International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Carbon-Aware Reinforcement Learning for Sustainable Task Scheduling in Edge Computing: A Review

Dr.Ankita Karale
Head of Department of Computer Engineering
Sandip Institute of Technology and Research Centre
Nashik, Maharashtra, India
ankita.karale@sitrc.org

Gargie Dharmadhikari
Department of Computer Engineering
Sandip Institute of Technology and Research Centre
Nashik, Maharashtra, India
gargiepd@gmail.com

Aditya Gunjal
Department of Computer Engineering
Sandip Institute of Technology and Research Centre
Nashik, Maharashtra, India
adityagunjal644@gmail.com

Sairam Kawade
Department of Computer Engineering
Sandip Institute of Technology and Research Centre
Nashik, Maharashtra, India
kawadesairam@gmail.com

Karan Chaudhary
Department of Computer Engineering
Sandip Institute of Technology and Research Centre
Nashik, Maharashtra, India
karanrank800@gmail.com

Abstract*—With the growing proliferation of IoT devices, intelligent sensing systems, and latency-sensitive applications, edge computing has emerged as an important computing paradigm for processing workloads closer to end users. However, task scheduling across heterogeneous edge nodes remains challenging because of varying workload characteristics, resource constraints, latency requirements, and the carbon intensity of electricity sources. Existing scheduling approaches commonly emphasize execution time, resource utilization, or energy efficiency, while carbon emissions are often considered separately. This paper presents a structured review of recent research on reinforcement-learning-based scheduling, energy-aware task management, workload clustering, and carbon-aware task offloading in edge computing. Based on the reviewed literature, key limitations are identified in carbon-intensity awareness, state-space scalability, joint optimization of latency and sustainability, and SLA protection. To address these gaps, a carbon-aware reinforcement-learning research framework is proposed that combines K-means-based task clustering, Q-learning, carbon-aware scheduling decisions, and SLA-aware task placement. The framework is designed to jointly consider latency, energy consumption, carbon impact, and quality-of-service requirements. A comparative evaluation methodology and research directions are also outlined to support future empirical validation of carbon-aware scheduling in heterogeneous edge environments.

Index Terms*—Edge computing, carbon-aware scheduling, reinforcement learning, Q-learning, task scheduling, workload clustering, K-means, carbon intensity, SLA-aware scheduling, sustainable computing.

I. Introduction

The rapid growth of IoT applications has increased the demand for low-latency and distributed computing. Edge computing addresses this requirement by placing computational resources closer to data sources and end users, reducing communication latency and network traffic compared with centralized cloud processing [2], [3]. However, edge environments contain heterogeneous and resource-constrained nodes, while incoming tasks differ in computational requirements, deadlines, priorities, and data sizes. Efficient task scheduling therefore remains a key challenge in edge computing [1], [4], [5].

Existing scheduling approaches primarily consider objectives such as latency, throughput, resource utilization, and energy consumption. However, lower energy consumption does not necessarily imply lower carbon emissions because electricity carbon intensity varies with location, energy source, and operating conditions [20]–[22]. Consequently, incorporating carbon intensity into scheduling decisions is important for reducing the environmental impact of distributed computing workloads. Reinforcement learning (RL) provides an adaptive approach for scheduling by learning decisions from interactions with changing system conditions and has been applied to task offloading and resource management in edge environments [6]–[11].

Despite these developments, the combined consideration of workload clustering, RL-based scheduling, carbon intensity, and service-level requirements remains an important research direction. This paper reviews existing approaches related to these areas and proposes a conceptual framework that integrates task clustering, Q-learning, carbon-aware scheduling, and SLA-aware task placement. The framework aims to provide a basis for jointly examining latency, energy consumption, carbon emissions, and SLA compliance in heterogeneous edge environments. The remainder of this paper is organized as follows: Section II reviews the literature, Section III presents the comparative analysis, Section IV discusses the strengths and limitations of existing approaches, Section V identifies research gaps, Section VI presents the proposed CRTSE framework, and Section VII concludes the paper.

