Shabina Modi, 2025, 13:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Deep Reinforcement Learning Approaches for Task Offloading in Mobile Edge Computing: A Critical and Systematic Review

Professor Shabina Modi, Ajinkya Shriram Gurav, Prasad Nagnath Londe, Pranit Dattatraya Patil, Prasanna Motiram Kasabe, Ajinkya Anil Dhane

Computer Science and Engineering
Rayat Shikshan Sanstha's Karmaveer Bhaurao Patil College of Engineering, Satara, Maharashtra, India

Abstract- This study focuses on developing a chatbot for a college website to enhance the user experience by automating responses to common inquiries, including FAQs, admissions, and student support. The backend uses Dialog flow, TensorFlow, and Python for NLP-based contextual intelligence, while the frontend uses HTML and CSS for smooth deployment. The system demonstrated the efficacy of AI-powered chatbots in enhancing accessibility, lowering workload, and offering ongoing support in academic settings with an 8.7/10 user satisfaction score and 91% response accuracy [1]– [5], [16].

Keywords: Artificial Intelligence, Chatbot, Dialogflow, NLP, Machine Learning, TensorFlow, and College Website Automation.

I. INTRODUCTION

Context: The Necessity of Edge Computing in the IoT and 6G Era

The modern landscape of computing is defined by the exponential growth of the Internet of Things (IoT) and the proliferation of complex, computationally intensive mobile applications, such as augmented reality, autonomous vehicles, and advanced smart healthcare systems [User Query Abstract, User Query Intro]. These applications share a critical requirement: extremely low latency processing, frequently demanding response times below 10 milliseconds. Traditional centralized Cloud computing, while offering vast computational capacity, fails to meet these stringent real-time requirements because the distant transmission of data introduces unacceptable delays and places enormous bandwidth burdens on the backhaul network [User Query Intro].

Mobile Edge Computing (MEC), also termed Fog Computing, serves as the critical architectural paradigm shift designed to address this challenge. MEC strategically deploys computing and storage resources closer to the end-users—at the edge of the mobile network, typically within cellular base stations or access points (APs) [User Query Intro]. By significantly reducing the physical distance data must travel, MEC dramatically cuts communication latency, thus enabling the real-time processing capability essential for next- generation applications.

The Core Problem: Optimal Task Offloading Decision Making

The effective operation of any MEC system hinges on efficient task ofloading strategies. Task ofloading is the fundamental decision process that determines where a computational task, generated by a mobile device or IoT sensor, should be executed. For a given task T, the of loading decision D resides in a tri-state space: $D \in \{0, 1, 2\}$, corresponding to Local Execution (0), Ofloading to a nearby MEC server (Edge Ofloading) (1), or Ofloading to the distant central Cloud (Cloud Ofloading) (2).

This decision process is inherently a multi-objective optimization challenge, where strategies seek to minimize a weighted function combining key performance metrics. The primary goals are minimizing the total Latency (L) and minimizing the Energy Consumption (E) of the mobile device. In complex systems, this may be encapsulated in a generalized Cost (C) or System Utility maximization. The successful deployment of next-generation IoT applications is predicated on

overcoming the systemic trade-offs between performance, privacy, and computational complexity inherent in these ofloading decisions [User Query Conclusion].

Scope and Structure of the Review

This systematic review provides a comprehensive categorization, comparison, and critical evaluation of state-of-the-art task ofloading strategies developed in recent years. The review focuses particularly on the shift from traditional mathematical modeling—specifically optimization theory and game theory—toward modern learning-based techniques, notably Deep Reinforcement Learning (DRL). The analysis identifies performance trade-offs, evaluates limitations of existing solutions, and outlines critical future research directions, emphasizing the necessity for secure, adaptive, and heterogeneous solutions suitable for dynamic network conditions and integration with future 6G technologies.

II. FUNDAMENTALS OF MOBILE EDGE COMPUTING ARCHITECTURES AND MODELING

The Three-Layered MEC Ecosystem

The MEC environment is fundamentally structured around three distinct computational layers, each serving a unique role in task processing:

1. Device Layer: This layer comprises the end-user equipment, including smartphones and various IoT sensors. These devices are the primary source of computational tasks but are typically constrained in terms of battery life, processing power, and storage capacity.
2. Edge Layer: This intermediate layer consists of MEC servers, which are powerful computation units strategically deployed geographically close to the device layer, often integrated within access points (APs) or cellular base stations (eNodeBs). The Edge Layer is characterized by high-bandwidth connectivity and provides low-latency computation services. Critically, these resources are finite and often shared among competing users, creating resource contention issues.
3. Cloud Layer: Representing the traditional centralized cloud data centers, this layer offers virtually limitless computational and storage capacity. However, due to the geographical distance, communication to this layer incurs the highest latency, limiting its suitability for real-time applications.

