis demonstrates the agent's ability to improve decision-making through reinforcement learning and dynamically optimize resource utilization. The DRL agent barely learns, returning an average of -150 cumulative rewards over 100 training episodes. The performance metrics, including energy consumption (45 kWh), SLA compliance (82%), and response time (230 ms), are relatively poor, suggesting suboptimal task scheduling. The beginning of this phase marks the exploration phase, during which the agent explores the environment's dynamics.

Excellent learning is evidenced by a cumulative reward of 50 after 500 episodes. Energy consumption: 35 kWh, SLA compliance: 91%, Response time: 190 ms. Conversely, these numbers indicate that the agent quickly learns to strike a balance between energy efficiency and SLA compliance. The agent reaches a high average cumulative reward of 150 after 1000 episodes. Staying at that level indicates convergence. Here, the performance saturates with optimization of energy (28 kWh), SLA compliance (97.5%), and response time (145 ms), which is the maximum level of preparation that the agent can achieve regarding task scheduling policies, as this stage indicates an optimal balance of resource execution and system performance. The convergence analysis illustrates the convergence and stability of the FedTaskRL training process. These results confirm that the framework can learn in complex federated cloud backgrounds to perform consistent and reliable task scheduling.

5.7. Comparison with State of the Art

In this section, the Fed-Bask method is compared with state-of-the-art techniques for task scheduling in the federated cloud environment, and the proposed FedTaskRL framework is evaluated against the best policies in terms of the monetary cost of self-organization. The analysis considers both SLA compliance and energy-aware metrics, highlighting the advantages of FedTaskRL. The study indicates that FedTaskRL outperforms the state-of-the-art methods in terms of energy consumption, SLA compliance, response time, and migration cost.

Table 6 presents a detailed comparison between the proposed FedTaskRL framework and five dominant DRL-based task scheduling algorithms: DRL-TS [41], A3C Scheduler [42], DRLIS [43], EdgeTimer [44], and MA-DRL [45]. The analysis focuses particularly on the key performance aspects related to federated clouds, including energy, SLA, response time, and migration. The best overall performance is achieved by FedTaskRL, with the lowest energy consumption of 28 kWh, indicating an efficient utilization of resources. This represents a 22% and 30% improvement under the DRL-TS (36 kWh) and A3C Scheduler (40 kWh). In comparison to the more closely performing MA-DRL (29 kWh), FedTaskRL aims to make it more energy efficient.

In terms of SLA satisfaction, FedTaskRL achieves a 97.5% satisfaction rate, surpassing that of all other methods. The second-best, MA-DRL, obtains 95%, and EdgeTimer gets 93%. This demonstrates that FedTaskRL is more effective at meeting service-level guarantees under dynamic workloads. Another crucial metric is response

time, and FedTaskRL tops the chart with 145 ms, followed by MA-DRL (150 ms) and EdgeTimer (160 ms). This can demonstrate that FedTaskRL can effectively address scheduling latency as required in real-time systems. The migration cost, which represents the operational overhead of a virtual machine, is lowest for FedTaskRL at $4.8, slightly larger than EdgeTimer at $5.5, and somewhat smaller than MA-DRL at $5.0. A smaller migration cost is unpacked in more optimal allocations of resources over space. In summary, FedTaskRL reliably outperforms baseline methods across both dimensions, which further indicates its robustness, generality, and optimization ability. Its holistic design, incorporating dynamic state-space update rules, multi-objective reward tuning, and scalable neural approximators as key components, yields powerful capabilities for real-time task scheduling under heterogeneous federation.

6. Discussion

The dynamic and diverse characteristics of federated cloud environments demand that task scheduling be intelligent and capable of coping with changing conditions in real-time, such as workload fluctuations, resource availability, and service requests. Another category for these workloads is traditional heuristic-based and rule-driven schedulers, which fail to be sufficiently adaptive and scalable for such scenarios. The recent development of deep reinforcement learning (DRL) provides feasible solutions for task offloading and resource allocation, but they are not free of defects. These challenges include poor state-space modeling, hard reward, a high migration overhead, and poor multi-objective optimization in decentralized settings. To this end, in this work, we propose FedTaskRL, an innovative federated DRL-based task scheduler with a few distinct features. Unlike traditional approaches, FedTaskRL features a dynamically made richer state space, a neural Q-learning architecture, and a composite reward function tailored to optimize energy consumption, SLA compliance, response time and migration cost jointly. The model can be trained in a federated setting where the data is not shared, maintaining the data locality and scalability. The experimental results evidence the effectiveness of FedTaskRL. From Table 6, we can see that it consistently achieves better performance than the state-of-the-art methods DRL-TS, A3C Scheduler, and MA-DRL in all metrics. The lowering of power usage (28 kWh), increase in SLA fulfilment (97.5%) and drop in migration cost ($4.8) reflect the efficiency of the proposed approach. These results prove the primary intuitive assumption that an improved DRL-based scheduler with a multi-objective reward and adaptable context awareness results in far better scheduling ability.

