Deep Q Networks with Centralized Learning over LEO Satellite Networks in a 6G Cloud Environment

Tiago Koketsu Rodrigues, and Nei Kato,
Graduate School of Information Sciences, Tohoku University, Sendai, Japan
Emails: {tiago.gama.rodrigues, kato}@it.is.tohoku.ac.jp

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10000709

***Abstract*—With 6G networks, we can expect more devices to start operating in remote areas, away from the conventional network infrastructure. With an increase in the number of devices, we should also see more data that needs to be processed and analyzed. An adequate response to this scenario is using satellite networks to reach remote devices and transfer the big data generated by them to be analyzed in cloud servers through Machine Learning models. However, while this is a good solution for data analysis, it can run into bottlenecks caused by long transmission in the limited channels of satellite networks. In this paper, we will model and analyze this service model, allowing us to identify the limitations of centralized learning over satellite networks. This study manages to determine when centralized learning with remote devices is a viable solution and when it needs to be complemented by other techniques due to poor performance caused by long transmission and overloaded communication channels.**

I. Introduction

With the advent of the 6th generation of computer networks (6G) will come the goal of maintaining a ubiquitous connection, which means bringing network access to even the most remote of areas [1], [2]. To fulfill this goal, the most appropriate tool is satellite networks. Low Earth Orbit (LEO) satellites in particular can cover a wide area while remaining close enough to the surface that access latency is, although not insignificant, manageable. With LEO satellites, rural areas, open ocean, disaster-stricken locations, i.e., areas where conventional network infrastructure is not fully available, can now have access to the Internet [2]–[4].

Devices in those locations, thus, can be used to collect data and run applications. Fig. 1 illustrates how satellites can bridge those devices to allow their data to reach cloud servers. However, this opens up another issue, where a massive amount of data will come from these remote parts of the network edge. As is usual with scenarios involving big data, using Machine Learning (ML) models to analyze the information is the best way to detect patterns and infer optimal actions [1], [5]–[8]. Thus, the data will have to be processed by servers to train ML models to make decisions for the applications and improve service, e.g., deciding what action to take in military missions or sensing expeditions through the sensed data sent by devices [1], [2]. The problem comes from transmitting the data from these devices to the servers. The latency in this communication is significant and can lead to the servers seeing infrequent updates on the environment and outdated information, which is inefficient in training the ML models [6], [9].

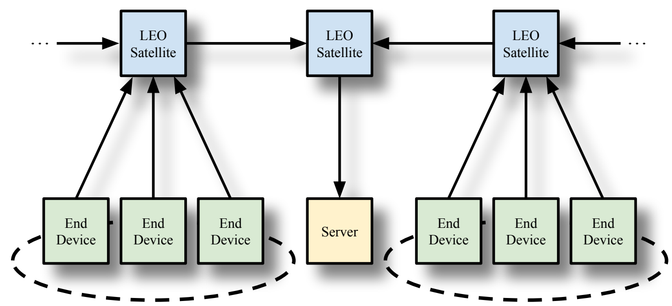

Fig. 1. Satellite networks can help end devices in remote areas send data to centralized cloud servers for processing.

A possible solution would be moving towards distributed learning, where the ML models are trained near the data sources [10]–[12]. As promising as this alternative is, there are problems with bias created by using a smaller sample of data, overhead caused by merging multiple ML models from different parts of the network, and, most importantly, distributed learning is more prone to getting stuck in local optima and less efficient in learning in general when compared to centralized learning [13]–[15]. Therefore, it should not be used as a catch-all solution. It is preferable to use centralized learning when possible while understanding the limitations of centralized solutions and using this understanding to complement the approach with distributed learning when needed.

Thus, in this paper, we will focus on studying centralized learning done through an LEO satellite network to analyze data from remote end devices. We will study the communication delay of this system and how it affects the training of the ML model to serve as a base for a framework that decides when centralized learning is not desirable. The remainder of the paper is organized as follows. Section II analyzes the literature and situates the contributions of this research within what is missing in the state of art. Section III gives a mathematical model for the transmission delay of the satellite network when supporting a centralized learning service. Section IV presents the performance of this centralized learning service model, analyzing how the delay affects the training of the ML model. Finally, Section V concludes the paper and provides some possible future research directions.