II. Literature Review

The reviewed studies can be grouped into four major research directions: reinforcement-learning-based scheduling, clustering-assisted scheduling, energy-aware resource management, and carbon-aware scheduling and offloading.

These approaches differ in their optimization objectives, decision mechanisms, and treatment of environmental sustainability.

A. RL-Based Schedulers for Latency and Throughput

Reinforcement learning has been widely investigated for adaptive task scheduling and resource allocation in edge and cloud environments. SCARL applies attention-based RL for multi-resource scheduling and improves performance over conventional queue-based approaches [1]. Fan et al. proposed CECS-A3C using an MDP and asynchronous advantage actor-critic learning for cloud-edge task scheduling [15]. Other studies have applied DRL to IoT scheduling and edge-cloud workflow management, addressing scalability, execution time, response time, and dynamic workload conditions [16], [22]–[24], [28]. RL-based approaches therefore provide adaptive scheduling decisions under changing system conditions; however, most primarily optimize performance or energy-related objectives rather than explicitly incorporating carbon intensity [1], [15], [16], [22]–[24], [28].

B. Clustering as a Scheduling Preprocessing Step

Task clustering can reduce scheduling complexity by grouping workloads with similar resource and execution characteristics. Ullah and Youn proposed a K-means-based approach that groups edge tasks according to resource requirements before scheduling [18]. Sheikh et al. further investigated dynamic K-means with fuzzy logic for fog environments containing tasks with varying characteristics [19]. These studies demonstrate the usefulness of clustering for resource management, but they do not jointly integrate clustered task states with carbon-aware reinforcement learning. This motivates the use of clustering as a preprocessing mechanism for reducing the scheduling state space.

C. Energy-Aware Reinforcement Learning

RL has also been applied to improve energy efficiency in distributed computing systems. Wang et al. investigated RL-based routing for wireless sensor networks to improve energy utilization [3], while Moghaddasi et al. applied DRL to task distribution across D2D, edge, and cloud environments with emphasis on energy consumption [25]. Federated RL has also been explored for fog computing, enabling collaborative learning while reducing the need to exchange raw data [27]. Although these approaches improve energy efficiency and resource utilization, energy minimization does not directly represent carbon emissions. Their findings therefore provide a basis for extending energy-aware RL toward carbon-aware scheduling.

D. Decentralized and Federated Optimization Patterns

Federated and decentralized learning approaches are relevant to edge environments because they can support distributed model training while reducing the need to share raw data. Martin et al. investigated federated DRL for collaborative network optimization, demonstrating the potential of decentralized learning for distributed systems [4]. Such approaches can improve scalability and privacy across multiple edge nodes, but their integration with carbon-aware task scheduling remains limited.

E. Carbon-Aware Scheduling and Offloading

Recent research has increasingly considered carbon emissions in computing-resource management. Liu et al. combined task clustering and reinforcement learning for edge task scheduling [5], while carbon-aware offloading and scheduling approaches have incorporated workload-related emissions and grid carbon intensity into decision-making [6], [7]. Other studies have investigated carbon-aware federated cloud scheduling [30] and reinforcement-learning-based task clustering [8].

Carbon-aware systems have also been developed for data centers and cloud platforms. Radovanović et al. investigated carbon-aware workload scheduling based on grid carbon intensity [9]. CASPER incorporates renewable-energy availability and SLA requirements into scheduling and resource allocation [10], while GreenScale considers the trade-off between system performance and carbon emissions [11]. EcoLife uses carbon information to adjust serverless execution timing [12], and Su et al. considered carbon cost in online management of learning and inference workloads [13]. Other work has examined carbon-aware capacity provisioning and multi-objective scheduling [29], [31]. These studies demonstrate the importance of carbon-aware decision-making, while also indicating a need for adaptive scheduling methods that jointly consider carbon intensity, workload dynamics, and service requirements.

