

Computation offloading strategy for LEO satellite edge computing based on graph attention imitation learning

Guisong Yang^{1,2}, Han Dai¹, Panxing Hang², Xingyu He^{1}*

¹University of Shanghai for Science and Technology, Shanghai, China

²National Key Laboratory of Space Intelligent Control, Beijing, China

*Corresponding Author. Email: sherri_he@163.com

Abstract. To address the issue of Quality of Service (QoS) degradation for mission-critical tasks in Low Earth Orbit (LEO) satellite edge computing under extreme congestion, this paper proposes an intelligent computation offloading architecture based on graph attention imitation learning. The proposed method models the satellite network as a time-varying graph to extract global topological features and innovatively introduces a QoS-aware action masking mechanism to forcibly preclude suboptimal decisions through strict physical constraints. Simulation results demonstrate that the proposed algorithm significantly reduces both the network-wide average latency and the 95th percentile tail latency. Furthermore, it achieves the highest success rate for high-priority tasks under heavy-load conditions. Ultimately, this architecture effectively overcomes the "herd effect" among local nodes, realizing superior global load balancing and absolute QoS guarantees for critical tasks within dynamic satellite networks.

Keywords: satellite edge computing, computation offloading, imitation learning, action masking

1. Introduction

With the rapid development of Sixth-Generation (6G) communication technology and Satellite-Terrestrial Integrated Networks (STIN), Low Earth Orbit (LEO) satellite communication networks have emerged as the core infrastructure for eliminating communication blind spots in remote and maritime regions, owing to their broad coverage, low latency, and robust survivability [1, 2]. Recently, inspired by Mobile Edge Computing (MEC) technology, Satellite Edge Computing (SEC) has been proposed. By equipping LEO satellites with computational capabilities, Internet of Things (IoT) devices and terrestrial terminals can directly offload computation-intensive tasks (such as remote sensing image processing and disaster early warning) to the orbital edge for real-time execution [3, 4]. This paradigm substantially alleviates the transmission bandwidth pressure on satellite-terrestrial links and significantly reduces end-to-end service latency [5]. Nevertheless, due to the high mobility of LEO satellites and the extreme constraints on onboard computing and storage resources, achieving the efficient offloading and collaborative scheduling of massive heterogeneous tasks within a highly dynamic and resource-scarce network topology remains a critical challenge that urgently needs to be addressed [6].

Most existing satellite computation offloading schemes rely on traditional heuristic algorithms or static optimization rules (e.g., greedy algorithms, random allocation). Although these methods are simple to deploy, they are highly susceptible to the "herd effect" when confronted with sudden traffic bursts, leading to local node resource depletion and severe queuing tail effects [7, 8]. To cope with dynamic and time-varying system states, Deep Reinforcement Learning (DRL) has been widely introduced in academia in recent years to optimize offloading decisions, such as utilizing Deep Deterministic Policy Gradient (DDPG) or Multi-Agent Reinforcement Learning (MARL) to handle mixed continuous and discrete action spaces [9, 10]. However, most existing intelligent scheduling strategies focus on a single objective, such as maximizing total system throughput or minimizing average latency [11]. Under extreme conditions of severe network congestion, they inevitably sacrifice latency-sensitive, high-priority critical tasks, thereby failing to meet the security requirements of highly reliable satellite communications [12].

To address complex and variable network topologies and overcome the bottleneck of slow convergence of traditional reinforcement learning in large state spaces, Graph Neural Networks (GNN) and Imitation Learning (IL) have begun to demonstrate tremendous potential in the field of computation offloading [13, 14]. GNNs can effectively extract high-order spatial features among nodes, while imitation learning significantly accelerates the convergence and online deployment of decision models through behavioral cloning [15]. To thoroughly resolve the issue of QoS degradation for critical tasks in sudden congestion scenarios, this paper proposes an intelligent computation offloading architecture based on Graph Attention Network and Imitation Learning (GAT-IL). First, we model the dynamic connections of the LEO satellite constellation as a time-varying graph, leveraging the powerful spatial feature aggregation capability of GAT to accurately capture the global topological structure and the distribution status of node computing resources. More crucially, to address the issues of "decision ambiguity" and "survivorship bias" inherent in pure imitation learning under extreme congestion scenarios, this paper innovatively introduces a QoS-Aware Action Masking mechanism. This mechanism establishes strict physical boundary constraints for high-priority tasks and directly truncates suboptimal decisions that cause long queues during the model's forward inference phase. This forcibly guides traffic to achieve global load balancing, thereby fundamentally eliminating the long-tail queuing phenomenon.

The main contributions of this paper are summarized as follows:

A computation offloading model for multi-priority tasks in dynamic LEO satellite networks is constructed, comprehensively considering inter-satellite link latency, onboard computational capacity constraints, and the strict tolerance time boundaries of heterogeneous tasks.

A GAT-IL intelligent offloading algorithm integrating global topology awareness is proposed. Furthermore, a QoS-aware action masking mechanism is innovatively designed, endowing the agent with the "hard constraint" capability to prioritize the survival rate of critical tasks under extreme congestion states.

