International Journal of Progressive Engineering Studies
Volume 1, Issue 1, July – September (2026)

RESEARCH ARTICLE

Dynamic Task Offloading Optimization in Mobile Edge Computing Using Graph-Attention Multi-Agent Reinforcement Learning

Hepsiba. D[1*], Aashika. T[2] and Dr. G Starlin Beaula[3]

[1]Dept. of Electronics and Communication Engineering, ACEW, Tamil Nadu, India
[2]Dept. of Electronics and Communication Engineering, ACEW, Tamil Nadu, India
[3]Dept. of Electronics and Communication Engineering, ACEW, Tamil Nadu, India
ibaheps@gmail.com

Abstract – Mobile Edge Computing (MEC) has emerged as an effective solution to support latency-sensitive and computation-intensive applications in Internet of Things (IoT) environments. However, efficient task offloading remains a challenging problem due to dynamic network conditions, heterogeneous device capabilities, and limited edge resources. Existing approaches, including heuristic methods, supervised learning, and deep learning models, cannot effectively adapt to dynamic network conditions, heterogeneous device capabilities, and multi-device resource contention. Single-agent reinforcement learning methods also fail to scale in environments where multiple devices compete for shared edge resources. To address these limitations, this paper proposes a Graph-Attention Multi-Agent Proximal Policy Optimization (GA-MAPPO) framework for intelligent and energy-efficient task offloading in MEC systems. In the proposed framework, each IoT device operates as an independent reinforcement learning agent, while a graph attention mechanism captures the relationships among devices, edge servers, and communication links. Proximal Policy Optimization ensures stable and efficient policy updates, allowing the system to adapt to changing workloads and network conditions. The multi-agent architecture supports collaborative decision-making, improving scalability and resource utilization across devices. Experimental results show that GA-MAPPO achieves 94% classification accuracy in offloading decisions and outperforms existing methods including DQN-based, PPO-based, MARL, and GNN-based approaches in terms of latency reduction, energy efficiency, and resource utilization. These results confirm that the proposed framework provides a scalable and adaptive solution for task offloading in next-generation MEC systems.

Keywords *– Mobile Edge Computing, Task Offloading, Graph Attention Network, Multi-Agent Reinforcement Learning, Proximal Policy Optimization, Energy Efficiency, Latency Optimization, Internet of Things*

1. INTRODUCTION

Mobile Edge Computing (MEC) has emerged as a promising technology to support the increasing demand for computation-intensive and latency-sensitive applications in modern Internet of Things (IoT) environments [11]. These devices are normally resource-constrained and have low processing power, storage and battery capacity [3]. Consequently, it becomes difficult to execute complicated applications like real-time analytics, augmented reality, autonomous decision-making, and intelligent monitoring when running locally [5]. The device resource constraints pose an interest in the form of effective computational assistance in proximity to final users [3]. Mobile edge computing resolves this issue by moving computing resources to the network edge, minimising the communication delay and making the system more efficient [12]. Thus, task offloading has become a viable method where the computational tasks are sent from the resource-constrained devices to the adjacent edge servers or cloud infrastructure [13]. This approach is used to minimise the delay in computation, enhance energy efficiency, and improve the performance of the whole system in IoT settings [1]. Efficient task offloading is a complicated and intricate issue despite the benefits of mobile edge computing [9]. The offloading decisions of the tasks should be based on various dynamic parameters such as network latency, bandwidth provision, device energy, task size, processing needs, and workload of edge servers [12]. These parameters are dynamic, as they change with the movement of devices, the fluctuations of networks and the changing user needs [22]. Conventional offloading methods of tasks are usually based on either heuristic-based or rule-based methods of making decisions based on predetermined thresholds [10]. These approaches are easy to use and simple to apply, but they are not very adaptable and cannot work in a dynamic environment [15]. In practice, in most situations, these fixed decision mechanisms are not able to react to sudden network jams, variable workload, or resource availability. As a result, there is a possibility of ineffective distribution of tasks and therefore more latency, energy use, and poor performance of the system [17]. Thus, to enhance the performance of the system and guarantee the effective use of resources in an environment of mobile edge computing, smart and dynamic offloading strategies for tasks are needed [19].

In order to overcome such shortcomings, machine learning approaches have gained broad usage over the last few years [5]. Decision trees, support vector machines, and deep neural networks are the techniques of supervised learning that have

been used to forecast the best decisions in task offloading using historical data [6]. These techniques enhance the accuracy of decisions as opposed to the conventional heuristic techniques [8]. However, supervised learning methods need labelled data to be trained, which is typically hard to find in real-world mobile edge computing settings [13]. Also, these models are usually trained in static settings and might not be able to adapt properly to changing environments [21]. The performance of static supervised learning models is poor when the conditions of a network vary or new devices are added to the system [9]. Moreover, hybrid deep learning models like convolutional neural networks, recurrent neural networks, and transformer-based models add overhead and complexity to computations [25]. This further complicates their use in resource-constrained edge environments with a preference for lightweight and adaptive solutions. Reinforcement learning has come out as a powerful alternative to resolve dynamic task offloading issues. Contrary to supervised learning, reinforcement learning allows agents to learn the best decision-making policy by interacting with the environment. Reinforcement learning agents in mobile edge computing monitor system state, take action and get rewarded according to the performance results. With time the agent becomes learned to optimise decision-making to perform better [30]. Deep reinforcement learning also complements this ability by using deep neural networks to enable the model to deal with complex and high-dimensional states [14]. This method allows dynamism in adjusting to fluctuating network conditions and resource availability. However, the reinforcement learning approaches of the past typically presuppose one-agent settings, in which a single device takes decisions on its own [17]. This assumption improves scalability and interaction limitations that do not model multiple devices that share a common resource [24]. With more devices, single-agent reinforcement learning methods are ineffective and result in inefficient resource allocation. [19]

Multi-agent reinforcement learning has been proposed as one of the solutions to these challenges. Multi-agent environments consist of a number of devices as autonomous agents, which learn through jointly finding optimal offloading policies of tasks [27]. Every agent monitors its local surroundings and communicates with other agents, which allows distributed decision-making and better scalability [17]. Multi-agent reinforcement learning is especially applicable in large-scale mobile edge computing systems, where multiple devices actively compete against each other in accessing common resources [23]. However, traditional multi-agent reinforcement methods tend to have difficulties with the representation of complicated relationships between devices, edge servers, and network conditions [17]. The absence of relational modelling lessens the accuracy of decisions and performance gains [18]. Thus, new modelling methods need to be applied to be able to model dependencies and interactions between system components [10]. Recently, graph attention networks have become an important focus of research in the modelling of complex relationships in distributed settings [18]. Graph attention mechanism devices and edge servers are nodes in a graph structure, and communication links are depicted by edges. Graph attention networks reproduce spatial and relational relationships among devices by allocating attention weights to various nodes [28]. This allows the system to rank valuable relationships and enhance decision-making performance [29]. The combination of graph attention networks and multi-agent reinforcement learning enables the system to use local device-related data and global network-related factors [4]. This fusion improves the level of accuracy in decision-making and dynamism to adapt to the evolving environment [18]. Consequently, graph attention-based multi-agent reinforcement learning models offer a viable remedy to the task of offloading in mobile edge computing environments [17]. Based on these issues, this paper presents a Graph-Attention Multi-Agent Proximal Policy Optimisation (GA-MAPPO) framework for energy-efficient task offloading in mobile edge computing environments [19]. The presented technique views each IoT device as a distinct agent and also uses graph attention networks to exemplify the association of devices and edge servers [25]. The problem of learning optimal task offloading strategies with a stable training performance is solved with the proximal policy optimisation algorithm [15]. By combining graph attention networks with multi-agent reinforcement learning, the proposed GA-MAPPO framework dynamically adapts to changing network conditions and improves decision accuracy. The suggested system offers a better level of scalability, less energy consumption, less latency and better system efficiency. Moreover, the framework enables distributed decision-making and smart resource allocation in high-scale mobile edge computing applications. The experiment findings prove that the proposed GA-MAPPO framework effectively enhances task offloading performance when compared to traditional approaches, which makes it an appropriate solution to next-generation IoT-enabled mobile edge computing systems.

2. RELATED WORKS

Mobile Edge Computing (MEC) has gained significant attention as an effective solution for addressing the increasing demand for low-latency and computation-intensive applications in Internet of Things environments [19]. As the number of smart devices and real-time applications is rapidly increasing, effective task offloading has become a vital part of edge computing [10]. Task offloading allows resource-limited devices to perform offload computational tasks to local edge servers or cloud environments, thus minimising processing latency and power. However, efficient task offloading strategies are difficult to design given a dynamic network state, heterogeneous devices, and limited edge resources. The first techniques of offloading tasks are primarily heuristic and rule-based. These methods are decided on preset thresholds like latency, bandwidth or device energy level. Although these techniques are easy to adopt, they are not efficient and could not work effectively in a dynamic environment [10]. The approaches are incapable of addressing complex decision-making situations, which leads to suboptimal resource

allocation and processing delay [11]. Moreover, the heuristic techniques frequently had a problem of scaling when there were two or more devices competing for the resources within a multi-device setup which was more sophisticated [13].

In order to eliminate these restrictions, optimisation-based offloading of tasks is proposed. These techniques are aimed at minimising latency, energy or cost through mathematical programming. Convex optimisation, game theory and multi-objective optimisation are some of the optimisation strategies used to enhance the decisions on task offloading [26]. Though they gave better performance than the heuristic methods, these approaches are typically computationally intensive and lack real-time flexibility [29]. Moreover, the optimisation-based solutions generally assumed a stable environment and failed to perform well in dynamic network situations or fluctuating workloads [27]. Machine learning techniques are introduced later to enhance task offloading decisions in mobile edge computing. sub strategies of task offloading forecasted on the basis of historical data are predicted with the help of supervised learning models. In classifying offloading decisions, different machine learning approaches are used, such as decision trees, support vector machines, and artificial neural networks [19]. These methods enhance the accuracy in prediction and minimise computation costs relative to the classical approaches to optimisation [30]. However, supervised learning models had to work with large labelled datasets and could not respond to changes in the environment in real-time [15].

