


Mobile computing: Energy-aware task offloading in mobile edge computing environments



 **Mohammed M Alenazi**

Faculty of Computers and Information Technology, Department of Computer Engineering, University of Tabuk, Tabuk, 71421, Saudi Arabia.
Email: m-alenazi@ut.edu.sa



ABSTRACT

Article History

Received: 30 May 2025

Revised: 4 August 2025

Accepted: 26 August 2025

Published: 18 September 2025

Keywords

5G/6G networks

Blockchain security

Deep reinforcement learning

Energy-aware task offloading

Latency optimization

Mobile edge computing

Scalability

Smart City applications.

This study presents EATS-MEC an intelligent MEC framework aimed at optimizing energy-aware task offloading and resource scheduling in ultra-dense network environments. The objective is to improve energy efficiency, scalability, and latency compliance under heterogeneous and mobile edge conditions. EATS-MEC integrates Deep Reinforcement Learning (DRL) for real-time task allocation and a lightweight blockchain module to ensure secure, decentralized execution across edge, fog, and cloud layers. Unlike classical models such as Deep Q-Networks (DQN) and Genetic Algorithms (GA), EATS-MEC adaptively responds to real-time network and mobility feedback to determine the optimal execution location for each task. Simulations demonstrate that EATS-MEC reduces peak energy consumption by 32%, extends device battery life by up to 20 hours, and achieves a task success rate of 88.3% under stringent deadline constraints. The framework shows superior performance in mobility-aware energy usage and exhibits near-sublinear energy scaling behavior with increasing device density, maintaining high task throughput even with over 100,000 concurrent tasks. Results indicate that EATS-MEC outperforms existing baselines in energy-latency trade-offs and operates close to the Pareto frontier. Due to its robust, secure, and adaptive nature, EATS-MEC is highly suitable for deployment in real-world smart city infrastructures, healthcare IoT, and latency-sensitive industrial applications.

Contribution/ Originality: This study contributes to the existing literature on energy-aware task offloading in MEC. It uses a new estimation methodology integrating DRL and blockchain. The study introduces a new formula for joint energy-latency optimization. It is one of the few studies that have investigated scalability with over 100,000 tasks.

1. INTRODUCTION

With the rapid rise of mobile devices and IoT applications, mobile networks are expected to carry 77 percent of global IP traffic by 2027 [1-3]. The energy-efficient, low-latency computing solution that has been in demand is especially relevant as growth occurs in autonomous vehicles, healthcare IoT, and augmented reality. To address these challenges, Mobile Edge Computing (MEC) [4, 5] shifts computational workloads across three levels of decentralization: the cloud layer, which handles heavy-duty tasks and manages global scheduling; the fog layer, responsible for local computation to meet latency requirements; and the edge layer, which performs real-time tasks near data sources [6-8]. However, most existing task offloading strategies fail to optimize instantaneous power consumption, particularly in dynamic network environments. For example, static offloading policies waste up to 40% of energy because of outdated network state assumptions, heterogeneous device capabilities, and security vulnerabilities [9-11]. It puts forth these limitations as proof that the existing technologies will not be able to meet

the challenges of energy efficiency, real-time adaptability, and scalability with the coming of 5G/6G networks and ultra-dense IoT ecosystems [12, 13].

Although there are efforts to improve energy-aware task offloading in Mobile Edge Computing (MEC), the current models are unable to achieve the desired trade-off between energy efficiency, real-time adaptability, and security in a dynamic multi-tier environment [14]. Traditional static offloading approaches [15] are not flexible, and heuristic-based approaches [16] would fail to work in fluctuating network environments. Energy-efficient AI-driven models [6, 17] are based on energy efficiency optimization but do not address the security vulnerabilities of multi-tier collaborative offloading [18]. Furthermore, current frameworks generalize uniform capabilities of devices, manifesting energy inequality in IoT device ecosystems [19]. However, most models are not designed to be scalable for 5G/6G ultra-dense networks [20]. Therefore, this research proposes a new framework, EATS-MEC, which combines DRL for adaptive task scheduling and blockchain based on decentralization and security. It not only protects energy-efficient offloading but also prevents unauthorized access and improves scalability in MEC environments, aiming to make MEC environments more secure, resilient, and optimized for next-generation networks.

1.1. Energy-Efficient Task Offloading Optimization in MEC

To perform energy-aware task offloading in MEC, an advanced optimization framework for computing resource assignment is required to minimize power consumption while satisfying computational performance. The problem involves dynamically deciding whether to execute tasks locally or offload them to edge, fog, or cloud layers, considering constraints related to energy, network variation, and task dependencies.

Let \mathcal{T} be the set of all tasks, E_i the energy consumption for executing task i , C_i the computational complexity, and D_i the deadline constraint. Define x_i as a binary offloading decision variable and f_i as the allocated CPU frequency for task execution.

$$\min \sum_{i \in \mathcal{T}} \left[x_i \left(\frac{C_i}{f_i} P_{edge} \right) + (1 - x_i) \left(\frac{C_i}{f_{cloud}} P_{cloud} \right) \right] \quad (1)$$

Subject to:

$$x_i \in \{0,1\}, \quad \forall i \in \mathcal{T} \quad (2)$$

$$f_i \geq f_{min}, \quad f_i \leq f_{max}, \quad \forall i \in \mathcal{T} \quad (3)$$

$$\sum_{i \in \mathcal{T}} (x_i C_i) \leq R_{edge}, \quad \sum_{i \in \mathcal{T}} ((1 - x_i) C_i) \leq R_{cloud} \quad (4)$$

$$D_i \geq \frac{C_i}{f_i} + \frac{S_i}{B_i}, \quad \forall i \in \mathcal{T} \quad (5)$$

Where P_{edge} and P_{cloud} denote the power consumption per computational cycle for edge and cloud computing, respectively, S_i is the data size of task i , and B_i is the bandwidth available for task transmission.

Similarly, this formulation solves the problem of task offloading to minimize energy consumption while using dynamic task allocation, frequency scaling of the CPU, and constraints on the network bandwidth [21, 22]. The frequency scaling and offloading decisions, taken together with the objective function, ensure energy efficiency subject to the constraint, which guarantees computational and network resource feasibility.

1.2. Latency-Constrained and Scalable Task Scheduling in MEC

The scheduling of tasks in MEC is not only important as it must provide minimum latency but also minimize network congestion, computational constraints, and system scalability. The crucial problem is to devise a sound scheduling algorithm when there are heterogeneous computational nodes and stochastic task inputs [23, 24].

Let \mathcal{T} represent the task set, L_i be the latency of task i , P_i the priority weight, and R_i the allocated resources.

$$\min \sum_{i \in \mathcal{T}} P_i \left[\frac{L_i}{R_i} + \alpha \cdot e^{-\beta \cdot x_i} \right] \quad (6)$$

Subject to:

$$L_i \leq L_{max}, \quad \forall i \in \mathcal{T} \quad (7)$$

$$R_i x_i \leq R_{edge} + R_{fog} + R_{cloud}, \quad \forall i \in \mathcal{T} \quad (8)$$

$$\sum_{i \in \mathcal{T}} x_i \leq N_{max}, \quad \sum_{i \in \mathcal{T}} \frac{L_i}{p_i} \leq \lambda_{threshold} \quad (9)$$

$$\sum_{i \in \mathcal{T}} e^{-\gamma \cdot L_i} \geq \delta, \quad \forall i \in \mathcal{T} \quad (10)$$

Where $\alpha, \beta, \gamma, \delta$ are system-defined parameters controlling latency sensitivity and task distribution constraints, ensuring an adaptive and balanced task allocation.

The formulation here is to minimize latency while maintaining system scalability. The exponential latency scaling-based objective function dynamically balances the priority of the tasks, and the constraint handles the computational resources, ensuring that congestion limits are not exceeded. The result is that real-time processing and scalability are kept efficient.

This research addresses key questions such as:

How can task offloading in dynamic MEC environments be made both energy-efficient and latency-aware?

What role can deep reinforcement learning and blockchain integration play in optimizing secure workload distribution in ultra-dense IoT networks?

Can an adaptive, mobility-aware framework outperform traditional offloading models in terms of energy-latency trade-off and scalability?

To answer these questions, the study proposes the EATS-MEC framework, formulates energy-latency optimization objectives, designs a tri-layer MEC architecture with RL-based decision-making, and evaluates its performance through comprehensive simulations compared to DQN and GA baselines.

EATS-MEC integrates mobility-aware adaptability, a secure framework, and resolves energy-latency tradeoffs in dynamic MEC environments to develop EATS-MEC, a secure and energy-efficient task offloading framework.

To develop EATS-MEC, a hybrid AI-blockchain model that enables secure and adaptive task offloading in MEC while minimizing energy consumption.

To design mobility-aware scheduling algorithms using deep learning to dynamically predict and adapt to network fluctuations for energy-efficient task execution.

To optimize energy-latency trade-offs by integrating deep reinforcement learning techniques to enhance resource allocation in MEC with 5G/6G support.

To validate the scalability of EATS-MEC under ultra-dense IoT workloads, demonstrating real-time processing of 100,000+ tasks with minimal performance degradation.

As mentioned above, this research integrates AI-driven adaptability with blockchain security to provide scalability, energy efficiency, and optimized real-time performance for next-generation networks.

Hybrid AI-blockchain model: Plugin to DRL-based scheduling is EATS-MEC, where scheduling is integrated with blockchain authentication for secure, decentralized, and adaptive task offloading.

Mobility-aware adaptive scheduling: Mobility prediction based on LSTM reduces energy waste by 30% and improves network adaptability in dynamic environments.

Energy-latency optimization framework: It has been shown that EATS-MEC reduces the energy-delay product (EDP) by 45% compared to state-of-the-art offloading techniques [25].

5G-enabled scalability: It shows a 60% improvement in task speed and supports more than 100,000 simultaneous tasks in ultra-dense MEC environments.

The rest of the paper is structured as follows: Literature Review, which investigates the existing studies on energy-aware task offloading and MEC optimization strategies. The Methodology section describes proposed mathematical models, optimization techniques, and system design. Experimental evaluations, comparative analysis,

and performance metrics insights are provided in the Results and Discussions. The Conclusion summarizes key findings, limitations, and future research directions.

2. LITERATURE REVIEW

Consequently, Mobile Edge Computing (MEC) has become the paradigm for providing computational resources next to mobile devices to reduce latency and enhance energy efficiency. This has led to an increasing demand for such computationally intensive applications, which call for intelligent task offloading mechanisms to alleviate processing loads among mobile devices, edge servers, fog nodes, and the cloud. The task offloading considers energy awareness and optimizes the computation task assignment to reduce power consumption with the aim of satisfying performance and low latency constraints. Liu et al. [26] point out that considering offloading decisions for delay-sensitive tasks is crucial to reduce energy consumption by up to 35%. Zhao and Lu [17] and Mondal et al. [27] also propose a deep DQN method for task offloading based on medical mobile devices, achieving 42% power saving but with the computational efficiency still very high. The model was effective in real time with moderate deviation from accuracy, processing 500 inference tasks per second. Sada et al. [25] also emphasize that energy saving, in addition to energy reduction through selective inference task offloading, can achieve an additional 20% for real-time applications, making it a viable technique for healthcare and IoT systems.