Key Decision Variables and Multi-Objective Optimization Metrics

Optimal ofloading policies are formulated by optimizing one or more decision variables against key performance metrics.

The principal metrics guiding the ofloading decision D are:

- Latency (L): This is the total time elapsed from the moment a task is generated until the computed results are successfully received by the mobile device. L encompasses several distinct components: the time required for the device to transmit the task data to the Edge or Cloud (transmission time), the time spent waiting for the MEC server's resources (queueing time), and the actual computational duration (computation time).
- Energy Consumption (E): This metric quantifies the power expended by the mobile device. This includes the energy consumed for local computation if the task is not ofloaded, or the energy required for wireless transmission if the task is sent to the Edge or Cloud. Minimizing E is vital for prolonging the battery life of resource- constrained IoT devices.
- System Utility (U): Often, strategies do not optimize L or E in isolation but rather combine them into a single metric, U. System utility is frequently defined as a weighted linear combination, $U = \alpha L + \beta E$, where α and β are weighting factors that reflect the system's priority (e.g., latency-sensitive applications would assign a much higher weight to α). Alternatively, utility can be defined as throughput maximization under strict Quality of Service (QoS) constraints on latency.

Task Modeling: Granularity and Dependencies

The complexity of the ofloading problem is heightened by the nature of the tasks themselves. Tasks can be modeled as coarse-grained (all or nothing ofloading) or fine- grained (partitioning the task into segments for local and remote execution). Furthermore, complex applications often involve tasks with inherent dependencies; for instance, Task A must be completed before Task B can begin. Handling such dependencies requires sophisticated, coordinated ofloading models. These advanced models must be capable of mapping dependent tasks to heterogeneous resources across multiple MEC servers simultaneously, especially in dense network scenarios.1 Modeling the ofloading problem in a multiple user, multiple MEC server environment, while accounting for task dependencies and resource competition, often necessitates framing the problem as a Markov Decision Process (MDP), paving the way for DRL solutions.1

Categorization of Task Offloading Strategies

Task ofloading methodologies can be systematically categorized based on the underlying mathematical or algorithmic tools used to determine the ofloading decision.

Optimization Theory-Based Strategies

Optimization-based approaches attempt to formalize the ofloading and resource allocation problem as a minimization problem, seeking to find the globally optimal set of decisions that satisfies network constraints [User Query Classification].

- **Core Techniques and Goal:** These strategies employ established techniques such as Linear Programming (LP), Non-Linear Programming, Mixed Integer Linear Programming (MILP), and Convex Optimization. The primary goal is minimizing a predefined objective function, typically the weighted sum of energy and latency ($E + \lambda L$).
- **Critique and Limitations:** While these methods, particularly Convex Optimization, can guarantee convergence to the optimal solution, they suffer from two critical drawbacks. First, they often require perfect network state information (e.g., channel conditions, server loads, processing speeds), an assumption that rarely holds true in dynamic wireless environments. Second, realistic ofloading problems often involve binary decisions (ofload or not) and resource allocation, leading to computationally prohibitive models like MILP. Although MILP-based approaches can achieve significant performance improvements—such as reducing latency by approximately 55% compared to Genetic Algorithms (GA) and 35% compared to Particle Swarm Optimization (PSO), and reducing the dropped task ratio by 70% over GA 2—their computational complexity means they cannot adapt quickly enough to the rapid, real-time changes characteristic of mobile networks.3

Game Theory-Based Strategies

Game theory provides a framework for modeling decision-making when multiple independent, rational entities (users or servers) interact and compete or cooperate for shared resources [User Query Classification].