Task offloading is also enhanced by deep learning-based methods that manipulate intricate feature representations. Convolutional neural networks, recurrent neural networks, and hybrid deep learning models are proposed to simulate the relationships among network parameters and offloading decisions [3]. These techniques delivered better results in complex settings where the patterns in data are nonlinear [22]. However, deep learning models tended to need a lot of training data and a lot of numbers to calculate [17]. Reinforcement learning is selected as a highly promising method to offload dynamic tasks in mobile edge computing. Using reinforcement learning, agents are able to learn the optimal policies by means of interacting with the environment and getting feedback according to the performance [6]. This is further improved by deep reinforcement learning, where neural networks are used in approximating value functions and policy functions [14]. These methods enhanced offloading decisions in tasks, as they took into account various measures of performance, including latency, energy usage, and resource usage [23]. The conventional reinforcement learning frameworks are more focused on single-agent environments and failed to effectively deal with multiple devices competing to share common resources [12].

Multi-agent reinforcement learning methods are proposed to resolve the issue of scalability. Multi-agent environments involve several devices that are independent agents and cooperate to make optimal decisions on task offloading [17]. Multi-agent reinforcement learning enhances scalability and aids distributed decision-making in large-scale mobile edge computing systems [20]. The techniques allow collaborative learning between devices and enhance performance of the system by avoiding resource contention [24]. To overcome these challenges, graph-based learning methods have been proposed in the recent past. Graph neural networks are used to provide a good framework to model the relationships between the devices, edge servers and communication links [8]. Graph attention mechanisms also complement this, as they add weight of importance to various nodes and connections within the network [2]. This is a method that enhances the accuracy of decision-making and performance of the system in a distributed environment [1]. The merging of graph learning with reinforcement learning in task offloading has been examined in recent works. This integration allows making dynamic decisions and taking into account the relationships between devices and network elements [4]. Graph-based reinforcement learning methods have shown better results in the area of reduction of latency, energy, and resources [16]. Scalability and adaptability of large-scale mobile edge computing settings are also improved by these techniques [28]. Even though these developments have been made, there are a number of challenges in the current practices. A lot of models continue to grapple with the challenge of trade-offs between various performance targets like latency, energy consumption, and system throughput [12]. There are also methods which have unstable training and slow convergence in dynamic environments [15]. This paper attempts to overcome these shortcomings by presenting a graph-attention multi-agent proximal policy optimisation framework to be used in offloading intelligent tasks in mobile edge computing environments [20].

3. PROPOSED FRAMEWORK

The rapid growth of mobile edge computing environments has significantly increased the demand for intelligent task offloading strategies that can efficiently manage dynamic network conditions and heterogeneous computing resources. Mobile edge computing has also become a solution to the weaknesses of the conventional cloud computing system, since it offers computational power nearer to end users. This method minimises latency and increases responsiveness and the overall service quality. However, mobile edge computing systems pose a number of novel challenges such as resource limitations, fluctuating network conditions, dynamic workloads, and energy constraints. The traditional methods of task offloading typically assume the use of fixed decision-making machinery or single-agent learning, which are inappropriate for managing these complex and large-scale settings. These traditional approaches do not cope with new conditions and tend to result in the inefficient use of resources, high latency, and high energy utilisation. Consequently, there is an increasing demand to have a smart architecture that can be adaptively determined, learn collaboratively, and manage resources efficiently in mobile edge computing systems. To overcome these issues, the current

paper presents a Graph-Attention Multi-Agent Proximal Policy Optimisation (GA-MAPPO) framework for energy-efficient task offloading in mobile edge computing systems. The proposed framework combines graph attention networks, multi-agent reinforcement learning, and proximal policy optimisation to enhance the decisions on task offloading and the overall system performance., as illustrated in Figure 1

The proposed framework is designed to overcome several limitations observed in existing task offloading approaches. Mobile edge computing environments include a variety of devices having different computing abilities, energy-related constraints, and communication needs. These devices can be smartphones, sensors, wearable devices, and IoT components, which create various workloads and demand effective processing. The traditional centralised systems find it difficult to handle these heterogeneous devices because they have limited scalability and high computational overhead. In centralised solutions, one controller makes all the decisions on behalf of all the devices, thus creating bottlenecks and resulting in inefficiency in the system. To solve this problem, the proposed GA-MAPPO framework uses a multi-agent architecture. Each device in this architecture is described as an autonomous agent, which can take intelligent decisions on offloading tasks. This distributed learning method enhances scalability and also allows devices to act independently with better system performance. Multi-agent reinforcement learning enables agents to learn optimal policies using local observations and interactions with the environment. This simplifies the system and increases flexibility to changing environments.

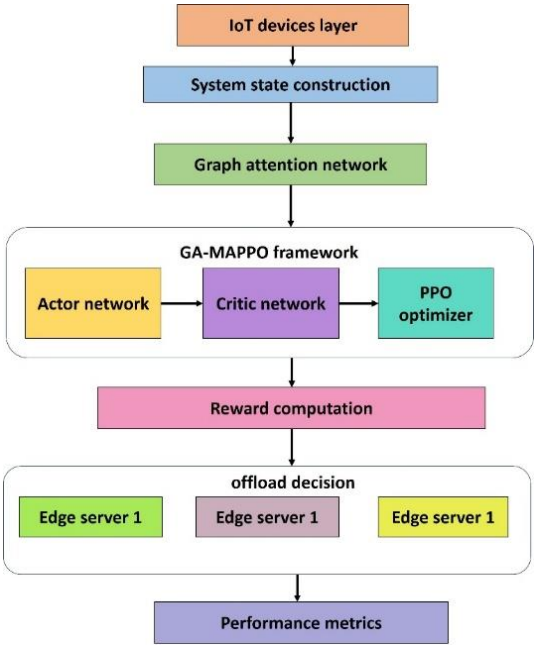

Figure 1 System Architecture of the Proposed GA-MAPPO Framework for Energy-Efficient Task Offloading in Mobile Edge Computing

The integration of the graph attention networks is also another important element of the proposed framework. Devices, communication links, edge servers, and communication links constitute a highly interconnected system in a mobile edge computing environment. The effectiveness of task offloading decisions not only depends on individual device conditions but also on the interrelations between devices and network resources. The classic forms of learning models tend to consider devices separately and do not reflect these relationships. However, interactions between several devices affect network congestion, resource availability, and communication delays. Therefore, relational information is introduced to improve decision accuracy. The proposed framework makes use of graph attention networks to model these relationships. The graph attention mechanism allows the system to extract relationships between devices, edge servers, and communication links. This enhances the decision-making process by taking into account both local and global network conditions. Also, graph attention networks place importance weights on the various nodes, and thus the model is able to prioritise essential connections. These weighted representations make the systems more efficient and improve the performance of offloading tasks.

Proximal policy optimisation is also proposed as the reinforcement learning algorithm in the offered framework. Reinforcement learning methods are common with decision-making in dynamic environments, and challenges encountered with them include unstable training and slow convergence. These problems are magnified in multi-agent setups where there are many agents who are learning concurrently. Proximal policy optimisation is chosen because of the stable learning behaviour and effective policy updates. This algorithm enhances the stability of learning through restrictions of the large policy changes and the gradual improvement of the policy performance. Proximal policy optimisation by balancing exploration and exploitation maximises training efficiency and minimises convergence time. Moreover, proximal policy optimisation encourages lifelong learning, which enables the system to change with varying network conditions and workload fluctuations. This flexibility renders the proposed architecture applicable to real-time mobile edge computing. Mobiles are usually run on limited battery capacity, and energy-conscious task offloading is therefore necessary. Over offloading of tasks or inefficient decision-making results may mean higher energy usage and less device life. The proposed GA-MAPPO model involves the use of energy consumption as one of the factors in decision-making. The system is dynamically set to process tasks locally or offload to edge servers by taking into account the energy constraints. This power-conscious approach will help to decrease the redundant offloading of tasks and enhance battery life. IoT devices, wearable systems, and mobile sensors are the domains where energy efficiency is especially significant because the power consumption directly influences usability and reliability. The proposed framework will increase the sustainability of devices

and improve the overall performance of the systems by maximising energy consumption.

The proposed framework also deals with latency reduction. A great number of mobile edge computing applications have a high demand for real-time processing and low latency. Fast response times are used in applications like augmented reality, autonomous driving, smart healthcare, and industrial automation. The effect of high latency is averse to the user experience and system reliability. The GA-MAPPO framework presented in the paper takes into account communication delay and processing delay in the determination of task offloading. The system reduces latency and enhances responsiveness by dynamically choosing the best places to execute. The ability facilitates real-time applications and improves user satisfaction. Moreover, latency-sensitive decision-making enhances the efficiency of the system by minimising the wasteful data transfer and maximising the use of resources. Another important aspect in mobile edge computing settings is resource utilisation. Efficient resource management is what guarantees that the computational resources are distributed among the devices efficiently. Edge servers are characterised by limited computational resources, and when they are not properly allocated, they can be overloaded and their performance may decrease. The proposed GA-MAPPO model is a dynamic resource allocation of the computational resources depending on workload conditions. This dynamic allocation of resources will avoid resource bottlenecks and will provide balance between tasks distributed among the edge servers. Scalability and system reliability are enhanced by efficiency in resource use. Moreover, the dynamic allocation will improve the flexibility of the system and will enable the framework to adjust to the different workloads and network conditions.

Another significant characteristic of the proposed framework is collaborative decision-making. In massive mobile edge computing systems, there can be numerous contending devices over scarce resources. Decision-making can result in conflicts over resources and poor performance of the system. Multi-agent architecture allows joint learning between devices. Agents exchange information and acquire collaborative policies to enhance efficiency in the system. This cooperative strategy minimises competition for resources and improves performance. Also, collaborative learning enhances equity in the distribution of resources, whereby all devices get the right computational support. The other significant benefit of the proposed GA-MAPPO framework is scalability. As the number of devices increases, the conventional centralised systems are inefficient and hard to manage. The multi-agent system enables the system to be scaled without adding computational complexity. Every agent develops on his or her own and helps in achieving improved global performance. The proposed framework is appropriate in large-scale IoTs due to this distributed learning. In large-scale sensor networks, industrial IoT systems, and smart cities, scalability is of particular significance.