II. Contributions to Literature

There has been plenty of research on the use of ML models to analyze big data generated by end devices [1], [5], [6], [8], [16]. Particularly, Deep Q Networks (DQN) have been used successfully lately due to their ability to react to changes in the environment in real-time, updating the model and moving towards more optimal actions in the applications [10]. Nonetheless, the consensus in ML literature is that centralized learning is not scalable due to the long latency needed to transmit data from the edge to the remote core [13], [17]–[19], which is particularly true in satellite networks [11]. In response to this, there have been extensive efforts to utilize distributed learning schemes, where the data is used to train multiple ML models on the edge of the network that are later merged to obtain a model with a global view of the data [5], [10], [15], [17]. Distributed learning has also been used in conjunction with satellite networks to bring big data analysis to remote areas, in response to the limitations of centralized learning [11], [12], [14]. However, as mentioned previously, distributed learning is not as efficient processing-wise as centralized learning, taking longer to reach optimum points, and edge servers are in general slower to process ML iterations.

Given this state of the art, the contributions of this paper are two-fold.

- We provide a model to estimate the communication delay associated with training an ML model with centralized learning over an LEO satellite network. This model can be used for measuring latency and this information can be used for understanding the cost of centralized learning.
- We provide a detailed study of how variables can affect not only the communication latency of centralized learning over an LEO satellite network but also how the efficiency of the ML model changes with this variation in latency. This study is important for understanding the effectiveness of centralized learning.

Both these points are novel and not found in the literature as there are no other dedicated studies on modeling and analysis of the centralized learning service model for satellite networks as far as we know. They are also important for deciding when to use centralized learning due to its processing benefits and when to use distributed learning to avoid the drawbacks of long latency (e.g., outdated information, infrequent model updates).

III. Centralized Training Delay

We assume that the end devices in our system model are divided into different cells. Additionally, each cell is covered by one, and only one, LEO satellite. Thus, there are equal amounts of satellites and cells. The cells and satellites are divided into orbits. Satellites in the same orbit rotate through the cells in their corresponding orbit. Additionally, we will assume the limitation that satellites can only communicate with their adjacent satellites in their orbit or entities in the cells they are currently overflying [20]. Each cell, in addition to end devices, houses one amplify-and-forward (AaF) relay that the devices use to communicate with the satellite and vice-versa [13]. Finally, this whole system is connected to a single centralized cloud server, which has a wired link to a base station per orbit. This base station is located in what we will call the origin cell of the orbit. For simplicity, we assume the origin cell of each orbit has no end devices. Besides the wired link between the base stations at origin cells and the server, all other communication is wireless and goes through NOMA channels, that avoid interference but require an open channel, as is expected of 6G networks [21]. Wireless transmission is dictated by the Shannon-Hartley theorem when determining the send rate (Eq. (1)), with the path loss being calculated through a millimeter-wave channel model (Eq. (2)) that is also mentioned as inherent to 6G environments [11], [22]. B is the bandwidth, A is the number of antennas at the transmitter, γ is the transmission power at the transmitter, G is the total antenna gain between transmitter and receiver, R is the Rayleigh fading coefficient, L is the path loss, N is the noise density, $q_{\text{intercept}}$ is the path floating intercept, q_{average} is the path loss average exponent, and Λ is the distance between transmitter and receiver.

$$S_x = B \cdot \left(1 + \frac{A \cdot 10^{\gamma+G+R-L-3}}{N \cdot B}\right) \tag{1}$$

$$L = q_{\text{intercept}} + 10 \cdot q_{\text{average}} \cdot \log_{10}\Lambda \tag{2}$$

Eq. (1) is applied to wireless channels. Meanwhile, for wired channels, we will assume that the medium is stable enough that the send rate can be pre-determined [21]. It is worth mentioning that, in both wired and wireless channels, the propagation delay also plays a role, being obtained by dividing the distance of transmission by the speed of light.