- **Core Techniques and Goal:** The goal is maximizing the “utility” (profit or benefit) for each user while driving the entire system toward a stable equilibrium, most commonly the Nash Equilibrium (NE). Techniques include Non-Cooperative Games, where users compete for finite MEC resources, and Cooperative Games. A particularly prevalent method is the Stackelberg Game, which models a hierarchical structure where the MEC server acts as the "leader" by setting resource pricing or allocation policies, and the mobile devices act as "followers" by adjusting their ofloading strategies in response.4
- **Critique and Limitations:** Game theory is excellent for modeling decentralized decision-making and resolving resource pricing issues in multi-user settings.4 By formulating the problem as a non-cooperative game, the existence of a Nash Equilibrium can be proved, and iterative algorithms can be proposed to obtain the solution, often using techniques like backward induction to verify the existence of a Stackelberg Equilibrium (SE).4 However, traditional game theory models struggle with rapid channel fluctuations and achieving fast convergence to

the optimal equilibrium in highly dynamic settings.3

Deep Reinforcement Learning (DRL)-Based Strategies

Machine Learning (ML) and, specifically, Deep Reinforcement Learning (DRL) represent the state-of-the-art solution for tackling the inherent complexity and dynamism of MEC ofloading [User Query Classification].

- **Core Approach:** DRL approaches model the environment as a Markov Decision Process (MDP). The DRL agent (which can be the mobile device, the MEC server, or a centralized controller) interacts with the dynamic network environment, learns an optimal ofloading policy through trial-and-error, and aims to maximize the cumulative reward (which is often the negative cost, representing minimized energy and latency).
- **Dominance and Techniques:** DRL and its extension, Multi-Agent DRL (MADRL), are increasingly dominant. An analysis of recent strategies reveals that DRL and MADRL collectively account for 36.26% of contemporary solutions for joint task ofloading and resource allocation problems, signifying a substantial increase in their adoption.7 Key DRL algorithms employed include Deep Q-Networks (DQN), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Optimization (PPO), depending on whether the action space (ofload decision and resource allocation) is discrete or continuous.

IV. CRITICAL PERFORMANCE ANALYSIS AND COMPARATIVE EVALUATION (THE DRL ADVANTAGE)

The comparative merits of the three ofloading categories reveal a clear trade-off between guaranteed mathematical optimality and adaptability to real-world operational dynamics.

Foundational Comparative Analysis: Optimality versus Adaptability

Optimization-based strategies provide a theoretical guaranteed global optimum but are often computationally intractable (NP-Hard problems) and require a complete, static snapshot of the network state. Game Theory strategies succeed in modeling rational, decentralized behavior and achieving a stable equilibrium but are slow to converge or adapt when channel conditions fluctuate rapidly. DRL, conversely, sacrifices the guarantee of absolute global optimality in favor of achieving real-time, robust, near-optimal performance and high adaptability in dynamic environments.8

A detailed comparison highlights this fundamental shift in research focus:

Table 1: Comparative Analysis of Foundational Task Ofloading Categories

Category	Core Optimization Goal	Optimality Guarantee	Adaptability to Dynamic Conditions	Decision Time Complexit y	Representativ e Mechanism
Optimization -Based	Global minimum (Energy $+\lambda$ Latency)	High (Guaranteed if convex/feasible)	Low (Requires near-perfect state; prone to failure)	High (MILP, often NP-Hard)	Convex Optimization, MILP [2]
Game Theory-Based	Nash/Stackelber g Equilibrium (Decentralized Utility)	Medium (Guarantees local equilibrium)	Medium (Reacts to user actions; slow convergence)	Medium (Iterative search algorithms)	Stackelberg Game [4]

DRL/ML-Based	Maximum Cumulative Reward (Policy Learning)	High (Near-optimal performance)	High (Learns dynamic policy online)	Low (After training), High (Training time)	PPO, DDPG, MADRL [7]

The superiority of DRL in handling real-world complexity is evident because DRL is more robust than heuristic algorithms, enabling it to make effective real-time online decisions based on learned policies.8 Traditional methods, including linear regression and dynamic programming, inherently possess disadvantages when applied to dynamic task-ofloading decisions in complex scenarios.3

In-Depth Comparison of DRL Variants for Offloading

The effectiveness of DRL hinges critically on selecting the appropriate algorithm based on the task's action space, which is often hybrid (discrete ofloading choice combined with continuous resource allocation).

Table 2: Performance Comparison of Core Deep Reinforcement Learning Algorithms in MEC

DRL Algorithm	Action Space Suitability	Convergen ce Speed	Stability and Robustness	Primary Trade-off/Challenge	Empirical Performanc e Insight
Deep Q-Network (DQN)	Discrete only	Fast initially	Medium (Q-value overestimati on)	Limited to binary/discret e ofloading decisions.	Outperform ed by modern algorithms in delay minimizatio n.[9]
Deep Determinis tic Policy Gradient (DDPG)	Continuous/H ybrid	Fast	Medium	High sensitivity to hyperparamet ers and noise; stability issues.[10]	Suitable for continuous resource allocation but less robust than PPO in dynamics.[11]
Proximal Policy Optimizati on (PPO)	Continuous/D iscrete/H ybrid	Excellent (Policy Gradient approach)	High (Trust region optimization)	Slightly higher per-step complexity; often requires centralized training.[11]	Demonstrat es significant improveme nt in minimizing processing delay and achieving stability.[9]