TABLE I
SUMMARY OF REVIEWED LITERATURE ON RL-BASED, CLUSTERING-BASED, AND CARBON-AWARE EDGE SCHEDULING

Ref	Study & Approach	Primary Focus	Carbon-Aware
[1]	Cheong et al., 2019 – SCARL (attention-based RL)	Multi-resource latency/throughput	No
[15]	Fan et al., 2020 – CECS-A3C (MDP + A3C)	Cloud-edge processing time	No
[16]	Sheng et al., 2021 – MDP + DRL	IoT scheduling scalability	No
[17]	Mao et al., 2016 – RL-based offloading	Energy-harvesting adaptation	No
[22]	Jayanetti et al., 2022 – DRL	Execution time & energy	No
[23]	Wang et al., 2024 – DRL scheduler	Response time (edge/fog)	No
[24]	Zhu et al., 2024 – RL workflow scheduling	Execution time	No
[28]	Mounesan et al., 2024 – RL scheduling	Volunteer edge-cloud throughput	No
[18]	Ullah & Youn, 2020 – K-means + M/M/c (KTCS)	Resource-based clustering	No
[19]	Sheikh et al., 2023 – Fuzzy K-means	Fog scheduling accuracy	No
[3]	Wang et al., 2024 – RL-based routing	WSN energy balance	No

Ref	Study & Approach	Primary Focus	Carbon-Aware
[25]	Moghaddasi et al., 2024 – DRL	D2D-edge-cloud energy	No
[27]	Azarkasb & Khasteh, 2025 – Federated RL	Fog energy & privacy	No
[4]	Martin et al., 2025 – Federated DRL	Cellular ENDC performance	No
[5]	Liu et al., 2025 – Clustering + RL (CRTSE)	Edge latency & efficiency	No
[6]	Wang et al., 2024 – Carbon-aware offloading	Data-transfer emissions	Yes
[7]	Gao et al., 2026 – Carbon-aware scheduling	Grid carbon intensity	Yes
[30]	Wang et al., 2023 – RL scheduling	Sustainable federated cloud	Yes
[8]	Leong et al., 2021 – RL-based clustering	Clustering optimization	No
[9]	Radovanović et al., 2023 – Heuristic platform	Datacenter emissions	Yes
[10]	Souza et al., 2023 – Casper (heuristic)	Renewable energy + SLA	Yes
[11]	Kim et al., 2023 – GreenScale (heuristic)	Performance–carbon trade-off	Yes
[12]	Jiang et al., 2024 – EcoLife (heuristic)	Serverless carbon timing	Yes
[13]	Su et al., 2023 – Online control (learning-based)	Colocated learning/inference carbon cost	Yes
[29]	Cao et al., 2025 – Digital twin	Adaptive capacity provisioning	Yes
[31]	Younesi et al., 2025 – Multi-objective scheduling	Energy & latency (IoT-CPS)	Partial
[14]	Yang et al., 2025 – Survey	Container orchestration trends	Yes

Note: "Carbon-Aware" indicates whether the study explicitly models or optimizes carbon intensity/emissions rather than energy consumption alone; "Partial" denotes studies that consider energy or environmental cost without direct grid carbon-intensity modeling.

III. Comparative Analysis

The reviewed studies were compared based on their primary optimization objectives, scheduling approaches, and treatment of carbon emissions. As summarized in Table I, RL-based approaches mainly target latency, throughput, execution time, and energy efficiency, whereas carbon-aware approaches explicitly consider carbon intensity or emissions in scheduling decisions [1], [6], [9]–[14], [20]–[24].

RL-based methods such as SCARL and CECS-A3C demonstrate the ability to adapt scheduling decisions to changing workloads and system conditions [1], [15]. Other studies extend RL to energy-aware routing, edge-cloud scheduling, and workflow optimization [3], [22]–[25], [28]. In contrast, several carbon-aware approaches rely on heuristic, rule-based, or optimization-based mechanisms to exploit variations in grid carbon intensity and renewable-energy availability [9]–[12], [14], [29]. These approaches demonstrate the feasibility of reducing carbon emissions but provide less emphasis on adaptive RL-based decision-making.

The comparison therefore indicates a gap between adaptive RL-based scheduling and explicit carbon-aware optimization. Combining workload clustering, reinforcement learning, and carbon-intensity information could provide a more integrated approach to balancing performance, energy consumption, and environmental impact.