Extensive simulation experiments are conducted, demonstrating that compared with traditional heuristic baseline algorithms and greedy algorithms, the proposed GAT-IL algorithm not only effectively reduces the average end-to-end latency of the system but also optimizes the success rate of high-priority tasks under extreme traffic pressure, exhibiting outstanding robustness and load-balancing capabilities.

2. System model and problem formulation

2.1. System model and problem formulation

This paper considers a Low Earth Orbit (LEO) satellite edge computing network composed of multiple orbital planes. The network adopts the Software-Defined Networking (SDN) paradigm for distributed management. In the time domain, due to the high-speed motion of the satellites, the inter-satellite physical distances and

topological connections are highly time-varying. Therefore, we discretize the system's operating time into a series of equal-length time windows $T_{win} = \{t_1, t_2, \dots, t_M\}$. Within any given time window $t \in T_{win}$, the satellite network topology can be modeled as a dynamic directed graph $G(t) = (V, E(t))$, where V represents the set of satellite nodes, and $E(t)$ denotes the set of Inter-Satellite Links (ISLs) in a connected state.

Regarding the control architecture, the nodes are divided into control satellites and computation satellites. Let $c \in V$ be a specific control node. At time t , the set of computation satellites scheduled and covered by this node is defined as the control domain $N_c(t)$. Computation-intensive and latency-sensitive tasks (e.g., real-time remote sensing analysis and emergency command processing) generated by terrestrial Internet of Things (IoT) terminals are first uploaded to the controller c via Ground-to-Satellite Links (GSLs). Acting as a regional brain, the controller c must designate the optimal offloading node $k \in N_c(t)$ for each task within an extremely short decision window, based on the intra-domain graph topology state. To focus on the core challenges of distributed inter-satellite computation offloading, this paper assumes that the data volume of the computation results is negligible, and therefore the latency incurred by downloading them back to the ground is omitted.

2.2. Communication and computation model

Assume that at time t , the control node c receives a set of pending tasks $U(t)$. For any task $u_i \in U(t)$, its heterogeneous characteristics are uniquely characterized by a quadruplet: $u_i \triangleq \{S_i, P_i, T_{tol,i}, L_i\}$. Here, S_i denotes the input data size of the task; $P_i \in \{1, 2, 3\}$ represents the priority label (where a higher value indicates a higher priority and greater sensitivity to latency); $T_{tol,i}$ is the maximum tolerance time of the task, beyond which the task is considered failed; and L_i denotes the geographical location information where the task was generated.

When the controller c decides to offload task u_i to a computation node k , its total end-to-end lifecycle latency $T_{total}^{i,k}$ consists of three tightly coupled physical stages: inter-satellite communication latency, dynamic queuing latency, and computation processing latency.

2.2.1. Inter-satellite communication latency

Based on queuing theory and electromagnetic wave propagation theory, the communication latency $T_{trans}^{i,k}$ for transmitting task data from the controller c to node k comprises the transmission delay, spatial propagation delay, and communication interface processing and queuing delay. The computation model can be formalized as follows:

$$T_{trans}^{i,k} = \frac{S_i}{B_{c,k}} + \frac{d_{c,k}(t)}{v_0} + \frac{Q_{comm}^c(t) + S_i}{C_{comm}} \quad (1)$$

where $B_{c,k}$ denotes the allocated Inter-Satellite Link (ISL) transmission bandwidth between the controller c and node k ; $d_{c,k}(t) = |L_c(t) - L_k(t)|_2$ represents their three-dimensional Euclidean distance at time t ; and v_0 is the speed of light in a vacuum. Furthermore, $Q_{comm}^c(t)$ denotes the volume of backlogged data currently in the controller's communication transmission queue, and C_{comm} represents the maximum processing rate of the communication network interface card.

2.2.2. Priority-aware computation processing latency

Upon the complete arrival of the data for task u_i at computation node k , node k performs an asymmetric allocation of computational resources based on the task's priority P_i . Let F_k denote the base computational capacity of node k , and the computing power allocation ratio function for tasks with different priorities is defined as $\rho(P_i)$:

$$\rho(P_i) = \begin{cases} 0.85, & \text{if } P_i = 3, \\ 0.65, & \text{if } P_i = 2, \\ 0.45, & \text{if } P_i = 1 \end{cases} \quad (2)$$

Therefore, the actual processing latency $T_{proc}^{i,k}$ of task u_i at node k is formulated as:

$$T_{proc}^{i,k} = \frac{S_i \cdot \omega}{\rho(P_i) \cdot F_k} \quad (3)$$

where ω denotes the computational intensity parameter required to process a single bit of data. This priority-based skewed computing power allocation mechanism ensures that urgent tasks can complete their physical execution within a shorter cycle.

2.2.3. Dynamic queuing latency

In scenarios characterized by extremely limited onboard computational capacity and dense task concurrency, queuing latency constitutes the core bottleneck leading to task timeouts. Let the absolute timestamp of the arrival of task u_i at the target node k be $t_{arr}^{i,k} = t + T_{trans}^{i,k}$. The task must wait in the task queue of node k until all preceding tasks have finished execution.