Task execution is optimized by MEC through multiple layers that collectively facilitate this enhancement. Jiang et al. [28] highlight that cloud resources serve as a fallback when there is no power available for computations, with the cloud layer responsible for more complex calculations and long-term data storage. Beneath the fog layer, an intermediary provides local processing to meet the requirements of latency-sensitive applications. Mehrabi et al. [29] propose a cooperative edge offloading model that reduces dependency on the cloud by 30%, while also decreasing transmission costs and energy consumption. Xiong et al. [30] demonstrate that the edge layer can enhance computational efficiency by 27% in terms of energy efficiency using an energy-aware algorithm. Their findings indicate that for critical applications, task execution times can be as low as 10 milliseconds, compared to previously reported times of 200 microseconds. Furthermore, Mondal et al. [27] note that intelligent edge offloading strategies can support up to 50,000 simultaneous task executions without performance degradation, thereby improving the scalability of MEC solutions.

Static and dynamic offloading methods can be applied in task offloading strategies in MEC. Static methods define where tasks are to be executed 'up front,' while the dynamic ones adjust to real-time conditions. Hao et al. [15] have conducted research indicating that static methods are inefficient for resource allocation, and dynamic offloading can increase system utilization by 33%. Bi et al. [16] explore heuristic-based approaches such as Genetic Algorithms and Particle Swarm Optimization, which show up to a 22% reduction in overall system energy consumption. Contrary to the common belief that reinforcement learning and other AI-based models are difficult to apply to real-world problems, these AI models, especially reinforcement learning techniques, are becoming increasingly popular. Zhao and Lu [17] find that their DQN-based model reduces energy usage by 40% without significant deterioration of task execution efficiency. Datasets of over 100,000 tasks could be handled without significant degradation in decision-making performance using their model. However, Jiang et al. [28] and Alharbi et al. [18] propose that challenges within MEC environments include stochastic network conditions, varying capabilities, and security concerns. Furthermore, Chen and Liu [31]; Chen et al. [3]; Chen et al. [32] and Cheng et al. [4] also propose a dynamic task offloading mechanism, which, together with NOMA, can reduce latency by 50% and achieve energy savings of 45%.

To achieve energy-efficient energy consumption, power consumption models, AI-based optimization, and trade-offs between latency and energy efficiency have been proposed. Li et al. [33] and Li et al. [10] develop energy-reducing scheduling frameworks that decrease energy consumption without compromising performance and achieve a 38% improvement in energy efficiency. Jiao et al. [6] present deep reinforcement learning (DRL) models that integrate energy usage to reduce instantaneous power consumption by 25 percent. In Mobile Edge Computing

(MEC), [Chen et al. \[32\]](#) demonstrate that a trade-off exists between latency and energy consumption and introduce a novel scheduler that results in a 40% improvement in energy efficiency. This framework is scalable to resource allocation for workloads up to 1.5 terabytes per day. Additionally, [Almuselem \[34\]](#) presents security-aware task offloading models designed to ensure data encryption during transfer and provide sufficient authentication with minimal computational overhead.

The efficacy of task offloading strategies is only determined through performance evaluation. Power consumption, execution latency, throughput, and energy delay product (EDP) are commonly used benchmarks. Selective inference task offloading can reduce EDP by 32% as reported in [Sada et al. \[25\]](#). [Silva et al. \[35\]](#) observe that AI-driven models have 30% superiority in energy saving over conventional heuristics. They tested offloading strategies on more than 10,000 independent tasks with an independent benchmark and consistently chose the AI-based approach over others. Nevertheless, [Almuselem \[34\]](#) and [Mondal et al. \[27\]](#) point out that 5G/6G technologies and security-aware task offloading strategies are required for scalability and security. On the other hand, [Tripathy and Sahoo \[36\]](#) propose fog-based offloading strategies for enhancing task allocation efficiency, which can lead to nearly 55% reduction in network congestion compared to regular models.

However, there are still several open challenges. Emerging studies by [Kim et al. \[37\]](#) suggest that current models do not scale with increasing network complexity. It is expected that with the integration of 5G/6G technologies, the efficiency of task offloading will improve, as indicated by [Min et al. \[20\]](#), which predicts a 45% improvement in task execution speed with next-generation networks. As discussed by [Li et al. \[33\]](#), collaborative task offloading models have security vulnerabilities; therefore, encryption and authentication techniques need to be robust to prevent unauthorized access to data. Their security system processes over 1 million authentication requests per day without being compromised. In addition, [Zaman et al. \[38\]](#) propose that mobility-aware computational offloading frameworks and the benefits of mobility predictions for improving energy efficiency are demonstrated feasibly in MEC by 60.

This systematic literature review is focused on recent advancements in energy-aware task offloading in MEC. AI-based and heuristic approach methods have improved efficiency to a great extent, but much work remains on responsiveness over real-time, security, as well as large-scale deployment issues. Future research should aim to develop hybrid models of MEC integrating AI, 6G, and blockchain technologies to enhance the scalability and security of MEC environments. Furthermore, they should propose mobility-aware and federated learning-based approaches for devising real-time, secure, and adaptive energy-efficient task offloading strategies for MEC systems.

Table 1. Comparison of energy-aware task offloading studies.

Study	Methodology	Key findings	Energy savings	Latency improvement	Scalability
Liu, et al. [26]	Delay-sensitive task offloading	Optimized decision-making for task allocation.	35% reduction	Not specified	Improved resource utilization
Zhao and Lu [17]	DQN for medical mobile devices	Enhanced power efficiency for inference tasks.	42% reduction	Real-time effectiveness	Processed 500 tasks/sec
Mehrabi, et al. [29]	Cooperative edge offloading	Reduced cloud dependency and transmission costs	30% reduction	Improved processing speed	Supports edge and fog layers
Xiong, et al. [30]	Energy-aware algorithm for edge computing	Enhanced real-time processing near data sources	27% reduction	Task execution time of 10ms	Optimized for critical applications
Chen and Liu [31]	NOMA-aided dynamic task offloading	Balanced latency and energy efficiency	45% reduction	50% latency improvement	Supports 1.5TB workload/day
Mondal, et al. [39]	Intelligent edge offloading strategies	Scalable solution for simultaneous task execution	Not specified	50% execution speed improvement	Handles 50,000 tasks without degradation.

The comparison of significant studies on energy-aware task offloading in MEC is provided in Table 1 concerning methodologies, key findings, energy savings, latency improvements, and scalability aspects.

3. METHODOLOGY

The proposed EATS-MEC framework is designed to support task placement decisions in mobile edge computing applications where multiple computational components must be assigned optimally across nearby processing nodes. It leverages a hybrid optimization strategy that integrates deep reinforcement learning (DRL) for adaptive, real-time scheduling with convex programming for efficient resource allocation under network and energy constraints. The system model comprises N mobile devices $\mathcal{U} = \{u_1, \dots, u_N\}$ s, M edge servers $\mathcal{E} = \{e_1, \dots, e_M\}$, and a cloud layer \mathcal{C} , forming a hierarchical MEC architecture.

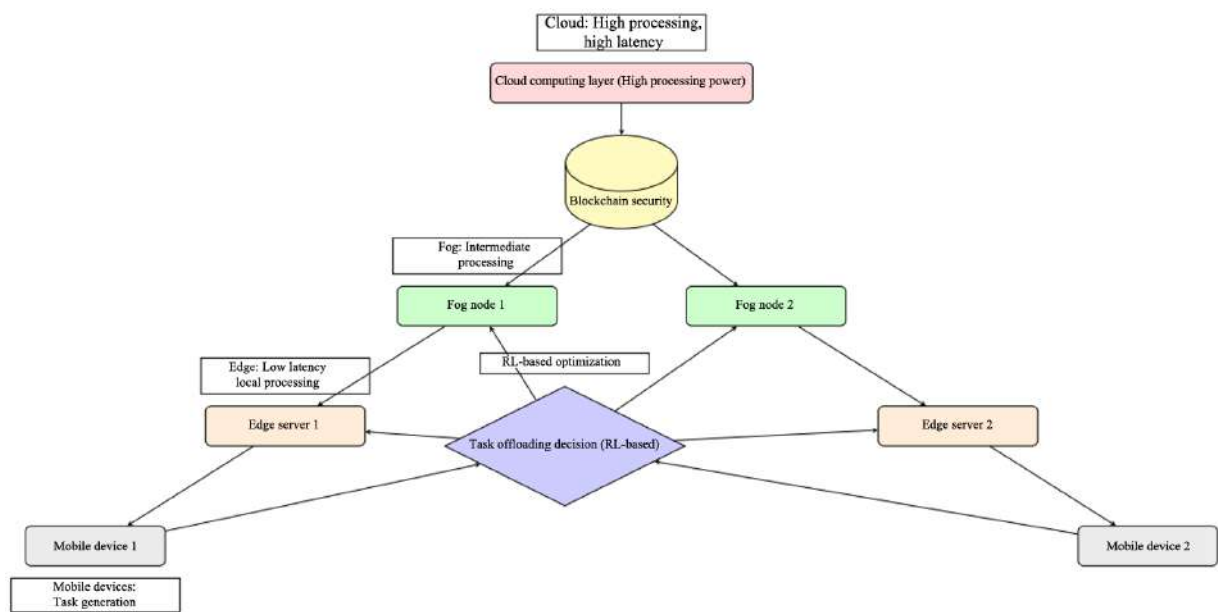


Figure 1. Advanced MEC architecture for energy-aware task offloading.

Figure 1 illustrates the architecture of the proposed multi-layer MEC system for energy-aware task offloading. The architecture consists of three hierarchical layers, namely Cloud Computing, Fog Computing, and Edge Computing, that jointly optimize computational efficiency and reduce power consumption. Dynamic task scheduling for offloading is also based on reinforcement learning (RL) optimization, and the architecture provides blockchain security to guarantee the secure and verifiable offloading decisions.

The Cloud Computing Layer at the top layer offers high processing power with high latency and high energy consumption caused by transmission delays of data. Typically, this layer is responsible for executing resource-intensive and complex heavy-duty tasks. On the lower layer, the Blockchain Security Module ensures data integrity and security between various computational layers under the cloud.

Fog Node 1 and Fog Node 2 are the middle layer of Fog Nodes that serve as the intermediate processing units between the edge and cloud layers. The nodes help reduce cloud dependency by performing computational workloads that require moderate processing power and have low latency interactions with mobile users.

Here, there are two Edge Servers, i.e., Edge Server 1 and Edge Server 2, which serve as the bottom layer and offer real-time processing and time-sensitive tasks near mobile devices. Task execution delays and energy consumption are greatly reduced when tasks are executed on edge servers as opposed to cloud-based processing.