IV. Critical Analysis of Existing Literature

A. Strengths of the Reviewed Body of Work

The reviewed literature demonstrates that reinforcement learning can support adaptive task scheduling in distributed edge and cloud environments. RL-based approaches address changing workloads and resource conditions while optimizing objectives such as latency, throughput, and energy efficiency [1], [15]–[17], [22], [23]. Clustering further reduces scheduling complexity by grouping tasks with similar characteristics before decision-making [18], [19]. Carbon-aware approaches demonstrate that considering grid carbon intensity and renewable-energy availability can reduce the environmental impact of computing workloads while maintaining service requirements [9], [10], [12].

B. Limitations and Methodological Concerns

Despite these advances, most RL-based scheduling approaches focus on latency, throughput, or energy efficiency without explicitly incorporating carbon intensity into the learning process [1], [2], [15]–[17], [22], [25], [28]. Conversely, several carbon-aware approaches rely on heuristic or optimization-based methods, which may be less adaptable to changing workloads and carbon-intensity conditions [9], [12], [14].

Another limitation is the lack of a common evaluation methodology. Differences in datasets, workload models, simulation environments, performance metrics, and baseline algorithms make direct comparison between studies difficult. In addition, carbon-aware scheduling often depends on external carbon-intensity data, making scheduling decisions sensitive to the timeliness and accuracy of these inputs [9], [13].

V. Research Gap Analysis

The comparative and critical analysis identifies four major gaps in existing carbon-aware scheduling research:

- Limited integration of carbon intensity with RL: Most RL-based scheduling methods primarily optimize latency, throughput, or energy consumption, while relatively few incorporate carbon intensity directly

into the state or reward function [1], [2], [15], [17], [22], [25], [28]. Carbon-aware approaches therefore remain largely dependent on non-learning methods [6], [9], [14].

- Limited integration of clustering and carbon-aware RL: Task clustering [18], [19] and RL-based scheduling [8], [15], [16] have largely been investigated as separate mechanisms. Their combined use with carbon-aware decision-making remains insufficiently explored.
- Adaptation and multi-objective decision-making: Existing carbon-aware approaches may depend on predefined rules or optimization models and may not adapt effectively to changing workloads and grid carbon intensity [9], [12], [14]. In addition, balancing carbon reduction with latency, energy consumption, and SLA requirements remains challenging.
- Scalability, data dependency, and validation: Tabular RL methods can become difficult to scale as the scheduling state space increases. Carbon-aware scheduling also depends on timely and accurate carbon-intensity information, while many approaches remain primarily simulation-based and lack large-scale real-world validation. These issues limit practical deployment and motivate further investigation of scalable and adaptive approaches.

VI. Proposed Framework: CRTSE

The proposed Carbon-aware Reinforcement Learning-based Task Scheduling for Edge Computing (CRTSE) framework integrates task clustering, reinforcement learning, and carbon-aware scheduling into a unified scheduling process. It consists of four functional layers: task profiling and clustering, Q-learning-based scheduling, carbon-aware decision-making, and cloud fallback with continuous feedback. The framework is further described through its reward formulation, Q-learning update rule, and evaluation methodology.