Let T_{end}^k denote the expected absolute completion time of the last task in the queue of node k . Then, the queuing latency $T_{queue}^{i,k}$ experienced by task u_i can be recursively calculated using the following equation:

$$T_{queue}^{i,k} = \max(0, T_{end}^k - t_{arr}^{i,k}) \quad (4)$$

Subsequently, the queue state of node k is synchronously updated in preparation for the arrival of the next task:

$$T_{end}^k \leftarrow t_{arr}^{i,k} + T_{queue}^{i,k} + T_{proc}^{i,k} \quad (5)$$

In summary, the total end-to-end latency incurred by offloading task u_i to computation node k is given by:

$$T_{total}^{i,k} = T_{trans}^{i,k} + T_{queue}^{i,k} + T_{proc}^{i,k} \quad (6)$$

2.2.4. Optimization problem formulation

In LEO satellite edge computing networks, the core objective of a control node is to determine the optimal computation offloading strategy for massive heterogeneous tasks within an extremely short decision window. To formulate this physical process into a rigorous mathematical optimization problem, we first define the system states and decision variables.

For the set of tasks $U(t)$ arriving at the control node c at time t , and the set of available computation nodes $N_c(t)$ within its control domain, a two-dimensional binary decision matrix $X = \{x_{i,k} \mid u_i \in U(t), k \in N_c(t)\}$ is defined. Here, the element $x_{i,k}$ denotes the atomic action of task allocation:

$$x_{i,k} = \begin{cases} 1, & \text{if task } u_i \text{ is offloaded to node } k \text{ for execution,} \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

The optimization objective of this paper is to achieve global network load balancing under extreme conditions of bursty traffic peaks, thereby minimizing the average end-to-end latency of the system; simultaneously, mandatory Quality of Service (QoS) guarantees must be provided for high-priority tasks. Traditional heuristic scheduling approaches often sacrifice a fraction of long-tail tasks in pursuit of minimizing the global latency, leading to queuing timeouts for high-priority tasks. To avoid this "herd effect," we integrate the task lifecycle latency with the maximum tolerance boundary.

The global optimization objective function of the system is defined to minimize the average processing cost across all tasks:

$$\frac{1}{|U(t)|} \sum_{u_i \in U(t)} \sum_{k \in N_c(t)} x_{i,k} T_{total}^{i,k} \quad (8)$$

Combining the communication model, the computation model, and the QoS requirements, the dynamic graph topology-aware task offloading optimization problem $P1$ can be rigorously formulated as:

$$(P1): \quad \frac{1}{|U(t)|} \sum_{u_i \in U(t)} \sum_{k \in N_c(t)} x_{i,k} T_{total}^{i,k} \quad (9)$$

$$s.t. \quad x_{i,k} \in \{0, 1\}, \quad \forall u_i \in U(t), \forall k \in N_c(t) \quad (10)$$

$$\sum_{k \in N_c(t)} x_{i,k} = 1, \quad \forall u_i \in U(t) \quad (11)$$

$$\sum_{u_i \in U(t)} x_{i,k} \cdot \rho(P_i) F_k \leq F_k^{max}, \quad \forall k \in N_c(t) \quad (12)$$

$$x_{i,k} T_{total}^{i,k} \leq T_{tol,i}, \quad \forall u_i \in \{u \mid P = 3\} \quad (13)$$

The aforementioned optimization problem $P1$ is subject to multiple stringent constraints regarding physical resources and Quality of Service (QoS):

Equation (10) and (11) Offloading atomicity constraints: These ensure that each computation task u_i must be allocated to one and only one unique computation node k within the control domain $N_c(t)$ for processing, rendering the tasks indivisible.

Equation (12) Physical constraint on computational capacity: This requires that for any computation node k , the total computational frequency allocated to all received tasks during the current decision cycle must strictly not exceed the maximum physical computational capacity F_k^{max} of its onboard processor, thereby preventing resource overload and node crashes.

Equation (13) QoS latency hard constraint: This serves as the core bottom-line constraint in our system model. It strictly mandates that for all critical tasks (priority $P_i = 3$), their total end-to-end latency $T_{total}^{i,k}$ must be less than or equal to their maximum tolerance time $T_{tol,i}$. Any offloading decision that violates this constraint shall be deemed invalid.

3. Intelligent computation offloading based on graph attention imitation learning

As discussed in the previous section, in the LEO satellite edge computing scenario, the optimization problem $P1$, which jointly addresses load balancing and critical task QoS guarantees, is a highly complex Mixed-Integer Non-Linear Programming (MINLP) problem. To overcome the bottlenecks of local optima and long-tail queuing faced by traditional heuristic algorithms within a millisecond-level decision window, this paper proposes a novel Graph Attention Imitation Learning (GAT-IL) framework, as shown in Figure 1. This framework leverages the powerful spatial topological representation capability of graph neural networks, efficiently fits expert knowledge through imitation learning, and innovatively introduces a QoS-aware action masking mechanism to guarantee the survival rate of critical tasks in the form of hard constraints.

