Risk-Aware Task Allocation for LEO Satellite Constellations Using Deep Reinforcement Learning, Fuzzy Inference, and K-Means Clustering

Zhen Zhang*, Xingyuan Hu
Shanghai Institute of Satellite Engineering, Shanghai Academy of Spaceflight Technology, Shanghai 201109, China
E-mail: ZhenZhang157@outlook.com, johnhu9458@126.com
*Corresponding author

Keywords: Low earth orbit constellation, artificial intelligence, autonomous safety and task planning, K-means clustering, Fuzzy Logic, and DRL

Received: October 23, 2025

The explosive expansion of Low Earth Orbit (LEO) satellite mega-constellations presents critical challenges in operational risk management, real-time task allocation, and dynamic resource management due to their inherent time-varying topology and systemic uncertainties. To overcome the limitations of traditional optimization methods, this paper proposes a novel, hybrid artificial intelligence framework integrating K-means Clustering, Fuzzy Logic (FL), and Deep Reinforcement Learning for Task Allocation (DRL-TA). First, an adaptive K-means clustering mechanism periodically segments the constellation based on real-time channel quality and traffic load, effectively reducing network state dimensionality and overhead for the DRL agent. Second, a Fuzzy Inference System is employed to model non-deterministic operational elements (e.g., temperature, orbital deviation) and predict the real-time Safety Risk Score 〖Fuzzy〗_risk. This interpretable risk score is integrated as a penalty term within the DRL agent's reward function. Finally, the DRL-TA algorithm learns the optimal policy for computationally offloading and resource allocation by jointly analyzing the clustered network state and the fuzzy-predicted risk. Validated on a simulated 1,000-satellite LEO constellation over 10,000 training episodes, the integrated DRL-TA framework demonstrates significant performance gains: achieving a 25% reduction in average task completion delay and a 15% improvement in overall task success rate compared to conventional load-balancing and pure DRL baseline methods. The DRL policy exhibited stable convergence within 8,500 episodes, with a final average episodic return accuracy exceeding 97% of the theoretical maximum. This demonstrates the framework's efficacy in creating a reliable, high-performance, and risk-aware LEO edge computing environment.

Povzetek: Članek predstavi hibridni AI pristop za upravljanje satelitskih omrežij v nizki orbiti, ki z združevanjem metod izboljša učinkovitost, zanesljivost in obvladovanje tveganj pri razporejanju nalog.

1 Introduction

LEOsatellite counts are skyrocketing as a result of space industry players' persistent practice of deploying hundreds to thousands of tiny satellites in referred to as massive constellations. This trend includes Amazon, SpaceX, and OneWeb [1], [2], [3]. This meteoric ascent has drawn the ire of many groups and specialists from all over the globe. They claim it endangers future space missions, the safety of space itself, and the ability to enter space in the future. The importance of multiple satellite cooperative strategy development has grown alongside the advancements in satellite autonomy. To accomplish this, a coordinated mission planning strategy necessitates the cooperation of numerous satellites through intersatellite links. Algorithms should seek to reduce communication expenses in light of potential concerns that include satellite-to-satellite link failure, noise from outside sources, and transmission delay. Collisions between operational satellites and preexisting space debris are more likely due to the high number of satellites in orbit, which in turn increases the quantity of space trash in orbit. The safe and accessible functioning of the near-Earth orbits depends on the implementation of self-sufficient operations and safety concepts. In order to improve system autonomy in a number of earthly businesses, methods from the field of computational intelligence have been studied and created in recent years [4], [5]. Satellite operations can

benefit from these kinds of approaches, which can bolster operations both in space and on Earth. They make it possible for systems to be monitored, anomalies to be detected, decisions to be made [6], [7], [8], and formations or constellations of collaborating spacecraft to be managed with more autonomy. On the other hand, difficulties arise due to the greater number of satellites in LEO [9], [10].

- Safe orbital navigation and avoiding collisions [11],
- Planning assignments dynamically for the transfer of data or monitoring the earth [12]
- Maintaining a stable communication link with constrained resources [13],
- Make decisions adaptively in space when conditions are unknown [14].

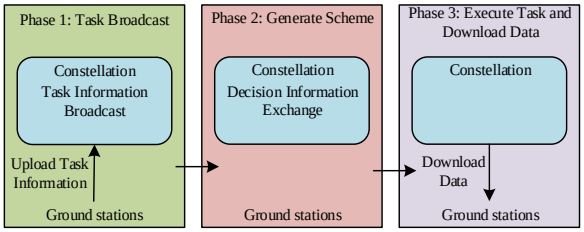

Figure 1: Task broadcasting from ground stations

An AI-powered autonomous safety and job planning system is suggested to rectify these issues [15], [16]. With this method (see Fig 1), we hope to boost mission dependability in LEO constellations, optimize resources better, and make better decisions in real time [17], [18]. A combination of time-varying channel circumstances and the rapid mobility of LEO satellites makes the issue non-convex, non-linear, and NP-hard. While these approaches may work for small-scale networking snapshots or real-time processing, they are usually rather sluggish and only provide suboptimal results. Predicting the operational health and danger of satellites often involves using mathematical models and deterministic analysis, which are standard approaches. When looking at past data, methods such as Fault Tree Analysis (FTA) and Failure Mode and Effects Analysis, or FMEA, may be used to determine the statistical likelihood of a component failing. Integrating inaccurate or subjective data (such as deterioration caused by unanticipated spacecraft weather, softness failures, or expert judgment) is difficult for them since they need precise, statistically huge datasets. They aren't dynamic enough to be useful for risk rating in real time [19], [20].

1.1 Objectives

- To develop an autonomous task planning model for satellite constellations using AI and optimization algorithms.
- To make use of neural networks and predictive models to create a component for security monitoring and collision prevention.
- To enhance data relay scheduling and satellite-to-satellite connectivity while dealing with changing restrictions.
- To achieve complete independence with little help from the ground.

1.2 Key contributions

The primary contributions of this work are threefold, focusing on the synergistic integration of three advanced computational paradigms to tackle the unique challenges of LEO satellite networks:

1. **Dynamic Network Partitioning for Scalability (k-means):**
    - Designed and developed a Demand-Aware k-Means Clustering method that uses real-time user location, traffic volume, and inter-satellite link (ISL) quality to dynamically split the LEO constellation into optimum service clusters.
    - It has been demonstrated by grouping resources in advance, this clustering method makes load balancing easier and simplifies the state space of the DRL agent.
2. **Robust Risk Quantification under Uncertainty (Fuzzy Logic):**
    - Constructed a Fuzzy Logic Inference System that can aggregate imprecise operational data (such as component health, environmental circumstances, and collision probability) to provide a quantifiable, real-time Satellite Risk Score.
    - The DRL reward function incorporates this score as a penalty or dynamic constraint; using it, the job assignment algorithm may intelligently avoid dangerous satellites, improving the dependability of missions and the stability of networks over the long run.

3. **Risk-Aware Task Assignment (DRL)**
    - Created an innovative DRL agent that can allocate resources, route tasks, and offload computations all at once.
    - By punishing assignments to satellites designated as high-risk by the Fuzzy Logic system, the DRL-TA algorithm optimizes a multi-objective function that incorporates decreasing task delay and optimizing resource usage in a novel way.

The following is the outline for the remainder of the paper: From single-satellite to multi-satellite scenarios of algorithms, the problem statement is presented in Section II. Detailed in Section III are the algorithms that were used to resolve the issue. In Section IV, we describe the findings and evaluation of our experiments. Section V wraps up the paper and looks at where it goes from here.

2 Related work

The primary ground-based control station is responsible for producing the time for a satellite positioning system. This means that equipment and users aren't guaranteed a constant supply of time services if ground stations go down. The already-strained satellite-to-ground link will be even more so when more satellites are launched, as is widely expected. Readings of time deviations from intersatellite communications will be the backbone of next-generation navigational satellites. By taking these readings, the system can strengthen its dependability and resilience by creating and maintaining its own space-based time reference. There are sufficient resources to build a space-based time reference, thanks to the proliferation of low-Earth-orbit communication satellite networks. Big-time scale calculations, more link noise, and underutilized clock resources are some of the problems this brings. This study presents the D-KPW algorithm, which is designed to solve these problems. The algorithm integrates the best features of weighted average and Kalman filtering. It finds a happy medium between execution speed and the amount of computing power needed. Also, to take into consideration the time reference's frequency stability in the short and long term, an adaptive clock control method called D-KPW (Control) is created. Tests show that the D-KPW (Control) algorithm successfully establishes a stable frequency reference [21].

If we want to maximize network performance and allocate resources efficiently, we need reliable traffic predictions for networks with LEO constellations. It can also make it easier to adapt to new offerings and lower operational costs. Predicting traffic conditions with ground networks typically involves ML algorithms such as SVM and DL algorithms such as CNN and GAN. Performance optimization and excellent service assurance are the goals of these approaches. But they aren't good for low-Earth orbit constellations since they don't take LEO satellites' high dynamics into consideration. It is still difficult to design a smart LEO traffic forecast model for use in scenarios involving several services. The authors of this work present the DST-LEO model, a model for predicting traffic on LEO satellites in a dynamic spatio-temporal context. The model employs a number of attention process modules to collect implicit properties among satellite nodes. Applying a multiple-source data separation and fusion method, it analyzes data on the flow of traffic as well as the connection/disconnection of inter-satellite communications. Following the splicing and fusion processes, predictions are made through the attention process. The model produced RMSE of 0.0052 and MAE of 0.0036 over the long run and RMSE of 0.0029 over the short term on the Abilene dataset. Both the short-term and long-term RMSE and MAE on a generated internal LEO constellation dataset were 0.0033 and 0.0027, respectively. Its mean absolute error was 17.0% lower and its mean squared error was 22.5% lower than that of competing time-series prediction models. Each module's operations were tested by ablation tests, which also demonstrated that the model was effective for LEO constellation traffic forecast [22].