Robustness is also enhanced in the proposed framework. Mobile edge computing environments are highly dynamic, with fluctuating network conditions and varying workloads. The mobile edge computing environments are dynamic, characterised by changes in the network conditions and different loads. Reinforcement learning makes the system update policies according to the new observations. Graph attention networks also increase robustness through capturing complicated relations within the environment. This combination enhances the reliability of the system and the predictability of the performance of the system in the varying conditions. Accuracy of the decisions is also enhanced by the proposed framework. Conventional rule-based systems fail to realise the dynamic behaviour of the system. Reinforcement learning enables the model to study the best policies by communicating with the environment. Graph attention networks lead to improved accuracy in decision-making because the relationships between devices are taken into account. These combinations enhance the offloading decisions and system performance.

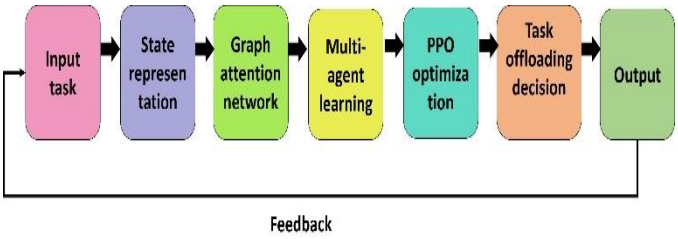

Figure 2 Workflow of the Proposed GA-MAPPO Framework for Task Offloading

In addition, the proposed GA-MAPPO framework reduces computational overhead. Deep learning models have a tendency to be computationally intensive. The proposed structure trades accuracy and computational efficiency with lightweight graph attention models and stable reinforcement learning models. This renders the system applicable in real-time applications and constrained environments. Overall, the proposed GA-MAPPO framework offers a smart, scalable, and resource-efficient task offloading solution to mobile edge computing settings.

4. METHODOLOGY

The proposed graph-attention multi-agent proximal policy optimisation is a framework derived to achieve intelligent offloading of tasks in mobile edge computing. The methodology is constructed as an entire process whereby inputs to the system are initially gathered by the IoT devices, communication networks, and edge servers and converted into a useful state representation, which is processed by graph attention learning and finally consumed by several reinforcement learning agents to make the most optimal offloading decisions. The model is not based on rigid guidelines. Rather, it acquires knowledge through constant engagement with the environment and refines its decision-

making in the long run. This methodology links all steps in a logical manner, and it begins with the data acquisition process and concludes with the optimised output of the task execution. The process-based design enhances flexibility, decreases latency, improves energy efficiency, and improves resource utilisation in fluid mobile edge computing.

4.1. Input Acquisition from Mobile Edge Computing Environment

The mobile edge computing environment input information is the initial step of the proposed methodology. Using this, numerous IoT devices continuously send computational tasks which are performed on the edge or offloaded to the edge server in the immediate neighbourhood based on the conditions of the system and resources available. These tasks are of different sizes and different computational complexity as well as have different latency requirements, and it is necessary to make efficient decisions. Simultaneously, the quality of the communication network and edge resource availability also have an effect on the choice. Thus, the framework initially collects the three key sources of system inputs, i.e., device-level information, network-level information, and edge-server-level information, as shown on Figure 3.

At the device level, there are inputs such as battery level, local processing power and queue backlog. The battery level also matters, as the devices that have low battery power might contemplate offloading tasks to save energy, and devices that have enough battery capacity can run tasks locally to save the communication time. The local computational capability is the processing power of the device, including CPU frequency, memory accessibility, and processing speed. More powerful devices can effectively do their work on-premises, whereas low-power devices can be offloaded. Moreover, the workload of the device is indicated by the number of tasks in the queue. A high queue size means that the workload is high and the process delay can be high in cases where local execution of tasks is done. Task offloading is useful to minimise the delay in these situations and enhance system performance.

Network-level inputs are bandwidth availability, transmission delay, and congestion status. These parameters define the quality of communication between the IoT devices and edge servers. The availability of bandwidth influences the speed of data transmission, whereas transmission delay influences how fast the tasks can be passed to the edge servers. Congestion status is used to check the situation in the network traffic, and the model is able to determine whether the network is congested or not. Unstable network conditions can raise the communication latency, and local execution can be more appropriate, but stable network conditions allow offloading of tasks efficiently. Processing capacity, current workload, and available resources are the inputs at the edge-server level. These parameters explain the capacity of edge servers to handle received tasks.

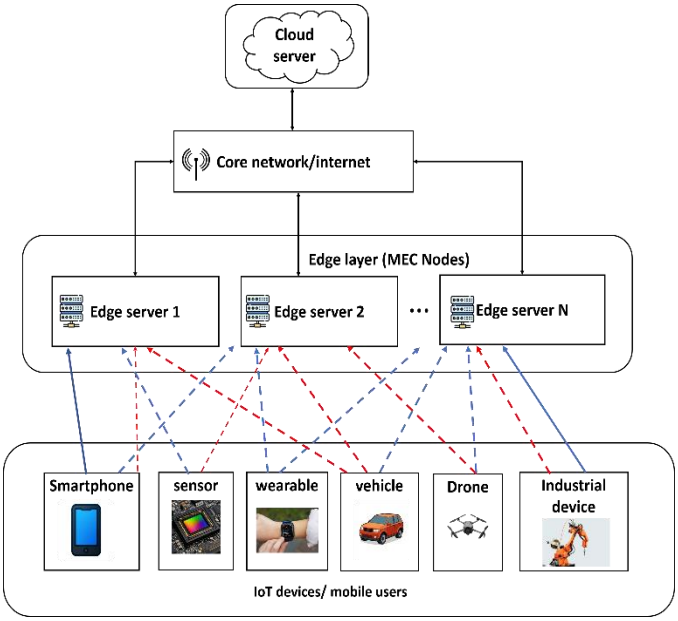

Figure 3 Mobile Edge Computing Architecture for Task Offloading

Task offloading is possible when the workload is low and the edge server has enough computation capacity. However, in case the server workload is high, the offloading can add delay. Thus, it is required to monitor the conditions of servers to achieve the optimal allocation of tasks. These inputs are the current operating state of the whole mobile edge computing space. The input data is updated regularly because the environment is dynamic and changes over time, and hence the model can respond to the current system conditions rather than relying on the old values. This is the entry point of the entire methodology in the sense that the quality of inputs that are gathered would have a direct influence on the quality of decisions that will be made in later stages. The proposed framework ensures the effective establishment of the intelligent offloading of tasks and effective management of resources by gathering correct and up-to-date information about the systems.

4.2. System State Construction and Feature Preparation

Once the raw input data is gathered, it is then converted into a structured state representation which can be interpreted by the learning agents, as show in Figure 4. Raw environmental data in its raw form is possibly not directly applicable to intelligent decision-making since this data is provided by different parties and may be in different quantities and degrees. Hence, this framework builds a multidimensional state vector which puts all the pertinent system parameters into a single representation. This state contains the device properties, task properties, network properties, and resource properties of servers. All the state parameters give context to the task offloading decision. Examples include how battery power can deter local execution and high network delay can reduce the value of offloading. In the same way, a server with lots of work

cannot be assigned to do other tasks despite having a great capacity of calculation.

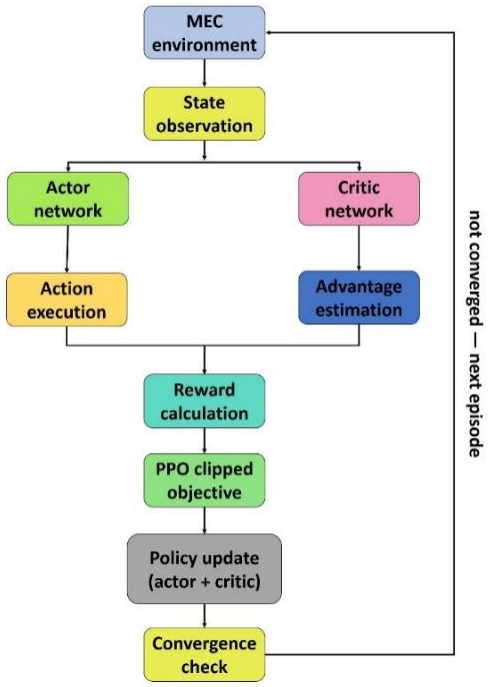

Figure 4 Proximal Policy Optimization (PPO) Based Policy Learning and Optimization Framework

The stability of learning is enhanced by feature normalisation followed by passing the state to the model. Normalisation is necessary to make sure that all features are in a similar range of numbers and no one parameter is overpowering the learning process. The significance of this stage of state preparation is that it converts the system input into a machine-readable form. Consequently, the agents will be able to understand the existing system situation clearly and make more trustworthy decisions on task offloading at the subsequent stage.

4.3. Relationship Modelling Through Graph Attention Mechanism

Once the system state is prepared, the relationships between devices and edge servers are detected through graph attention learning based on the proposed methodology. Devices are not used in isolation in a real mobile edge computing environment. The availability of neighbouring devices, edge server availability, shared communication links and dynamically varying workloads affect their performance. These interactions make it complex to depend on the other, which needs to be monitored when making decisions to offload tasks. The conventional methods of offloading tasks usually consider devices as isolated and do not consider these interdependencies, which may result in inefficient resource distribution and cause higher latency. To address this shortcoming, the proposed structure represents the mobile edge computing environment in the form of a graph that can help to effectively establish the relationship between all the components of the system.