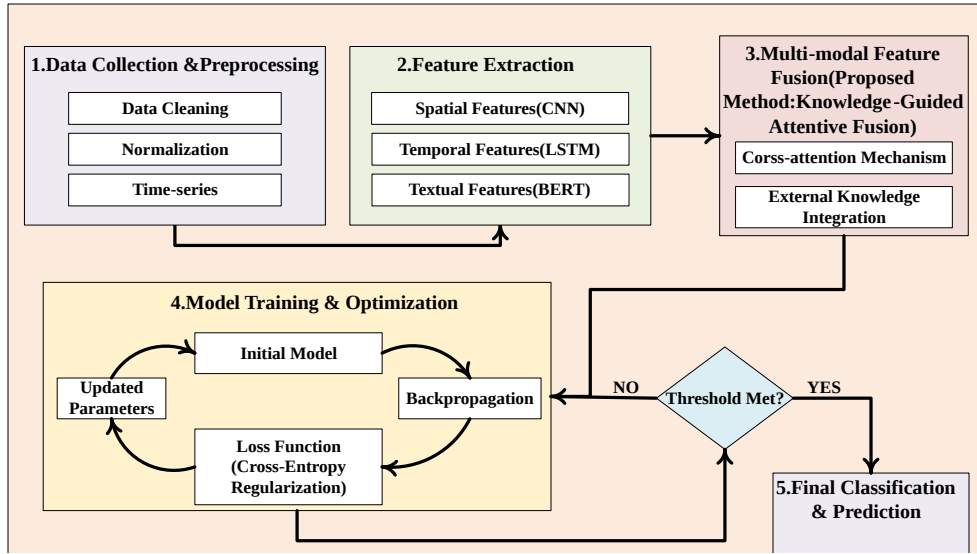


Figure 1. Algorithm framework diagram

3.1. Subsection dynamic topological representation and state extraction

Due to the continuous and relatively high-speed motion of the LEO satellite constellation, the number of candidate computation nodes and their physical states faced by the control node are highly time-varying. To enable the neural network to handle dynamically varying input dimensions and accurately capture inter-satellite topological correlations, we first model the control domain under the current time window t as a local directed fully connected graph $G_c(t) = (V_c, E_c)$. In this graph, V_c contains the set of all currently available candidate computation satellites within the control domain, while E_c represents the set of logical feature interaction edges among the candidate nodes.

For any given pending task u_i to be offloaded, the control node collects the multi-dimensional heterogeneous features of all candidate computation nodes $k \in V_c$ in real time, thereby constructing the raw state feature vector $x_k = [d_{c,k}, q_k, \eta_k, P_i]^T \in R^4$. This vector encompasses the physical spatial distance $d_{c,k}$ between the controller and the target computation node, the number of backlogged tasks q_k in the target node's current task queue, the ratio of remaining available computing power $\eta_k = F_k^{curr} / F_k^{max}$ reflecting the resource health of the node, and the priority weight P_i of the task to be allocated.

Considering the significant differences in physical dimensions across the various dimensions of these feature vectors, feeding them directly into the neural network is highly prone to triggering gradient explosion and feature vanishing phenomena. Therefore, during the model's forward inference phase, we first perform Local Z-score Normalization on the features of the graph nodes based on the statistical distribution within the current control domain. The normalized feature \tilde{x}_k is calculated as shown in the following equation:

$$\tilde{x}_k = \frac{x_k - \mu_c}{\sigma_c + \epsilon} \quad (14)$$

where μ_c and σ_c are the mean and standard deviation vectors of the raw features of all candidate nodes within the current control domain, respectively, and ϵ is a infinitesimally small positive constant introduced to prevent zero-division errors. The normalized feature tensor X will serve as the standard input for the subsequent GAT model.

3.2. Data-driven expert imitation learning framework

To overcome the bottleneck of slow convergence inherent in traditional reinforcement learning when dealing with massive action spaces, this paper constructs a data-driven expert imitation learning paradigm. During the offline training phase, the system executes a greedy heuristic expert algorithm under an ideal, congestion-free network state to generate a substantial volume of high-quality task offloading trajectories as expert labels. These labels are subsequently utilized to train the GAT model for end-to-end behavioral cloning. The core advantage of the GAT model lies in its ability to dynamically evaluate and compute the importance weights among nodes via a self-attention mechanism, thereby maintaining robust permutation invariance within the highly dynamic LEO satellite topology.

In the graph attention layer, the relative attention importance coefficient $e_{k,j}$ between computation node k and its neighbor node j is first derived through a shared linear transformation and an activation function:

$$e_{k,j} = \text{LeakyReLU}(a^\top [W\tilde{x}_k \| W\tilde{x}_j]) \quad (15)$$

Subsequently, a Softmax function is employed to globally normalize this coefficient, yielding the final attention weight $\alpha_{k,j}$:

$$\alpha_{k,j} = \frac{\exp(e_{k,j})}{\sum_{l \in V_c} \exp(e_{k,l})} \quad (16)$$

where W represents the learnable weight matrix, a is the parameterized vector of the attention mechanism, and $\|$ denotes the feature concatenation operation.