End devices constantly sense the environment around them and perform actions. They utilize an ML model that takes as input features in the environment and outputs the action that is expected to have the highest reward. This action may modify the environment. For each action taken, the device records the state before and after the action, the chosen action, and the reward received. This data is sent to the AaF relay of the cell, relayed to the satellite in view, and then forwarded through inter-satellite links (ISLs) until it reaches the satellite over the origin cell of the orbit. From there, it is forwarded to the cell base station and sent to the server, where it is used to train an ML model there. Periodically, this ML model is broadcast back to the end devices (taking the reverse path used by the device-generated data) so that the end devices have a better model. In this paper, we will focus on the transmission of data from the device to the server, as it is used to update the ML model. If this transmission takes too long, the model will be seldom updated and/or updated with old information, making the action taken by the device sub-optimal.

We will divide the data transmission into two phases. The fronthaul transmission (FTX) will go from the data leaving the end device until it first arrives at an LEO satellite. The

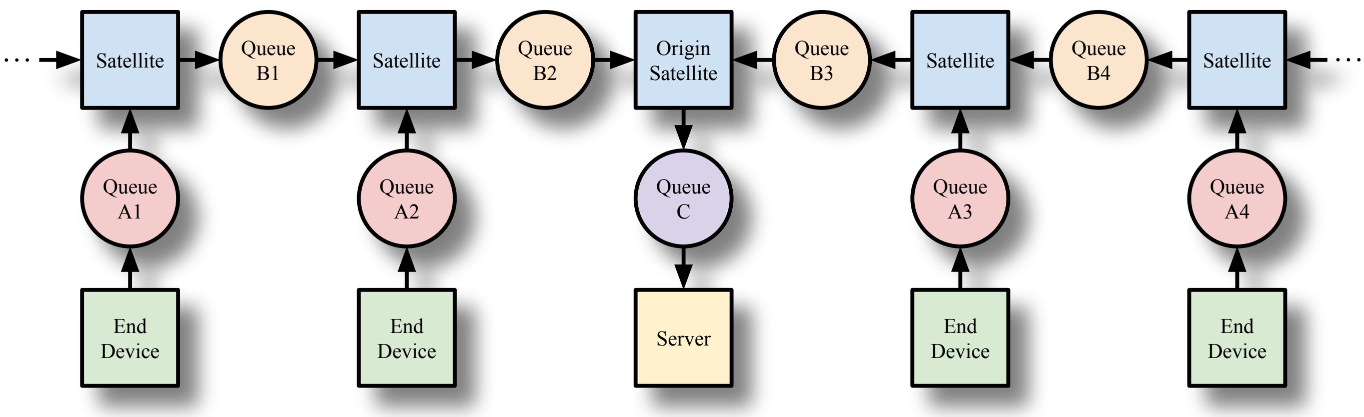

Fig. 2. How data from multiple end devices in different cells go through different queues to reach the central server.

backhaul transmission (BTX) will go from the first arrival at a satellite until the data arrives at the central server. In this transmission, there are 3 important types of queues. The first one, Q^m_{FTX}, corresponds to the end devices of cell m waiting for an available channel to send to the AaF relay and the satellite in view of m (we will consider the communication in FTX as one single link, instead of a link between device and relay and another between relay and satellite, for simplicity). The second queue, Q^n_{ISL}, corresponds to waiting for the ISL between satellite n and the next-hop satellite to be available to send the device data toward the origin cell satellite. The final queue, Q^p_{origin}, corresponds to the transmission from the satellite in view of the origin cell of orbit o to the ground base station in the cell, waiting for a channel to be available for such link. Fig. 2 shows how data has to go through these queues between the end devices and the cloud server; queues labeled A correspond to Q^p_{origin}, queues labeled B correspond to Q^n_{ISL} and queue C is Q^p_{origin}. Additionally, we will represent each queue by a tuple $(k^i_x, \mu^i_x, \lambda^i_x)$ of, respectively, the number of channels, the time needed for transmission to be completed once channel access is gained, and the arrival rate of data packets. Note that all these queues are M/M/k queues. As such, we can calculate the estimated total time spent on the queue for each data packet (Eq. (3)) as well as the exit rate of each queue (Eq. (4). Eq. (5) and Eq. (6) are auxiliary values used in the calculation.