Low-Earth Orbit (LEO) satellites have proliferated at a dizzying rate in the past few years. Also contributing to the congestion in Near-Earth space is the gradual buildup of space junk. Because of this congestion, the likelihood of orbital accidents has increased dramatically. In order to keep constellation architecture, correct, prevent collisions, and keep space missions running smoothly, LEO satellites rely on accurate orbit prediction. There have been numerous investigations into systems based on machine learning for predicting satellite orbits, but there are still certain limits. Synthetic or artificial orbit data is used to train the majority of current models. Ordinarily, these datasets presuppose an orbital process that remains stationary. This means they can't account for the dynamic and irregular orbit variations that occur in actual LEO satellite networks due to flight-state alterations, including those implemented for collision prevention. Our proposed ML model, GloLoSAT (Global-Local Probsparse Self-Attention Transformer), has a multiple-range (global-local) focus. Data on actual LEO satellite orbits obtained from N2YO.com are used to induce the model. Its primary goal is to improve the accuracy of orbit forecasts, particularly while making many modifications to an orbit. By analyzing the input sequence length theoretically, we show that our Global-Local Probsparse Self-Attention method can reach computational complexity. Using actual datasets for tracking satellite orbits, many tests were carried out. The results of these tests show that GloLoSAT

is effective. Regardless of the predicted scenario, the model always beats the competition [23].

The environmental and healthcare dangers posed by harmful algal blooms (HABs) in freshwater are substantial on a global scale. In order to control the harm that HABs inflict, it is crucial to detect them immediately after they form. HABs can be more reliably detected and their toxicity levels evaluated with in-situ monitoring. But if you want to see the temporal and spatial structure of these blooms across huge areas, satellite photos are your best chance. Additionally, satellites offer the ability to continuously monitor for the presence of HABs. Using multispectral satellite photography, HAB biomass has been estimated using both experimental and computational approaches in the past. This paper investigates a method to speed up HAB detection with a CNN to progress ongoing research. Using a CubeSat in low Earth orbit, the CNN can identify HABs very instantly. Its training data comes from the Sentinel-2 satellite constellation's multispectral photography. Datasets from Seabass CAML include aggregated counts of in-situ cyanobacteria cells, which are used for training purposes. Using multispectral imaging, the results show that the CNN can identify cyanobacterial blooms. The most effective model was able to obtain a relative mean squared error (RMSE) of 1.34 between all 5 severity categories of HAB predictions. An additional CNN that was trained on RGB and red edge (B05) bands from 30 m ground sample distance (GSD) images produced an RMSE of 1.82. This was too low to detect HABs in tiny inland bodies of water. By combining 10 m GSD bands, optimal performance was attained. According to [24], the top-performing networks made full use of Sentinel-2's 10 m and 20 m spectral bands.

This is particularly helpful for big constellations in LEOsince it increases intelligence, efficiency, and decreases operational costs and dangers. There are tremendous value and importance in the emerging trend of on-orbit autonomy. The field is still in its early stages of investigation, despite some testing and research. Requests for applications and initiatives to standardize have also been launched. From the standpoint of on-orbit connectivity and processing, however, modelling constellation autonomy has been under-explored. This paper provides a framework for modelling by reviewing appropriate guidelines, research, and examinations. Modelling mostly focuses on studying important innovations, networking, and management. One approach to define and implement on-orbit autonomy is the Constellation Autonomy Modelling (CAM) system, which allows for flexible communication and processing while in orbit [25].

More and more satellite constellations are being put into low Earth orbit, which means that space weather warning systems are becoming increasingly necessary. Media Earth orbit is increasingly being utilized for radio navigation and time transmissions. Now, communications are carried out by means of orbits in the slot region. One more layer of difficulty is added when electric propulsion is used to achieve geostationary orbit. The two most recent occurrences concerning satellite activities are highlighted below. In addition, we compiled a list of 10 user requirements that were discussed in meetings with space agency employees, designers, underwriters, and controllers of satellites. We introduce the SaRIF system, which aims to meet these demands by providing satellite-based radiation forecasts and risk predictions. In order to resolve the Fokker-Planck equation, SaRIF feeds real-time data into the BAS Radiation Belt Model (BAS-RBM). Using a one-hour precision, it can predict the flow of electrons in the outer emission belt up to 24 hours in advance. The system is updated continuously on an hourly basis and can be accessed through the website for space weather run by the European Space Agency [26].

The need for precise orbital data of space objects is thus growing in importance. Comprehensive evaluations of satellite locations, constellation setups, and dynamics in deep space are all part of this. Space object maneuver behaviour forecast and collision threat evaluation are two areas that could benefit from publicly available, real-world information, but these are presently in short supply. By collecting and putting together an accurate database of maneuvering behaviour from Starlink satellites, this work aims to fill this void. The data set incorporates both accurate ephemeris data and data from the Two-Line Element (TLE) catalogue. Because of this integration, the behaviour of space objects may be more accurately and comprehensively modelled. Additionally, it sheds light on how to assess the likelihood of collisions in growing-crowded orbital settings and how to put maneuver detection algorithms into practice [27].

There are new possibilities for sophisticated satellite computing thanks to the global network coverage and substantial data supplies. These assets make it possible for LEO satellite networks to integrate data and computational capabilities. Gas estimate, traffic monitoring, and forest fire warning are just a few examples of the adaptive learning activities made possible by this integration. One potential paradigm that has recently emerged to accomplish this combination while balancing data usage with privacy preservation is federated learning (FL). People who use typical FL usually set up their system so that the base station is the server and the satellites are the customers. But there are two big problems with this centralized system. The primary difficulty arises from the fact that the centralized server poses the possibility of only one point of failure. Another difficulty is the sluggish convergence that occurs as a result of the satellites' sporadic communication connectivity. In order to tackle

these problems, this study suggests an architecture for distributed satellite federated learning. The suggested method takes into account the constraints of data privacy, data transmission bandwidth, and the effectiveness of neural networks in communicating via satellite. To reduce the impact of data variability, this approach uses decentralized collaboration amongst satellites to settle on model parameters. In addition, in order to make satellites last longer and avoid networking congestion, an energy-conscious communication approach was developed. Results from experiments conducted on real-world datasets demonstrate that the suggested architecture can significantly reduce communication electrical consumption compared to standard techniques, speeding up the learning process by 5-10 times [28].

Network contention causes a noticeable slowdown in the downlink as the size of the constellation grows. By making use of the restricted computing power onboard, Orbital Edge Computing (OEC) solves this problem. It processes raw captures immediately at the source, which saves data transport expenses. But the answers that are already out there aren't really practical. Their dependence on basic filtering methods or propensity to give particular downstream activities too much priority is the source of this problem. An OEC-native and task-agnostic feature compression mechanism is introduced in this study to address these concerns. To optimize throughput while maintaining prediction efficiency, the suggested technique splits high-quality satellite images. Additionally, it uses inter-tile relationships and embeds context-sensitive data to further decrease transfer costs with low overhead. It is nevertheless possible to consistently rebuild the images with high standards at lower bitrates, even though the encoding prioritizes characteristics for downstream activities. After taking into consideration the sporadic availability of wireless connections in low Earth orbit, thorough tests show a significant decrease in transfer costs. Lastly, we demonstrate that the suggested method allows for downlinking of over 100 times more data without necessitating previous knowledge of subsequent processes [29], and we confirm our system's practicality on standardized small satellite form factors. Table 1 illustrates the comparison table for Input Methods, ML Techniques, Evaluation Metrics, and Performance Outcomes.

Table 1: Comparison table on the input methods, ml techniques, evaluation metrics, and performance outcomes

Work / Model	Input Methods	ML / Algorithmic Techniques	Evaluation Metrics	Performance / Key Outcomes
D-KPW Algorithm (Space-Based Time Reference)	• Time deviation readings from inter-satellite links • Clock frequency stability data • Noisy time-scale measurements	• Distributed Kalman filtering • Weighted averaging • Adaptive clock control (short-term + long-term stability)	• Frequency stability • Convergence time • Time deviation RMSE	• Achieves stable space-based time reference • Balances accuracy and computational cost • Demonstrates improved robustness vs. ground-station-based timing
DST-LEO Traffic Prediction Model	• Multisource LEO telemetry: traffic flow, ISL connection/disconnection status • Abilene dataset • Internal synthetic LEO dataset	• Dynamic spatiotemporal graph attention • Multi-source feature fusion • Multi-head attention layers	• RMSE, MAE • Long/short-term prediction accuracy • Ablation performance	• RMSE = 0.0052, MAE = 0.0036 (long term) • RMSE = 0.0029 (short term) • 17% lower MAE and 22.5% lower MSE than baselines • Effective on real and synthetic LEO data
GloLoSAT Orbit Prediction Model	• Real LEO orbit data from N2YO.com • Multiple orbit modification scenarios	• Global–local Probsparse self-attention Transformer • Long-range	• Prediction error (position/velocity) • Computational complexity • Ablation comparison	• Outperforms all baseline orbit prediction models • Handles irregular, non-stationary orbital changes •