To further enhance the stability of the model in representing complex topologies, this paper adopts a Multi-Head Attention mechanism. By concatenating the outputs of M independent attention heads and applying the ELU non-linear activation function, we obtain the final context-aggregated feature representation h_k for node k :

Upon completing the information aggregation within the graph space, to preserve the original absolute hard metrics of the node itself, we perform a residual concatenation of the normalized input feature \tilde{x}_k and the high-dimensional context feature h_k . The resulting comprehensive feature is subsequently fed into a Multi-Layer Perceptron (MLP) evaluation network to compute the final offloading preference score s_k :

$$s_k = W_{out} \text{Dropout} \cdot \text{ReLU}(W_{hidden}([\tilde{x}_k \| h_k] + b_{hidden})) + b_{out} \quad (17)$$

The score s_k in the above equation precisely maps the implicit expected comprehensive benefit of selecting node k as the offloading target for task u_i , given the current global topology and resource distribution state of the satellite network.

3.3. QoS-aware action masking mechanism for critical tasks

Although the aforementioned GAT-IL model demonstrates excellent policy fitting capabilities under normal network loads, when the LEO satellite network encounters extreme bursty traffic peaks, the input states often exceed the data distribution boundaries of offline training (Out-of-Distribution, OOD). Under such extreme conditions, pure imitation learning is highly susceptible to compounding errors, causing the neural network to blindly mimic static rules. This exacerbates long-tail queuing and can even result in the massive loss of critical tasks due to timeouts. To thoroughly resolve this pain point and strictly guarantee the QoS latency hard constraint in problem $P1$, this paper innovatively proposes a QoS-Aware Action Masking mechanism.

This masking mechanism acts as the underlying security barrier of the decision-making system. By directly superimposing physical hard-line constraints at the forward output end of the neural network, it forcibly vetoes suboptimal decisions that may lead to task failures. Specifically, we design action masks in two dimensions.

3.3.1. Computational capacity depletion mask

When the remaining available computing power ratio η_k of a candidate node k drops below the warning threshold of 10%, there is a high probability that this node will cause severe processing timeouts for subsequent tasks. The computational capacity mask function $M_k^{(1)}$ is defined as follows:

$$M_{k^{(1)}} = \begin{cases} -\infty, & \text{if } \eta_k < 0.10, \\ 0, & \text{otherwise} \end{cases} \quad (18)$$

3.3.2. High-priority task queuing mask

For highly latency-sensitive, high-priority critical tasks ($P_i = 3$), they exhibit a zero-tolerance characteristic towards queuing congestion. To forcibly break the "herd effect" at congested nodes and guide high-priority traffic toward idle edge satellites, this paper strictly prohibits allocating high-priority tasks to nodes where the current number of backlogged tasks $q_k \geq 2$. The queuing mask function $M_k^{(2)}$ is defined as follows:

$$M_{k^{(2)}} = \begin{cases} -\infty, & \text{if } P_i = 3 \text{ and } q_k \geq 2, \\ 0, & \text{otherwise} \end{cases} \quad (19)$$

Integrating the aforementioned multi-dimensional hard constraints, the raw score s_k output by the GAT evaluation network is forcibly corrected by this mechanism into a final score \widetilde{s}_k that incorporates physical constraints:

$$\widetilde{s}_k = s_k + M_k^{(1)} + M_k^{(2)} \quad (20)$$

On this basis, the control node selects the node k^* with the highest corrected score as the final offloading target:

$$k^* = \arg(\widetilde{s}_k) \quad (21)$$

Furthermore, the system possesses an extreme fault-tolerant degradation capability. When the entire network falls into extreme congestion, causing all candidate nodes within the control domain to trigger negative infinity penalties, the algorithm automatically falls back to the original model scores, ignoring the action masks (i.e., $k_{fallback}^* = \argmax s_k$), to seek the maximization of the survival probability among suboptimal solutions. By introducing this action masking mechanism, the algorithm proposed in this paper not only preserves the global topology awareness capability of the neural network but also endows the decision-making system with absolute engineering robustness, thereby realizing a powerful guarantee for the survival rate of high-priority tasks from the underlying logic.

4. Performance evaluation

To verify the effectiveness and robustness of the proposed intelligent computation offloading algorithm based on Graph Attention Imitation Learning (GAT-IL), this section conducts a comprehensive performance evaluation within an LEO satellite edge computing simulation environment that accounts for realistic orbital dynamics and bursty traffic peaks. Furthermore, an in-depth comparison and analysis of various core network metrics are provided.

4.1. Simulation parameter settings and comparative baselines

This paper constructs a discrete-event simulation platform for LEO satellite networks with time-varying topological characteristics based on Python. In the simulation scenario, the control node receives heterogeneous computation-intensive tasks uploaded from ground stations and distributes them for offloading

to available computation nodes within its control domain based on the instantaneous graph topology. To fully validate the superiority of the proposed algorithm, we select three representative traditional scheduling strategies as comparative baselines: Random Allocation Algorithm (Random), where the controller blindly and randomly selects an available computation satellite within the domain for task offloading; Greedy Distance Algorithm (Greedy-Distance), a strategy that solely considers the physical spatial topology, consistently scheduling tasks to the computation node with the shortest physical Euclidean distance while completely ignoring its resource queuing state; Heuristic Baseline Algorithm (Heuristic Baseline), which is the static multi-factor weighted expert algorithm used to generate the training labels for imitation learning. The GAT-IL algorithm proposed in this paper deeply clones the aforementioned expert experience and superimposes strict QoS-aware action masking hard constraints.