By addressing the major limitations in the previous models, FedTaskRL pushes forward the frontier of intelligent federated task scheduling. There are profound implications of the above framework for the emerging real-time cloud-edge orchestration in our nextgeneration distributed systems. Limitations of the study and future work are discussed in Section 5.1.

6.1. Limitations of the Study

Although the FedTaskRL framework is a significant step forward in task scheduling in a federated cloud environment, some limitations remain. This evaluation first utilizes synthetic workload traces, which, although they represent real workloads, may still not fully reflect the complexity of cloud operations in the wild. Secondly, such a study is limited to task scheduling, where other problem domains, such as fault tolerance and security, are much more critical. Third, the overhead of training the DRL model is significant, especially in large-scale environments, and needs further improvement for practical applications. Section 6.1 discusses these challenges and opportunities for extending the framework's capabilities to address them fully.

7. Conclusion and Future Work

In this paper, we introduced FedTaskRL, to the best of our knowledge, the first DRL-based task scheduling framework for federated cloud environments. To this end, the framework addresses some of the paramount challenges (e.g., dynamic workload management, energy efficiency, and SLA compliance) through novel aspects (e.g., dynamic state-space updates, a well-crafted reward function, and a transferable and scalable Q-value approximation via neural networks). We experimentally demonstrate that FedTaskRL significantly improves energy consumption, SLA guarantee, response time, and migration cost compared to state-of-the-art approaches, thereby showcasing its efficacy and scalability in modern federated cloud systems. Several aspects can be further explored in future research directions within the FedTaskRL framework. We will enhance our framework with real-world datasets and deploy it in prototypical federated cloud environments to validate its practical applicability. In future designs, fault tolerance and security mechanisms can be incorporated into X-PANDA to enable it to overcome these limitations and address more comprehensive operational challenges. Moreover, minimizing the computational overhead of DRL training will further improve its scalability for larger-scale systems. Another promising direction is to generalize the framework to include multi-objective optimization or optimization in terms of multiple competing resources, such as cost, energy, and latency variables, to be jointly optimized. Finally, the framework can also be applied to other dispersed computing paradigms, such as edge computing, which could expand its usefulness and relevance.