$$T^i_x = \frac{W^i_x}{(1-\rho^i_x)\cdot k^i_x \cdot \mu^i_x} + \frac{1}{\mu^i_x} \tag{3}$$

$$Z^i_x = \frac{1-\rho^i_x}{\lambda^i_x + \mu^i_x} + \frac{\rho^i_x \cdot k^i_x}{\mu^i_x} \tag{4}$$

$$\rho^i_x = \frac{\lambda^i_x \cdot \mu^i_x}{k^i_x} \tag{5}$$

$$W^i_x = \frac{\frac{\left(k^i_x \cdot \rho^i_x\right)^{k^i_x}}{k^i_x!}}{\sum_{j=0}^{k^i_x-1}\left(\frac{(k^i_x\cdot\rho^i_x)^j}{j!}\right)\cdot(1-\rho^i_x) + \frac{(k^i_x\cdot\rho^i_x)^{k^i_x}}{k^i_x!}} \tag{6}$$

For Q^m_{FTX}, k^m_{FTX} corresponds to the number of antennas in a LEO satellite (Eq. (7)). μ^m_{FTX} is the time needed to transmit the data packet from the end device to the LEO satellite, which itself depends on the send rate of this link (Eq. (8)). λ^m_{FTX} corresponds to the data generation rate of the end devices (Eq. (9)). A_{LEO} is the number of antennas in an LEO satellite, α is the data packet size, S_{FTX} is the send rate in FTX, D_m is the number of devices in m, and β is the average data generation rate of a single device.

$$k^m_{\text{FTX}} = A_{\text{LEO}} \tag{7}$$

$$\mu^m_{\text{FTX}} = \frac{\alpha}{S_{\text{FTX}}} \tag{8}$$

$$\lambda^m_{\text{FTX}} = \frac{D_m}{\beta} \tag{9}$$

For Q^n_{ISL}, k^n_{ISL} corresponds to the ISL link itself (Eq. (10)). μ^n_{ISL} is the time needed to transmit a data packet between two satellites, which depends on the send rate of the ISL (Eq. (11)). Lastly, λ^n_{ISL} is obtained as a sum from the data packets coming from Q^m_{FTX} while n is in view of m and the data packets being forwarded from other satellites to n (Eq. (12)). S_{ISL} is the throughput at the ISL and $\dot{N}$ is the set of all satellites that send packets directly to n (adjacent satellites that are further from the original cell).

$$k^n_{\text{ISL}} = 1 \tag{10}$$

$$\mu^n_{\text{ISL}} = \frac{\alpha}{S_{\text{ISL}}} \tag{11}$$

$$\lambda^n_{\text{ISL}} = Z^m_{\text{FTX}} + \sum_{\dot{n}\in\dot{N}} Z^{\dot{n}}_{\text{ISL}} \tag{12}$$

Finally, for Q^p_{origin}, k^p_{origin} is the number of antennas in the LEO satellite (Eq. (13)). μ^p_{origin} is how long it takes to transmit a data packet from the satellite to the ground base station (Eq. (14)). $\lambda^p_{\text{origin}}$ comes from the exit rate of all satellites that send

TABLE I
BASIC PARAMETERS FOR A 6G SATELLITE NETWORK

Number of orbits	2
Satellites per orbit	11
Cells per orbit	11
End devices per cell	50
Average data generation time	3min
Training time	1ms
Timeslot length	5min
Total observation time	3h
Latency from base station to server	100ms
Number of antennas (satellite)	48
Number of antennas (base station)	64
Packet size (ML model)	125B
Packet size (experience tuple)	12.5kB