4.2. Model training and convergence analysis

Before deploying the GAT-IL algorithm to the dynamic scheduling system, we first verify the fitting capability and convergence characteristics of the graph attention imitation learning network during the offline phase. As shown in Figure 2, the evolution of the training loss and validation accuracy of the model over 250 epochs is presented. It can be observed that in the initial training phase (the first 50 epochs), as backpropagation proceeds, the training loss of the model exhibits an extremely steep downward trend. This indicates that the GAT, based on the multi-head attention mechanism, can rapidly capture the spatial topological correlations and heterogeneous feature distributions among satellite nodes.

Subsequently, the training loss stabilizes after 100 epochs, converging to an extremely low level; simultaneously, the behavioral cloning accuracy on the validation set climbs steadily, ultimately forming a stable plateau at a high level close to 1.0. This convergence curve fully demonstrates that GAT-IL can accurately learn and replicate the expert policy under congestion-free states without exhibiting obvious overfitting degradation phenomena, thereby laying a solid representational foundation for the subsequent online dynamic masking inference.

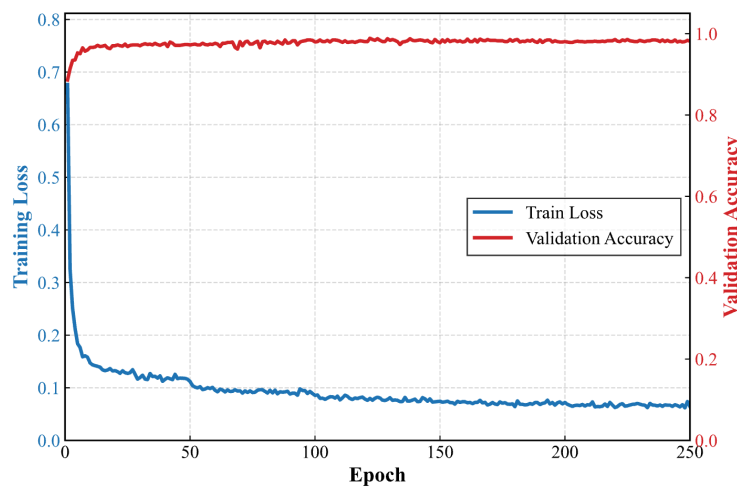


Figure 2. Model convergence graph

4.3. End-to-end experimental performance analysis

Reducing the end-to-end processing latency of massive tasks is one of the core objectives in satellite edge computing. Figure 3 presents a detailed comprehensive comparison of the various algorithms in terms of end-to-end latency metrics. Specifically, Figure 3(a) illustrates the average latency performance for tasks of

different priorities. It is noteworthy that for the extremely latency-sensitive, high-priority tasks ($Priority = 3$), the Greedy-Distance algorithm achieves seemingly the lowest average latency (8.0s), whereas the proposed GAT-IL algorithm records 9.5s.

However, this superficial phenomenon stems from a typical "survivorship bias": the greedy algorithm blindly pushes tasks to the nearest nodes, causing a massive number of tasks to experience congestion and timeout, which are subsequently dropped by the system (and thus fail to be accounted for in the successful latency statistics). In contrast, to maximally preserve high-priority tasks, the GAT-IL algorithm actively triggers the action masking mechanism, diverting a portion of the traffic to slightly more distant idle nodes. At the cost of a marginal sacrifice in physical transmission latency, it thoroughly circumvents the fatal queuing latency. For medium- and low-priority tasks, the latency performance of the GAT-IL algorithm comprehensively outperforms both the random algorithm and the heuristic baseline.

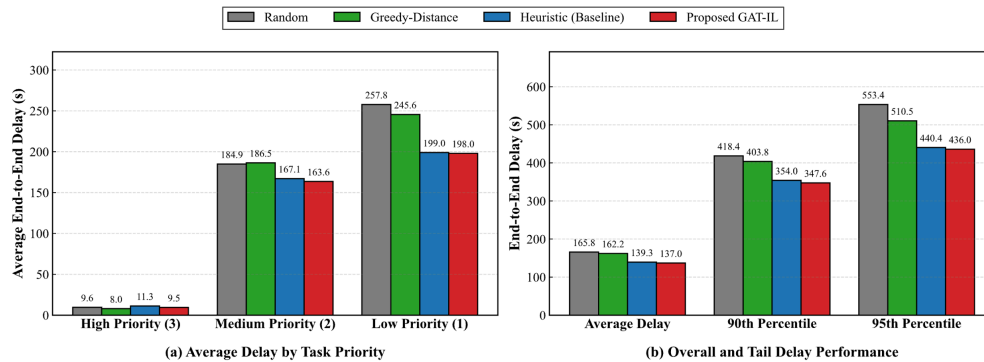


Figure 3. Comparison of end-to-end latency performance

Figure 3(b) further reveals the latency distribution characteristics of the overall system from a macroscopic perspective. In the comparison of the network-wide average latency, the 90th percentile, and the 95th percentile tail latencies, the GAT-IL algorithm consistently demonstrates a significant leading advantage. Particularly concerning the 95th percentile metric, which measures the long-tail effect of extreme network congestion, GAT-IL (436.0s) achieves a substantial reduction compared to Random (553.4s) and Greedy-Distance (510.5s). This robustly corroborates that GAT-IL, by virtue of its global attention receptive field and underlying masking hard constraints, can effectively break the "herd effect" at local nodes, realizing deep load balancing of computing resources across the entire network.