This graph depiction considers IoT devices and edge servers to be nodes, and communication links between them are depicted as edges. System information in each node includes device workload, battery level, computational capability, server capacity, and network conditions. The edges are communication quality between nodes, bandwidth, latency, and signal strength. This organised representation helps the system to model relations among devices and servers in a better way. Using this graph, the framework reflects the dynamic relationships and dependencies that are present within the mobile edge computing environment. These representations allow the system to examine the impact of one device or server on another, thus enhancing the accuracy of task allocation decisions. Once the graph structure is built, a graph attention mechanism is used to learn the significance of every node and its surrounding relationships. In the graph attention mechanism, unlike the traditional graph-based methods that consider the neighbours of a node equally, the weights of the individual nodes are allocated differently according to their relevance to the decision in question. This concentration on learning aids the system to concentrate on the significant devices and edge servers and minimises the role of irrelevant nodes.

For example, an edge server with a large computation capacity and low communication delay might be assigned a large weight on the attention list relative to a server with a heavy load or remote location. Otherwise, high-workload devices that have restricted battery capacity can be given preference in the decision-making. These attention weights are learned automatically by the model throughout the training process, which enables the framework to keep up with the dynamic changes in the environment. The graph attention mechanism can extract meaningful features that enhance the contextual awareness by focusing on the most relevant relationships. In the process, the framework creates a better representation of the system state that reflects the properties of the individual devices as well as their relationships with their neighbours. This rich feature representation is further passed to the decision-making module, where it is processed further. The proposed methodology helps to increase the accuracy of task allocation, decrease latency, and improve the overall system performance because of the integration of graph attention learning. This action is very important towards enhancing decision-making, as it gives a better insight into the complicated interactions within the mobile edge-computing environment.

4.4. Action Space Definition and Offloading Decision Generation

Once the state of the system has been optimised with the help of graph attention learning, the next step is to specify the available actions and produce a task offloading decision, as shown in Figure 5.

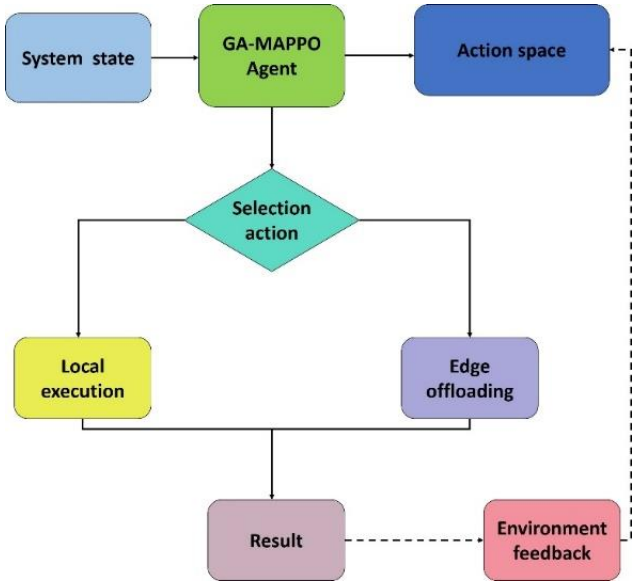

Figure 5 GA-MAPPO Based Task Offloading Decision Process

In the presented framework, every agent has to select a single course of action for every task presented to it. The possible actions typically involve either running the task on the local processor of the IoT device or offloading the task to an edge server of choice. This action space is used to reflect actual implementation decisions in mobile edge computing systems. The decision is not made randomly. Rather, it is founded on the processed state information that is given by the former stages.

The agent analyses the existing battery life, communication time, bandwidth and server workload and then decides an action. In the event that local execution would be expensive in terms of delay or energy consumption, the model might favour offloading. Conversely, when network conditions are not good or the server chosen is overloaded, the model can instead opt to use local execution. Through this, the action generation stage becomes the centre of decision-making of the methodology. It converts the studied system state to an execution plan. The step is extremely crucial since it links the knowledge of the environment with the task allocation result. The framework, therefore, starts shifting to output generation rather than the input analysis.

4.5. Reward Calculation for Evaluating Decision Quality

After an action is chosen and implemented, the system monitors the result of a decision and calculates a reward value. One of the most significant elements of the proposed methodology is the reward function since it helps the agents to receive feedback about the effectiveness of their decisions. Under the proposed framework, the agents constantly engage with the mobile edge computing environment by making decisions of either performing tasks within the mobile environment or transferring them to the edge servers. The system reviews the outcome every time an action is taken based on the key performance indicators of the time taken to complete the task, the amount of energy used, and the resource used. Using these results, a reward value is derived to lead the learning process. Within this paradigm, the reward is modelled by integrating several performance goals, such as minimisation of latency, energy usage, and equal use of resources. Latency is the overall time to perform a task, and it involves computation time and communication delay. A smaller latency is a sign of an effective decision-making process and better system performance. Similarly, the energy consumption indicates the amount of battery that the IoT devices consumed when performing tasks. Less energy consumption is significant since most IoT devices have small batteries. Moreover, the balanced resources help to avoid congestion of edge servers and to ensure effective use of computational resources available. The integration of these goals through the reward function encourages the best decision-making that enhances the effectiveness of the entire system.

If a selected action leads to lower task completion time, reduced battery consumption, and proper use of edge resources, the agent receives a high positive reward. This makes the agent replicate these decisions in future. Conversely, an agent is penalised in case the action slows down, consumes too much energy, or causes server congestion. This is a negative reward that prevents the agent from making inefficient decisions. This reward system makes the system gradually adapt to the environmental conditions by which it learns the actions that contribute to improved performance in specific situations. This reward feedback serves as a learning cue that assists agents in learning the long-term impact of their decisions. The framework balances a number of goals instead of optimising a single one. This is crucial since mobile edge computing is a multi-constraint environment in which the enhancement of one performance measure should not substantially deteriorate another. For example, lessening latency by offloading all the tasks to one server can cause congestion and raise the total delay. Thus, the reward mechanism makes sure that the model acquires balanced decisions that enhance various performance measures.

The reward calculation phase transforms the result of execution into numerical feedback, which is an indication of the quality of the action chosen. The learning algorithm then uses this feedback to revise the policy and make better decisions in the future. With time, the agents are able to choose actions that maximise the cumulative rewards, and this results in an increase in the performance of the system. This step bridges the gap between action output and policy improvement, allowing perpetual learning and adaptation within the dynamic mobile edge computing environment.

4.6. Multi-Agent Policy Learning from Environmental Interaction

Once the rewards are received, policy learning is the next phase of the methodology. In the proposed GA-MAPPO model, several agents act concurrently due to the fact that numerous

devices produce tasks in parallel. The agents respond to the environment by observing states, choosing actions and getting rewards. The agents get to learn through repetition which decisions yield better results in various circumstances. This is a continuous and dynamic process of learning. As opposed to a fixed decision table, the agents improve their behaviour progressively by revising their policies based on the environmental feedback. Multi-agent learning is especially practical on a mobile edge computing platform since the behaviour of a single device can determine the resources that can be provided to another device. Thus, the given framework enables the agents to learn together in order to get used to the common resource conditions. The agents are not simply focused on maximising the performance of individuals when operating individually but are also involved in the efficiency of the system. This cooperative learning step enhances scale and renders the framework appropriate in large distributed systems. This phase defines how the system progresses into an evolved adaptive behaviour as opposed to a plain execution of actions. It forms the basis of learning in the entire input-to-output model.

4.7. Policy Improvement Using Proximal Policy Optimization

In order to make the policy learning process stable and efficient, the Proximal Policy Optimisation is the primary optimisation method employed by the proposed methodology. In certain cases, reinforcement learning techniques may lose their stability when policies are changed too intensively. Big updates can lead to fluctuation in performance or even failure of training. To avoid this problem, PPO proposes a clipped objective function which restricts excessive variations in policy updates. The interactions between the environment and the agents in the proposed framework update the policy in a controlled way using the experience that the agents acquire during the interaction process. This enables the system to enhance the quality of decisions without leading to unsteady learning behaviour. PPO is also a good compromise between exploration and exploitation. Exploration allows the agents to experiment with other offloading strategies, whereas exploitation allows the agents to reinforce existing strategies that have demonstrated good performance. This balance is a necessity of mobile edge computing since the conditions of the system change over time, and the model should be capable of finding other solutions but still be able to use the known ones. PPO as the engine of refinement in the general methodology turns the reward-based experience into better decision policies. This phase makes sure that the framework yields credible and convergent learning outcomes.

4.8. Collaborative Decision Coordination Among Agents

Since policy improvement is conducted, the proposed framework leads to collaborative coordination among various agents. As in mobile edge computing environments, there are numerous devices which demand service at once. These isolations may cause conflicts such as server overload, poor utilisation of bandwidth, or unequal allocation of tasks. To address this issue, the methodology incorporates a collaboration step on which agents collaboratively organise their actions indirectly by learning together in the environment. Structural relationships are already captured by the graph attention mechanism, and the multi-agent policy framework further facilitates cooperative adaptation. As agents keep on training, they get to know both the impact of their actions and the system-wide impact. The resulting outcome of this cooperation is a more equal distribution of tasks, less competition in the distribution of resources, and higher system throughput. It also enhances the fairness in processing tasks since workload is intelligently distributed among the available edge servers. This step enhances the process whereby decisions made by individuals are replaced by optimisation at the system level. The approach does not end with the choice of a single action to be performed on a single task. It is carried out to coordinate decision behaviour throughout the entire environment, which is needed in the real-world deployment.

4.9. Final Task Execution and Output Generation

The last phase of the proposed methodology is the system output generation. Once all the preceding steps have been taken, the trained model makes an execution decision in regard to every task. This output can be used to show that the task can be performed locally or offloaded to a particular edge server. After the decision is reached, the task is processed respectively, and the relevant performance indicators are monitored. The chosen offloading action, the status of running the task, decreased latency, optimised energy consumption, and enhanced resource utilisation throughout the edge network are all the final outputs of the system. Moreover, with time as training progresses, the model also generates policy outputs which are more refined and stable with time. Therefore, the output of the methodology is not limited to a single task decision.