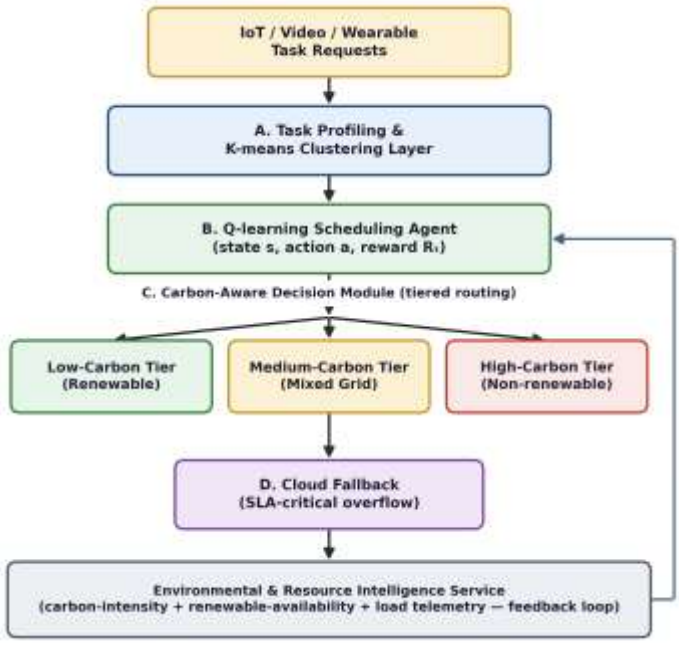


Fig. 1. CRTSE architecture: task profiling and clustering feed a Q-learning scheduler, which — guided by a carbon-aware decision module — routes work across low-, medium-, and high-carbon edge tiers, with cloud fallback for SLA-critical overflow and continuous feedback from an environmental and resource intelligence service.

A. Task Profiling and Clustering Layer

Incoming tasks are characterized using parameters such as payload size, computational requirement, latency deadline, dependency, and priority. K-means clustering is then applied to group tasks with similar scheduling characteristics before scheduling [8], [18], [19]. This preprocessing reduces the scheduling state space and enables the RL agent to make decisions more efficiently.

B. Reinforcement Learning Scheduling Agent

A Q-learning agent observes the current system state, including server queue length, processor utilization, energy consumption, and carbon intensity. It selects an appropriate edge server for each task cluster and receives a reward based on the scheduling objectives. Through repeated interaction with the environment, the agent updates its Q-values and improves subsequent scheduling decisions [6], [16].

C. Carbon-Aware Decision Module

The framework classifies edge nodes according to their current carbon intensity and energy source. Nodes with lower carbon intensity are preferred, while higher-carbon nodes are selected when required to satisfy latency or SLA constraints. Carbon information is incorporated into the scheduling decision and reward mechanism to balance performance and environmental objectives [10], [11].

D. Cloud Fallback and Feedback

When an edge node cannot satisfy the required SLA, the task can be forwarded to a cloud data center. The framework also receives updated information on renewable-energy availability, processor utilization, and grid carbon intensity. This feedback allows scheduling decisions to adapt to changing resource and environmental conditions.

E. Multi-Objective Reward Formulation

The proposed reward function jointly considers execution latency, energy consumption, carbon emissions, and SLA violations. To account for differences in scale, the latency, energy, and carbon-emission terms are normalized before aggregation:

$$R_t(i,j) = -(\alpha \cdot \hat{T}_total(i,j) + \beta \cdot \hat{E}_consumed(i,j) + \gamma \cdot \hat{C}_emission(i,j)) - P_SLA \quad (1)$$

where $\hat{T}$_total, $\hat{E}$_consumed, and $\hat{C}$_emission denote normalized latency, energy consumption, and carbon emissions, respectively, and $\alpha + \beta + \gamma = 1$. Carbon emissions are estimated from the energy consumption and the carbon intensity of the electricity source:

$$C_emission(i,j) = E_consumed(i,j) \times C_intensity(j)$$

The SLA penalty is defined as:

$$P_SLA = \lambda \max(0, (T_total(i,j) - D_i) / D_i)$$

where D_i is the maximum allowable completion time for task i, and λ controls the penalty for violating the SLA deadline.

F. Q-Learning Update Rule

The Q-learning agent continuously updates its action-value function by applying the temporal-difference learning rule shown in Equation (3). During each scheduling iteration, the agent evaluates the reward received from its current action and uses this information to improve future scheduling decisions. This iterative learning process enables the scheduler to gradually identify policies that achieve an effective balance between latency, energy consumption, carbon emissions, and SLA compliance.

$$Q(s, a) \leftarrow Q(s, a) + \eta[R_t + \delta \max_{a'} Q(s', a') - Q(s, a)] \quad (3)$$

In Equation (3), η represents the learning rate, which determines the influence of newly acquired information during model training, whereas δ denotes the discount factor that controls the importance assigned to future rewards. The variables s' and a' correspond to the next system state and the available actions in that state. To balance exploration and exploitation, the proposed framework employs an ε-greedy strategy, allowing the agent to explore alternative scheduling decisions while still selecting actions that have previously produced favorable outcomes. This strategy improves learning efficiency and helps the scheduler adapt to dynamic edge computing environments.