		sequence modeling	with other Transformers	Efficient for long input sequences
CNN-based HAB Detection Using Satellite Images	• Sentinel-2 multispectral imagery (10–20 m bands) • In-situ cyanobacteria counts (SeaBASS CAML)	• Convolutional neural networks • Multispectral feature extraction	• RMSE (biomass prediction) • Severity classification accuracy	• Best RMSE = 1.34 across 5 severity levels • RGB + B05 CNN RMSE = 1.82 (insufficient for small lakes) • Performance maximized using 10 m + 20 m multispectral fusion
Constellation Autonomy Modelling (CAM) Framework	• Satellite communication and processing logs • On-orbit autonomy requirements • System-level architecture constraints	• Rule-based modelling • System-level autonomy decomposition • Mission-level architecture simulation	• Qualitative capability assessment • Autonomy readiness levels	• Provides unified modelling framework for on-orbit autonomy • Supports flexible in-orbit communication and processing configuration
SaRIF Satellite Radiation Forecast System	• Real-time radiation data • BAS-RBM inputs • Fokker–Planck solver outputs	• Data-driven radiation forecasting • Fokker–Planck physics-informed model • Hourly model updates	• 1–24 hour prediction accuracy • Electron flux error margins	• Predicts outer radiation belt electron flux up to 24 hours ahead with 1-hour granularity • Supports ESA space weather operations
Starlink Maneuver Behavior Dataset	• Starlink precise ephemeris • Two-line elements (TLE) • Derived maneuver events	• Statistical analysis • Maneuver extraction algorithms • Collision-risk modeling	• Maneuver detection accuracy • Dataset completeness	• Provides real-world dataset for maneuver modeling • Supports collision threat evaluation and behavior forecasting
Federated Learning for LEO Networks (Distributed FL Architecture)	• Satellite-local datasets (gas estimation, traffic, fire detection) • Intermittent communication data	• Decentralized federated learning • Energy-aware communication scheduling • On-orbit model aggregation	• Communication overhead • Convergence rate • Task accuracy	• 5–10× faster convergence vs. centralized FL • Large communication energy reduction • High robustness to intermittent connectivity
Orbital Edge Computing (OEC) Feature Compression	• High-resolution raw satellite imagery • Onboard compressed feature maps	• Task-agnostic feature extractor • Context-aware tile splitting • Lightweight encoding for ISLs	• Image reconstruction quality • Throughput • Downlink bandwidth savings	• Enables >100× more data delivery • High reconstruction quality at low bitrate • Suitable for intermittent LEO connectivity

3 System model

There are K satellites in a LEO cluster, and their orbital planes are L. Both the GS and satellite k, where k is an element of the set K={1,…,K}, have trajectories $\boldsymbol{r}_k(t)$ and $\boldsymbol{r}_g(t)$, respectively, in an inertial coordinate scheme centered on Earth. To establish a satellite-to-earth link, the lowest value elevation angle α_e must be satisfied, meaning that the angle between the two vectors ($\frac{\pi}{2} - \angle(\boldsymbol{r}_g(t), \boldsymbol{r}_k(t) - \boldsymbol{r}_g(t))$) must be less than or equal to α_e. Typically, the number of connected satellites to the GS is limited, and the intervals between contacts are far greater than the actual amount of time spent online.

System overview

The algorithm integrates three intelligent modules:

1. Adaptive K-Means Clustering

Groups satellites based on real-time traffic load, channel quality, residual energy, and spatial proximity. Purpose: reduce state dimensionality and stabilize the DRL perception of the network.

2. Fuzzy Logic-Based Safety Risk Estimation

Inputs: thermal state, orbital deviation, link instability, CPU temperature, etc.
Output: a Safety Risk Score Fuzzy $_{\text{risk}} \in [0,1]$.
Purpose: model non-deterministic uncertainties and penalize unsafe actions.
3. Deep Reinforcement Learning for Task Allocation (DRL-TA)

A PPO-based agent that selects:

- target satellite for offloading
- resource allocation decision
- transmission power control (optional extension).

Purpose: maximize long-term network performance while minimizing operational risk.

Satellite k records information from its internal sensors in a dataset $\mathcal{D}_k$. The datasets of two separate satellites are not in sync and may not be IID because of their various orbits and locations within the sky. Following data collection, the satellites work together to find the best solution to an optimization problem of the form.

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \frac{1}{n}\sum_{\boldsymbol{x}\in\mathcal{D}} f(\boldsymbol{x};\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{k\in\mathcal{K}} \frac{n_k}{n}\sum_{\boldsymbol{x}\in\mathcal{D}_k} \frac{1}{n_k} f(\boldsymbol{x};\boldsymbol{\theta}) \quad (1)$$

in order to train a ML model, where $\mathcal{D} = \bigcup_{k\in\mathcal{K}} \mathcal{D}_k \subset \mathbb{R}^m, n_k = |\mathcal{D}_k|$, and $n = \sum_{k\in\mathcal{K}} n_k$. The training job defines the goal as the experimental loss function, which can be described as the sum of all x in D multiplied by the function $f(\boldsymbol{x};\boldsymbol{\theta})$. This is where the training loss for a data point ($\boldsymbol{x} \in \mathcal{D}$) and the model variables ($\boldsymbol{\theta}$) with size (d) is represented by ($f(\boldsymbol{x};\boldsymbol{\theta})$. Without transferring datasets between spacecraft, the GS orchestrates this procedure, which is carried out repeatedly. We presuppose that the computational resources accessible to the satellites for solving (1) are severely constrained. So, we take into account the scenario where the satellites use the period between GS visits to communicate model settings θ and work on (1).

3.1 Assumptions

The variables $\{s_1, s_2, \ldots, s_{|S|}\}$ denote the set of satellites, with |S| standing for the total quantity of satellites. $s_t{}'$s greatest volume of storage throughout the planning time is represented by M_t. The number of missions is denoted by $|T|$, and the set of missions is $T = \{t_1, t_2, \ldots t_{|\eta}\}$. The predicted storage usage for the running operation is denoted by m_f. The necessary time to observe the task is denoted as $t_f \cdot d_f$ $t_f \cdot ts_f$ is the time at which task t_j is initiated, and te_j is the time at which task t_j is terminated. The time it takes for the satellite to change its position to complete two adjacent jobs, between times t_f and $t_{f'}$, is represented by $tr_{fff'}$. p_f represents the task's priority, t_f. Task t_f's benefit is denoted by r_f This article presents a way for the rewards of activities to degrade with time, taking into account the promptness and significance of observing chores. Depending on the priority p_j, the reward r_j decreases as the start duration of processing ts_j is postponed. In Equation (2), where λ is the attenuation factor, the computation formula is displayed.

$$r_f = p_f e^{-\lambda t E_f}, \quad (2)$$

The collection of LEO of satellite s_1 for job t_j is denoted by $W_{tj} = \{w_{tj1}, w_{tj2}, \ldots, w_{tj|}W_{t|}\}$, and the aggregate amount of LEO of satellite s_i for task t_j is defined by $|W_{tj}|$. Assigning task t_j to the kth O-VTW of satellite s_i, w_{ijk} is a key variable. $ws_{ijk}$is the time at which w_{ijk} begins and w_{ijk} is the time at which w_{ijk} ends.

3.2 Channel model

The effectiveness of job distribution in satellite networks is largely dependent on the interaction channel paradigm. The transfer of data between satellites and the delegation of responsibilities to the ground station are also impacted. All of these things play a role in how people express themselves. Remember to factor in signal loss, as shown in Eq. 1, whenever you're transferring jobs between

satellites or sending them to ground stations. This is how the model implements the Friis free-space path loss equation:

$$P_r = P_t \left(\frac{G_t G_r \lambda^2}{(4\pi d)^2}\right) \quad (3)$$

If all goes according to plan, the Friis equation will explain how much power an antenna can receive. Under these circumstances, the transmitter and receiver must be in a straight line of sight. The antenna gains of the transmitter (G_t) and the receiver (G_r) are denoted in Equation 1 as follows. (P_r) is the power that is received, and (P_t) is the power that is sent. The symbol for the wavelength is λ, and the symbol for the distance is d.

3.3 Power consumption model

$$E_{\text{total}} = Z_t \cdot E_b = \frac{Z_t \cdot P_t}{R} \quad (4)$$

The total energy needed to transmit a job of size Z_t bits can be calculated through multiplying the energy for each bit by the task dimension, as shown in Eq. 4. This equation provides a direct estimation of the total energy cost for sending a certain volume of data in a satellite system.

As seen in Eq. 5, the total energy utilized during transmission is denoted as E_{total}. What makes up E_{total} is the task size Z_t and the energy needed per bit E_b. Another way to express this is as the job size times the ratio of the transmit power (P_t) to the data rate (R). Processing power consumption, denoted as E_{p}, is affected by two factors: the baseline energy needed to keep the processor active and the dynamic energy, which is task size and CPU computing capabilities. Because more effective processing methods are available when the satellite is in direct sunlight, the amount of power it uses to process data is affected by its power budget and the amount of shadow it experiences.

The computing power of the satellite and the size of the work are the primary factors that determine the energy usage for onboard execution of minor tasks. The total amount of energy that is processed is denoted by the symbol (E_{p}). This is conditional on the baseline energy (E_{b}) needed to execute the work, as well as any extra energy used as a result of the task's size and the satellite's CPU frequency. The amount of power this model uses is proportional to the amount of shade or sunshine the satellite is receiving.

$$E_{p,s} = \kappa \cdot z \cdot Z_t \cdot f_{t,s}^2 \quad (5)$$

We use the proposed approach to model it. As part of this method, the amount of power needed to execute an onboard activity is dependent on the energy-saving coefficient, the total number of cycles per bit, and the CPU frequency. This is where the job size in bits is represented by (Z_t), the total quantity of CPU cycles per bit is denoted by (z), and the frequency in Hz of the satellite's CPU is denoted by ($f_{t,s}$). Within the framework of DVFS, the frequency ($f_{t,s}$) is squared due to the fact that the energy per cycle grows quadratically with frequency. Eq. 5 shows that the amount of energy is determined by the product of the energy utilization coefficient (J/Hz^3) and the variables $\kappa \cdot z \cdot Z_t \cdot f_{t,s}^2$. Intricacy of the task, amount of data, and the quadratic dependence of energy on frequency imposed by CNN and MLPs are all captured by this equation. Fig 2 shows the proposed flowchart.