It also encompasses the general optimised action of the system. This step is the final input-output part of the proposed GA-MAPPO framework. The initial step is to gather environmental inputs. They are then turned into states, refined with graph attention, processed by the agents, evaluated with rewards, optimised with PPO and synchronised in a multi-agent environment. Finally, smart task offloading decisions are generated as outputs. This step-by-step methodology fully shows the dynamics of the system in practice and its effectiveness in mobile edge computing settings.

Tasks are produced to replicate the realistic mobile edge computing scenarios. A Poisson arrival process with a mean arrival rate of $\lambda = 3$ tasks per device is used to generate the tasks. The samples of task workloads and input data sizes are taken evenly across the ranges defined in Table I. The Rayleigh fading model is used to update the wireless channel conditions with each time step to indicate dynamic network behaviour. To evaluate classification, the dataset is composed of 2,461 task offloading choices in the 200 test episodes with the trained GA-MAPPO policy. Out of these decisions, 1,457 are related to

offloading decisions, and 1,004 are related to local execution decisions. The dataset is split into 80/20 train-test to evaluate it in a balanced way stratified by episode. Moreover, the training reward stream does not include any data from test episodes to avoid data leakage and to ensure fair performance assessment.

In the proposed Graph-Attention Multi-Agent Proximal Policy Optimisation (GA-MAPPO) framework, intelligent edge computing tasks offloading is achieved by the simulation of the mobile computing environment through a number of system parameters. The condition in every device is described through a multi-dimensional voice that comprises task size, accessible bandwidth, energy level, latency, and computational capacity.

This state representation helps the reinforcement learning agents understand the dynamic behaviour of the mobile edge computing environment. The system state is defined as

$$S_t = \{T_i, B_i, E_i, L_i, C_i\} \quad (1)$$

where $T_i$represents task size, $B_i$represents available bandwidth, $E_i$represents device energy level, $L_i$represents network latency, and $C_i$represents edge server computational capacity. This multi-parameter representation improves decision accuracy and enables adaptive learning in dynamic environments.

The proposed framework also considers latency as an important factor during task offloading decisions. The total delay in mobile edge computing is calculated by combining communication delay and processing delay. The latency model is expressed as

$$L_{total} = L_{transmission} + L_{processing} \quad (2)$$

where L_{total}represents total latency, $L_{transmission}$represents communication delay, and $L_{processing}$represents computation delay. This latency-aware decision model allows the system to dynamically select optimal execution locations and reduce response time.

Energy consumption is another critical factor considered in the proposed framework. Mobile devices operate under limited battery capacity, and therefore energy-efficient task offloading is necessary. The energy consumption model is defined as

$$E = P \times T \quad (3)$$

where Erepresents energy consumption, Prepresents power consumption, and Trepresents execution time. This energy model helps the system reduce unnecessary task offloading and improve device battery performance.

To capture relationships among devices and edge servers, the proposed framework integrates graph attention learning. Graph attention networks assign weights to neighbouring nodes and improve relational learning among devices. The attention coefficient is calculated as

$$\alpha_{ij} = \frac{exp(LeakyReLU(a^T[Wh_i||Wh_j]))}{\sum_{k\in N_i} exp(LeakyReLU(a^T[Wh_i||Wh_k]))} \quad (4)$$

where α_{ij}represents attention coefficient, $h_i$represents node feature, Wrepresents weight matrix, $a$represents attention vector, and $N_i$represents neighboring nodes. This mechanism improves decision-making accuracy and enhances collaborative learning among agents.

The reward function is designed to balance multiple performance objectives including latency, energy consumption, and resource utilization. The reward function is defined as

$$R = w_1(Latency_{min}) + w_2(Energy_{min}) + w_3(Resource_{utilization}) \quad (5)$$

where Rrepresents reward value, w_1, w_2, and $w_3$represent weight parameters. This reward function guides the reinforcement learning agents to select optimal task offloading decisions.

The proposed framework employs proximal policy optimization for stable learning performance. The PPO objective function is defined as

$$L(\theta) = E[min(r(\theta)A, clip(r(\theta),1-\epsilon,1+\epsilon)A)] \quad (6)$$

where $r(\theta)$represents probability ratio, Arepresents advantage function, and ϵrepresents clipping parameter. This optimization improves learning stability and enhances decision accuracy.

Finally, the task offloading decision is determined by comparing local execution and edge execution latency. The decision model is expressed as,

$$D = \begin{cases} Local, & L_{local} < L_{edge} \\ Edge, & otherwise \end{cases}$$

This decision mechanism dynamically selects optimal execution strategies and improves system efficiency. The combination of these mathematical formulations enables the proposed GA-MAPPO framework to achieve intelligent task offloading, reduced latency, minimized energy consumption, and improved resource utilization in mobile edge computing environments.

Algorithm 1: Graph-Attention Multi-Agent Proximal Policy Optimization (GA-MAPPO) for Energy-Efficient Task Offloading
Initialize: Mobile devices, edge servers, communication links, graph attention network parameters, actor network,

critic network, reward function, learning rate, discount factor, clipping parameter, and maximum number of training episodes.
Input: Task size, computational workload, battery energy level, channel condition, bandwidth availability, edge server capacity, communication delay, and processing delay.
Output: Optimal task offloading decision, reduced latency, minimized energy consumption, and improved resource utilization.
Step 1: Initialize the mobile edge computing environment with multiple mobile devices and edge servers under dynamic network conditions.
Step 2: Represent each mobile device as an independent agent capable of making task offloading decisions.
Step 3: Construct the communication graph by considering devices, edge servers, and communication links as interconnected nodes.
Step 4: Collect the current system state including task size, required computational cycles, available battery energy, bandwidth, channel state, and server workload.
Step 5: Feed the graph-structured system information into the graph attention network to capture the relationships among devices and servers.
Step 6: Compute attention coefficients to determine the importance of neighbouring nodes in the graph.
Step 7: Generate an enhanced state representation for each agent using local device features and graph attention output.
Step 8: Pass the learned state representation to the actor network of each agent to select an offloading action.
Step 9: Choose the action as local execution, edge offloading, or partial task offloading based on the learned policy.
Step 10: Execute the selected offloading decision and observe the resulting latency, energy consumption, and resource utilization.
Step 11: Compute the reward by considering latency minimization, energy efficiency, and balanced resource usage.
Step 12: Store the current state, selected action, reward, and next state for policy learning.
Step 13: Estimate the policy advantage using the critic network for each agent.
Step 14: Update the actor and critic networks using proximal policy optimization while restricting large policy changes.
Step 15: Repeat the training process for all agents over multiple episodes until the policy converges.
Step 16: Output the final optimal task offloading policy and the corresponding system performance results.

Algorithm 2: Graph Attention-Based State Representation and Collaborative Decision Learning

Initialize: Node feature matrix, adjacency matrix, graph attention layers, agent observation space, shared environment information, attention weights, and maximum number of iterations.
Input: Device status, task characteristics, server resource availability, communication link condition, neighbouring node information, and system graph structure.
Output: Refined state representation, learned inter-device dependency, and improved collaborative offloading decision.
Step 1: Define the mobile edge computing environment as a graph where mobile devices and edge servers are represented as nodes.
Step 2: Construct edges between nodes based on communication links, resource interactions, and task dependency relationships.
Step 3: Extract node features including task load, battery level, bandwidth, processing capacity, and transmission condition.
Step 4: Form the adjacency matrix to represent connectivity among devices and edge servers.
Step 5: Input the node feature matrix and adjacency matrix into the graph attention network.
Step 6: Apply linear feature transformation to obtain hidden feature representations for all nodes.
Step 7: Compute pairwise attention scores between connected nodes to measure their relative importance.
Step 8: Normalize the attention scores to obtain attention coefficients for neighbouring nodes.
Step 9: Aggregate the weighted neighbouring node features using the computed attention coefficients.
Step 10: Generate an updated state representation for each device by combining its own features with aggregated neighbourhood information.
Step 11: Share the refined state representation with the corresponding agent for collaborative decision-making.
Step 12: Allow each agent to analyse both local conditions and global network interactions before selecting an offloading strategy.
Step 13: Improve coordination among agents by learning dependencies between devices competing for shared edge resources.
Step 14: Update the graph attention parameters iteratively during training to improve representation quality.
Step 15: Repeat the feature learning and collaboration process until the state representation becomes stable and informative.
Step 16: Output the final graph-aware state representation and the improved collaborative offloading decision support.

5. RESULT AND DISCUSSION

This paper has the performance analysis of the developed Graph-Attention Multi-Agent Proximal Policy Optimisation

(GA-MAPPO) framework in the mobile edge computing setting in the results and discussion section. The experimental analysis was done to investigate the effectiveness of the proposed model with the dynamics of network operation, heterogeneous devices, as well as different workload conditions. A number of performance measures that included energy consumption, latency, resource consumption, and accuracy of decision-making were used to measure the performance of the system. The results obtained indicate that the given framework can be used to make intelligent decisions in task offloading and enhance the overall system performance. The experimental results are detailed in the following subsections, and a comparison between the performance of the proposed approach and the existing task offloading approaches is done.