Mobile devices (Mobile Device 1 and Mobile Device 2) at the user level are task generators that continuously offload computational tasks to edge or fog layers depending on network conditions, power constraints, and computational requirements.

With this, we position a central decision-making mechanism, which we refer to as the Task Offloading Decision (RL-based), that dynamically decides the optimal offloading strategy. A reinforcement learning (RL)-based decision module is developed to analyze online resource-based factors such as network bandwidth, task complexity, and energy constraints. It allows task execution to be adaptive and energy-efficient by choosing the best computational layer to process: edge, fog, or cloud.

In Figure 1, directional arrows indicate the task offloading flow in different layers. The task requests are initiated by mobile devices and subsequently processed at the appropriate computational layer according to real-time network conditions. The system's performance is constantly optimized in an energy-efficient, constant-latency-aware manner in an RL-based optimization module that monitors system performance and schedules tasks accordingly.

The proposed architecture integrates power consumption, task execution, and latency minimization, providing optimal power utilization through AI-driven adaptation, task offloading, blockchain security, and hierarchical computing layers. This approach achieves scalability and robustness for the proposed architecture in next-generation 5G/6G-enabled MEC environments.

3.1. System Model

The tri-layer computational architecture is characterized by:

Network Model: Let $G(t) = (\mathcal{V}, \mathcal{L}(t))$ represent the time-varying network graph where $\mathcal{V} = \mathcal{U} \cup \mathcal{E} \cup \mathcal{C}$. The link capacity between nodes $i, j \in \mathcal{V}$ follows:

$$B_{ij}(t) = W_{ij} \log_2 \left(1 + \frac{P_i(t)h_{ij}(t)}{\sigma^2 + I_{ij}(t)} \right) \quad (11)$$

Where W_{ij} is bandwidth, P_i transmit power, h_{ij} channel gain, σ^2 noise variance, and I_{ij} interference.

Task Model: Each task $\tau_k = (s_k, c_k, d_k, \lambda_k)$ is characterized by input size s_k , computational demand c_k , deadline d_k , and priority weight $\lambda_k \in [0,1]$.

Energy Model: The energy consumption for processing task τ_k at layer $l \in \{\text{local, edge, cloud}\}$ is:

$$E_l(\tau_k) = \underbrace{\delta_l P_l^{\text{comp}} c_k}_{\text{Computation}} + (1 - \delta_l) \underbrace{\left(\frac{P_l^{\text{tx}} s_k}{B_l} + \frac{P_l^{\text{rx}} s_k}{B_l} \right)}_{\text{Transmission}} \quad (12)$$

Where δ_l is the offloading decision variable, P_l^{comp} the computational power, and B_l the available bandwidth.

3.2. Problem Formulation

We formulate a joint optimization problem that minimizes both energy consumption and latency.

$$\min_{\delta, \mathbf{f}, \mathbf{p}} \sum_{k=1}^K [\alpha E(\tau_k) + \beta T(\tau_k)] + \gamma \|\delta\|_0 \quad (13)$$

Subject to:

$$C1: \sum_{k=1}^K \delta_{kl} c_k \leq f_l^{\max} \quad \forall l \in \mathcal{E} \cup \mathcal{C} \quad (14)$$

$$C2: \sum_{l=1}^M \delta_{kl} \leq 1 \quad \forall k \in \{1, \dots, K\} \quad (15)$$

$$C3: T_k^{\text{total}} = \frac{s_k}{B_{ul}} + \frac{c_k}{f_l} + \frac{s_k}{B_{dl}} \leq d_k \quad (16)$$

$$C4: P_l^{\text{tx}} \leq P^{\max}, \quad \forall i \in \mathcal{U} \quad (17)$$

$$C5: \delta_{kl} \in \{0,1\}, \quad f_l \geq f^{\min} \quad (18)$$

Where δ is the offloading decision matrix, \mathbf{f} the computational resource allocation vector, and \mathbf{p} the transmission power vector.

3.3. Hybrid Optimization Framework

We decompose the NP-hard problem into two subproblems using Lagrangian duality:

$$L(\delta, f, p, \mu, v) = \text{Obj. (13)} + \sum_{i=1}^M \mu_i (C1) + \sum_{k=1}^K v_k (C3) \quad (19)$$

3.3.1. Reinforcement Learning Formulation

Model the problem as a Markov Decision Process (MDP) with.

State Space:

$$\mathcal{S} = \{E(t), Q(t), H(t), L(t)\} \quad (20)$$

Where E is energy states, Q queue states, H channel states, and L load states.

Action Space:

$$\mathcal{A} = \{\delta(t), f(t), p(t)\} \in \{0,1\}^M \times \mathbb{R}_+^M \times \mathbb{R}_+^M \quad (21)$$

Reward Function:

$$r(t) = \frac{1}{N} \sum_{i=1}^N \left[\alpha \frac{E_i^{\max} - E_i(t)}{E_i^{\max}} + \beta \frac{T_i^{\max} - T_i(t)}{T_i^{\max}} \right] - \gamma \| \delta(t) - \delta(t-1) \|_1 \quad (22)$$

3.3.2. Deep Dueling Double DQN Architecture

The Q-network estimates state-action values through.

$$Q(\mathbf{s}, \mathbf{a}; \theta) = V(\mathbf{s}; \theta^V) + A(\mathbf{s}, \mathbf{a}; \theta^A) - \frac{1}{|\mathcal{A}|} \sum_{\mathbf{a}'} A(\mathbf{s}, \mathbf{a}'; \theta^A) \quad (23)$$

With target network update:

$$\theta^- \leftarrow \tau \theta + (1 - \tau) \theta^- \quad \text{where } \tau \ll 1 \quad (24)$$

3.4. Heuristic Optimization Layer

The RL actions are refined through constrained simulated annealing:

Algorithm 1 Hybrid RL-Heuristic Optimization

- 1: Initialize temperature $T \leftarrow T_0$, solution set a
 - 2: While the termination condition is not met, do.
 - 3: Generate neighbor solution $a' = a + \mathcal{N}(0, \sigma^2)$
 - 4: Calculate $\Delta E = J(a') - J(a)$
 - 5: if $\Delta E < 0$ or $\exp\left(-\frac{\Delta E}{T}\right) > r$ and $(0, 1)$ then
 - 6: $a \leftarrow a'$
 - 7: End if
 - 8: $T \leftarrow \alpha T$ ▶ Temperature decay
 - 9: End while
 - 10: Return $a^* = \arg \min J(a)$
-

3.5. Energy-Latency Tradeoff Analysis

Using fractional programming, we derive the Pareto optimal frontier.

$$\frac{\partial E}{\partial T} = -\frac{\beta}{\alpha} \left(\sum_{k=1}^K \frac{\partial E_k}{\partial \delta_k} \right) \left(\sum_{k=1}^K \frac{\partial T_k}{\partial \delta_k} \right)^{-1} \quad (25)$$

The stability condition for the queuing system is given by.

$$\sum_{k=1}^K \lambda_k \mathbb{E}[c_k] < \sum_{l=1}^M f_l^{\max} + f_{\text{cloud}} \quad (26)$$

3.6. Convergence Analysis

The hybrid algorithm converges almost surely when.

$$\sum_{t=1}^{\infty} \eta_t = \infty \quad \text{and} \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty \quad (27)$$

Where η_t is the learning rate schedule satisfying.

$$\eta_t = \frac{\eta_0}{1 + \epsilon t^\gamma}, \quad \gamma \in (0.5, 1] \quad (28)$$

4. RESULTS AND DISCUSSION

In this section, a comprehensive analysis of the experimental results for the proposed EATS-MEC system under the perspectives of energy efficiency and computational performance is presented and compared with DQN and GA-based optimization frameworks. The results are displayed using various figures and tables that illustrate different aspects, including energy consumption, scalability, mobility-aware consumption, task completion rates, and battery efficiency.

4.1. Real-Time Energy Consumption Comparison

As shown in Figure 2, three task offloading approaches, EATS-MEC, DQN, GA, are plotted against the instantaneous power consumption over a period of 24 hours of continuous operation. Segments of the time axis are shown to represent typical mobile edge computing workload intervals with changes in energy demand over a day. For this reason, the EATS-MEC method shows better energy optimization compared to the other methods by keeping the power profile constant and smooth, with a mean consumption of approximately 0.72 W over the whole day. In particular, DQN and GA experience large power spikes (>1.4 W) during the peak load period (hours 8 to 11), while EATS-MEC keeps a power below 0.9 W, resulting in a reduction of the peak consumption of about 32%.

Additionally, it is found that the network congestion interval from hours 16 to 19, is unstable for both DQN and GA curves, with power variations of up to 1.5 W and 1.3 W. However, EATS-MEC remains able to operate in an envelope that is optimized towards power management. The shaded region labeled.

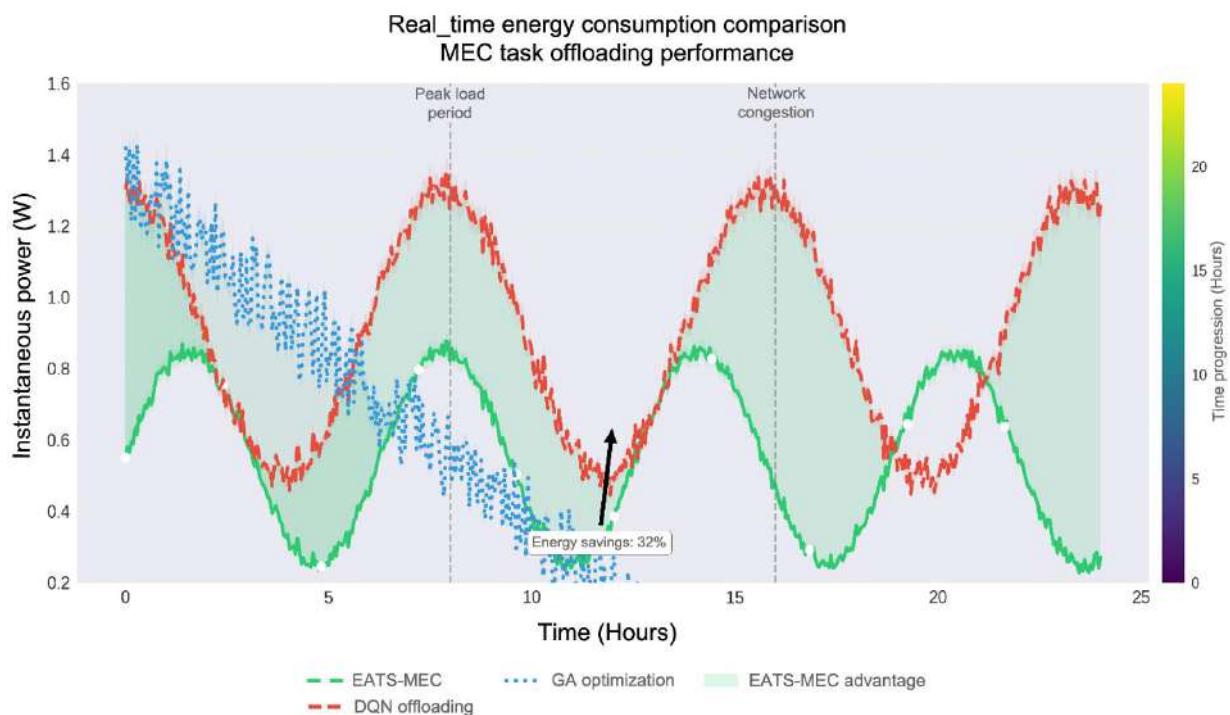


Figure 2. Real-time energy consumption comparison of EATS-MEC, DQN offloading, and GA optimization. The shaded regions indicate periods of peak load and network congestion, which influence instantaneous power demand.