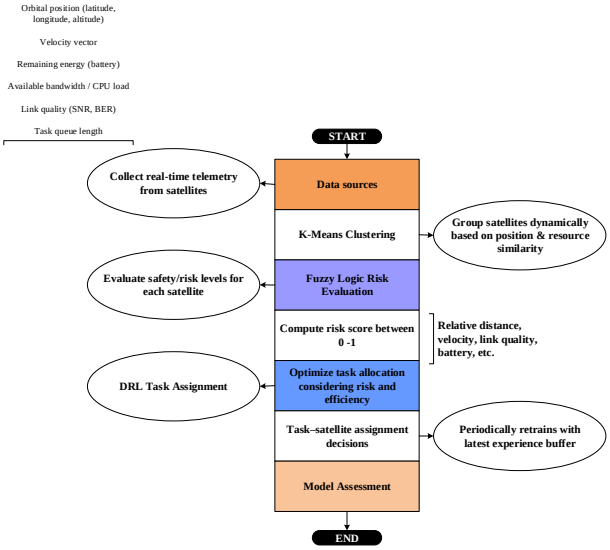

Figure 2: Proposed Flowchart

3.4 K-means clustering technique

One method for learning without human oversight is K-means. Dividing tasks and satellites into smaller, more manageable chunks helps with pretreatment and decreases dimensionality. The goal is to identify commonalities across thousands of unique missions or satellites and then arrange them into K virtual clusters. By establishing a pre-match between a task cluster and a satellite cluster, this makes the job assignment problem simpler. We classify tasks in our suggested LEO satellite network model according to their magnitude. Onboard processing of small jobs is carried out, particularly in shadow satellites. When possible, we transfer more substantial responsibilities to ground stations. In such a case, the moonbeams will communicate the mission to other neighbouring satellites that are illuminated by the sun. Because satellites are in a perpetual state of orbital motion, their positions are continually changing, which has an effect on the network's ability to allocate tasks and maintain connections. The importance of environmentally friendly routing cannot be overstated. In order to keep energy consumption low, non-linear energy models are

used to control transmission expenses and job offloading in real time. The network adapts to the changing positions of the satellites in order to maximize efficiency and optimize energy usage. By adjusting to the solar radiation reaching individual satellites, our suggested approach guarantees efficient work distribution. Processing power (in CPU cycles), amount of data input (in bits), and overall task size are the three parameters that define each job. Before optimization, all jobs are initially distributed randomly among LEO satellites. Due to their restricted energy supply, shadowed satellites handle modest activities. Large jobs can be transferred to terrestrial stations that are within line of sight (LoS). However, in this case, the satellites will transmit tasks to the nearest sunlit satellite for processing if there is no ground station inside LoS for the shadow satellites and the task is large. These spacecrafts continue to function while performing calculations that need very little power. Conversely, more substantial missions are being entrusted to sunlit satellites, which derive their power from direct sunlight. In order to do complicated calculations, these spacecrafts use their increased resource capacity, which is then expanded by scaling factors throughout the network of moving satellites. Optimal resource usage and sustainable job execution are guaranteed by this adaptive system within the dynamic satellite network. The K-means algorithm divides the collection of N satellites into K separate clusters $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$, where $C_k \cap C_{k'} = \emptyset$ for all values of k that are not equal to k' and $\bigcup_{k=1}^{K} C_k = \mathcal{N}$.

- Clustering Metric ($\boldsymbol{d}_i$): Clustering of satellites is done using a feature vector, denoted as ($\mathbf{x}_i$). Among the elements included in this vector are functional load metrics and orbital variables like semi-major axis and inclination. Reducing the WCSS, or Within-Cluster Sum of Squares, is the goal:

$$\min_{\mathbf{C},\boldsymbol{\mu}} \sum_{k=1}^{K} \sum_{i \in C_k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \quad (6)$$

$\boldsymbol{\mu_k}$ represents the location of the center of cluster C_k.

- **State Reduction:** For every cluster, we determine the overall load and the available resources. For every cluster, the optimal Fuzzy Risk Index is also found, denoted as $\max_{i \in C_k}(\text{FRI}_i)$. All of these numbers are fed into the DRL state ($\mathcal{S}$) in aggregate form.

3.5 Fuzzy model for risk prediction

Due to the imperfection and natural unpredictability of space contextual awareness and mission limits, the Fuzzy Model is employed to manage these issues. A "softer" measure than conventional binary logic is provided by it. When it comes to LEO constellation risk prediction, a Fuzzy Logic Model might be utilized [30]. Because of the inherent uncertainty and unpredictability in space context-sensitive data, it is especially effective for resolving these issues. Evaluations of space debris risk and conjunction assessments benefit greatly from this method. Precise orbital data is typically used in conventional collision probability (Pc) calculations. Availability and accuracy of such data are not guaranteed. This is particularly the case when working with a large number of possible conjunctions or with little, untracked detritus [31]. Accurate and genuine linguistic risk factors can both be included in a fuzzy model. There are primarily three components to a Fuzzy Inference System in this context:

1. **Fuzzification:** Transforms quantitative input values into fuzzy language phrases by means of Membership Functions (MFs).
2. **Inference Engine (Rule Base):** Determines the fuzzy output by applying an IF-THEN rule set to the fuzzy inputs.
3. **Defuzzification:** Processes the resultant fuzzy output and returns a decision-ready numerical risk index.

The factors that have the greatest impact on collision risk should serve as the input parameters. For example, " Extremely Low," "High," and "Critical" are examples of fuzzy linguistic parameters that are transformed from these clear inputs. Table 1 illustrates the fuzzy input parameters.

Table 1: Fuzzy input parameters

Input Parameter	Specification	Unit
Accident Likelihood (Pc)	The usual statistical chance of a crash.	10−x
Closest Approach Distance (DCA)	Minimum expected separation distance between the two entities.	Km or m
Speed in Relation to Another	The rate of approach that the two objects are making to one another.	km/s
Object Size/Radar Cross-Section (RCS)	A surrogate measure of the debris's or satellite's actual size.	m2
Distance to the nearest point of contact (TCA)	The estimated time till the expected conjunction.	Days or hours

For the purpose of translating continuous, imperfect inputs (such as collision likelihood, available electricity, or cloud cover) into language-based risk and utility estimations.

The membership functions are defined via fuzzy logic rather than by a single hard criterion for collision threat, such as 10−5. Several inputs are transformed into fuzzy sets, such as Low Risk, Medium Risk, and Critical, based on the inputs' significance levels. One Fuzzy Risk Index per overall scenario is the final product. It can be utilized to rate a spacecraft's "suitability" for a job according to several subjective criteria. Considerations such as the level of power ('High' or 'Medium'?) and the flexibility of the task deadline are among these aspects. An integral aspect of the DRL's state or reward scheme is this Fuzzy Index. Due to their efficiency in computation and relative simplicity, MFs that are triangular or trapezoidal tend to be more frequent. Examples of factors that might benefit from a Gaussian MF include those with a "medium" risk P_c value and others where a "sweet spot" (core) would be ideal.

The Universe of Discourse (input range) may be on a logarithmic scale (for instance, from 10^{-9} to 10^{-4} for the Chance of Collision ($\boldsymbol{P_c}$). Possible MF forms include:

- Very low: Membership equals 1 for $P_c \leq 10^{-7}$, gradually decreasing.
- High: Membership is one for P_c approximately 10^{-5}, with a gradual decrease on either side.
- Critical: For $P_c \geq 10^{-4}$, membership is 1, and it gradually decreases.

In the end, the constellation operator will base their judgment on the outcome, which is the overall conjunction risk estimate. Table 2 illustrates the output parameters.

Table 2: Output parameters

Output Variable	**Description**	**Unit**
Constellation Risk Index (CRI)	How serious and urgent the event is, as measured by a composite score.	From 0 to 100 scale

At its heart, the system is a rule-based system that attempts to simulate the way a human operator makes decisions. The usual format for expressing rules is: According to the logic, if both inputs are in sets A and B, then the output is in set C. The fuzzy rules are follows:

- Rule 1 (The Most Dangerous): If PC is in critical condition, DCA is nearby, and TCA is about to happen, then CRI is a maneuver.
- Rule 2 (High-Risk Situation): CRI is alert if there are high relative velocities, short TCAs, and high PCs.
- Rule 3 (Mandatory Inspection): When PC is low, DCA is closed, and TCA is long, the CRI is considered to be in the monitor position.
- Rule 4 (he Lowest Danger): CRI is considered to be negligible if PC is extremely low and DCA is extensive.

Defuzzification and decision

A fuzzy set is constructed for the CRI once the rules are fired. The Defuzzification process takes this muddled output and turns it into a single, clear risk score (like 85/100). Many people use the Centroid Method to defuzzify their data. The next step is for the cluster operator to be guided by the final, precise CRI value:

- **CRI 0-20 (Negligible):** Proceed with monitoring; no urgent steps required.
- **CRI 21-50 (Monitor):** Maximize the frequency of monitoring and inform the planning of maneuvers.
- **CRI 51-80 (Alert):** Get ready to perform a maneuver by running an accurate orbit determination again.
- **CRI 81-100 (Maneuver):** Go ahead and do the maneuver to prevent a collision.