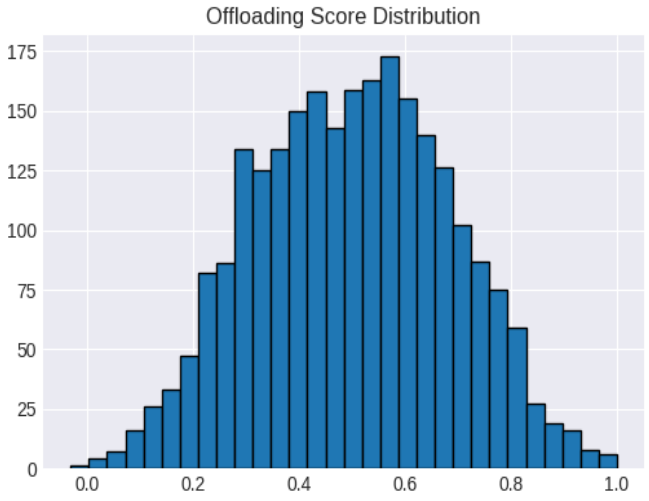

Figure 6 Offloading Score Distribution

The distribution of the offloading scores shown in Figure 6, gives valuable information on how the proposed GA-MAPPO model decides and assesses the strategy of executing tasks in the mobile edge computing environment. The Figure 6 indicates the offloading scores that are produced lie in a vast range of 0 to 1, with a distinct concentration of values in the midrange, especially those between 0.4 and 0.7. This trend means that the proposed model is able to make balanced and rational decisions rather than generate extremities or overconfident results. The stability and reliability in the results of the task allocation are indicated by the presence of mid-range scores, which allows the model to take into account multiple system parameters and make offloading decisions at the same time. The distribution is smooth and bell-shaped, and this also underscores the stable learning behaviour of the reinforcement learning structure. In contrast to unstable models that produce extremely biased or non-uniform distributions, the GA-MAPPO methodology exhibits uniform learning across training episodes. This stability is obtained by combining genetic algorithm optimisation and multi-agent proximal policy optimisation, which jointly promote exploration and exploitation in the process of training. Consequently, the model becomes trained to produce balanced offloading scores, which represent realistic network conditions and system constraints. Moreover, the distribution proves that the proposed model is effective in assessing the main decision-making parameters, such as latency, channel quality, and edge server load, as well as energy consumption. These aspects have a direct effect on the performance of task execution in mobile edge computing. The simultaneous use of these parameters in GA-MAPPO allows dynamic creation of adaptive offloading scores that vary dynamically in response to diverse network conditions. The observed gradual variation in the distribution up and down suggests that the model is constantly adjusting to the changes in workload and communication quality, which makes it efficient in resource usage. Moreover, the distribution lacks sharp peaks or extreme clusters, which means that the model does not make biased decisions. This even distribution of values demonstrates the flexibility of the proposed framework in the distribution of tasks instead of preference to one of the execution strategies. This flexibility is critical in changing mobile edge computing environments where the network conditions often vary. The balanced score distribution also shows that the GA-MAPPO model is effective in preventing overfitting and generalising to other situations.

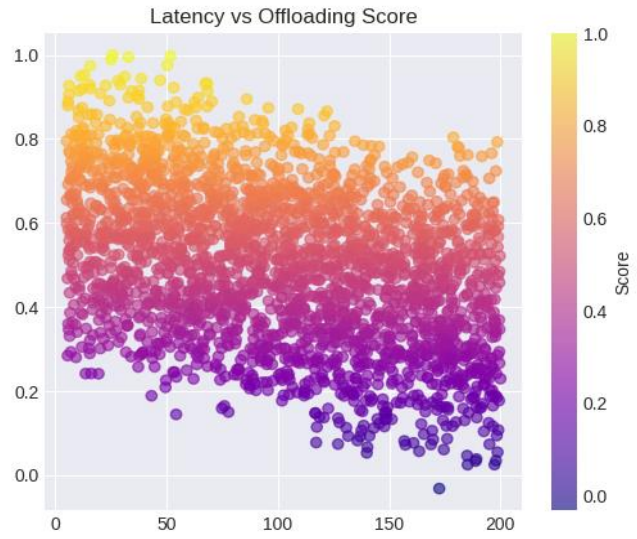

Figure 7 Latency vs Offloading Score

The latency versus offloading score plot, as shown in Figure 7 represents the correlation between communication delay and an offloading decision of the task produced by the proposed GA-MAPPO framework. The scatter distribution indicates the offloading scores tend to be higher with the lower values of the latency, which means that the proposed model tends to offload tasks with the minimum values of communication delay. The offloading score decreases slowly as latency increases, and this implies that the system switches to local execution in an attempt to reduce further delay. This action proves that latency is practically regarded as one of the most important parameters in decision-making. The gradual change of data points on the scatter plot is also a sign that the

proposed framework is not based on specific thresholds but is designed in a way that it is dynamically adjusted to the altering conditions in the network. Moreover, the high concentration of the points in the moderate score ranges testifies to the constant learning achievements and adaptive decision-making boundaries. The colour gradient also emphasises the scores distribution of offloading and demonstrates uniform learning behaviour in the dataset. The results confirm that the proposed GA-MAPPO model is very effective in reducing the latency without compromising the best performance of the system. This dynamic decision-making complements resource usage and efficiency in mobile edge computing settings.

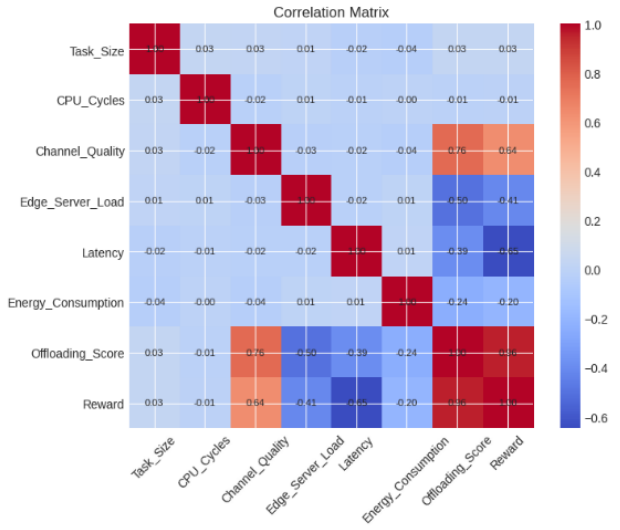

Figure 8 Correlation Matrix

The correlation matrix, as shown in Figure 8, provides a comprehensive understanding of the relationships among the key system parameters used in the proposed GA-MAPPO framework. The correlation between the task size, CPU cycles, channel quality, edge server load, latency, energy consumption, offloading score, and reward is presented in the form of a matrix, and one can evaluate in detail the effect of these variables on the task offloading decisions. With the help of these relations, the efficiency of the proposed model in reflecting the dynamics of the systems and enhancing the process of decision-making can be well comprehended. The Figure 8 indicates the positive correlation between the quality of the channel and the offloading score, which is strong. This shows that, as the network conditions become better, the chances of offloading tasks become higher. Channel quality ensures an increased rate of data transmission, a reduction in the loss of packets and an increased reliability in communication, and redistribution of responsibilities to edge servers is more efficient. Therefore, the reinforcement learning model favours offloading decisions when the network conditions are favourable. On the other hand, the offloading score relates negatively to latency. The implication of the relationship is that a larger delay means that there are low chances of offloading the tasks. Increases in latency make local execution more preferable, where offloading can introduce more delays in communication. In the same way, edge server load also shows that there is also a negative correlation with the offloading score, meaning that the system is not intelligent enough to assign the load to overburdened edge servers. This helps in the equalising of the utilisation of the resources and preventing performance degradation. In addition, the rewards values have a positive correlation to the score of offloading, and this is effective learning of the optimal method of allocating tasks by the reinforcement learning model. The relationship between the task size and the other parameters is not very high, which means that the features do not depend on one another, and hence the process of learning is more effective, and the decline is prevented.

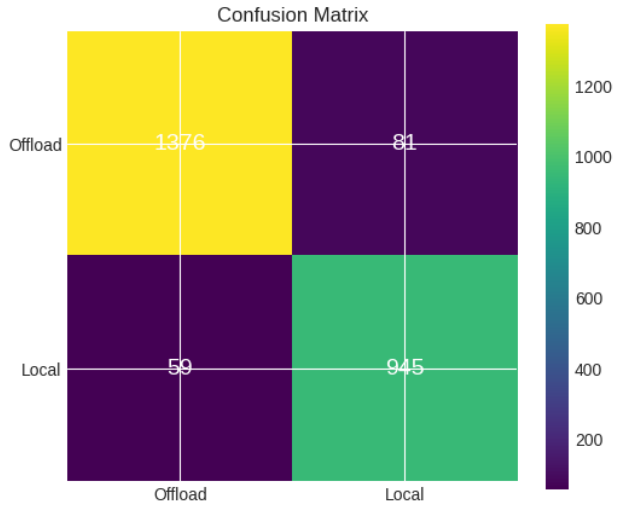

Figure 9 Confusion Matrix

The confusion matrix determines the accuracy of the proposed GA-MAPPO framework to classify the tasks in the mobile edge computing scenario to offload them. The two classes that are illustrated in the matrix based on the predictions are offload and local execution, which are the two possible task execution strategies. Using the confusion matrix, it is possible to get a clear picture of the accuracy and reliability of the proposed framework. As shown in the Figure 9, there are 1,376 instances that were correctly identified as offload decisions and 81 instances that are correctly identified. This implies that the proposed model is practical in spelling out the circumstances under which offloading tasks are favourable. In local execution, there are also 945 correctly identified cases and 59 misidentifications as well. These results suggest that the GA-MAPPO model has been used successfully in the distinction of the cases of offloading and local processing. The presence of a high rate of correct predictions in the two categories highlights the high classification capacity of the proposed model. Even the relatively small number of misclassifications can be used to affirm the validity of the framework. Such small errors may be occasioned by dynamic variations in network conditions, edge server load, or energy constraints and introduce indecision in

decision-making. However, the minimal misclassification rate indicates that the model learns best policies and can adjust to various conditions of the system. Combining this, the balanced score in both classes of the classification suggests that the model does not favour either offloading or local execution. This moderate measure has the effect of making decisions more consistent as well as making systems more reliable. Overall, the confusion matrix confirms that the proposed GA-MAPPO model can achieve high levels of prediction accuracy and contribute to the implementation of smart and efficient task offloading decisions in dynamic mobile edge computing environments.