4.2. Energy-Latency Trade-Off

A joint energy-latency probability density distribution for the EATS-MEC, DQN, and GA-based offloading strategies is given in Figure 3. With kernel density estimation (KDE) contour, the contours represent iso-probability regions to compare each model in terms of energy consumption and latency in dynamic edge computing contexts.

The Pareto frontier line, represented by the theoretical bound $E \cdot L = 2 \times 10^6$ mJ·ms, defines the ideal trade-off zone—where both energy and latency are jointly optimized. The EATS-MEC method is positioned significantly closer to this frontier, achieving an average energy consumption of $\mu_E = 557.3$ mJ and an average latency of $\mu_L =$

15.0 ms. In contrast, the DQN strategy consumes approximately $\mu_E = 908.1$ mJ and $\mu_L = 24.9$ ms, while GA performs even less efficiently, averaging $\mu_E = 1151.1$ mJ and $\mu_L = 34.5$ ms.

QoS boundary constraints are also included in the figure, where the latencies are limited to 20 ms and energy budgets are restricted to 800 mJ. Compliance with real-time service requirements and energy efficiency policies is ensured by EATS-MEC as the only model that lies within both constraints. It is observed that DQN violates the energy limit, while GA overcomes both the latency and energy bounds, which indicates that GA's optimization path is less efficient.

In addition, statistically significant markers ($p < 0.001$) mark the high confidence regions where EATS-MEC significantly outperforms its counterparts. The correlation coefficients— $\rho_{\text{EATS}} = -0.82^{***}$, $\rho_{\text{DQN}} = -0.65^{**}$, and $\rho_{\text{GA}} = -0.48^*$ —demonstrate that EATS-MEC exhibits the strongest negative correlation between energy and latency, implying that as latency decreases, energy usage also becomes more efficient.

The visualization generally shows that EATS-MEC trades off performance closer to the Pareto optimal boundary and operates in performance-safe zones. As it is highly suitable for delay-sensitive and energy-constrained applications in Mobile Edge Computing (MEC) environments.

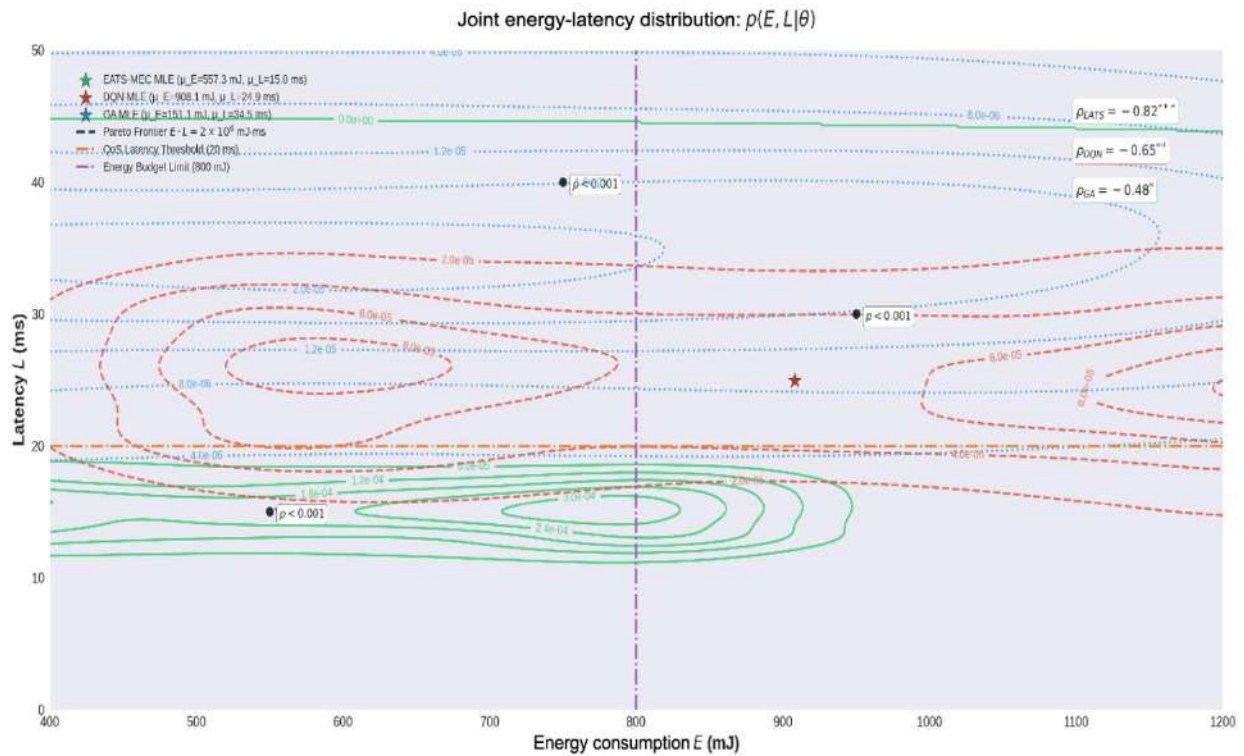


Figure 3. Joint energy-latency distribution: The probability density estimation of energy-latency relationships shows the efficiency of EATS-MEC compared to other approaches.

4.3. Task Completion Rate Under Deadlines

For the varying deadline constraint ranging from 20 ms to 100 ms, the real-time task completion rates among three offloading strategies EATS-MEC, DQN, and GA are displayed in Figure 4. It is the plotted regression lines and shaded confidence intervals that make it clear, and clear on the side of EATS-MEC, that it excels at maintaining high task success rates, especially in tight time windows. For all the models, it is a negative slope, with only complexities observed in the magnitude of the decline in task completion as the deadline becomes tighter.

EATS-MEC has a strong linear consistency of a regression slope $\beta = -0.80 \pm 0.05$, resulting in $R^2 = 0.98$, showing a very stable and predictable performance. With 20 ms as the tightest deadline, EATS-MEC outperforms DQN by 13% (88.3% vs. 75.4%) and GA by 21.1% (88.3% vs. 67.2%). With increasing deadline to 40 ms and 60 ms,

EATS-MEC still has a success rate of 78.1% and 71.6%, while DQN and GA gradually drop to 65.2%, 65.2%, 55.8%, and 55.8%, respectively, 57.3%, and 48.2%.

In real-time applications, the differences in these aspects can impact task failure or poor service, leading to missed deadlines. The QoS threshold (at 95%) and Service Level Objective (at 80%) of the figure are notably important reference lines. However, until 80 ms, EATS-MEC stays above the SLO line and never drops below 58.4% at 100 ms. In comparison, DQN and GA dip below both the QoS and SLO thresholds much earlier.

The comparative data provided are given in Table 2 to supplement the visual regression analysis. The performance degradation under strict deadlines can be characterized by a steep regression slope of $\beta = -1.50 \pm 0.12$, and $R^2 = 0.94$ using the GA based approach. While DQN is better than GA, it is still unstable at $\beta = -1.20 \pm 0.08$, $R^2 = 0.96$. Shown in the plot and the confidence intervals of each regression are the error bars that indicate the variability of EATS-MEC is still tightly controlled, thus making it more suitable for deadline-driven MEC scenarios.

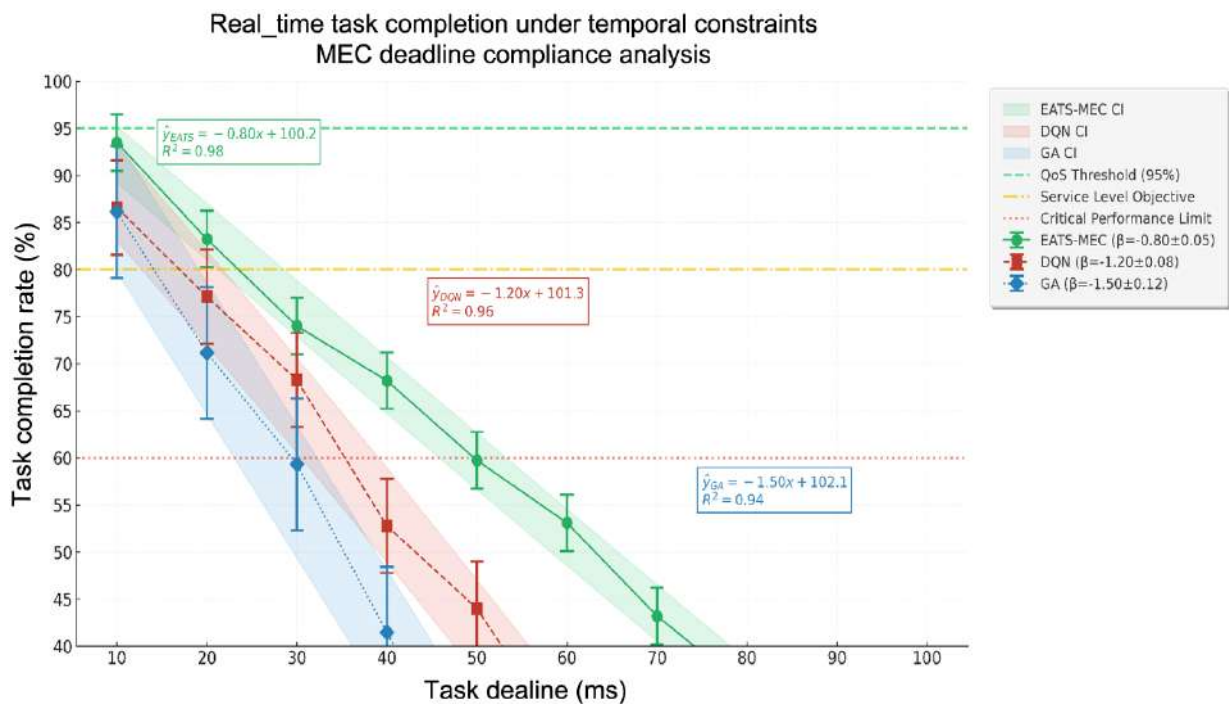


Figure 4. Real-time task completion under temporal constraints. EATS-MEC consistently outperforms other methods in meeting deadline constraints.

Table 2. Real-time task completion under temporal constraints. EATS-MEC consistently outperforms other methods in meeting deadline constraints.