The analysis demonstrates that PPO has emerged as the most favored algorithm for complex MEC ofloading due to its stability and robustness.10 While DQN is suitable only for discrete action spaces and DDPG is adept at continuous control tasks, PPO provides versatility and consistent performance.10 Recent empirical results indicate that PPO algorithms can quickly achieve the optimal policy for ofloading in dynamic environments. When compared against benchmarks like DDPG and DQN, PPO demonstrates significant improvement in minimizing processing delay and offers superior stability and faster convergence.9 The enhanced stability of PPO stems from its mechanism of constraining policy updates within a "trust region," which prevents unstable fluctuations and catastrophic performance degradation during the learning process in volatile wireless channels.11

Hybrid Strategies: Combining DRL and Game Theory

A significant advancement involves hybrid strategies that strategically combine DRL's adaptability with Game Theory's ability to model decentralized, rational interactions. This combination addresses the complexity of multi-user systems where competition for shared resources is high.

This approach resolves the convergence-decentralization dilemma by assigning the dynamic elements to the DRL agent and the economic/rational elements to the game model. For instance, in one sophisticated approach, a PPO reinforcement learning framework was employed to handle the dynamic selection of MEC servers, while a two-step optimization based on Game Theory was used to determine the size of the ofloaded data and the pricing of computing services.12 The optimal values for data size and pricing were determined by achieving the Nash Equilibrium of the strategy game between end-users. This combined approach was proven to have better performance than existing solutions in terms of both convergence time and stability.12

Empirical studies corroborate the effectiveness of such hybrid systems. Models combining DRL with Nash Equilibrium game theory have shown substantial performance gains over traditional DDPG and DQN algorithms: energy consumption was reduced by 26.4%, latency decreased by 6.87%, and overall cost was minimized by 7.41%.13

V. FRONTIER CHALLENGES AND FUTURE RESEARCH TRAJECTORIES

Despite the maturation of DRL methodologies, the transition to next-generation ecosystems presents four critical unresolved challenges that define the current research frontier.

Handling Dynamic Network States and User Mobility

Existing models often rely on assumptions of stable channel conditions, which fail when faced with the volatile, dynamic nature of real-world wireless channels and high device mobility.

The future trajectory must focus on developing robust and lightweight ML models that can adapt their ofloading policies in real-time with minimal re-training overhead. This necessitates developing DRL models that incorporate techniques such as meta-learning or transfer learning to rapidly assimilate new network states. Furthermore, incorporating mechanisms like Graph Attention Networks (GAT) within PPO (e.g., GAT-based EPS-PPO) has been shown to outperform DDPG and similar methods by ensuring adaptability, scalability, and robustness, specifically against failures in dynamic and uncertain MEC environments.11 The goal is to evolve ofloading from reactive decision-making to a proactive policy that anticipates network changes.

Security and Privacy Preservation in Offloading

Ofloading sensitive tasks, particularly those involving private user data (e.g., smart healthcare, vehicular data), to an untrusted Edge server generates serious privacy and security concerns. Addressing this requires a dual approach to security: securing the model training phase and securing the runtime computation phase.

Table 3: Security and Privacy Frameworks for MEC Ofloading

Challenge Addressed	Solution/Technique	Mechanism of Privacy Preservation	Impact on Performance	Integration Context

Model Training Privacy (Data Silos)	Federated Learning (FL)	Decentralized training; only model updates (parameters) are shared, not raw data.[14]	Communication overhead during aggregation; challenges with Non- IID data.	Training the DRL ofloading policy across multiple devices.[15]
Runtime Computation Privacy	Homomorphic Encryption (HE)	Enables computation directly on encrypted task data at the untrusted MEC server.[17]	High computational overhead for encryption/decryption ; limited complex operations.	Securing sensitive data processing (e.g., healthcare) after the ofload decision.
Efficiency and Privacy	Federated Knowledge Distillation (FKD/ KECO)	Transfers learned knowledge from complex models via soft labels, protecting underlying data while reducing communicatio n size.[14]	Requires careful distillation strategy design.	Optimizing large-scale model deployment in resource-constrained Edge environments.[14]