4.4. QoS guarantee capability and task success rate

Under extreme network traffic peaks, the ability to guarantee the survival rate of critical tasks is a decisive metric for evaluating the engineering robustness of a scheduling algorithm. Figure 4 depicts the evolutionary trend of the cumulative number of successfully completed high-priority tasks over the simulation period. In the initial phase of the simulation (prior to approximately 250s), the network load is relatively light, and the cumulative curves of all algorithms highly overlap. However, as time progresses and congestion intensifies, the Greedy-Distance and Random algorithms rapidly deviate from the optimal envelope and exhibit evident growth stagnation due to local queuing explosions. Although the heuristic baseline maintains a relatively good growth trend, it remains constrained by the limitations of static rules. In contrast, the GAT-IL algorithm proposed in this paper (represented by the red line) exhibits the most robust and steep climbing slope throughout the middle and late stages. Ultimately, it establishes an overwhelming lead by the end of the simulation, successfully completing the maximum number of high-priority tasks.

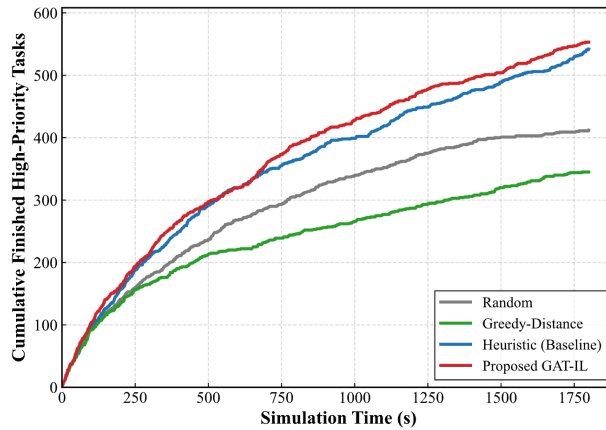


Figure 4. Evolution of cumulative high-priority task completion

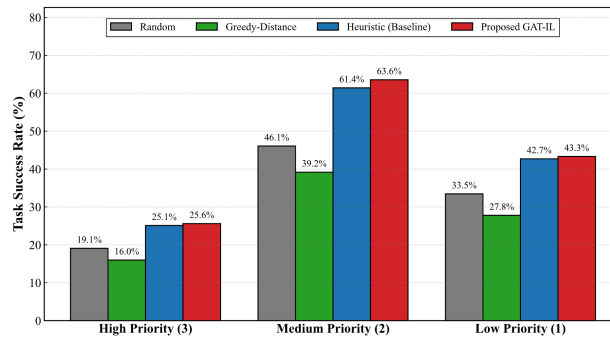


Figure 5. Task success rate across different priority levels

To more intuitively deconstruct the service guarantee capabilities of various algorithms under different QoS requirements, Figure 5 compiles the final successful execution rates of tasks across all priority levels. The data indicates that under the same high-intensity network load, the success rates of the traditional Random and Greedy algorithms experience a severe collapse across all priority levels (for instance, the Greedy algorithm only achieves a 16.0% survival rate for high-priority tasks). In contrast, the GAT-IL algorithm proposed in this paper exhibits extremely clear QoS-skewed guarantee characteristics: it not only overwhelmingly surpasses traditional spatial greedy strategies in the success rates of medium- and low-priority tasks but also achieves the highest overall success rate of 25.6% for the most critical high-priority tasks ($Priority = 3$). This result thoroughly corroborates the correctness of the theoretical design in Section 3 from a macroscopic statistical perspective: by forcibly severing the allocation paths of high-priority tasks to highly backlogged nodes (queue length ≥ 2) at the decision-making terminus, the QoS-aware action masking mechanism endows the GAT with exceptionally strong risk-aversion capabilities, successfully defending the lifeline of critical tasks in harsh network edge environments.

5. Conclusion

Aiming at the dynamic topology and extreme congestion issues in LEO satellite edge computing networks, this paper proposes an intelligent computation offloading algorithm based on Graph Attention Imitation Learning (GAT-IL). To overcome the limitations of traditional static heuristic strategies, which are prone to triggering the "herd effect" and long-tail queuing, this paper innovatively introduces a QoS-aware action

masking mechanism. By imposing physical hard constraints, it forcibly blocks suboptimal offloading paths from the underlying decision-making logic. Extensive simulation results demonstrate that the proposed algorithm not only significantly reduces the network-wide average latency and the 95th percentile tail latency, achieving efficient global load balancing, but also provides absolute Quality of Service (QoS) guarantees for high-priority critical tasks under severe heavy-load conditions, attaining the optimal task success rate overall. Future research will further introduce Multi-Agent Reinforcement Learning (MARL) mechanisms, with the expectation of realizing efficient cross-domain computational power orchestration and distributed collaborative scheduling within more complex LEO constellation topologies.