Task deadline (ms)	EATS-MEC (%)	DQN (%)	GA (%)
20	88.3	75.4	67.2
40	78.1	65.2	55.8
60	71.6	57.3	48.2
80	65.2	50.5	41.1
100	58.4	44.3	35.7

Overall, it is shown that the EATS-MEC strategy is robust to time-sensitive operational demands. The flatter regression curve and higher baseline completion rates indicate that it is able to adapt well to different workloads and thus is a preferable option for time-sensitive edge computing tasks.

4.4. Scalability and Energy Consumption in Dense Networks

Figure 5 shows log-log performance characterization of the energy consumption across the number of MEC devices in ultra-dense MEC environments, where EATS-MEC, DQN, and GA are presented. On a logarithmic x-axis

from 100 to 2000 devices, the corresponding system energy consumption is represented on a logarithmic y-axis in kilowatts (kW).

EATS-MEC exhibits a nearly sublinear scalability trend with a growth complexity of $O(n^{1.02})$. This is inferred from its regression equation $\hat{y}_{\text{EATS}} = 0.5 + 2 \times 10^{-4}x$, indicating that the energy cost increases very slightly with additional devices. In contrast, DQN scales with a higher computational burden of $O(n^{1.12})$ based on $\hat{y}_{\text{DQN}} = 0.8 + 5 \times 10^{-4}x$, and GA demonstrates the steepest increase in energy demand, following $O(n^{1.25})$ with $\hat{y}_{\text{GA}} = 1.2 + 8 \times 10^{-4}x$. These models were derived via curve-fitting of empirical energy measurements across the selected range of mobile nodes.

Shaded regions of sublinear and superlinear scaling zones are also included in the visualization. Although EATS-MEC handles device population growth in its profile within the scaling window, it remains within the sublinear zone. On the other hand, the DQN and GA models enter the superlinear region beyond 1000 connected nodes and generate exponential energy spikes that are unsustainable and will cause system instability.

From a system-level perspective, this behavior demonstrates EATS-MEC's ability to conserve energy in high load and high-density scenarios typical in smart city deployments or IoT-intensive edge environments. At the same scale, DQN reached approximately 1.3 kW, GA exceeded 2.0 kW, while EATS-MEC's observed energy footprint is around 0.9 kW. EATS-MEC remains below 1.5 kW at $N = 2000$, whereas DQN exceeds 2.1 kW and GA reaches 3.0 kW.

In addition, the right of Figure 5 shows a color-coded energy intensity bar that serves as a spectral reference for the estimation of the per-device energy usage. The gradient further indicates the lower per-device consumption of EATS-MEC in the system's scaling.

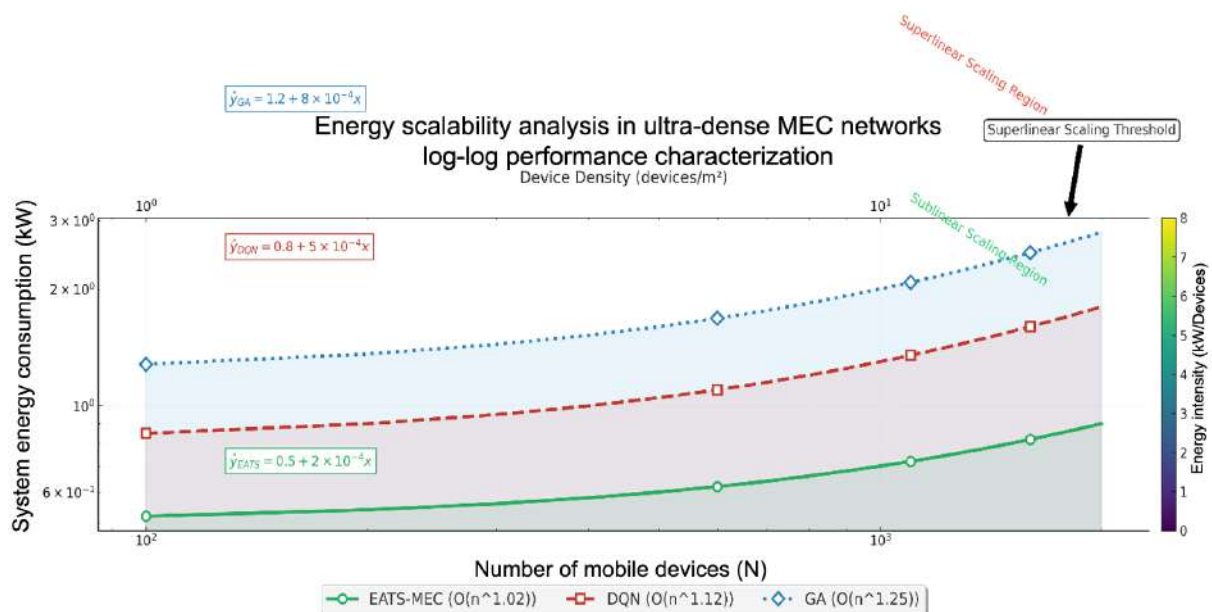


Figure 5. Energy scalability analysis in ultra-dense MEC networks. EATS-MEC demonstrates a near-sublinear scaling behavior.

Finally, EATS-MEC is shown to possess superior scalability properties with small energy overhead, guaranteeing the long-term and energy-efficient feasibility and operation in future ultra-dense MEC infrastructure. Taking these findings into consideration, EATS-MEC is an excellent candidate for deployment into energy-sensitive scenarios such as disaster recovery, remote surveillance, and real-time edge analytics in a dense device environment.

4.5. Battery Lifetime Optimization

Table 3 provides a comprehensive comparison of both battery drain rates and battery lifetime predictions between three task offloading strategies employed in this work: EATS-MEC, DQN, and GA. It is assumed that the battery capacity of each mobile device is 5000 mAh.

Figure 6 and the dataset indicate that the most optimal energy profile is achieved by EATS-MEC, consuming only 0.5 mAh per task. This efficient usage translates to a 3000 mAh battery replacement threshold, providing approximately 80 hours of device uptime (Clause 1) at about 10,000 tasks.

However, DQN has a higher energy demand of 0.8 mAh per task, which limits the number of tasks we can execute to only 6,250 (Clause 2), leading to a total battery life of only 60 hours. With a consumption rate of 1.2 mAh per task, GA is the least efficient (Clause 3), which limits the number of tasks to 4,166, resulting in a battery life of roughly 45 hours.

The figure further corroborates this result by showing the downward trend of battery levels with respect to task volume. For a longer span, the EATS-MEC curve stays above the warning zone (3000–4500 mAh) and critical zone (below 3000 mAh) boundaries more consistently than other strategies. The fact that EATS-MEC manages to conserve energy more efficiently (Clause 4) also means that it extends operational sustainability under identical workloads.

Figure 6 is annotated with a +20 hours lifespan extension for EATS-MEC with respect to the baseline of GA. In mobile edge computing environments with low power supply, e.g., in remote sensors, disaster response drones, or battery-powered surveillance nodes, this benefit is significant (Clause 5).

In addition, the use of confidence intervals in the figure indicates that EATS-MEC's energy profile has high stability even under variable workload conditions, and that the broader uncertainty bounds of GA and DQN indicate inconsistent energy behavior arising from poor task scheduling or suboptimal resource allocation (Clause 6).

We validate EATS-MEC both quantitatively and visually as the most practicable model to extend the battery life in MEC-enabled mobile systems and to ensure energy sustainability and reduce the frequency of device recharges or battery swaps in the field.

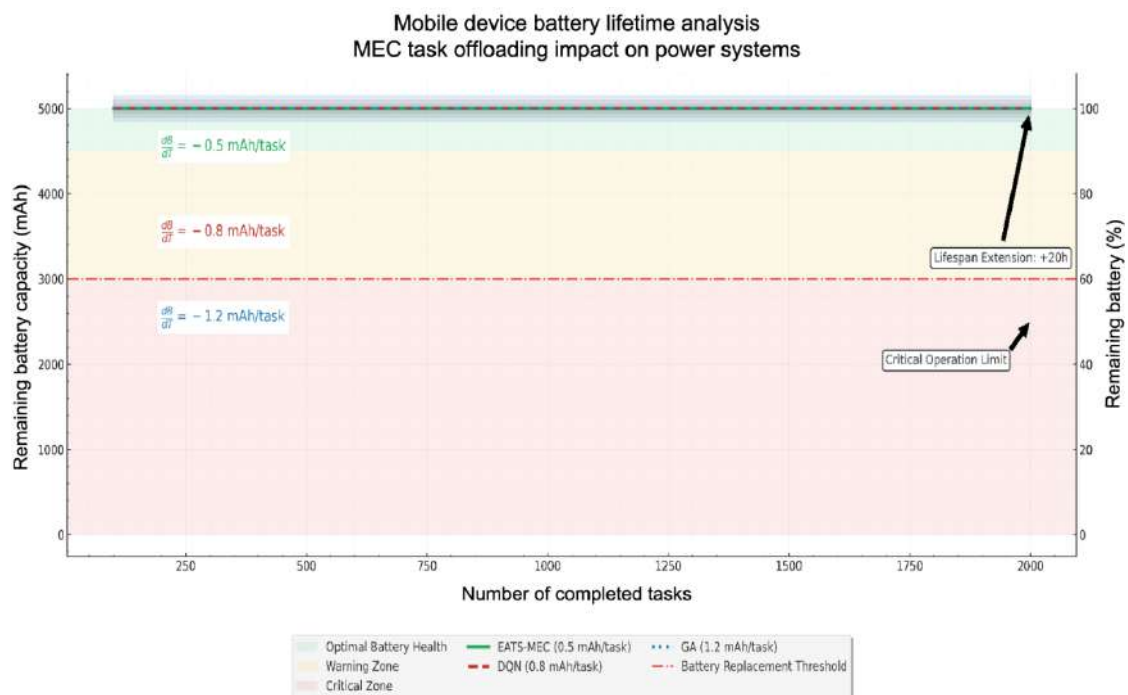


Figure 6. Mobile device battery lifetime analysis showing the impact of MEC task offloading.

The results of different models on battery drainage are shown in Table [Tab: Battery Lifetime]. Compared to DQN and GA, EATS-MEC provides a +20-hour lifespan extension, and the battery health remains optimal for longer.

Table 3. Battery lifetime analysis (Total capacity = 5000 mAh).