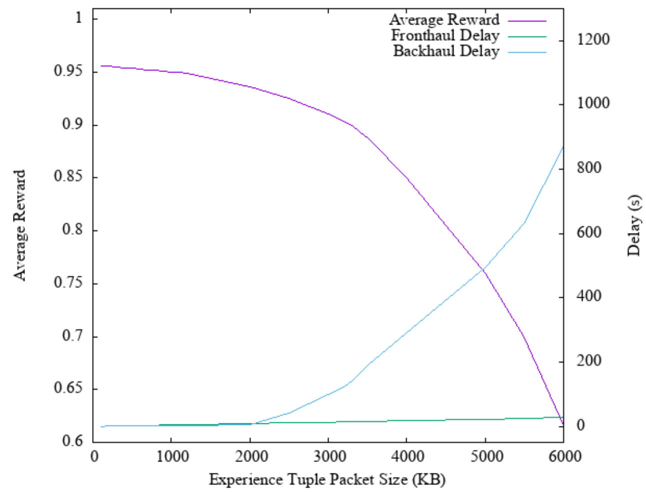

Fig. 3. The effects that payload size have on delay and reward.

packets directly to the satellite in view of the origin cell (Eq. (15)). S_{BS} is the send rate between the LEO satellite above the origin cell and its ground base station, $\dot{p}$ is the set of all satellites adjacent to the origin cell satellite in orbit p.

$$k^p_{\text{origin}} = A_{\text{LEO}} \tag{13}$$

$$\mu^p_{\text{origin}} = \frac{\alpha}{S_{\text{BS}}} \tag{14}$$

$$\lambda^p_{\text{origin}} = \sum_{n \in \dot{p}} Z^n_{\text{ISL}} \tag{15}$$

Thus, it is now trivial to estimate the transmission delay of each data packet, by adding the expected total queue system time at each queue the data packet will pass through, plus the propagation time at each link, and finally the wired link transmission delay between the origin cell base station and the server (which itself, being a wired communication, should be dominated by propagation).

IV. SERVICE MODEL PERFORMANCE

It is notable that, according to the model in the previous section, data packets must go through multiple queues. This makes the system particularly vulnerable to high loads since queues with full occupation will cause an exponential growth in the delay as the queues just get longer and longer. To further analyze this phenomenon, we measure the delay, both FTX and BTX, while varying the workload. This variation can be done by increasing the data packet size, which increases how long each transmission takes. Another possibility is increasing the number of individual data packets, either by having more end devices or more frequent data generation by each device. The results shown in this section use as parameter values the numbers shown in Table I, unless the graphs directly say otherwise. These values are what is expected to be seen in 6G satellite networks [11], [23]–[25]. To measure the reward, we had the end devices traverse through identical mazes, going from an initial position to a goal position, where after each step they get rewarded a value between 0 and 1 proportional to their distance to the goal position. The values in the graph are the average reward of the devices at the end of the observation.

The ML model used is a DQN, due to its ability to adapt to the states of the end devices as they perform actions and change position through reinforcement learning [10]. The complexity and training time of a DQN depends on the neural network layout and the problem tackled. An in-depth analysis is out of scope for this paper, but experiments show that for the maze traversal 2 hidden layers with 100 nodes each and a training batch size of 10 experiences yield good results. As explained in the previous section (and illustrated in Fig. 1), end devices are deployed in cells, whereas the cells and satellites form circular orbits. The location of the end devices is decided via random distribution inside its cell. The results shown come from the average of 100 different simulations.

In Fig. 3, we gradually increase the data packet size and measure transmission delay and average reward. We can see that, for small packet sizes, there is an insignificant variance in both FTX and BTX delay, both staying at low values. This results in a steady stream of data to the central server that allows the model to be trained well and the end devices to achieve near-maximum reward. However, after 2000KB, we can see the effects of the queues becoming overloaded, as the BTX delay starts to quickly grow. Due to this, the central server will get data packets less often, which is reflected in a similarly steady decrease in the rewards. This situation worsens as the data packet grows even further. Meanwhile, as there is only one queue in FTX, the increase in FTX delay remains insignificant for the whole observation interval, indicating that the overload happens from the ISLs forward.