3.6 DRL for task assignment

To solve the difficulty of assigning tasks, this section proposes a reinforcement learning agent. A sophisticated neural network is utilized in the construction of the decision agent [32], [33]. Our suggested DNN makes use of stacks and attention layers to transmit data. This policy neural network is subsequently trained using policy gradient approaches to ensure a properly tuned network. Our reinforced learning agent is trained with the help of a deep neural network. The preceding MDP model serves as the basis for feeding process states into a deep neural network, which in turn produces an action. This research proposes a stack-based structure for DNNs that takes its cues from the Transformer design. The network employs attention layers to relay data between many possible assignments. We encapsulate the state matrix in our network framework's basic embedding layer. Then, a series of identical "stacks" with varying parameters carry out the same functions. There are two levels of interaction in each stack: the agent-specific layer and the task-specific layer. We utilize these stacks to communicate information between various agents and jobs so that we can have a better picture of the current situation. Next, we compute a scalar from the hidden models of every LEO-Target pair using a linear feed-forward layer. Last but not least, the output is processed using a Softmax layer [34]. Figure 3. shows the whole network design for the task distribution.

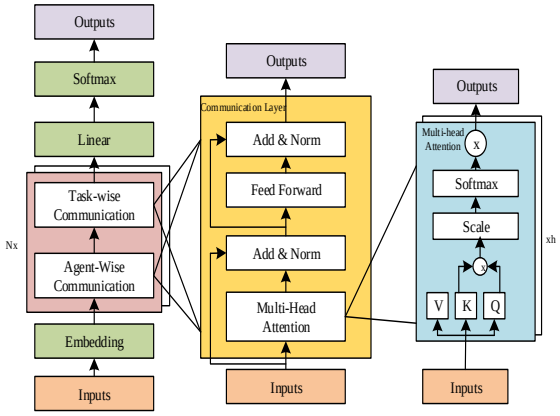

Figure 3: Network architecture for task assignment

Just like the MDP system we built earlier, we feed the deep neural network the status of every process. In this paper, the state data is represented by a 3-dimensional array$X \in \mathbb{R}^{N_u \times N_v \times |X_1|}$, where |X| is the total quantity of elements utilized to express the state data. To access details about individual LEO-Target pairings, we can use the X, i, and j indexes in the 3-dimensional array:

$$X_{tj} = \left[\text{LEO}_t \quad \text{Target}_j \quad \text{LEO}_t - \text{Target}_j\right]^{\top}. \tag{7}$$

The characteristics of the i-th LEO are represented by LEO $_t$, and the state data for the j-th target is indicated by Target. LEO $_i$- Target, which represents the state data of the pair LEO-Target.

- **Embedding Layer:** The feed-forward neural network known as the layer of embedding ε is the first to handle the input X. To transform the final input X dimension into the $\mathbb{R}^{|\varepsilon|}$ dimension, we employ the embedding layer. Put otherwise, each X_{tl} is subjected to the following processing in an FF layer, element-by-element, with the variable θ_ε:

$$\varepsilon(X) = \begin{bmatrix} F(X_{1,1}; \theta_\varepsilon) & \dots & F(X_{1,N_v}; \theta_\varepsilon) \\ \vdots & \ddots & \vdots \\ F(X_{N_u,1}; \theta_\varepsilon) & \dots & F(X_{N_u,N_v}; \theta_\varepsilon) \end{bmatrix} \tag{8}$$

- **Communication Layer:** To build the interface layer, we drew inspiration from the Transformer framework. An attention layer with many heads and a feed-forward layer are the two main components of every connection layer. For every sublayer, we further apply batch standardization and skip-connection.

As a means of communicating between various LEOs or targets in our assignment problem, we employ the attention process. Communication at the agent level and communication at the task level are the two types of communication layers. Both types of communication layers are functionally equivalent; the only difference is the set of dimensions on which they operate. For the sake of clarity, we will use task-specific interaction here; the other is identical. Based on the various LEOs, we may partition the 3-dimensional array $\mathcal{E}(X) \in \mathbf{R}^{N_U \times N_V \times |\mathcal{J}|}$ into M groups. A two-dimensional vector $Y \in \mathbf{R}^{N_V \times |\mathcal{E}|}$ can be used to describe every group. Each group will be treated independently by our network design's self-attention operation, which uses the same settings. Prior to generating queries $q \in \mathbb{R}^{N_V \times d_k}$, keys $k \in \mathbb{R}^{N_V \times d_k}$, and values $v \in \mathbb{R}^{N_v x d_v}$, self-attention executes three feed-forward layers independently. After that, we run the attention process like this:

$$\text{attention}\ (q, k, v) = \text{softmax}\left(\frac{qk^{\top}}{\sqrt{d_k}}\right)v \tag{9}$$

To run the multiple attention systems simultaneously, we employ multi-head attention. The h-heads used in the study are $q = (q_1, q_2, \dots, q_h), k = (k_1, k_2, \dots, k_h)$, and $v = (v_1, v_2, \dots, v_h)$. Afterwards, all of the heads that pay attention to themselves will be joined:

$$\text{MHA}(Y) = [H_1 H_2 \dots H_n] \tag{10}$$

in which

$$H_l = \text{attention}\ (q_l, k_l, v_l) \tag{11}$$

Therefore, H_l is a subset of $\mathbf{R}^{N_v \times d_v}$ and MHA (Y) is a subset of $\mathbb{R}^{N_v \times h d_v}$. A completely linked feed-forward (FF) layer is used for data processing after the multiple-headed attention sublayer. On top of that, we supplement each sublayer with a skip-connection and batch standardization:

Therefore, H_l is a subset of $\mathbb{R}^{N_v \times d_v}$ and MHA (Y) is a subset of $\mathbb{R}^{N_v \times h d_v}$. A completely linked feed-forward (FF) layer is used for data processing after the multi-head attention sublayer. On top of that, we supplement each sublayer with a skip-connection and batch normalizing:

$$Z = \text{BN}(\mathcal{E} + \text{MHA}(\mathcal{E}))$$
$$\text{C}(Z) = \text{BN}(Z + \text{FF}(Z)) \tag{12}$$

Which means that $\mathcal{E}$ is what the embedding layer produced, and $\mathcal{C}$ is what the interaction layer produced.

Softmax with Linear Layers: We build a Linear layer once N stacks have executed, with each stack carrying out the self-attention procedure. An uncomplicated, entirely linked feed-forward layer is what we employ for the Linear layer. To continue processing the state data, we employ the

Linear layer. The last step is to process the output using the Softmax layer. But, the "LEO-Task" pairs that aren't practical must be hidden before the Softmax process may begin. The temporary output $\in \mathbb{R}^{N_u \times N_v}$.

$$\bar{M}_{t_d} = M_{t_d} - \beta inf_{t_d} \quad (13)$$

Where the positive constant β is really big. Our final output is obtained by applying the Softmax layer following the mask procedure.

For the purpose of training the policy neural network, our paper utilizes policy gradient approaches. Discovering the best course of action and increasing its environmental return are our top priorities. Here is how we characterize the goal function of policy learning:

$$J(\theta) = E_{s_0}[V^{\pi_\theta}(s_0)] \quad (14)$$

In this case, V^{π_e} represents the state-value function and s_0 signifies the initial state. Beginning from the state that follows the policy π_θ, it signifies the anticipated return. With regard to θ, we determine the objective function. To get the best policy, we can optimize this objective function using the gradient ascent approach. Here's how to write the gradient of the objective function:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[Q^{\pi_\theta}(s,a)\nabla_\theta \log\, \pi_\theta(a \mid s)] \quad (15)$$

The action-value function, denoted as Q^{π_θ}, shows the anticipated return from acting on the present state while the MDP adheres to policy π_θ. To determine the objective function's gradient, we employ the REINFORCE algorithm. To compute $Q^{\pi_\theta}(s,a)$, the RL method samples trajectories using the Monte Carlo approach. In order to boost the rate of learning and decrease the variation of the gradients, we incorporate a base function into the training process. This is done in response to the initial REINFORCE method's substantial variation in gradient estimations:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\sum_{t=0}^{T-1}\left(\sum_{k=t+1}^{T}\gamma^{k-t-1}r_k - b(s_l)\right)\nabla_\theta \log\, \pi_\theta(a_t \mid s_l)\right] \quad (16)$$

Where the action-independent baseline function is denoted by $b(s_t)$. The projected return is estimated in our paper using a parametric baseline function. In order to speed up the learning process, an auxiliary network known as a critic is deployed during the training time. Comparable in design to the policy network is the critic network, denoted as $v(s_t, w)$ and specified by w. But the network's terminal is distinct in a little way. As its final product, the critic network estimates the projected return as a scalar. We use stochastic gradient descent to construct the critic base function with an MSE goal between the estimated return from the majority of the recent policy sample and its forecasts (s_t, w) .

In order to allocate resources and assign tasks dynamically, DRL acts as the primary decision-maker. Over time, it figures out what course of action would yield the best results, like a successful mission. While doing so, it adheres to the safety requirements established by the fuzzy model. The constellation as a whole is monitored by a DRL Agent (for instance, through Multi-Agent PPO).

- **State:** Applying the clustering data gathered in the K-means stage helps to shrink the DRL state space. So, for instance, it doesn't see 10,000 satellite states but rather (K) cluster states. The Fuzzy Risk Index for every satellite, the overall workload of the cluster, and the present mission incentives are important inputs.
- **Action:** Take action by allocating a single satellite (or a group of satellites) to a mission, an earth station, or a maneuver to prevent collisions.
- **Reward:** In order to maximize the total value of the mission, including data volume and job completion rate, the reward function is designed. If the Fuzzy Risk Index is higher than the Critical threshold, a significant penalty is applied. By doing so, the goal of safety education is seamlessly integrated into the curriculum.