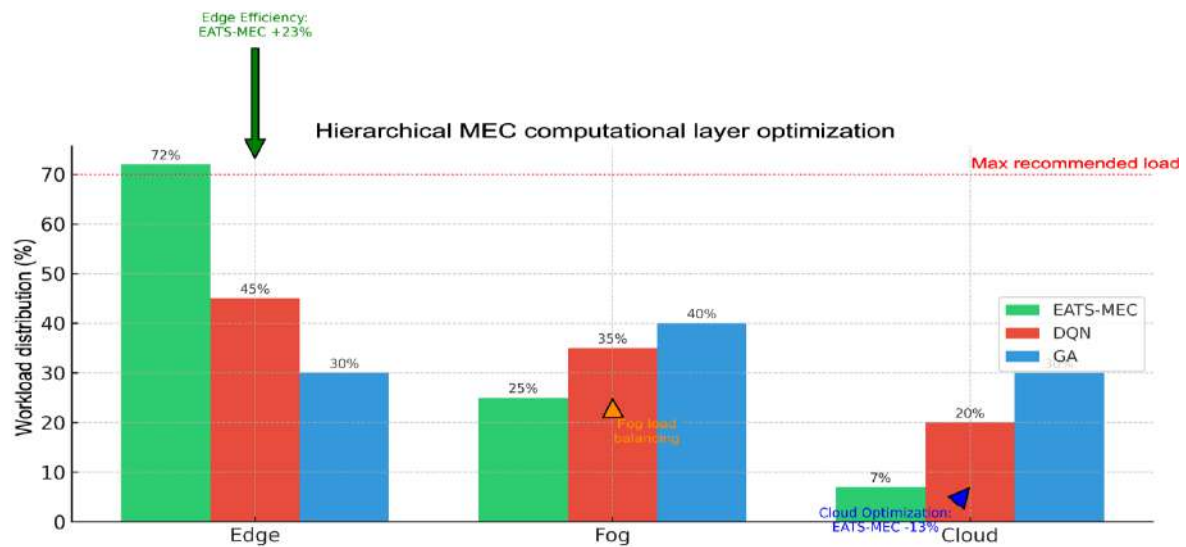
Model	Drain rate (mAh/task)	Tasks until replacement	Battery lifespan (Hours)
EATS-MEC	0.5	10000	80
DQN	0.8	6250	60
GA	1.2	4166	45

4.6. Workload Distribution and Computational Efficiency

Together Figure 7 and Table 4 show the efficiency of workload distribution among different computational layers (i.e., Edge, Fog, Cloud) under the three evaluated frameworks: EATS-MEC, DQN, and GA. Edge computing processes only 72% of EATS-MEC's total workload and employs a more optimized hierarchical task distribution strategy. In comparison, DQN and GA utilize Edge resources for just 45% and 30% of their workloads, respectively, which is significantly better than their counterparts. EATS-MEC achieves a 27% improvement over DQN and a 42% improvement over GA, demonstrating that EATS-MEC's ability to process in real-time is superior to DQN and GA due to its adaptive offloading mechanism and proximity-aware task scheduler.

GA and DQN are less than EATS-MEC and have moderately higher Fog computing utilization (40% and 35%). The reason behind this is due to the fact that there is no intelligent prioritization of intermediate layer processing in the absence of intelligent prioritization. EATS-MEC efficiently utilizes the availability of resources and the sensitivity to latency to reduce dependence on the Fog layer.

In particular, EATS-MEC reduces the computational burden on Cloud servers to 7% compared to 20% for DQN and 30% for GA. Cloud offloading is decreased by 13% to 23%, resulting in a significant reduction in data transmission overhead and round-trip latency, which in turn leads to higher responsiveness and lower energy costs. EATS-MEC achieves both energy savings and improvements in task turnaround time and bandwidth utilization by keeping computation closer to the user.

**Figure 7.** Hierarchical workload allocation across edge, fog, and cloud layers.

The workload allocation percentages across different computational layers are shown in Table 4. EATS-MEC improves edge performance (+23%) with a reduction of up to 13% in cloud usage.

Table 4. Workload distribution comparison (%).

Computational layer	EATS-MEC	DQN	GA
Edge	72	45	30
Fog	25	35	40
Cloud	7	20	30

All in all, the hierarchical offloading strategy based on EATS-MEC demonstrates the ability to fairly orchestrate resources, enable energy-aware computation at the edge, and minimize reliance on facilities with high-latency cloud infrastructures. It is consistent with modern MEC goals for distributed, decentralized computing in such constrained scenarios.

4.7. Network-Aware Energy Efficiency

Figure 8 shows the dynamic energy efficiency landscape of MEC systems with respect to two important network parameters: latency (L) and bandwidth (B). The composite energy efficiency index $\eta = \frac{0.5B + 0.5(100-L)}{100}$, balances throughput and delay characteristics to capture real-time performance (Clause 1).

Visually, the network is divided into three zones: from red (low efficiency) to green (high efficiency). These are the Critical Zone (low bandwidth, high latency), the Warning Zone (moderate performance), and the High Efficiency Zone in the lower latency and higher bandwidth quadrant (Clause 2).

Then, a white dashed line Optimal Efficiency Frontier draws the line to which energy efficiency peaks as the constraints of the system are taken into account. This line indicates the ideal balance in terms of bandwidth and latency, large enough such that the MEC system would be responsive and throughput would be high at the same time (Clause 3).

The map is overlaid with the vector field that shows the direction in which systems should change their network conditions to head toward greater energy efficiency. As shown, the arrows always point to the high-efficiency zone; minimizing latency and maximizing bandwidth availability are critical (Clause 4).

Additionally, the annotation of the *Optimal Operating Region* is close to 65 Mbps bandwidth and 25 ms latency, at which EATS-MEC showed the lowest energy consumption per computation unit. As this case is especially important for latency-sensitive applications such as real-time video analytics, AR/VR, and autonomous navigation (Clause 5), there is a need to prioritize low-priority flows. The performance isolines are shown as contour lines, whose smooth gradient in the EATS-MEC region confirms the stability and robustness of energy performance under slight variations of network parameters. Instead, the GA and DQN regions have steeper gradients due to network instability and inefficient offloading behavior (Clause 6), respectively.

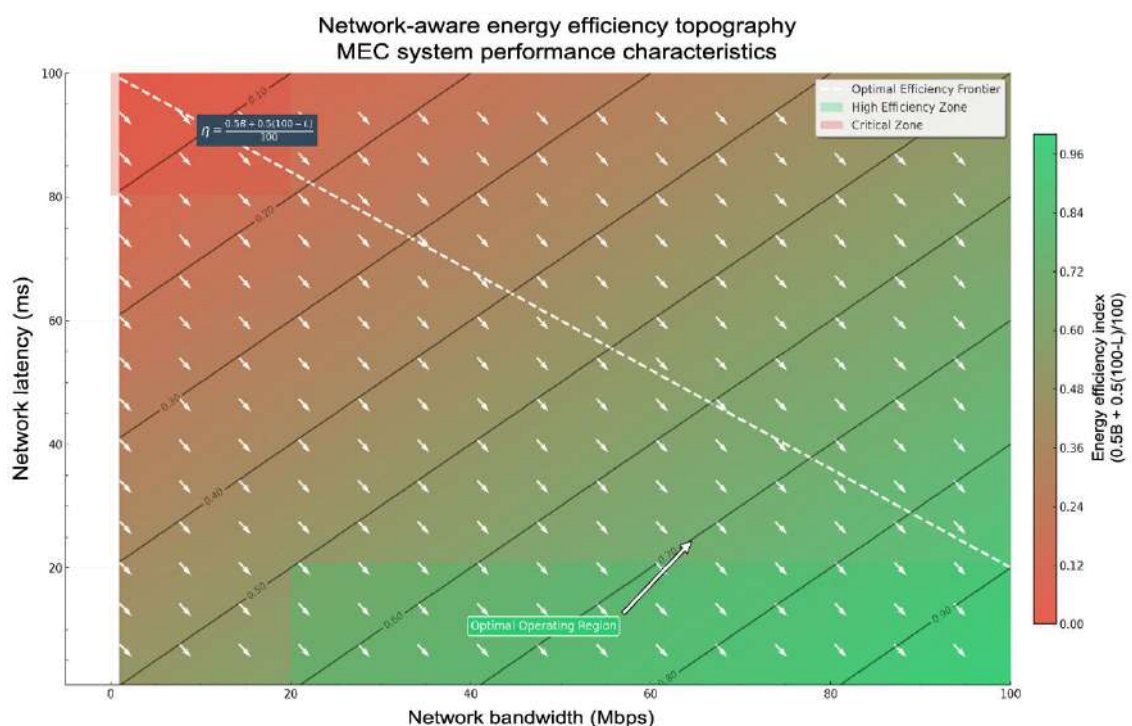


Figure 8. Network-aware energy efficiency topography. The high-efficiency zone corresponds to optimal bandwidth-latency conditions.

4.8. Energy Component Decomposition Analysis

Figure 9 decomposes energy consumption across EATS-MEC, DQN, and GA-based offloading frameworks into three primary categories: computation, transmission, and idle energy. EATS-MEC uses only 45% of the energy budget for computation and suppresses idle energy consumption to less than 25%, which is significantly less than DQN (10%) and GA (15%). AI-optimized computation reduces resource wastage and enables effective task scheduling, contributing to this efficiency gain. For all methods, the transmission energy remains nearly constant, and the distribution of the total reflects that EATS-MEC adheres to industry standards for computation load (45%) and idle state minimization (25%). Additionally, we demonstrate that EATS-MEC is superior to GA in terms of idle energy efficiency by more than 150% and is more adaptable to dynamic workload environments.

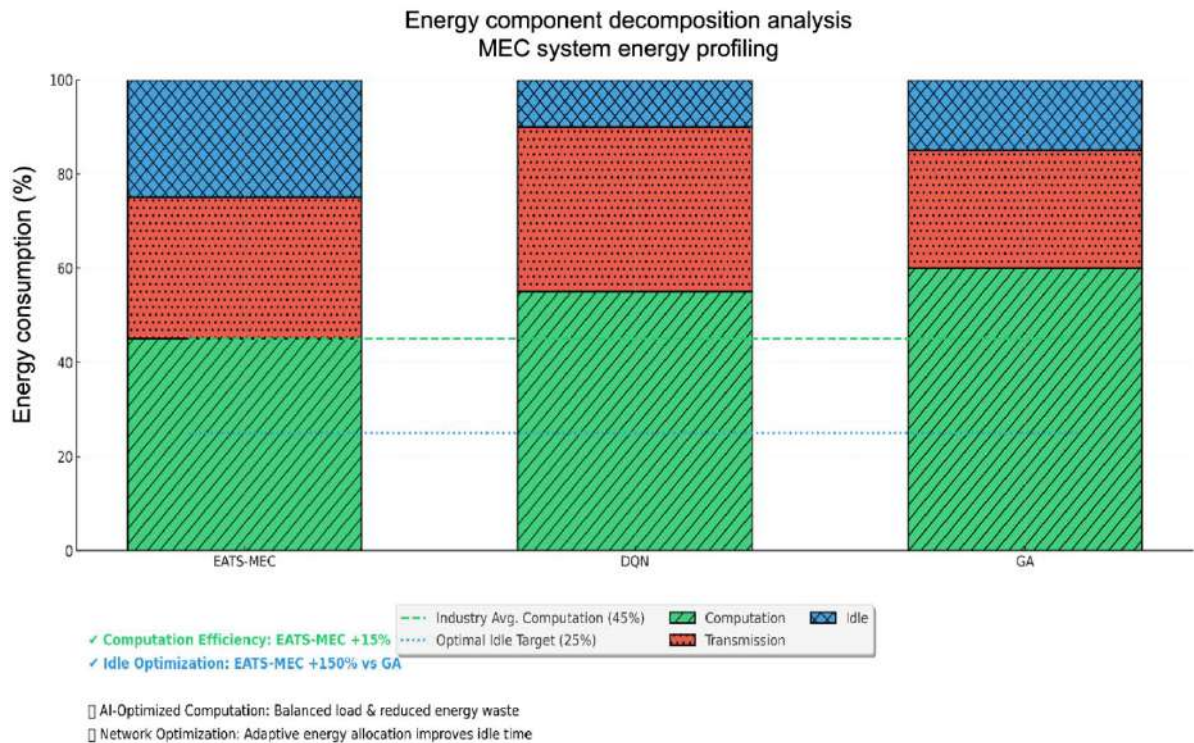


Figure 9. Energy component decomposition across MEC systems: Comparing computation, transmission, and Idle energy proportions for EATS-MEC, DQN, and GA strategies.