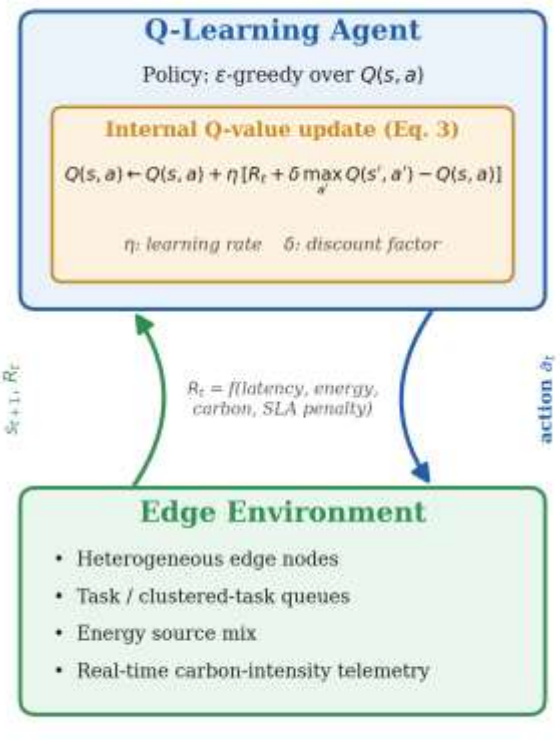


Fig. 2. Closed-loop agent–environment interaction of the CRTSE Q-learning scheduler: the agent selects an action through an ε-greedy policy over Q(s,a), the edge environment returns the next state and a reward reflecting latency, energy consumption, carbon emissions, and SLA compliance, and the reward drives the Q-value update of Eq. (3).

G. Proposed Evaluation Methodology

Since CRTSE is presented as a conceptual framework, its performance can be evaluated using discrete-event simulation with heterogeneous edge nodes and representative IoT workloads. The framework can be compared with latency-first greedy, static carbon-aware, and RL-based scheduling approaches. The evaluation should consider SLA compliance rate, carbon emissions per task, average task latency, energy consumption, and cloud fallback rate. Carbon-intensity data can be obtained from historical or real-time grid carbon-intensity datasets. Sensitivity analysis can further examine the effect of reward weights, workload intensity, and carbon-intensity variations on scheduling performance.

TABLE II
PROPOSED EVALUATION METRICS FOR CRTSE

Metric	Definition	Target Direction
SLA compliance rate	Percentage of tasks completed within deadline D_i	Maximize
gCO2 / task	Mean carbon emitted per completed task	Minimize
Mean latency	End-to-end delay across all tasks	Minimize
Energy per task	kWh consumed per completed task	Minimize
Cloud fallback rate	Percentage of tasks routed to cloud backup	Minimize

In addition to the evaluation metrics in Table II, sensitivity analysis of the weighting coefficients α, β, and γ can be performed to examine the trade-off among latency, energy consumption, and carbon emissions. Varying these coefficients can show how different objective priorities affect scheduling behavior, carbon emissions, and SLA compliance.

VII. Conclusion

This review examined recent research on reinforcement-learning-based and carbon-aware task scheduling in edge computing. The analysis identified key challenges related to carbon-intensity awareness, scalability, multi-objective optimization, and SLA protection. To address these gaps, the proposed CRTSE framework integrates K-means-based task clustering, Q-learning, carbon-aware scheduling, and SLA-aware task placement within a unified framework. The reward formulation jointly considers latency, energy consumption, carbon emissions, and SLA compliance. Since CRTSE is presented as a conceptual framework, its effectiveness requires validation through simulation and real-world experiments. Future research can investigate deep reinforcement learning, federated learning, predictive carbon-intensity models, and large-scale real-world validation to improve the practical applicability of carbon-aware edge computing.

References

[1] M. Cheong, H. Lee, I. Yeom, and H. Woo, "SCARL: Attentive Reinforcement Learning-Based Scheduling in a Multi-Resource Heterogeneous Cluster," IEEE Access, vol. 7, pp. 153432–153444, 2019.