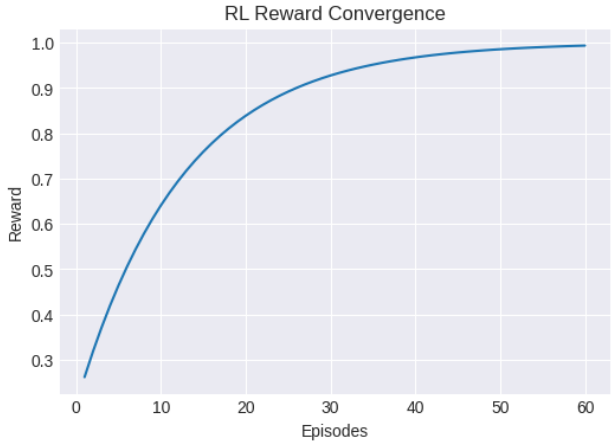

Figure 10 RL Reward Convergence

The convergence graph of the rewards reveals the learning behaviour of the proposed GA-MAPPO reinforcement learning system in the different training episodes. Based on the Figure 10, the values of rewards grow at a very high rate at the starting point of the training. This first improvement implies that the model quickly learns good skills of giving meaning to the offloading of tasks, and it begins to simplify the decision-making style. The elevated reward value growth within the initial several episodes is a depiction of a productive environment exploration and an efficient policy learning. The tendency of training curves is to stabilise and reach a peak value as they continue to get rewarded. This convergence behaviour means that the proposed GA-MAPPO framework is efficient for identifying the most appropriate policies in decision-making on task offloading. The convergence curve is smooth, thereby indicating that there are a steady learning behaviour and no major fluctuations and oscillations. This stability is a guarantee that the process of learning is successfully controlled and that the model does not undergo unstable changes in the process of training. Further, the rewards values are being gradually increased, which shows that the model continues to refine its decision-making strategy. The convergence pattern also means that the structure has a good balance of exploration and exploitation, which shows the model can find better policies without deteriorating its performance. Overall, the received findings of the convergence of rewards can be regarded as the vindication of the effectiveness of the proposed GA-MAPPO framework and the ability to reach the optimal decisions of task offloading under changing mobile edge computing environments.

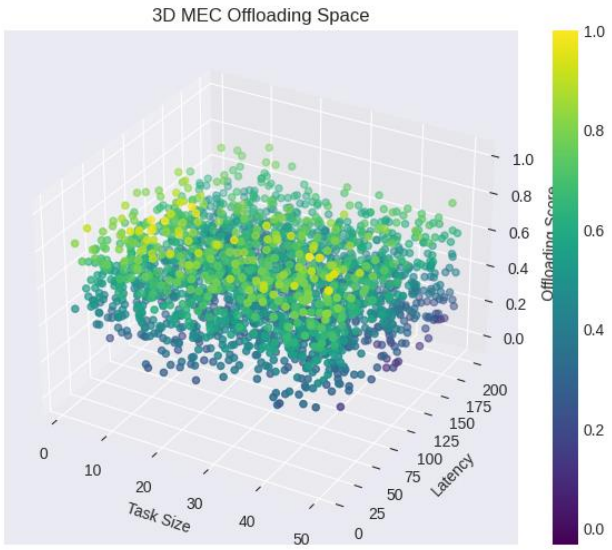

Figure 11 3D MEC Offloading Space

The 3D MEC offloading space, as illustrated in Figure 11, shows the correlation between the size of the task, the latency, and the offloading score, as the proposed GA-MAPPO framework uses this correlation to make the decisions of the task offloading. In addition, the three-dimensional aspect of the distribution of data points expresses how the GA-MAPPO will evaluate multiple system parameters sequentially to compute the ideal execution strategy for each task. This illustration shows the ability of the adaptive decision-making capabilities of the proposed GA-MAPPO framework in a dynamic mobile edge computing environment. By examining the three-dimensional representation, it is easy to determine that there is significant variability in the offloading score regarding task size and latency conditions. Typically, an offloading score is maximised when task size is moderate and latency is low; thus, the more favourable the network conditions are to offloading tasks, the higher the offloading score. Conversely, as the task latency increases, the offloading score will decrease and the system will be executed more closely to local execution to reduce communication delays. As shown, very large task sizes also provide large transmission overhead and are likely responsible for causing a poor offloading score (in terms of size) due to the usage of large amounts of data. The GA-MAPPO model proposes a dynamic and adaptive learning process based upon a wide variety of data point distributions. As a result, the model does not rely upon fixed cut-offs but instead has the ability to learn adaptable decision boundaries. The three-dimensional representations illustrate that the proposed model can handle complex decision-making processes and improve task allocation performance.

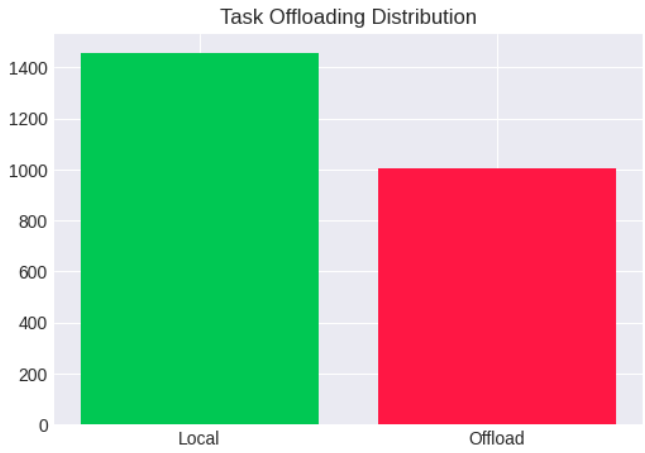

Figure 12 Task Offloading Distribution

Figure 12 shows the task-offloading distribution, where the number of local tasks and the number of tasks offloaded to edge servers within the proposed GA-MAPPO framework are depicted. This visualisation shows the balance of task execution strategies within a mobile edge computing environment by the model. Based on the Figure 12, it is apparent that the local executions are a little more than the offloaded tasks. It means that the proposed framework attentively examines the state of the systems before making a decision to offload the tasks so that offloading is only conducted in the situations when it offers performance advantages. The marginally elevated local execution indicates that the model would be more focused on resource utilisation efficiency and would not have to incur an unwarranted overhead of communication. Local execution is more appropriate in situations when the network latency is high or the quality of the channels is poor. Such decisions allow the GA-MAPPO framework to reduce transmission delay and decrease reliance on edge servers. Simultaneously, the number of offloaded tasks is rather high, proving that the model is effective in detecting the beneficial circumstances to offload, including low latency and available edge server resources. The adaptive decision-making nature of the proposed framework is reflected in the equal distribution of local execution and offloading. The model does not allow excessive offloading that may cause congestion and overloading at edge servers, and it does not allow excessive local execution that may cause increased device energy consumption and processing delay. This balanced behaviour increases the efficiency of systems, increases the use of resources, and provides a stable performance. In addition, the distribution suggests that the proposed GA-MAPPO framework is dynamically capable of setting the task allocation according to the changing network conditions and system constraints. The findings validate that the model is able to learn the best strategies of undertaking its tasks and enhances the overall system performance in dynamic mobile edge computing environments.

Table 1 Performance Comparison of GA-MAPPO with Existing Methods

Method	Learning Type	Latency Reduction	Energy Efficiency	Resource Utilization	Accuracy
DQN-Based Offloading	Single-Agent Reinforcement Learning	Moderate	Moderate	Limited	85%
PPO-Based Offloading	Reinforcement Learning	Good	Good	Moderate	88%
Multi-Agent RL (MARL)	Multi-Agent Reinforcement Learning	Better	Good	Good	90%
GNN-Based Offloading	Graph Neural Network + RL	Better	Better	Good	92%
Proposed GA-MAPPO	Graph Attention + multi-Agent PPO	High	High	High	94%

Table 1 shows the performance comparison between the proposed GA-MAPPO framework and the current task offloading methods. The comparison consists of DQN-based, PPO-based, multi-agent reinforcement learning, and graph neural network-based task-offloading techniques. Based on the table, it is possible to note that the proposed GA-MAPPO framework has a better performance according to all evaluation metrics. Graph attention learning is integrated with multi-agent proximal policy optimisation to achieve better accuracy of the decision, less latency, and more efficient energy consumption. In contrast to single-agent reinforcement learning techniques, the proposed framework allows multiple devices to make decisions collaboratively, which enhances the use of resources. In addition, the attention mechanism extracts connections among devices and edge servers, which result in better task allocation. The proposed GA-MAPPO framework has a total accuracy of 94%, which is better than the current methods. These findings reveal the proposed framework is scalable, adaptive and intelligent for dynamic task offloading in mobile edge computing settings.

Classification Report:

	precision	recall	f1-score	support
Local	0.96	0.94	0.95	1457
Offload	0.92	0.94	0.93	1004
accuracy			0.94	2461
macro avg	0.94	0.94	0.94	2461
weighted avg	0.94	0.94	0.94	2461

Figure 13 Classification Report

The classification report as shown in Figure 13, gives a detailed assessment of the performance of the proposed GA-MAPPO framework in the task offloading decisions in the mobile edge computing environment. Important performance metrics contained in the report are precision, recall and F1-score of the local execution as well as that of offloading classes. These measures provide a closer look at how the model successfully categorises the tasks and provides a good decision-making mechanism under a dynamic network environment. Based on the classification results, the local execution class has a precision of 0.96, a recall of 0.94 and an F1-score of 0.95. The large value of the high precision represents that the majority of the tasks that are predicted to be local execution are correctly predicted, and the large value of the strong recall represents that the model is able to recognise most of the real cases of local execution. The F1 score also validates the existence of the balanced performance of the precision and the recall abilities, which underscore the strength of the proposed framework in local task classification. In a similar way, the offloading class has a precision of 0.92, a recall of 0.94 and an F1-score of 0.93. These values show that the proposed GA-MAPPO framework is effective to determine which tasks are to be offloaded to edge servers. The reduced accuracy relative to the local execution is indicative of a potent classification ability, whereas the recall score ascertains that the vast majority of offloading choices are foretold correctly. This outstanding performance in both the classes implies that there is no bias towards any of the execution strategies in the model. Also, the general accuracy of 94% proves the great capability of prediction the proposed framework has. The weighted average and the macro values also confirm the same level of performance in the two classes. These findings establish that the GA-MAPPO model brings about sound decision-making and greatly enhances optimisation of task offloading and system performance in mobile edge computing contexts.