References

- [1] Li, L., Zhang, H., Sun, C., Guan, W., & Zhang, M. (2026). Artificial intelligence-enabled optimization mechanisms for space-air-ground integrated networks in 6G. *Chinese Journal of Engineering*, 48(2), 316–330. <https://doi.org/10.13374/j.issn2095-9389.2025.08.08.003>
- [2] Lyu, T., Xu, Y., Liu, F., Xu, H., & Han, Z. (2025). Task offloading and resource allocation for satellite-terrestrial integrated networks. *IEEE Internet of Things Journal*, 12(1), 262–275. <https://doi.org/10.1109/JIOT.2024.3465656>
- [3] Zhang, Y., Chen, C., Liu, L., Lan, D., Jiang, H., & Wan, S. (2023). Aerial edge computing on orbit: A task offloading and allocation scheme. *IEEE Transactions on Network Science and Engineering*, 10(1), 275–285. <https://doi.org/10.1109/TNSE.2022.3207214>
- [4] Yang, X., & Hong, B. (2022). Cost-efficient task offloading for satellite edge computing systems. In *2022 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1–5). Shenzhen, China: IEEE. <https://doi.org/10.1109/ISNCC55209.2022.9851711>
- [5] Wu, H., Yang, X., & Bu, Z. (2024). Task offloading with service migration for satellite edge computing: A deep reinforcement learning approach. *IEEE Access*, 12, 25844–25856. <https://doi.org/10.1109/ACCESS.2024.3367128>
- [6] Zhang, R., & Zhao, B. (2023). Task offloading and resource allocation for cloud-edge collaboration in low earth orbit satellite networks. In *2023 IEEE 23rd International Conference on Communication Technology (ICCT)* (pp. 764–769). Wuxi, China: IEEE. <https://doi.org/10.1109/ICCT59356.2023.10419596>
- [7] Xia, J., Wang, P., Li, B., & Fei, Z. (2022). Intelligent task offloading and collaborative computation in multi-UAV-enabled mobile edge computing. *China Communications*, 19(4), 244–256. <https://doi.org/10.23919/JCC.2022.04.018>
- [8] Tang, Q., Fei, Z., & Li, B. (2022). Distributed deep learning for cooperative computation offloading in low earth orbit satellite networks. *China Communications*, 19(4), 230–243. <https://doi.org/10.23919/JCC.2022.04.017>
- [9] Zhang, H., Liu, R., Kaushik, A., & Gao, X. (2023). Satellite edge computing with collaborative computation offloading: An intelligent deep deterministic policy gradient approach. *IEEE Internet of Things Journal*, 10(10), 9092–9107. <https://doi.org/10.1109/JIOT.2022.3233383>
- [10] Jia, M., Zhang, L., Wu, J., Guo, Q., Zhang, G., & Gu, X. (2025). Deep multiagent reinforcement learning for task offloading and resource allocation in satellite edge computing. *IEEE Internet of Things Journal*, 12(4), 3832–3845. <https://doi.org/10.1109/JIOT.2024.3482290>
- [11] Wen, W., Cui, H., & He, T. (2025). Multi-layer reinforcement learning assisted task offloading in satellite edge computing. *IEEE Transactions on Vehicular Technology*, 74(4), 6561–6572. <https://doi.org/10.1109/TVT.2024.3516081>
- [12] Wei, W., Hu, H., Yan, C., Wang, Y., & Jiang, B. (2025). ST-DSQN: A task offloading method for low earth orbit satellite networks. In *2025 6th International Conference on Computer Engineering and Application*

- (*ICCEA*) (pp. 606–611). Hangzhou, China: IEEE. <https://doi.org/10.1109/ICCEA65460.2025.11103023>
- [13] Sun, Z., Mo, Y., & Yu, C. (2023). Graph-reinforcement-learning-based task offloading for multiaccess edge computing. *IEEE Internet of Things Journal*, 10(4), 3138–3150. <https://doi.org/10.1109/JIOT.2021.3123822>
- [14] Li, C., Huang, P., Luo, Z., & Zhang, C. (2025). Graph-based reinforcement learning for privacy-preserving task offloading in satellite-terrestrial networks. In *2025 International Conference on Satellite Computing (Satellite)* (pp. 1–4). Yantai, China: IEEE. <https://doi.org/10.1109/Satellite67108.2025.11430442>
- [15] Yu, S., Chen, X., Yang, L., Wu, D., Bennis, M., & Zhang, J. (2020). Intelligent edge: Leveraging deep imitation learning for mobile edge computation offloading. *IEEE Wireless Communications*, 27(1), 92–99. <https://doi.org/10.1109/MWC.001.1900232>