[2] X. Lu, C. Liu, S. Zhu, Y. Mao, P. Lio, and P. Hui, "RLPTO: A Reinforcement Learning-Based Performance-Time Optimized Task and Resource Scheduling Mechanism for Distributed Machine Learning," IEEE Trans. Parallel Distrib. Syst., vol. 34, no. 12, 2023.

[3] Y. Wang, K. Deng, Y. Ren, J. Chan, G. Tian, and C. S. Jensen, "Reinforcement Learning for Tackling Energy-Saving and Energy-Balance Dilemma of Cluster-Based Routing Protocols in WSNs," IEEE Access, vol. 12, 2024.

[4] A. Martin, I. de-la-Bandera, A. Mendo, J. Outes, J. Ramiro, and R. Barco, "Federated Deep Reinforcement Learning for ENDC Optimization," IEEE Trans. Mobile Comput., vol. 24, no. 6, 2025.

[5] H. Liu, L. Tian, and M. Guo, “CRTSE: Clustering and Reinforcement Learning Based Task Scheduling Algorithm for Edge Computing,” IEEE Trans. Cloud Comput., vol. 14, no. 1, pp. 129–145, 2025, doi: 10.1109/TCC.2025.3640958.

[6] L. Wang et al., "Carbon-Aware Task Offloading for Sustainable Edge Computing," IEEE Internet Things J., 2024.

[7] G. Gao et al., "Carbon-Aware Task Scheduling for Cloud-Edge IoT Systems," IEEE Internet Things J., 2026.

[8] C. P. Leong, C. S. Liew, C. S. Chan, and M. H. U. Rehman, “Optimizing Workflow Task Clustering Using Reinforcement Learning,” IEEE Access, vol. 9, pp. 110614–110626, 2021.

[9] A. Radovanović, R. Koningstein, I. Schneider, B. Chen, A. Duarte, B. Roy, D. Xiao, M. Haridasan, P. Hung, N. Care, and S. Talukdar, "Carbon-Aware Computing for Datacenters," IEEE Trans. Power Syst., vol. 38, no. 2, pp. 1270–1280, 2023.

[10] A. Souza, S. Jasoria, B. Chakrabarty, A. Bridgwater, A. Lundberg, F. Skogh, A. Ali-Eldin, D. Irwin, and P. Shenoy, "Casper: Carbon-Aware Scheduling and Provisioning for Distributed Web Services," in Proc. 14th ACM Int. Green and Sustainable Computing Conf., 2023, pp. 67–73.

[11] Y. G. Kim, U. Gupta, A. McCrabb, Y. Son, V. Bertacco, D. Brooks, and C.-J. Wu, "GreenScale: Carbon-Aware Systems for Edge Computing," arXiv:2304.00404, 2023.

[12] Y. Jiang, R. Basu Roy, B. Li, and D. Tiwari, "EcoLife: Carbon-Aware Serverless Function Scheduling for Sustainable Computing," in Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis (SC), 2024.

[13] S. Su, Z. Zhou, T. Ouyang, R. Zhou, and X. Chen, "Learning to Be Green: Carbon-Aware Online Control for Edge Intelligence with Colocated Learning and Inference," in Proc. IEEE 43rd Int. Conf. Distributed Computing Systems (ICDCS), 2023.

[14] J. Yang, Z. Saad, J. Wu, X. Niu, H. Leung, and S. Drew, "A Survey on Task Scheduling in Carbon-Aware Container Orchestration," arXiv:2508.05949, 2025.

[15] Q. Fan, Z. Li, and X. Chen, "Deep Reinforcement Learning Based Task Scheduling in Edge Computing Networks," in Proc. IEEE/CIC Int. Conf. Communications in China (ICCC), 2020.

[16] S. Sheng, P. Chen, Z. Chen, L. Wu, and Y. Yao, "Deep Reinforcement Learning-Based Task Scheduling in IoT Edge Computing," Sensors, vol. 21, no. 5, art. 1666, 2021.

[17] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices," IEEE J. Sel. Areas Commun., vol. 34, no. 12, pp. 3590–3605, 2016.