References

[1] Prashanth Choppara, and S. Sudheer Mangalampalli, "Adaptive Task Scheduling in Fog Computing Using Federated DQN and K-Means Clustering," *IEEE Access*, vol. 13, pp. 75466-75492, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[2] Guruh Fajar Shidik et al., "Novel Unsupervised Cluster Reinforcement Q-Learning in Minimizing Energy Consumption of Federated Edge Cloud," *IEEE Access*, vol. 13, pp. 92577-92595, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[3] Jinming Wang et al., "Deep Reinforcement Learning Task Scheduling Method for Real-Time Performance Awareness," *IEEE Access*, vol. 13, pp. 31385-31400, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[4] S. Nambi, and P. Thanapal, "EMO-TS: An Enhanced Multi-Objective Optimization Algorithm for Energy-Efficient Task Scheduling in Cloud Data Centers," *IEEE Access*, vol. 13, pp. 8187-8200, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[5] Xiaojing Chen et al., "Toward Dynamic Resource Allocation and Client Scheduling in Hierarchical Federated Learning: A Two-Phase Deep Reinforcement Learning Approach," *IEEE Transactions on Communications*, vol. 72, no. 12, pp. 7798-7813, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[6] Mazin Abed Mohammed et al., "Federated-Reinforcement Learning-Assisted IoT Consumers System for Kidney Disease Images," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 4, pp. 7163-7173, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[7] Somayeh Kianpisheh, and Tarik Taleb, "Collaborative Federated Learning for 6G With a Deep Reinforcement Learning-Based Controlling Mechanism: A DDoS Attack Detection Scenario," *IEEE Transactions on Network and Service Management*, vol. 21, no. 4, pp. 4731-4749, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[8] Rongqi Zhang et al., "Federated Deep Reinforcement Learning for Multimedia Task Offloading and Resource Allocation in MEC Networks," *IEICE Transactions on Communications*, vol. E107-B, no. 6, pp. 446-457, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[9] Tai Manh Ho, Kim-Khoa Nguyen, and Mohamed Cheriet, "Federated Deep Reinforcement Learning for Task Scheduling in Heterogeneous Autonomous Robotic System," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 1, pp. 528-540, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[10] Hojjat Baghban et al., "Edge-AI: IoT Request Service Provisioning in Federated Edge Computing Using Actor-Critic Reinforcement Learning," *IEEE Transactions on Engineering Management*, vol. 71, pp. 12519-12528, 2024. [CrossRef] [Google Scholar] [Publisher Link]
[11] Zhu Tianqing et al., "Resource Allocation in IoT Edge Computing via Concurrent Federated Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1414-1426, 2022. [CrossRef] [Google Scholar] [Publisher Link]
[12] Wei Huang et al., "FedDSR: Daily Schedule Recommendation in a Federated Deep Reinforcement Learning Framework," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3912-3924, 2023. [CrossRef] [Google Scholar] [Publisher Link]
[13] Peiying Zhang et al., "Deep Reinforcement Learning Assisted Federated Learning Algorithm for Data Management of IIoT," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8475-8484, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[14] S.R. Shishira, and A. Kandasamy, "BeeM-NN: An Efficient Workload Optimization Using Bee Mutation Neural Network in Federated Cloud Environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 3151-3167, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[15] Xu Zhao et al., "Low Load DIDS Task Scheduling Based on Q-Learning in Edge Computing Environment," *Journal of Network and Computer Applications*, vol. 188, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[16] N. Yuvaraj, T. Karthikeyan, and K. Praghash, "An Improved Task Allocation Scheme in Serverless Computing Using Gray Wolf Optimization (GWO) Based Reinforcement Learning (RIL) Approach," *Wireless Personal Communications*, vol. 117, pp. 2403-2421, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[17] Zheyi Chen et al., "Adaptive and Efficient Resource Allocation in Cloud Datacenters Using Actor-Critic Deep Reinforcement Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1911-1923, 2022. [CrossRef] [Google Scholar] [Publisher Link]
[18] G. Matheen Fathima, and L. Shakkeera, "Efficient Task Scheduling and Computational Offloading Optimization with Federated Learning and Blockchain in Mobile Cloud Computing," *Results in Control and Optimization*, vol. 18, pp. 1-16, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[19] Gaith Rjoub et al., "Enhanced Dynamic Deep Q-Network for Federated Learning Scheduling Policies on IoT Devices Using Explanation-Driven Trust," *Knowledge-Based Systems*, vol. 318, pp. 1-17, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[20] Amjad Iqbal, Mau-Luen Tham, and Yoong Choon Chang, "Double Deep Q-Network-Based Energy-Efficient Resource Allocation in Cloud Radio Access Network," *IEEE Access*, vol. 9, pp. 20440-20449, 2021. [CrossRef] [Google Scholar] [Publisher Link]
[21] Tinghao Zhang, Kwok-Yan Lam, and Jun Zhao, "Deep Reinforcement Learning Based Scheduling Strategy for Federated Learning in Sensor-Cloud Systems," *Future Generation Computer Systems*, vol. 144, pp. 219-229, 2023. [CrossRef] [Google Scholar] [Publisher Link]
[22] Zhiyu Wang, Mohammad Goudarzi, and Rajkumar Buyya, "TF-DDRL: A Transformer-Enhanced Distributed DRL Technique for Scheduling IoT Applications in Edge and Cloud Computing Environments," *IEEE Transactions on Services Computing*, vol. 18, no. 2, pp. 1039-1053, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[23] Mahdi Safaei Yaraziz, and Richard Hill, "A Review of Resource Allocation for Maximizing Performance of IoT Systems," *IEEE Access*, vol. 13, pp. 98426-98451, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[24] Himani Chaudhary et al., "Advanced Queueing and Scheduling Techniques in Cloud Computing Using AI-Based Model Order Reduction," *Discover Computing*, vol. 28, pp. 1-40, 2025. [CrossRef] [Google Scholar] [Publisher Link]