Model Training Privacy via Federated Learning (FL): Federated Learning is critical for privacy-preserving model training. FL enables multiple mobile devices to collaboratively update the parameters of a centralized DRL ofloading model without ever exposing their local private data.14 This addresses the problem of data silos and ensures that the model learns from a diverse dataset while maintaining user confidentiality. FL frameworks are vital for applications like Vehicular Edge Computing (VEC) in 6G networks.15

Runtime Computation Privacy via Homomorphic Encryption (HE): To protect the confidentiality of the task data itself during processing on an untrusted MEC server, advanced cryptographic techniques are essential.18 Homomorphic Encryption (HE) has emerged as a transformative solution, permitting the server to perform computations directly on encrypted data. This ensures data confidentiality throughout the entire computation process, regardless of whether the Edge server is compromised.17 Additive Homomorphic Encryption is specifically employed in secure aggregation steps within privacy-preserving decentralized learning frameworks.14

Efficiency and Scalability via Federated Knowledge Distillation (FKD): To minimize communication overhead and computational burden, especially in resource-constrained environments, Federated Knowledge Distillation (FKD), such as the KECO framework, is utilized. This method transfers compressed knowledge from complex "teacher" models to lightweight "student" models via soft labels, protecting the underlying data while significantly reducing the load on network bandwidth and resources.14

Addressing Resource Heterogeneity and Task Dependencies

Real-world MEC systems are characterized by heterogeneous hardware (e.g., CPUs, GPUs, FPGAs) and complex multi-part applications requiring tasks with sequential dependencies. Simple ofloading models cannot handle this complexity.

The future must center on creating coordinated ofloading models that can map dependent tasks to heterogeneous resources across multiple MEC servers simultaneously. This requires modeling the multi-user, multi-MEC server system as a sophisticated MDP, explicitly accounting for task dependencies and resource competition.1 Collaborative Multi-Agent DRL (MADRL) architectures are essential here, as they allow multiple agents (users or servers) to learn decentralized but coordinated policies that maximize global system utility, effectively managing resource utilization in a smart-device-centric MEC system.19

Integration with Next-Generation (6G) Networks

Future 6G networks are projected to feature ultra-dense deployment, intelligent reflecting surfaces, and new communication paradigms. Ofloading strategies must evolve to leverage these capabilities. The primary future focus is designing proactive offloading policies that anticipate user movement and network state changes using highly accurate predictive models. For Vehicular Edge Computing (VEC), a critical component of 6G, this requires efficient computational ofloading and resource management using FL to handle high communication costs and privacy protection.15 DRL algorithms will be tasked with dynamic allocation of transmission power and computing assignment ratios, especially in collaborative ofloading scenarios where local training is assisted by neighboring Roadside Units (RSUs).16 This optimization enhances resource utilization and task completion efficiency necessary for sub-millisecond 6G requirements.

VI. CONCLUSION

Task ofloading remains the cornerstone for achieving low-latency and energy-efficient computation within Mobile Edge Computing environments. This systematic review confirms a clear and decisive trend away from reliance on static Optimization Theory and pure Game Theory models toward intelligent, learning-based strategies. Deep Reinforcement Learning, particularly the use of stable and robust algorithms like Proximal Policy Optimization (PPO), and hybrid models integrating DRL with Game Theory, consistently outperform traditional benchmarks by providing the necessary adaptability to handle dynamic network conditions and complex resource coupling.

The successful migration to future 6G ecosystems and the deployment of sensitive Industrial IoT applications depend on resolving several critical trade-offs. Future research must prioritize the development of secure ofloading policies utilizing Federated Learning for training privacy and Homomorphic Encryption for runtime computation confidentiality.

Simultaneously, the field requires scalable Multi-Agent DRL architectures capable of managing resource heterogeneity and complex task dependencies across ultra-dense, collaborative MEC deployments. By focusing on these secure, resilient, and adaptive strategies, the fundamental bottlenecks of distributed computing can be overcome, realizing the full potential of edge intelligence.

REFERENCES

1. https://ieeexplore.ieee.org/iel8/6287639/10380310/10749809.pdf
2. https:/arxiv.org/pdf/2407.11155
3. https:/www.mdpi.com/2076-3417/12/21/11260
4. https://.ieee.org/iel7/6287639/9312710/09525401.pdf
5. https://w ww.mdpi.com/1424-8220/25/17/5286
6. https://www.computer.org/csdl/journal/sc/2025/01/10783060/22xb4JLB94Y
7. https://ieeexplore.ieee.org/document/10386861/