Next, we increase the number of end devices per cell, as seen in Fig. 4. When we have too few devices, there is very little data for the ML model to be trained, which results in poor performance and bad rewards. This changes as we use more devices, giving more experience to the model and allowing the end devices to obtain near-maximum reward. However, on the other extreme, too many devices mean more data packets, which in turn overload the queues. This is

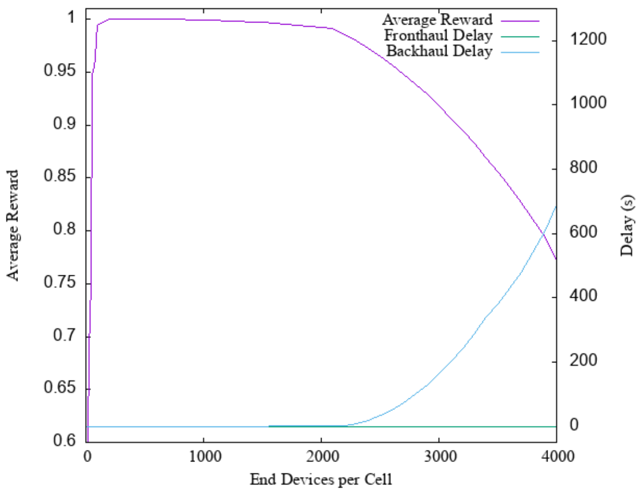

Fig. 4. The effects that the number of devices have on delay and reward.

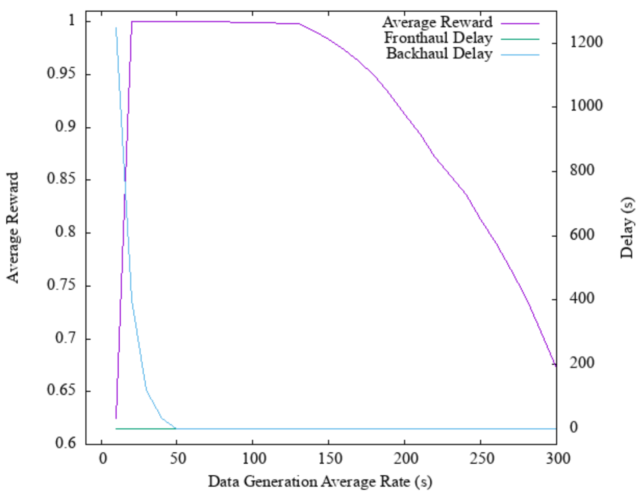

Fig. 5. The effects that data generation rate has on delay and reward.

evidenced by how, with more end devices, after around the 2000 devices per cell mark, we see a quick increase in the BTX delay. Consequently, the data packets take longer to reach the server, which observes a delayed report of the device environments and trains the ML model with this outdated data set. The result is a corresponding decrease in the achieved reward for the end devices. As observed previously, there is no significant change in the FTX delay, indicating that the queues being overloaded are in the BTX, showing the vulnerability of the system from the ISLs forward.

Finally, we evaluate the performance of the system while varying the frequency with which each device sends data. Results can be seen in Fig. 5, which changes how long on average each device takes to generate one data packet. As expected, the curves have similar behavior as Fig. 4. If the frequency of data generation is too high, BTX delay is also very high. This is caused by the system having too many data packets, with the data packets not being able to leave the queue before new packets arrive, overloading the queues' occupation. The result is that data packets take too long to reach the server and the model, as before, is trained with outdated data and consequently causes the devices to obtain less than optimal rewards. By lowering the frequency a bit, we can relieve the queues and allow the ML model to be trained with more up-to-date data. This allows the devices to reach near maximum reward. But, if the data generation becomes too infrequent, even though the delay does not change, there is not enough data to properly train the model and the result is also a worse performance. As before, FTX delay does not see any significant changes, showing once more that the queues in the ISLs forward are the ones more heavily affected.