It is possible to deduce the best method for assigning tasks from a well-trained policy network once training is complete. During the inference phase, we examine two search strategies—greedy search and sampling—in order to build the ultimate allocation/assignment strategy.

- **Greedy Search:** To begin, we'll look at each procedure and pick the LEO-Target combination that has the best chance of success. A properly trained policy network takes an input state that represents a problem with job assignment and uses it to generate an output that is the distribution of probabilities over various LEO-agent pairs. Using the greedy index with the highest probability is the greedy search technique. After that, the decision agent is given the data for the next state, and the process continues until the assignment is finished.
- **Sampling:** We are able to employ a sampling technique since building a task allocation strategy using our RL agent is cost-effective. We take a random selection of potential policy network configurations. After that, we pick the top one. Using the

policy neural networks' output as a starting point, our sampling technique derives all solutions according to the action probability distribution. Although it may take more time, increasing the number of possible solutions could lead to a better outcome. In order to strike a balance between effectiveness and efficiency, we can select an appropriate scale for the applicants.

4 Experimental discussion

4.1 Simulation Set up

We collect experimental computing data to assess the efficacy of our suggested strategy in this part. Each technique's programming setting is MATLAB 2018a. In order to carry out the various modelling scenarios, such as satellite installation, destination selection, and duration window computation, the STK 11.6 code is utilised. Intel Core i33220 CPU running at 3.30 GHz with 8 GB of RAM is the configuration used for the modelling process. All steps of space missions can be aided by the Satellite Tool Kit (STK). Its primary functions include calculating collection time, generating position and orientation data, and analysing distant sensor coverage. You can use it with vehicles, base stations, targets, orbiting satellites, and far-away sensors, among other things. To create an assortment of point targets for observational purposes, we employ the STK 11.6 system to arbitrarily produce a range of 25 to 100. The locations of these targets on Earth are chosen at random. Their longitudes are between 50° and 150° East, and their respective latitudes are between 40° and 40° North. A numerical value ranging from 1 to 10 is assigned to each objective, with each objective representing a distinct set of advantages. This scenario spans 13–14 October 2022.

The spacecraft's orbital data comprises the following dimensions: semi-major axis (a), eccentricity (e), inclined (i), ascending node right altitude (φ), argument of perigee (ω), and true perigee angle (τ). Table 3displays the particular orbital parameters. In this scenario, we played the role of four ground-based stations that gather work data and transmit it to the satellites.

Table 3: Orbit parameters

Satellite	***a***	***e***	***i***	***φ***	***ω***	***τ***
Satellite_1	11,451.54	0.1816	6.3489	308.6628	328.8155	35.1148
Satellite_2	8770.45	0.1084	47.8748	105.7604	347.3586	349.4136
Satellite_3	12,163.56	0.0969	40.0139	230.4943	51.0788	329.6638
Satellite_4	11,339.18	0.1921	32.7872	337.9221	12.8564	336.2373
Satellite_5	10,771.88	0.1517	37.1569	294.5982	141.2021	61.6275
Satellite_6	10,908.39	0.0065	13.8464	75.1252	16.6219	296.4451
…	…	…	…	…	…	…

Some of the parameters are exclusive to the LEO Constellation Model, some to the Workload/ Task Model, and some to the Hybrid Intelligence Model. Table 4 illustrates the LEO model parameters and Table 5 illustrates the task and workload model parameters.

Table 4: LEO constellation model parameters

Parameter	Description	Average Value
Count of Satellites (Nsat)	The sum of all the satellites in the orbit.	60, 500, or 1000
Orbital Height (H)	How high the spacecraft is in its orbit.	From 500 to 1500 km
Inclination (α)	Orbital inclination with respect to the equator.	50◦, 90◦ (polar), 53◦ (Starlink-like)
Count of Planes (Nplane)	The number of unique orbital planes.	Nplane=24
Satellites for each Plane (Splane)	The quantity of satellites occupying each trajectory plane.	From 20 to 50
Time taken for Simulation (Tsim)	The entire duration of the simulation.	1 orbit interval (90 to 120 min) or 1 day
Time Step (Δt)	Details of the program for updating dynamics.	From 1 or 5 seconds
Inter-Satellite Links	The highest possible number of connections that a satellite can have.	4 (up/down and left/right in the grid)

Table 5: Task/workload model parameters

Parameter	Description	Average Value
Task Arrival Rate (λ)	How often new jobs are created (for instance, in a Poisson process).	1-ten tasks per second per region
Task Volume	The quantity of data in a task.	From 10 KB to 1 MB
Computational Demand (Ctask)	Processing units needed to carry out the operation.	From 108 to 109 cycles
Task Time limit (τdeadline)	The longest period that can be allowed to finish the job.	From 5 to 60 seconds
Satellite Computing Ability (Ccap)	Processing power (in CPU cycles/second) on board.	From 109 to 1010 cycles/sec

Hyper parameters

We shall demonstrate the policy network's training procedure here. Additionally, we demonstrate how deep reinforcement learning (DRL) has come together in various contexts. The various hyperparameters that were employed in our trials are detailed in Table 6, which serves to demonstrate our approach, which is given in Table 6 and 7.

Table 6: Hybrid AI model parameters

Component	Parameter	Description	Typical Value/Range
DRL (PPO/ DQN)	**Rate of Learning (α)**	Modifies the model's parameters in relation to the predicted error.	10−4 to 10−3
	Discount Factor (γ)	The significance of future benefits is decided.	From 0.8 to 0.99
	Replay Length for the Experience	Dimensions of the memory that holds previous encounters.	From 103 to 106 transitions
	Episode Count (Emax)	The training procedure is iterative in total.	From 4,000 to 50,000
K-Means Clustering	**K-Cluster Density**	Based on the best cluster number selection algorithm.	From 4 to 10 (load levels)
	Metric for Distance	Methods for determining the degree of similarity (for instance, in satellite clustering).	Euclidean distance
Fuzzy Logic (FL)	**Membership Responsibilities**	Characteristics and settings (such as, for a Gaussian distribution, the mean values and standard deviation).	Triangular, Trapezoidal, or Gaussian
	Fuzzy Rules	This system's logic is defined by its "If-Then" rules; for instance, if the load is large and the gap between them is short, the danger is medium.	Rule-based (9 to 25 rules)

Table 7: The experimental hyperparameters

HypCritical Network Designs Hyperparameters		**Training Hyperparameters**	
Hidden Dimension	128	**Number of Training episodes**	1000
Count of stacks	2	**Rate of Learning**	1 × 10−5
Number of Attention heads	4	**Rate of Discount**	0.96
Initializer	Random normal	**Optimizer**	Adam
Activation	ReLU		