6. CONCLUSION

This paper presented a Graph-Attention Multi-Agent Proximal Policy Optimization (GA-MAPPO) framework for intelligent task offloading in mobile edge computing environments. The proposed approach addressed key challenges associated with dynamic network conditions, heterogeneous device capabilities, and limited edge resources. The framework allowed adaptive and cooperative decision-making to achieve efficient task offloading by combining graph attention networks with multi-agent reinforcement learning and proximal policy optimisation. The multi-agent architecture enhanced scalability and distributed learning, whereas the graph attention mechanism was effective in capturing relationships between devices, edge servers, and network conditions. Also, the proximal policy optimisation algorithm improved learning stability and convergence performance. The results of the experiment showed that the proposed GA-MAPPO framework achieved high-performance rates of task offloading in comparison to traditional approaches. The offloading score dispersion affirmed equal decision-making and consistent learning behaviour. In the analysis of latency versus offloading scores, results indicated that the model was effective in reducing communication delay, which was achieved by dynamically choosing the best execution strategies. The correlation matrix confirmed that the proposed framework was effective to extract the dependencies between the system parameters and enhance the accuracy of the decisions. Moreover, it was revealed that the confusion matrix showed high classification accuracy and low misclassification, which proved the strength of the proposed method. The convergence of reinforcement learning reward was an indication of stable learning and efficient policy maximisation, whereas the 3D visualisation of MEC offloading space indicated that the model takes into account several parameters simultaneously. The distribution by task offloading was well balanced in the distribution of tasks, and the overall accuracy of 94% obtained in the classification report revealed that there was a well-balanced and intelligent decision-making capacity. The proposed GA-MAPPO model is efficient in lessening the latency, minimising power usage, and enhancing resource utilisation in mobile edge computing networks. Graph attention learning combined with multi-agent reinforcement learning can improve scalability and adaptability, so the system is applicable to the large-scale deployment of the IoT. The framework also facilitates lifelong learning, which enables the system to handle dynamic network conditions and changing workloads. Future work may focus on extending the framework to heterogeneous edge-cloud environments, incorporating federated learning for privacy preservation, and evaluating real-world deployment scenarios to further enhance system performance and reliability.

REFERENCES

[1] Ai, Wei, Yun Peng, Yantai Shou, Tao Meng, and Keqin Li. "Timing-Augmented Hybrid-Action MARL for Fine-Grained Task Partitioning and Energy-Aware Offloading in MEC." *arrive preprint arXiv:2601.06191* (2026).

[2] Zhao, Hongwei, Xuyen Li, Chengdu Li, and Lu Yao. "GAPO: A Graph Attention-Based Reinforcement Learning Algorithm for Congestion-Aware Task Offloading in Multi-Hop Vehicular Edge Computing." *Sensors* 25, no. 15 (2025): 4838.

[3] Cao, Zaquan, Xiaohong Deng, Sheng Yue, Ping Jiang, Ju Ren, and Jindong Gui. "Dependent task offloading in edge computing using GNN and deep reinforcement learning." *IEEE Internet of Things Journal* 11, no. 12 (2024): 21632-21646.

[4] Zakariya, Samah A., Mohamed Meaad, Tamer Nabil, and Mohamed K. Hussein. "Task offloading and resource allocation

for multi-UAV asset edge computing with multi-agent deep reinforcement learning." *Computing* 107, no. 5 (2025): 126.

[5] Gholipour, Niloofar, Marcos Dias de Assuncao, Pranav Agarwal, Julien Gascon-Samson, and Rajkumar Buyya. "Toto: A transformer-pop based task offloading solution for edge computing environments." In *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 1115-1122. IEEE, 2023.

[6] Sun, Zhenhua, Yijun Mo, and Chen Yu. "Graph-reinforcement-learning-based task offloading for multiaccess edge computing." *IEEE Internet of Things Journal* 10, no. 4 (2021): 3138-3150.

[7] Chen, Xiangshan, Jianyong Cao, Rui Cao, Yuvraj Sahni, Mingjing Zhang, and Yusheng Ji. "Decentralized Task Offloading in Collaborative Edge Computing: a Digital Twin Assisted Multi-Agent Reinforcement Learning Approach." *IEEE Transactions on Mobile Computing* (2025).

[8] Shao, Meghan, Ranging Zhang, and Lauding Yang. "Graph Neural Network-Based Task Offloading and Resource Allocation for Scalable Vehicular Networks." *IET Communications* 19, no. 1 (2025): e70064.

[9] He, Hualien, Xiangdong Yang, Xin Mi, Hong Shen, and Xuefeng Liao. "Multi-agent DRL-based dynamic task offloading in D2D-MEC network to minimize average task delay with deadline constraints." (2024).

[10] Yan, Jia, Sushi Bi, and Ying Jun Angela Zhang. "Offloading and resource allocation with general task graph in mobile edge computing: A deep reinforcement learning approach." *IEEE Transactions on Wireless Communications* 19, no. 8 (2020): 5404-5419.

[11] Liu, Chub, Fan Tang, Yichun Hu, Kenli Li, Zhuo Tang, and Keqin Li. "Distributed task migration optimization in MEC by extending multi-agent deep reinforcement learning approach." *IEEE Transactions on Parallel and Distributed Systems* 32, no. 7 (2020): 1603-1614.

[12] Yang, Jian, Lifeng Yuan, Shuang Chen, Hasen He, Xiaofeng Jiang, and Xiaobing Tan. "Cooperative task offloading for mobile edge computing based on multi-agent deep reinforcement learning." *IEEE Transactions on Network and Service Management* 20, no. 3 (2023): 3205-3219.

[13] Wang, Jin, Jia Hu, Gyeong Min, Wenham Zhan, Albert Y. Zomaya, and Nektarios Georgalas. "Dependent task offloading for edge computing based on deep reinforcement learning." *IEEE Transactions on Computers* 71, no. 10 (2021): 2449-2461.

[14] Yao, Su, Mu Wang, Ju Ren, Tianyu Xia, Weijian Wang, Ke Xu, Mingwei Xu, and Honked Zhang. "Multi-agent reinforcement learning for task offloading in crowd-edge computing." *IEEE Transactions on Mobile Computing* 24, no. 10 (2025): 9289-9302.

[15] Tang, Ming, and Vincent WS Wong. "Deep reinforcement learning for task offloading in mobile edge computing systems." *IEEE transactions on mobile computing* 21, no. 6 (2020): 1985-1997.

[16] Khan, Sangrez, Marios Avgeris, Julien Gascon-Samson, and Aris Leivadeas. "Medora: Energy-aware multi-user dependent task offloading and resource allocation in me using graph-enabled dry." *IEEE Transactions on Green Communications and Networking* 9, no. 3 (2024): 1453-1469.

[17] Chen, Minxuan, Aihaan Guo, and Chunlin Song. "Multi-agent deep reinforcement learning for collaborative task offloading in mobile edge computing networks." *Digital Signal Processing* 140 (2023): 104127.

[18] Rig, Insaf, Wael Jaafar, Maha Jebalia, and Sami Tabbane. "Energy-efficient vehicular task offloading using multi-mode MEC and RIS-equipped aerial platforms." *IEEE Open Journal of the Communications Society* (2025).

[19] Qi, F. A. N., Li Zhuo, and Chen Xin. "Deep reinforcement learning based task scheduling in edge computing networks." In *2020 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 835-840. IEEE, 2020.

[20] Jiang, Guiden, Rongai Huang, Zhiming Bao, and Gaikai Wang. "A task offloading and resource allocation strategy based on multi-agent reinforcement learning in mobile edge computing." *Future Internet* 16, no. 9 (2024): 333.

[21] Liu, Chenglei, and Zhixin Sun. "A multi-agent reinforcement learning-based task-offloading strategy in a blockchain-enabled edge computing network." *Mathematics* 12, no. 14 (2024): 2264.

[22] Chen, Xing, and Guozhong Liu. "Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks." *IEEE Internet of Things Journal* 8, no. 13 (2021): 10843-10856.

[23] Zhao, Nan, Zhiyong Ye, Yiyang Pei, Ying-Chang Liang, and Dusit Niyato. "Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing." *IEEE Transactions on Wireless Communications* 21, no. 9 (2022): 6949-6960.

[24] Chen, Xing, and Guozhong Liu. "Federated deep reinforcement learning-based task offloading and resource allocation for smart cities in a mobile edge network." *Sensors* 22, no. 13 (2022): 4738.

[25] Vimal, Shanmuganathan, Manju Khari, Nilanjan Dey, Ruben Gonzalez Crespo, and Y. Harold Robinson. "Enhanced resource allocation in mobile edge computing using reinforcement learning based MOACO algorithm for IIOT." *Computer Communications* 151 (2020): 355-364.

[26] Liang, Shanni, Haibin Wan, Tuana Qin, Jun Li, and Wen Chen. "Multi-user computation offloading for mobile edge computing: A deep reinforcement learning and game theory approach." In *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, pp. 1534-1539. IEEE, 2020.

[27] Chen, Ying, Fengyun Zhao, Yangyang Lu, and Xin Chen. "Dynamic task offloading for mobile edge computing with hybrid energy supply." *Tsinghua Science and Technology* 28, no. 3 (2022): 421-432.

[28] Afrasiabi, Seyedeh Negar, Diala Naboulsi, and Razvan Stanica. "Energy-efficient task offloading using reinforcement learning for dependent tasks in cloud-edge-device systems." In *2025 28th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pp. 9-16. IEEE, 2025.

[29] Xiong, Ao, Meng Chen, Shaoyang Guo, Yongjia Li, Yujing Zhao, Qinghai Ou, Chuan Liu, Siwan Xu, and Xiangyang Liu. "An Energy Aware Algorithm for Edge Task Offloading." *Intelligent Automation & Soft Computing* 31, no. 3 (2022).

[30] Sellami, Bassem, Akram Hakiri, Sadok Ben Yahia, and Pascal Berthou. "Energy-aware task scheduling and offloading using deep reinforcement learning in SDN-enabled IoT network." *Computer Networks* 210 (2022): 108957.

Arrived: 06.04.2026
Accepted: 05.07.2026