V. Conclusion and Future Directions

As seen from the results, we can safely conclude that centralized learning through satellite networks can reliably train ML models to reach near maximum rewards. However, the system is also susceptible to poor performance in some scenarios. First, if there is not enough data, the average reward achieved goes down. However, this is more a characteristic of ML models than caused by the network architecture. On the other hand, if there is too much workload in the system, the training of the ML model is also subpar and leads to sub-optimal performances. The results show that the areas more easily affected by this are indeed the ISLs sections of the satellite network and parts of the service model beyond that. Thus, we can conclude that, in certain scenarios, centralized learning through satellite networks alone is not capable of reaching acceptable results. As FTX is not as easily affected as ISLs, one alternative to solve this issue is complimenting centralized learning with distributed solutions that rely less on the BTX and long transmissions between satellites. In the future, we would like to experiment with this, using the models here to estimate when centralized learning may fail and, in those cases, test and analyze whether distributed learning supported by the 6G LEO satellite network assumed in this paper will complement the centralized learning.

Acknowledgment

This work was conducted under the national project, Research and Development of Ka-Band Satellite Control for Various Use Cases as part of the Research and Development for Expansion of Radio Wave Resources (JPJ000254), supported by the Ministry of Internal Affairs and Communications (MIC), Japan.

References

[1] M. Centenaro, C. E. Costa, F. Granelli, C. Sacchi, and L. Vangelista, "A Survey on Technologies, Standards and Open Challenges in Satellite IoT," *IEEE Communications Surveys and Tutorials*, vol. 23, no. 3, pp. 1693–1720, May 2021.

[2] J. Zhang, X. Zhang, P. Wang, L. Liu, and Y. Wang, "Double-Edge Intelligent Integrated Satellite Terrestrial Networks," *China Communications*, vol. 17, no. 9, pp. 128–146, September 2020.

[3] S. Zhang, G. Cui, Y. Long, and W. Wang, "Joint Computing and Communication Resource Allocation for Satellite Communication Networks with Edge Computing," *China Communications*, vol. 18, no. 7, pp. 236–252, July 2021.

[4] Z. Song, Y. Hao, Y. Liu, and X. Sun, "Energy-Efficient Multiaccess Edge Computing for Terrestrial-Satellite Internet of Things," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14 202–14 218, March 2021.

[5] S. Hong and J. Chae, "Distributed Online Learning With Multiple Kernels," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, August 2021, Early Access.

[6] X. Cao, J. Zhang, and H. V. Poor, "Constrained Online Convex Optimization With Feedback Delays," *IEEE Transactions on Automatic Control*, vol. 66, no. 11, pp. 5049–5064, November 2021.

[7] M. Xiong, Y. Li, L. Gu, S. Pan, D. Zeng, and P. Li, "Reinforcement Learning Empowered IDPS for Vehicular Networks in Edge Computing," *IEEE Network*, vol. 34, no. 3, pp. 57–63, June 2020.

[8] T. K. Rodrigues, K. Suto, J. Liu, and N. Kato, "Machine Learning Meets Computation and Communication Control in Evolving Edge and Cloud: Challenges and Future Perspective," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 1, pp. 38–67, March 2020.

[9] X. Cao, J. Zhang, and H. V. Poor, "Impact of Delays on Constrained Online Convex Optimization," in *Proceedings of the 2019 53rd Asilomar Conference on Signals, Systems, and Computers*, November 2019, pp. 1578–1581.

[10] I. Saeed, T. Alpcan, S. M. Erfani, and M. B. Yilmaz, "Distributed Nonlinear Model Predictive Control and Reinforcement Learning," in *Proceedings of the 2019 Australian New Zealand Control Conference (ANZCC)*, November 2019, pp. 1–3.