[18] I. Ullah and H. Y. Youn, "Task Classification and Scheduling Based on K-Means Clustering for Edge Computing," Wireless Pers. Commun., vol. 113, pp. 2611–2624, 2020.

[19] M. S. Sheikh, R. N. Enam, and R. I. Qureshi, "Machine Learning-Driven Task Scheduling with Dynamic K-Means Based Clustering Algorithm Using Fuzzy Logic in FOG Environment," Frontiers in Computer Science, vol. 5, 2023.

[20] F. Khoramnejad and M. Erol-Kantarci, "On Joint Offloading and Resource Allocation: A Double Deep Q-Network Approach," IEEE Trans. Cogn. Commun. Netw., vol. 7, no. 4, pp. 1126–1141, 2021.

[21] A. Bhushan, A. K. Singh, S. Sachan, and S. Mishra, “Improving Energy Efficiency in IoT Edge Computing Through Dynamic Deep Q-Network Based Resource Management,” Software: Practice and Experience, 2025, doi: 10.1002/spe.70017.

[22] A. Jayanetti, S. Halgamuge, and R. Buyya, "Deep Reinforcement Learning for Energy and Time Optimized Scheduling of Precedence-Constrained Tasks in Edge-Cloud Computing Environments," Future Gener. Comput. Syst., vol. 137, pp. 14–30, 2022.

[23] Z. Wang, M. Goudarzi, M. Gong, and R. Buyya, "Deep Reinforcement Learning-Based Scheduling for Optimizing System Load and Response Time in Edge and Fog Computing Environments," Future Gener. Comput. Syst., vol. 152, pp. 55–69, 2024.

[24] K. Zhu, Z. Zhang, S. Zeadally, and F. Sun, "Learning to Optimize Workflow Scheduling for an Edge-Cloud Computing Environment," IEEE Trans. Cloud Comput., 2024.

[25] K. Moghaddasi, S. Rajabi, F. S. Gharehchopogh, and A. Ghaffari, "An Advanced Deep Reinforcement Learning Algorithm for Three-Layer D2D-Edge-Cloud Computing Architecture for Efficient Task Offloading in the Internet of Things," Sustain. Comput.: Informatics and Systems, vol. 43, art. 100992, 2024.

[26] Y. Wang and S. Sun, "Dynamic Task Scheduling in Wireless Edge Computing Using Deep Reinforcement Learning with Ordinal Optimization," J. Wireless Commun. Netw., vol. 2025, art. 96, 2025.

[27] S. K. Azarkasb and S. H. Khasteh, "Fog Computing Tasks Management Based on Federated Reinforcement Learning," J. Grid Comput., vol. 23, no. 1, 2025.

[28] M. Mounesan, M. Lemus, H. Yeddulapalli, P. Calyam, and S. Debroy, "Reinforcement Learning-Driven Data-Intensive Workflow Scheduling for Volunteer Edge-Cloud," in Proc. IEEE 8th Int. Conf. Fog and Edge Computing (ICFEC), 2024, pp. 79–88.

[29] Z. Cao, R. Wang, X. Zhou, R. Tan, Y. Wen, Y. Yan, and Z. Wang, "Adaptive Capacity Provisioning for Carbon-Aware Data Centers: A Digital Twin-Based Approach," IEEE Trans. Sustain. Comput., vol. 10, pp. 730–743, 2025.

[30] Z. Wang, S. Chen, L. Bai, J. Gao, J. Tao, R. R. Bond, and M. D. Mulvenna, "Reinforcement Learning Based Task Scheduling for Environmentally Sustainable Federated Cloud Computing," J. Cloud Comput., vol. 12, no. 1, art. 174, 2023.

[31] A. Younesi, E. Oustad, M. Abolnejadian, M. Ansari, and A. Ejlali, “MoTiCPS: Energy Optimization on Multi-Objective Task Scheduling in IoT-Integrated Cyber-Physical Systems,” IEEE Trans. Sustain. Comput., vol. 10, no. 4, pp. 744–755, 2025, doi: 10.1109/TSUSC.2024.3525090.