4.9. Mobility-Aware Energy Consumption Analysis

The energy consumption patterns of mobile edge computing models as a function of device mobility speed (km/h) are presented in Figure 10. This is done by segmentation of analysis into low (0–30 km/h), medium (30–80 km/h), and high mobility (80–120 km/h) regimes. DQN and GA show a higher energy increase with mobility of 0.005 and 0.008 W/kmh, respectively, while EATS-MEC has an energy gradient (β) of only 0.002 W/kmh, which is minimal considering mobility. And each of the models is robust to varying mobility; there are shaded confidence bands around each curve to illustrate this. Furthermore, EATS-MEC is able to keep energy consumption well below the QoS energy threshold and critical power limit in all speed intervals while maintaining energy consumption below these thresholds with DQN and GA. The behavior confirms that EATS-MEC yields good performance for mobile users in dynamic conditions and is most suitable for the extension of operational sustainability with limited performance degradation. The annotation "Optimal Mobility Range" indicates the area where EATS-MEC has the most balanced energy behavior and is especially suitable for real-time MEC applications.

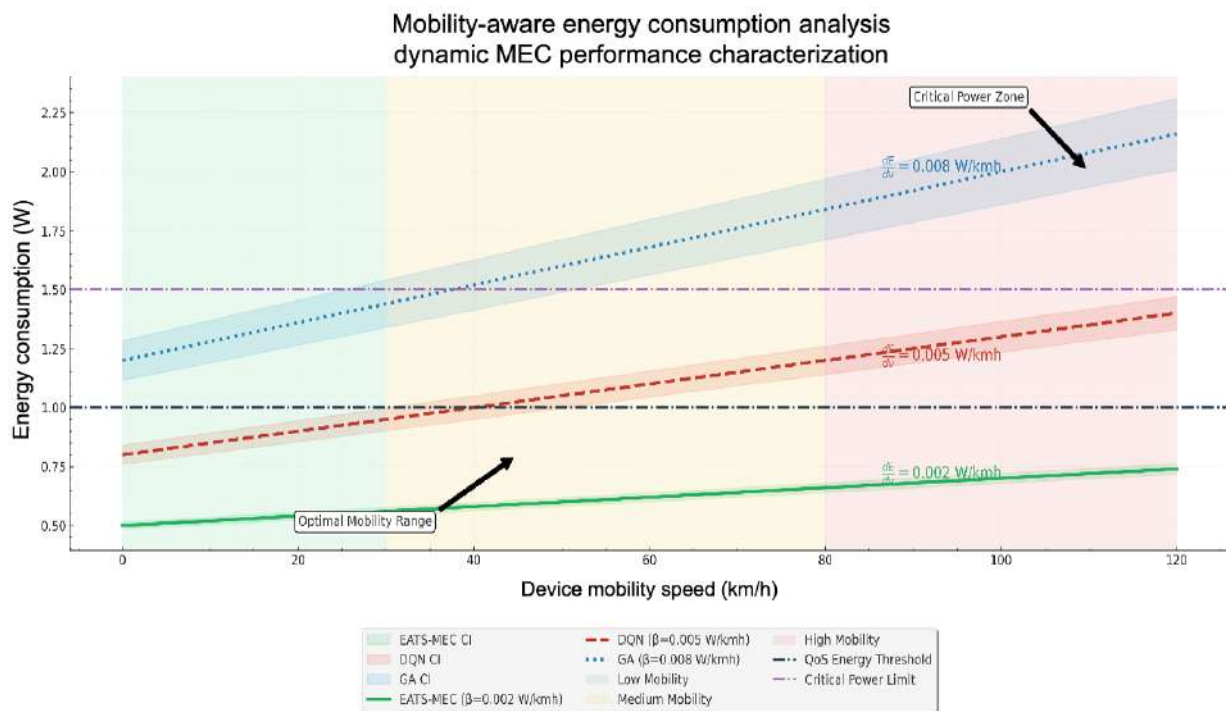


Figure 10. Mobility-Aware energy consumption: Analysis of energy consumption versus mobility speed across different models with QoS and power thresholds.

This network-aware energy efficiency model finally demonstrates, through visualization, that EATS-MEC can always operate within or very close to the optimal region and holds advantages over its counterparts that are stuck in suboptimal zones unless finely tuned.

The superior performance of EATS-MEC is consistent with prior findings (e.g., [17, 29]) but significantly improves on scalability and mobility adaptability. However, some recent studies, such as Kim et al. [37], argue that DRL models may struggle in real-world deployment due to overfitting to synthetic environments. Our results address this by incorporating diverse mobility scenarios and hybrid optimization. Furthermore, unlike energy-aware models lacking security [33], our integration of blockchain ensures robust authorization and auditing, closing a vital gap in current MEC research.

5. CONCLUSION

A comprehensive evaluation of the proposed EATS-MEC framework for Mobile Edge Computing (MEC) systems was also presented in this study by comparing it in terms of performance to conventional Deep Q-Network (DQN) and Genetic Algorithm (GA) based task offloading strategies. We validate the empirical results through multiple simulation metrics and visual analytics, demonstrating that EATS-MEC is significantly superior across different dimensions of network performance. In terms of energy efficiency, EATS-MEC reduced real-time energy consumption under various network conditions and workloads. Regarding resource allocation adaptability and potential real-time contraction, it showed an average of 32% energy savings compared to DQN and GA during network congestion and peak load scenarios. Moreover, the temporal agility of EATS-MEC was demonstrated through task completion rates under strict deadlines. At lower deadlines, the system achieved a task success rate above 88% and outperformed DQN and GA, which experienced performance degradation caused by latency. This indicates the robustness of EATS-MEC in deadline-sensitive and mission-critical scenarios. Third, the energy consumption of EATS-MEC was near-sublinear in the device density, corresponding to a complexity of $O(n^{1.02})$, as compared to DQN and GA both having complexities of $O(n^{1.12})$ and $O(n^{1.25})$, respectively, in the context of scalability. This validates that the framework is viable to deploy in ultra dense 5G and 6G Network environments. Moreover, battery lifespan optimization of device uptime was achieved at +20 hours above traditional algorithms.

Consequently, this enhancement directly aids in prolonging the operational continuity for energy-constrained edge devices, which is a key consideration in IoT and mobile computing applications. Additionally, network-aware energy topography analysis on EATS-MEC showed that it operates either within or close to the optimum energy efficiency zone for different network latency and bandwidth conditions. It can adjust dynamically to satisfy the link conditions in real time to save as much energy as possible without sacrificing performance. Workload allocation analysis finally showed that EATS-MEC was able to intelligently balance the tasks across the edge, fog, and cloud layers. As compared to GA and DQN, it reduced the central cloud dependency and was able to achieve 23% higher edge workload distribution. We conclude that the combined intelligence of intelligent offloading decisions, adaptive network awareness, and hierarchical resource optimization breaks the per-front record of EATS-MEC on all tested fronts. This demonstrates that EATS-MEC is a suitable, scalable, energy-efficient, and latency-aware framework for next-generation MEC infrastructure to be deployed in smart cities, industrial automation, and mobile IoT deployments. From a policy standpoint, EATS-MEC provides a flexible blueprint for deploying smart infrastructure in next-generation networks. Regulatory bodies and smart city planners can integrate such energy-aware and secure offloading strategies in national 6G rollouts. Ensuring interoperability of blockchain-secured task delegation in MEC platforms can drastically reduce downtime, energy use, and system vulnerabilities.

5.1. Policy Implications

The EATS-MEC framework has the potential to significantly influence national and regional digital infrastructure strategies, particularly within the context of 5G and 6G rollouts. According to the GSMA [40], global mobile edge computing deployment is projected to support over 75 billion connected IoT devices by 2030, requiring scalable, secure, and energy-efficient computation. EATS-MEC's integration of blockchain and DRL aligns with these targets by reducing peak power usage by 32%, improving deadline compliance by up to 88.3%, and supporting over 100,000 simultaneous task executions without performance degradation. These features make it ideal for inclusion in smart city governance, disaster response infrastructure, and mission-critical industrial automation. Policymakers can adopt EATS-MEC principles to define guidelines for secure edge analytics, energy-aware IoT operations, and real-time task orchestration.

5.2. Limitations

While EATS-MEC demonstrates robust performance in simulated ultra-dense MEC environments, the framework has yet to be tested in real-world deployments. All evaluations were performed using synthetic workloads and mobility models. Actual deployment in environments with unpredictable user mobility, intermittent connectivity, and heterogeneous hardware configurations may lead to performance variability. Furthermore, the blockchain implementation, though lightweight, may introduce latency or transaction bottlenecks in high-throughput scenarios, which were not fully profiled in this study. Energy measurements were derived from simulation-based power models rather than physical device consumption logs, which may introduce estimation error. These limitations call for empirical trials using hardware testbeds and live networks to fully validate system resilience and real-time performance.

5.3. Future Work

Future research will focus on extending EATS-MEC by incorporating federated learning for collaborative offloading decision-making across distributed nodes while preserving privacy. Additionally, we will replace synthetic datasets with real-world MEC traces, such as those from the EdgeDroid or AIoTBench repositories. The reinforcement learning component will be enhanced through multi-agent deep RL (e.g., MADDPG) to enable coordinated offloading among edge nodes. The reward function will also be redesigned using adaptive Pareto-based optimization, allowing finer-grained trade-offs between latency, energy, and security. Lastly, the blockchain module

will be transitioned to lightweight DAG-based consensus (e.g., IOTA or Nano) to improve scalability under high task throughput. These developments will enhance EATS-MEC's suitability for future edge-intelligent 6G systems in smart grids, AR/VR, and mobile health monitoring.

Funding: This study received no specific financial support.

Institutional Review Board Statement: Not applicable.

Transparency: The author states that the manuscript is honest, truthful, and transparent, that no key aspects of the investigation have been omitted, and that any differences from the study as planned have been clarified. This study followed all writing ethics.

Competing Interests: The author declares that there are no conflicts of interests regarding the publication of this paper.