[25] Ashwin Singh Slathia et al., "SHERA: SHAP-Enhanced Resource Allocation for VM Scheduling and Efficient Cloud Computing," *IEEE Access*, vol. 13, pp. 92816-92832, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[26] Reena Panwar, and M. Supriya, "RLPRAF: Reinforcement Learning-Based Proactive Resource Allocation Framework for Resource Provisioning in Cloud Environment," *IEEE Access*, vol. 12, pp. 95986-96007, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[27] Yang Yu, and Xiaoqing Tang, "Hybrid Centralized-Distributed Resource Allocation Based on Deep Reinforcement Learning for Cooperative D2D Communications," *IEEE Access*, vol. 12, pp. 196609-196623, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[28] Hafiz Muhammad Fahad Noman et al., "FeDRL-D2D: Federated Deep Reinforcement Learning- Empowered Resource Allocation Scheme for Energy Efficiency Maximization in D2D-Assisted 6G Networks," *IEEE Access*, vol. 12, pp. 109775-109792, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[29] James Adu Ansere et al., "Optimal Computation Resource Allocation in Energy-Efficient Edge IoT Systems With Deep Reinforcement Learning," *IEEE Transactions on Green Communications and Networking*, vol. 7, no. 4, pp. 2130-2142, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[30] Yuao Wang et al., "Cooperative End-Edge-Cloud Computing and Resource Allocation for Digital Twin Enabled 6G Industrial IoT," *IEEE Journal of Selected Topics in Signal Processing*, vol. 18, no. 1, pp. 124-137, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[31] Sowmya Madhavan et al., "Cybertwin Driven Resource Allocation Using Optimized Proximal Policy Based Federated Learning in 6G Enabled Edge Environment," *Digital Communications and Networks*, pp. 1-15, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[32] Antonio Scarvaglieri, Sergio Palazzo, and Fabio Busacca, "A Lightweight, Fully-Distributed AI Framework for Energy-Efficient Resource Allocation in LoRa Networks," *Proceedings of the IEEE/ACM 16th International Conference on Utility and Cloud Computing*, Taormina, Messina, Italy, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[33] Jiayin Zhang et al., "Elastic Task Offloading and Resource Allocation Over Hybrid Cloud: A Reinforcement Learning Approach," *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 1983-1997, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[34] Duc-Dung Tran et al., "Multi-Agent DRL Approach for Energy-Efficient Resource Allocation in URLLC-Enabled Grant-Free NOMA Systems," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 1470-1486, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[35] Syed Usman Jamil, M. Arif Khan, and Sabih Ur Rehman, "Resource Allocation and Task Off-Loading for 6G Enabled Smart Edge Environments," *IEEE Access*, vol. 10, pp. 93542-93563, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[36] Bushra Jamil et al., "IRATS: A DRL-Based Intelligent Priority and Deadline-Aware Online Resource Allocation and Task Scheduling Algorithm in a Vehicular Fog Network," *Ad Hoc Networks*, vol. 141, pp. 1-19, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[37] Palash Roy et al., "Distributed Task Allocation in Mobile Device Cloud Exploiting Federated Learning and Subjective Logic," *Journal of Systems Architecture*, vol. 113, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[38] P. Nehra, and Niththa Kesswani, "Efficient Resource Allocation and Management by Using Load Balanced Multi-Dimensional Bin Packing Heuristic in Cloud Data Centers," *Journal of Supercomputing*, vol. 79, pp. 1398-1425, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[39] Lujie Tang et al., "Joint Optimization of Network Selection and Task Offloading for Vehicular Edge Computing," *Journal of Cloud Computing*, vol. 10, pp. 1-13, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[40] Shashank Swarup, Elhadi M. Shakshuki, and Ansar Yasar, "Task Scheduling in Cloud Using Deep Reinforcement Learning," *Procedia Computer Science*, vol. 184, pp. 42-51, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[41] Abdelkarim Ben Sada et al., "Multi-Agent Deep Reinforcement Learning-Based Inference Task Scheduling and Offloading for Maximum Inference Accuracy under Time and Energy Constraints," *Electronics*, vol. 13, no. 13, pp. 1-27, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[42] Yijun Hao et al., "EdgeTimer: Adaptive Multi-Timescale Scheduling in Mobile Edge Computing with Deep Reinforcement Learning," *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, Vancouver, BC, Canada, pp. 671-680, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[43] Zhiyu Wang et al., "Deep Reinforcement Learning-Based Scheduling for Optimizing System Load and Response Time in Edge and Fog Computing Environments," *Future Generation Computer Systems*, vol. 152, pp. 55-69, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[44] Shreshth Tuli et al., "Dynamic Scheduling for Stochastic Edge-Cloud Environments Using A3C Learning and Residual Recurrent Neural Networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 3, pp. 940-954, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[45] Shuran Sheng et al., "Deep Reinforcement Learning-Based Task Scheduling in IoT Edge Computing," *Sensors*, vol. 21, no. 5, pp. 1-19, 2021. [CrossRef] [Google Scholar] [Publisher Link]