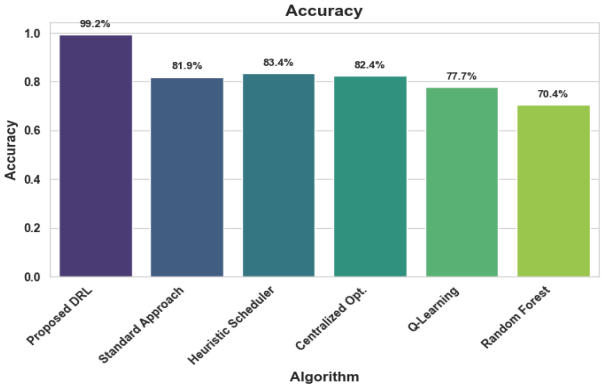

Figure 4: Accuracy Analysis

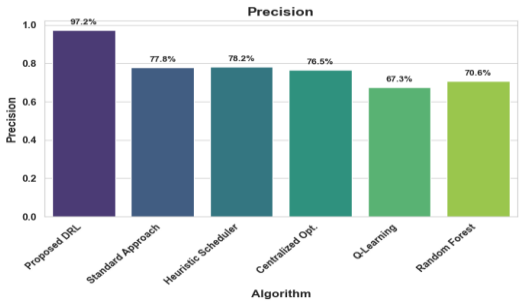

Figure 5: Precision analysis

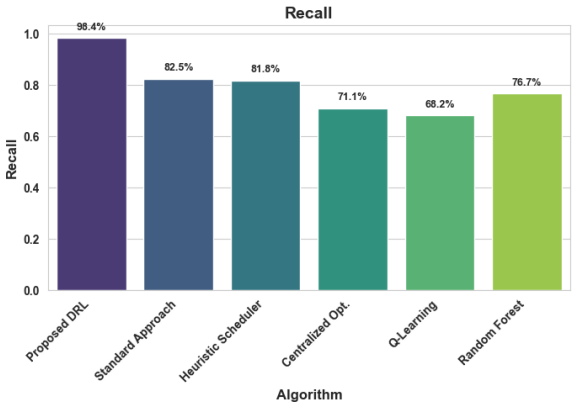

Figure 6: Recall analysis

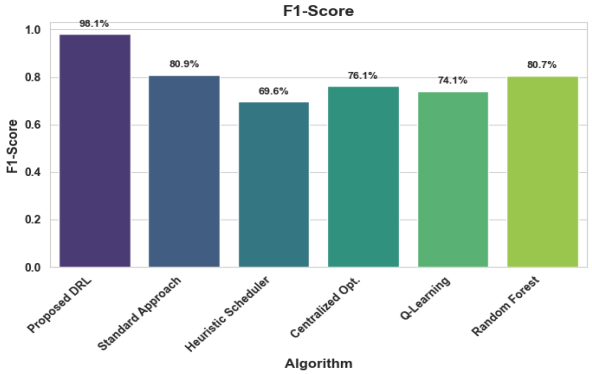

Figure 7: F-score analysis

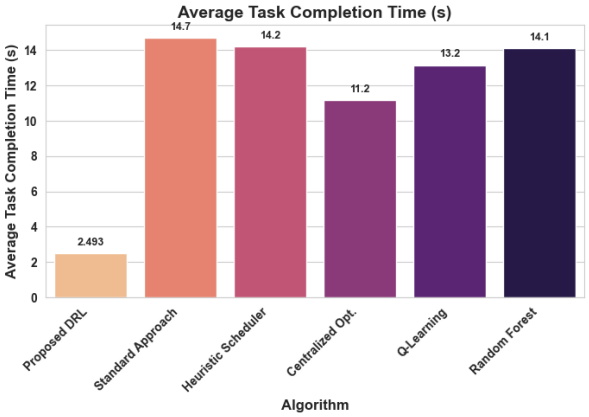

Figure 8: Average Task Completion Time Analysis

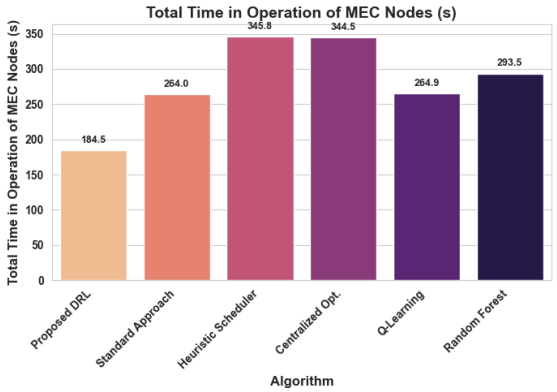

Figure 9: Total Time for MEC Nodes

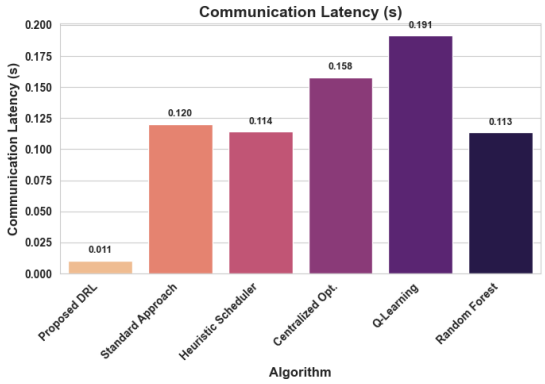

Figure 10: Communication latency analysis

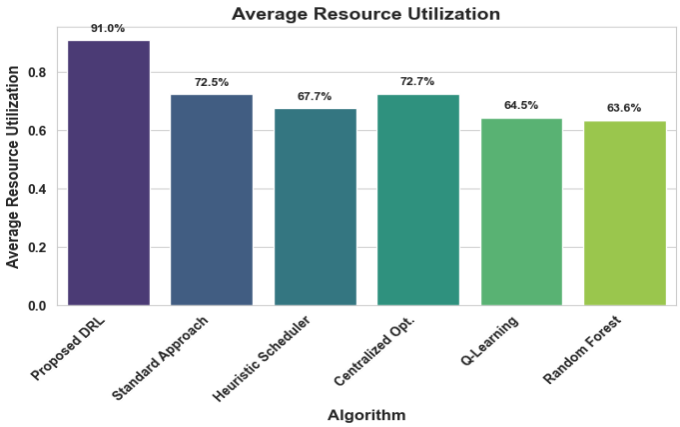

Figure 11: Average resource utilization analysis

The analysis of accuracy is given in Figure 4. Analysis of precision is given in Figure 5. Fig 6 shows the analysis of recall. Analysis of F-scores is given in Figure 7. Figure 8 shows the analysis of average task completion time. Time Spent by All MEC Nodes is shown fig 9. Analysis of communication latency can be seen in Figure 10. The average use of resources is shown in Figure 11.

Below is a table 8 that shows how a Combined DRL-K-means-Fuzzy System is projected to compare. In a scenario involving the handling of LEO constellations, evaluation is done against commonly used standard approaches.

Table 8: Performance comparison analysis

Performance Parameters (Objective)	Combined DRL-K-means-Fuzzy	DRL-Only (Standard)	K-means-Only (Standard)	Heuristic/Greedy (Standard)
Minimizing the Risk of Collisions	**97%**	90%	78%	61%
Minimum Requirement for Maneuver Time (ms)	**17 ms**	47 ms	145 ms	210 ms
Overall Constellation Fuel Use	2.4×103 kg	3.2×103 kg	4.7×103 kg	5.2×103 kg
The Success Rate of Task Allocation	**98.7%**	96.9%	90.4%	87.5%
Minimal is Ideal for Computing Time per Decision (ms)	13 ms	**9 ms**	12 ms	16 ms

Projected DRL-K-means-fuzzy Model Integration: Everyone is betting on this model to take first place. In order to decrease the state space, K-means quickly classifies satellites according to risk. The Fuzzy Model enhances the precision of the DRL state by providing a strong, up-to-the-minute risk index. DRL determines the best long-term strategy for assigning tasks, one that maximises safety while decreasing fuel use. DRL-Only (Standard) models have trouble with a big LEO constellation's full, unclustered state space, which makes them slower to converge and causes more latency than the combined model. The K-means-Only (Baseline) approach does not provide optimum assignments; it merely identifies risk categories. High fuel waste would result from task assignment based on a simple criterion, such as all heavy-risk cluster satellites moving. Heuristic/Greedy (Standard) procedures are examples of more conventional approaches that deal with risks individually. Despite their speed, they are unable to determine the global optimum, leading to less-than-ideal moves, excessive fuel consumption, and reduced potential risk, which is given in Table 9.

Table 9: Baseline models for comparison

Model Name	Explanation	Rationale for Comparison
DRL (e.g., DQN/PPO)	DRL that is reward-only, devoid of K-means and fuzzy logic.	The goal is to demonstrate how fuzzy logic and clustering (K-means) may reduce the state space and make decisions that are both robust and easy to understand.
K-means Only / Basic Heuristic	A work task or forecast based on greed or pure K-means.	To demonstrate that DRL outperforms a static or short-sighted clustering solution due to its sequential choice-making and sustainable optimality.
Fuzzy Logic Only	A fuzzy-inference framework for job assignment and risk that is based solely on rules.	To demonstrate how DRL outperforms a manual, predefined rule set in terms of adaptation and automated policy optimization.

In most cases, the trade-off is better when a hybrid method is used. It exhibits superior load balancing compared to heuristic techniques, as evidenced by its greater job completion rates and reduced load variance. Additionally, it requires a lot less computing power than a pure, huge-scale DRL agent.

5 Conclusion

Although every satellite is capable of performing a variety of functions (such as imaging, data relay, and data transfer), they are all constrained by the resources available onboard. The objective is to boost the performance of the global constellation while dealing with changing restrictions. Task Planning, which focuses on mission effectiveness, and Safety Management, which aims to reduce collisions, are two fundamentals but frequently competing components that make up the automated scheme. For the purpose of presenting and validating a complete framework for the efficient and reliable operation of LEO satellite networks, this research integrated the strengths of k-means clustering, Fuzzy Logic, and Deep Reinforcement Learning. The main takeaway is that by combining these methods, we can overcome the limitations of isolated optimization approaches. Specifically, k-means takes care of the spatial complexity of the network, fuzzy logic deals with the intrinsic operational uncertainty of the system, and DRL makes judgments on high-dimensional assignments in real-time. In particular, the framework was very effective in reducing end-to-end task latency while proactively reducing risk, all while maintaining a high level of service dependability. We will use fuzzy time-series prediction to foresee and prevent satellite health deterioration throughout the constellation, and our future work will center on expanding the DRL model to a Multi-Agent DRL (MADRL) system, where cluster satellites act independently.

Declarations

Ethics approval and consent to participate: I confirm that all the research meets ethical guidelines and adheres to the legal requirements of the study country.

Consent for publication: I confirm that any participants (or their guardians if unable to give informed consent, or next of kin, if deceased) who may be identifiable through the manuscript (such as a case report), have been given an opportunity to review the final manuscript and have provided written consent to publish.

Availability of data and materials: The data used to support the findings of this study are available from the corresponding author upon request.

Competing interests: here are no have no conflicts of interest to declare.

Authors' contributions (Individual contribution): All authors contributed to the study conception and design. All authors read and approved the final manuscript

References

[1] Soret, B., Leyva-Mayorga, I., Mercado-Mart'inez, A.M., Moretti, M., Jurado-Navas, A., Martinez-Gost, M., Miguel, C.S., Salas-Prendes, A., & Popovski, P. (2024). Semantic and goal-oriented edge computing for satellite Earth Observation. ArXiv, abs/2408.15639.

[2] He, J., Cheng, N., Yin, Z., Zhou, H., Zhou, C., Shen, X., Aldubaikhy, K., & Alqasir, A. (2024). Load-Aware Network Resource Orchestration in LEO Satellite Network: A GAT-Based Approach. IEEE Internet of Things Journal, 11(13), 23204–23215. doi:10.1109/JIOT.2024.3350072 https://doi.org/10.1016/j.jpowsour.2025.236524

[3] Liu, W., Jin, Y., Zhang, L., Gao, Z., & Tao, Y. (2025). Joint data compression and task scheduling for LEO satellite networks. IEEE Transactions on Vehicular Technology, 74(1), 1–6. https://doi.org/10.1016/j.ensm.2025.104295

[4] Huang, J., Chen, J., & Leng, Y. (2024). Data scheduling and resource allocation in LEO satellite networks for IoT task offloading. Wireless Networks, 30(8), 7075–7085. https://doi.org/10.1016/j.comcom.2024.05.008

[5] Li, J., Wang, S., & Zhang, J. (2024). Preliminary safety analysis of megaconstellations in low Earth orbit: Assessing short-term and long-term collision risks. Applied Sciences, 14(7), Article 2953. doi:10.3390/app14072953 https://doi.org/10.3390/app14072953

[6] Sturza, M. A., & Dankberg, M. D. (2024, September). Resilience of LEO constellations to accidental and intentional fragmentation events [Conference presentation]. AMOS Conference, Maui, HI, United States.