[11] B. Zhao, G. Ren, X. Dong, and H. Zhang, "Distributed Q-Learning Based Joint Relay Selection and Access Control Scheme for IoT-Oriented Satellite Terrestrial Relay Networks," *IEEE Communications Letters*, vol. 25, no. 6, pp. 1901–1905, June 2021.

[12] K. Li, H. Zhou, Z. Tu, W. Wang, and H. Zhang, "Distributed Network Intrusion Detection System in Satellite-Terrestrial Integrated Networks Using Federated Learning," *IEEE Access*, vol. 8, pp. 214 852–214 865, December 2020.

[13] K. Liu, W. Quan, D. Gao, C. Yu, M. Liu, and Y. Zhang, "Distributed Asynchronous Learning for Multipath Data Transmission Based on P-DDQN," *China Communications*, vol. 18, no. 8, pp. 62–74, August 2021.

[14] D. Li, H. Wang, Z. Yang, Y. Gu, and S. Shen, "An Online Distributed Satellite Cooperative Observation Scheduling Algorithm Based on Multiagent Deep Reinforcement Learning," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 11, pp. 1901–1905, July 2021.

[23] S. Hendaoui, A. Mannai, and N. Zangar, "Cognitive CQI/5QI Based Scheme for Software Defined 5G Hybrid Satellite-Terrestrial Network:

[15] Z. Huang, F. Liu, M. Tang, J. Qiu, and Y. Peng, "A Distributed Computing Framework Based on Lightweight Variance Reduction Method to Accelerate Machine Learning Training on Blockchain," *China Communications*, vol. 17, no. 9, pp. 77–89, September 2020.

[16] X. Cao, J. Zhang, and H. V. Poor, "A Virtual-Queue-Based Algorithm for Constrained Online Convex Optimization With Applications to Data Center Resource Allocation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 4, pp. 703–716, August 2018.

[17] M. Chen, D. G. abd K. Huang, W. Saad, M. Bennis, A. V. Feijan, and V. H. Poor, "Distributed Learning in Wireless Networks: Recent Progress and Future Challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, October 2021.

[18] Z. Wang, J. Qiu, Y. Zhou, Y. Shi, L. Fu, W. Chen, and K. B. Letaief, "Federated Learning via Intelligent Reflecting Surface," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 808–822, February 2022.

[19] L. Cui, Y. Qu, G. Xie, D. Zeng, R. Li, S. She, and S. Yu, "Security and Privacy-Enhanced Federated Learning for Anomaly Detection in IoT Infrastructures," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3492–3500, May 2022.

[20] Z. Zheng, N. Hua, Z. Zhong, J. Li, Y. Li, and X. Zheng, "Time-Sliced Flexible Resource Allocation for Optical Low Earth Orbit Satellite Networks," *IEEE Access*, vol. 7, pp. 56 753–56 759, May 2019.

[21] T. K. Rodrigues, J. Liu, and N. Kato, "Offloading Decision for Mobile Multi-Access Edge Computing in a Multi-Tiered 6G Network," *IEEE Transactions on Emerging Topics in Computing*, June 2021, Early Access.

[22] T. K. Rodrigues, K. Suto, and N. Kato, "Edge Cloud Server Deployment with Transmission Power Control through Machine Learning for 6G Internet of Things," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 4, pp. 2099–2108, December 2021.

Slicing for Ultra Reliability And Video Congestion Offloading," in *Proceedings of the 2020 International Symposium on Networks, Computers and Communications (ISNCC)*, October 2020, pp. 1–7.

[24] I. Leyva-Mayorga, B. Soret, M. Roper, D. Wubben, B. Matthiesen, A. Dekorsy, and P. Popvoski, "LEO Small-Satellite Constellations for 5G and Beyond-5G Communications," *IEEE Access*, vol. 8, pp. 184 955–184 964, October 2020.

[25] T. K. Rodrigues, J. Liu, and N. Kato, "Application of Cybertwin for Offloading in Mobile Multiaccess Edge Computing for 6G Networks," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16 231–16 242, July 2021.