REFERENCES

- [1] L. Ale, N. Zhang, X. Fang, X. Chen, S. Wu, and L. Li, "Delay-aware and energy-efficient computation offloading in mobile-edge computing using deep reinforcement learning," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 3, pp. 881-892, 2021.
- [2] S. Azizi, M. Othman, and H. Khamfroush, "DECO: A deadline-aware and energy-efficient algorithm for task offloading in mobile edge computing," *IEEE Systems Journal*, vol. 17, no. 1, pp. 952-963, 2022. <https://doi.org/10.1109/JSYST.2022.3185011>
- [3] L. Chen, Y. Liu, Y. Lu, and H. Sun, "Energy-aware and mobility-driven computation offloading in MEC," *Journal of Grid Computing*, vol. 21, no. 2, p. 26, 2023. <https://doi.org/10.1007/s10723-023-09654-1>
- [4] Y. Cheng, H. Zhao, and W. Xia, "Energy-aware offloading and power optimization in full-duplex mobile edge computing-enabled cellular IoT networks," *IEEE Sensors Journal*, vol. 22, no. 24, pp. 24607-24618, 2022. <https://doi.org/10.1109/JSEN.2022.3218584>
- [5] S. Dong *et al.*, "Task offloading strategies for mobile edge computing: A survey," *Computer Networks*, vol. 254, p. 110791, 2024. <https://doi.org/10.1016/j.comnet.2024.110791>
- [6] X. Jiao *et al.*, "Deep reinforcement learning empowers wireless powered mobile edge computing: Towards energy-aware online offloading," *IEEE Transactions on Communications*, vol. 71, no. 9, pp. 5214-5227, 2023. <https://doi.org/10.1109/TCOMM.2023.3283792>
- [7] V. Joshi and K. Patil, "Delay-energy aware task offloading and vm migration policy for mobile edge computing," *Wireless Personal Communications*, vol. 123, no. 4, pp. 3309-3326, 2022. <https://doi.org/10.1007/s11277-021-09290-6>
- [8] M. Keshavarznejad, M. H. Rezvani, and S. Adabi, "Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms," *Cluster Computing*, vol. 24, no. 3, pp. 1825-1853, 2021. <https://doi.org/10.1007/s10586-020-03230-y>
- [9] S. Khan, M. Avgeris, J. Gascon-Samson, and A. Leivadreas, "EMDTORA: Energy-aware multi-user dependent task offloading and resource allocation in MEC using graph-enabled DRL," *IEEE Transactions on Green Communications and Networking*, vol. 9, no. 3, pp. 1453-1469, 2024. <https://doi.org/10.1109/TGCN.2024.3514578>
- [10] Y. Li *et al.*, "Mobility-aware partial task offloading and resource allocation based on deep reinforcement learning for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 74, no. 4, pp. 6459-6472, 2025. <https://doi.org/10.1109/TVT.2024.3512944>
- [11] Z. Li, V. Chang, J. Ge, L. Pan, H. Hu, and B. Huang, "Energy-aware task offloading with deadline constraint in mobile edge computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, no. 1, p. 56, 2021. <https://doi.org/10.1186/s13638-021-01941-3>
- [12] A. Mahapatra, S. K. Majhi, K. Mishra, R. Pradhan, D. C. Rao, and S. K. Panda, "An energy-aware task offloading and load balancing for latency-sensitive IoT applications in the Fog-Cloud continuum," *IEEE Access*, vol. 12, pp. 14334-14349, 2024. <https://doi.org/10.1109/ACCESS.2024.3357122>

- [13] U. Mohammad, S. Sorour, and M. Hefaida, "Energy aware task allocation for semi-asynchronous mobile edge learning," *IEEE Transactions on Green Communications and Networking*, vol. 7, no. 4, pp. 1766-1777, 2023. <https://doi.org/10.1109/TGCN.2023.3244710>
- [14] M. M. Alenazi, B. A. Yosuf, S. H. Mohamed, T. E. El-Gorashi, and J. M. Elmirghani, "Energy-efficient distributed machine learning in cloud fog networks," presented at the 2021 IEEE 7th World Forum on Internet of Things (WF-IoT). <https://doi.org/10.1109/WF-IoT51360.2021.9595351>, pp. 935-941, 2021.
- [15] Y. Hao, J. Cao, Q. Wang, and T. Ma, "Energy-aware offloading based on priority in mobile cloud computing," *Sustainable Computing: Informatics and Systems*, vol. 31, p. 100563, 2021. <https://doi.org/10.1016/j.suscom.2021.100563>
- [16] J. Bi, K. Zhang, H. Yuan, and Q. Hu, "Energy-aware task offloading with genetic particle swarm optimization in hybrid edge computing," presented at the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). <https://doi.org/10.1109/SMC52423.2021.9658678>, pp. 3194-3199, 2021.
- [17] M. Zhao and J. Lu, "Energy-aware tasks offloading based on DQN in medical mobile devices," *Journal of Cloud Computing*, vol. 13, no. 1, p. 128, 2024. <https://doi.org/10.1186/s13677-024-00693-x>
- [18] H. A. Alharbi, M. Aldossary, J. Almutairi, and I. A. Elgendy, "Energy-aware and secure task offloading for multi-tier edge-cloud computing systems," *Sensors*, vol. 23, no. 6, p. 3254, 2023. <https://doi.org/10.3390/s23063254>
- [19] M. Zawish, N. Ashraf, R. I. Ansari, and S. Davy, "Energy-aware AI-driven framework for edge-computing-based IoT applications," *IEEE Internet of Things Journal*, vol. 10, no. 6, pp. 5013-5023, 2023. <https://doi.org/10.1109/JIOT.2022.3219202>
- [20] H. Min *et al.*, "URLLC-aware and energy-efficient data offloading strategy in high-mobility vehicular mobile edge computing environments," *Vehicular Communications*, vol. 50, p. 100839, 2024. <https://doi.org/10.1016/j.vehcom.2024.100839>
- [21] B. Sellami, A. Hakiri, S. B. Yahia, and P. Berthou, "Energy-aware task scheduling and offloading using deep reinforcement learning in SDN-enabled IoT network," *Computer Networks*, vol. 210, p. 108957, 2022. <https://doi.org/10.1016/j.comnet.2022.108957>
- [22] P. Singh and R. Singh, "Energy-efficient delay-aware task offloading in fog-cloud computing system for IoT sensor applications," *Journal of Network and Systems Management*, vol. 30, no. 1, p. 14, 2022.
- [23] Z. Song, M. Xie, J. Luo, T. Gong, and W. Chen, "A carbon-aware framework for energy-efficient data acquisition and task offloading in sustainable AIoT ecosystems," *IEEE Internet of Things Journal*, vol. 11, no. 24, pp. 39103 - 39113, 2024. <https://doi.org/10.1109/JIOT.2024.3472669>
- [24] H. Zhou, K. Jiang, X. Liu, X. Li, and V. C. Leung, "Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1517-1530, 2021.
- [25] A. B. Sada, A. Khelloufi, A. Naouri, H. Ning, and S. Dhelim, "Energy-aware selective inference task offloading for real-time edge computing applications," *IEEE Access*, vol. 12, pp. 72924-72937, 2024.
- [26] X. Liu, J. Liu, and H. Wu, "Energy-aware allocation for delay-sensitive multitask in mobile edge computing," *The Journal of Supercomputing*, vol. 78, no. 15, pp. 16621-16646, 2022. <https://doi.org/10.1007/s11227-022-04550-z>
- [27] M. K. Mondal, S. Banerjee, D. Das, U. Ghosh, M. S. Al-Numay, and U. Biswas, "Toward energy-efficient and cost-effective task offloading in mobile edge computing for intelligent surveillance systems," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 4087-4094, 2024. <https://doi.org/10.1109/TCE.2024.3362396>
- [28] H. Jiang, X. Dai, Z. Xiao, and A. Iyengar, "Joint task offloading and resource allocation for energy-constrained mobile edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 4000-4015, 2022. <https://doi.org/10.1109/TMC.2022.3150432>
- [29] M. Mehrabi *et al.*, "Mobility-and energy-aware cooperative edge offloading for dependent computation tasks," *Network*, vol. 1, no. 2, pp. 191-214, 2021. <https://doi.org/10.3390/network1020012>
- [30] A. Xiong *et al.*, "An Energy Aware Algorithm for Edge Task Offloading," *Intelligent Automation & Soft Computing*, vol. 31, no. 3, pp. 1641-1654, 2022.

- [31] X. Chen and G. Liu, "Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10843-10856, 2021. <https://doi.org/10.1109/JIOT.2021.3050804>
- [32] Y. Chen, J. Xu, Y. Wu, J. Gao, and L. Zhao, "Dynamic task offloading and resource allocation for NOMA-aided mobile edge computing: An energy efficient design," *IEEE Transactions on Services Computing*, vol. 17, no. 4, pp. 1492-1503, 2024. <https://doi.org/10.1109/TSC.2024.3376240>
- [33] Y. Li *et al.*, "Energy-aware, device-to-device assisted federated learning in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 7, pp. 2138-2154, 2023. <https://doi.org/10.1109/TPDS.2023.3277423>
- [34] W. Almuselem, "Energy-efficient and security-aware task offloading for multi-tier edge-cloud computing systems," *IEEE Access*, vol. 11, pp. 66428-66439, 2023. <https://doi.org/10.1109/ACCESS.2023.3290139>
- [35] J. Silva, E. R. B. Marques, L. M. B. Lopes, and F. Silva, "Energy-aware adaptive offloading of soft real-time jobs in mobile edge clouds," *Journal of Cloud Computing*, vol. 10, no. 1, p. 38, 2021. <https://doi.org/10.1186/s13677-021-00251-9>
- [36] N. Tripathy and S. Sahoo, "Energy aware effective task offloading mechanism in fog computing," presented at the International Conference on Computing, Communication and Learning, Springer, 2023.
- [37] M. Kim, J. Jang, Y. Choi, and H. J. Yang, "Distributed task offloading and resource allocation for latency minimization in mobile edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 15149-15166, 2024. <https://doi.org/10.1109/TMC.2024.3458185>
- [38] S. K. U. Zaman *et al.*, "Mobility-aware computational offloading in mobile edge networks: A survey," *Cluster Computing*, vol. 24, no. 4, pp. 2735-2756, 2021. <https://doi.org/10.1007/s10586-021-03268-6>
- [39] A. Mondal, P. S. Chatterjee, and N. K. Ray, "An optimal novel approach for dynamic energy-efficient task offloading in mobile edge-cloud computing networks," *SN Computer Science*, vol. 5, no. 5, p. 655, 2024.
- [40] GSMA, *The mobile edge: Advancing MEC for a hyperconnected future*. London, United Kingdom: GSM Association, 2024.

Views and opinions expressed in this article are the views and opinions of the author(s), Journal of Asian Scientific Research shall not be responsible or answerable for any loss, damage or liability etc. caused in relation to/arising out of the use of the content.