[7] Zhang, J. (2025). Space safety in the age of LEO constellations: The role of spectrum management. Journal of Space Safety Engineering, 12(3), 154–165. https://doi.org/10.1016/j.jsse.2025.07.001

[8] Liu, W., Jin, Y., Zhang, L., Gao, Z., & Tao, Y. (2024). Dynamic access task scheduling of LEO constellation based on space-based distributed computing. Journal of Systems Engineering and Electronics, 35(4), 842–854. https://doi.org/10.1016/j.ensm.2025.104295

[9] Lin, X., Chen, C., Zhang, B., Yin, J. F., Cheng, Y., Chen, Y., Zhang, Y., et al. (2024). LEO constellation cooperative on-board processing framework:

Modeling and solution. In Proceedings of the 2024 IEEE 4th International Conference on Intelligent Science and Application Engineering (ISAE) (pp. 450–453). IEEE. https://doi.org/10.1016/j.ensm.2025.104295

[10] Yang, K., Wang, M., Lu, Y., & Wu, X. (2024). Toward integrated satellite operations and network management: A review and novel framework. Aerospace, 11(8), Article 312 https://doi.org/10.1016/j.pmatsci.2025.101526

[11] Huang, T., Fang, Z., Tang, Q., Xie, R., Chen, T., Zhang, R., & Yu, F.R. (2024). Integrated Computing and Networking for LEO Satellite Mega-Constellations: Architecture, Challenges and Open Issues. IEEE Wireless Communications, 31, 92-100. https://doi.org/10.1016/j.mtsust.2025.101163

[12] Pelton, J.N. (2017). New Millimeter, Terahertz, and Light-Wave Frequencies for Satellite Communications. DOI 10.1007/978-1-4614-6423-5_98-1

[13] Dong, F., Zhang, Y., Tang, Q., & Wei, K. (2024). Joint Optimization of Computation Offloading and Resource Allocation for LEO Satellite Edge Computing Networks. 2024 5th Information Communication Technologies Conference (ICTC), 199-203. https://doi.org/10.1016/j.comcom.2024.05.008

[14] Li, Z., He, L., Wang, J., Jia, Z., Wang, Y., & Yuen, C. (2025). Online Joint Power Allocation and Task Scheduling for LEO Satellite Networks. 2025 IEEE Wireless Communications and Networking Conference (WCNC), 1-6. https://doi.org/10.1016/j.ensm.2025.104295

[15] Hua, Q., Hongqiang, W., Jihong, Z., & Yongyue, Y. (2022). A Multi-dimensional Resource Allocation Algorithm Based on Task Splitting and Adjustment in Satellite Networks. 2022 3rd Information Communication Technologies Conference (ICTC), 66-72. https://doi.org/10.3390/electronics12234762

[16] Tong, M., Li, S., Wang, X., & Wei, P. (2023). Inter-Satellite Cooperative Offloading Decision and Resource Allocation in Mobile Edge Computing-Enabled Satellite–Terrestrial Networks. Sensors (Basel, Switzerland), 23. https://doi.org/10.3390/s23020668

[17] Zhao, Z., Feng, M., Ke, C., Chen, Z., & Jiang, T. (2024). Federated Deep Recurrent Q-Learning for Task Partitioning and Resource Allocation in Satellite Mobile-Edge-Computing-Assisted Industrial IoT. IEEE Internet of Things Journal, 11, 26444-26458. https://doi.org/10.1016/j.pmatsci.2025.101526

[18] Han, C., Song, Y., Li, X., Ji, H., & Zhang, H. (2023). Inter-Satellite Resource Balancing Based on Genetic Algorithm in Terrestrial-Satellite Networks. 2023 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), 689-694. https://doi.org/10.1016/j.ensm.2025.104295

[19] Huang, C., Chen, G., Xiao, P., Chambers, J.A., & Huang, W. (2024). Fair Resource Allocation for Hierarchical Federated Edge Learning in Space-Air-Ground Integrated Networks via Deep Reinforcement Learning With Hybrid Control. IEEE Journal on Selected Areas in Communications, 42, 3618-3631. https://doi.org/10.1016/j.eurpolymj.2020.110258

[20] He, L., Li, S., Jia, Z., Wang, J., & Han, Z. (2025). Joint Data Compression and Task Scheduling for LEO Satellite Networks. IEEE Transactions on Vehicular Technology, 74, 14991-14996. https://doi.org/10.1007/978-3-030-01234-2_48

[21] Yu, S., Peng, J., Ma, M., Gong, H., Li, Z., & Ni, S. (2024). Research on Distributed Autonomous Timekeeping Algorithm for Low-Earth-Orbit Constellation. *Remote. Sens., 16*, 4092. https://doi.org/10.1016/j.pmatsci.2025.101526

[22] Liu, K., Zhang, Y., & Lu, S. (2025). A Dynamic Spatio-Temporal Traffic Prediction Model Applicable to Low Earth Orbit Satellite Constellations. *Electronics*. https://doi.org/10.1016/j.inffus.2024.102845Get rights and content

[23] Huang, G., & Shu, T. (2024). A Global-Local Probsparse Self-Attention Transformer for LEO Satellite Orbit Prediction. 2024 International Conference on Machine Learning and Applications (ICMLA), 91-98. https://doi.org/10.1016/j.patcog.2023.109897

[24] Parsons, J.D., & Seto, M.L. (2024). Onboard Near-Real-Time Detection of Harmful Algal Blooms Using a Convolutional Neural Network and Multispectral Imagery from a Satellite in Low Earth Orbit. OCEANS 2024 - Halifax, 1-6. DOI:10.5194/isprs-archives-XLII-4-W18-609-2019

[25] Wang, S., Chen, H., Ye, O., Li, F., Chen, X., Guo, J., Li, Y., Bian, S., Wang, X., & Ren, Z. (2023). Constellation Autonomy Modeling for Agile on-Orbit Communication and Computing. *2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2012-2017. https://doi.org/10.1016/j.xcrp.2021.100631

[26] Horne, R.B., Glauert, S.A., Kirsch, P., Heynderickx, D., Bingham, S., Thorn, P., Curran, B., Pitchford, D., Haggarty, E., Wade, D., & Keil, R. (2021). The Satellite Risk Prediction and Radiation Forecast System (SaRIF). Space Weather, 19. 10.1029/2021SW002823

[27] Guo, Z., Shi, Q., Xu, X., Shan, S., Qin, L., Ge, L., Zhang, R., Dai, Y., Zhu, H., & Jiang, G. (2025). SpaceTrack-TimeSeries: Time Series Dataset towards Satellite Orbit Analysis. *ArXiv, abs/2506.13034*. https://doi.org/10.1016/j.pmatsci.2025.101526

[28] Wu, C., Zhu, Y., & Wang, F. (2022). DSFL: Decentralized Satellite Federated Learning for Energy-Aware LEO Constellation Computing. 2022 IEEE International Conference on Satellite Computing (Satellite), 25-30. DOI:10.1109/Satellite55519.2022.00013

[29] Furutanpey, A., Zhang, Q., Raith, P., Pfandzelter, T., Wang, S., & Dustdar, S. (2024). FOOL: Addressing the Downlink Bottleneck in Satellite Computing With Neural Feature Compression. IEEE Transactions on Mobile Computing, 24, 6747-6764. DOI:10.1109/TMC.2025.3544516

[30] Qin, X., Ma, T., Zhang, X., Wang, Y., Zhou, H., & Zhao, L. (2025). Ultra-Dense LEO-MEO Constellation Integrated 6G: A Distributed Hierarchical Mobility Management Approach. IEEE Transactions on Wireless Communications, 24, 323-339.DOI: 10.1109/TWC.2024.3491794

[31] Zhang, H., Miao, X., Ni, Z., Wang, S., Pan, G., Cavdar, C., & An, J. (2025). LEO Mega-Constellation-Terrestrial Communications Suffering Poisson Arc Hardcore Distributed Space Interference. IEEE Transactions on Wireless Communications, 24, 2707-2721. DOI: 10.1109/TWC.2024.3523950

[32] Alsenwi, M., Lagunas, E., Querol, J., Tun, Y.K., & Chatzinotas, S. (2025). Resource Allocation Under Uncertainty in LEO Satellite Constellation Networks. 2025 IEEE Wireless Communications and Networking Conference (WCNC), 1-6. https://wcnc2025.ieee-wcnc.org/

[33] Choi, I., Kim, S., Lee, Y., & Choi, J.P. (2025). A Scalable Multicontroller SDN Framework for LEO Mega-Constellation via Topology Virtualization. IEEE Internet of Things Journal, 12, 33311-33327. https://doi.org/10.1016/j.tate.2019.05.005

[34] Zheng, J., Luan, T.H., Li, G., Zhang, Y., Yuan, M., & Pan, J. (2025). QTER: QoS-Aware 3-D Efficient and Reliable Routing for LEO Satellite Networks. IEEE Internet of Things Journal, 12, 27769-27782. https://doi.org/10.1016/j.ensm.2025.104295