Adapting to Dynamic LEO-B5G Systems: Meta-Critic Learning Based Efficient Resource Scheduling

Yaxiong Yuan, *Student Member, IEEE*, Lei Lei, *Member, IEEE*, Thang X. Vu, *Member, IEEE*, Symeon Chatzinotas, *Senior Member, IEEE*, Osvaldo Simeone, *Fellow, IEEE*, and Sumei Sun, *Fellow, IEEE*

***Abstract*—In beyond 5G systems, low earth orbit (LEO) satellite-assisted communications have been widely considered to provide wide coverage and cost-efficient data services. Such dynamic space-terrestrial networks have exponentially increased the degrees of freedom in network management. In this paper, we address two practical issues for an over-loaded LEO-terrestrial system. First, how to efficiently schedule resources to serve the massive connected users, such that more data and users can be delivered/served. Second, how to make the algorithmic solution more resilient in adapting to wireless dynamic environments. In solution development, we propose an iterative suboptimal algorithm to provide an offline benchmark. To adapt to unforeseen variations, we propose an enhanced meta-critic learning algorithm (EMCL), where a hybrid neural network for parameterization and Wolpertinger policy for action mapping are designed in EMCL. The results demonstrate EMCL's effectiveness and fast-response capabilities in over-loaded systems and in adapting to dynamic environments than previous actor-critic and meta-learning methods.**

***Index Terms*—LEO satellites, resource scheduling, reinforcement learning, meta-critic, dynamic environment.**

I. Introduction

In practical 5G networks, the massive number of users to be served and their increasing demands for high-data-rate services can lead to terrestrial base stations (BSs) becoming over-loaded, which in turn results in degraded user experience, e.g., longer delay in requesting data services or lower data rate [1]. Compared to terrestrial networks, the integration of satellites, e.g., low earth orbit (LEO) satellites, and 5G systems is considered as a promising solution to provide cost-efficient data services [2]. SpaceX and OneWeb launch thousands of LEO satellites to deploy a dense constellation to support seamless communications for remote areas or disaster scenarios and offload traffic from terrestrial networks [3].

The solutions for terrestrial network optimization and resource management might not be directly applied to integrated satellite-terrestrial systems [4]. In the literature, tailored schemes have been investigated to improve the networks' performance. In [5], the authors proposed a user scheduling scheme to maximize the sum-rate and the number of accessed users by utilizing the LEO-based backhaul. In [6], a joint power allocation and user scheduling scheme was proposed to maximize the network throughput in hierarchical LEO systems with the constraint of transmission delay. In [7], the authors developed a joint resource block allocation and power allocation algorithm to maximize the total transmission rate for LEO systems. It is worth noting that the resource optimization problems in LEO-terrestrial networks are typically combinatorial and non-convex. The conventional iterative optimization methods, e.g., in [5]–[7], are unaffordable for real-time operations due to their high computational complexity.

A. Related Works: State-of-the-art and Limitations

Towards an efficient solution, various learning techniques are widely studied. Compared to supervised learning, reinforcement learning (RL) learns the optimal policy from observed samples without preparing labeled data. As one of the promising RL methods, deep reinforcement learning (DRL) can adopt deep neural networks (DNNs) for parameterization and rapid decision making. Recent works have applied RL/DRL for resource management in LEO-terrestrial systems. In [8], to maximize the achievable rate in LEO-assisted relay networks, a DQN-based algorithm was proposed to make the online decisions for link association. The authors in [9] adopted multi-agent reinforcement learning to minimize the average number of handovers and improve the efficiency of channel utilization for LEO satellite systems. In [10], the authors applied an actor-critic (AC) algorithm to LEO resource allocation, such as beam allocation and power control. The above RL algorithms in practical LEO systems are limited by the following issue. That is, the performance of a learning model largely depends on the data originated from the experienced samples or the observed environment, but the wireless environment is highly complex and dynamic. When network parameters vary dramatically, the performance of the learning models can be degraded. To remedy this, one has to re-collect a large number of training data and re-train the learning models, which is time-consuming and inefficient to adapt to fast variations [11].

To address this issue, a variety of studies focus on how to make the learning models quickly respond to dynamic environments. Transfer learning applies the knowledge acquired from a source learning task to a target learning task to speed up

The work has been supported by the ERC project AGNOSTIC (742648), by the FNR CORE projects ROSETTA (C17/IS/11632107), ProCAST (C17/IS/11691338) and 5G-Sky (C19/IS/13713801), and by the FNR bilateral project LARGOS (12173206).

Yaxiong Yuan, Lei Lei, Thang X. Vu, and Symeon Chatzinotas are with the Interdisciplinary Centre for Security, Reliability and Trust, Luxembourg University, 1855 Kirchberg, Luxembourg (e-mail: yaxiong.yuan@uni.lu; lei.lei@uni.lu; thang.vu@uni.lu; symeon.chatzinotas@uni.lu).

O. Simeone is with King's Communication, Learning & Information Processing (KCLIP) Lab, Department of Engineering, King's College London, United Kingdom (e-mail: osvaldo.simeone@kcl.ac.uk).

S. Sun is with the Institute for Infocomm Research, Agency for Science, Technology, and Research, Singapore 138632 (e-mail: sunsm@i2r.a-star.edu.sg).

the re-training process and reduce the volume of the collected new data sets [12]. The performance of transfer learning is limited by finding correlated tasks. Another approach, joint learning, aims at obtaining a single model that can be adapted to dynamic environments by optimizing the loss function over multiple tasks [13]. Besides, continual learning can also accelerate the adaptation to the new learning task by adding the experienced data from the previous tasks to the re-training data set, thus avoiding completely forgetting previously learned models [14]. However, joint learning and continual learning might have good learning performance in average but their generalization abilities for specific tasks are limited if different tasks are highly diversified [15]. In contrast, meta-learning extracts meta-knowledge and achieves good performance for specific tasks without requiring the related source tasks. The authors in [16] proposed a model-agnostic meta-learning algorithm (MAML) to obtain the model's initial parameters as meta-knowledge to quickly adapt to new tasks. In [17], an algorithm combining actor-critic with MAML (AC-MAML) was developed to learn a new task from fewer experience data sets. In [18], the authors proposed a promising meta-critic learning framework with better performance than conventional AC and AC-MAML, In [19], a meta-learning-based adaptive sensing algorithm was proposed, which determines the next most informative sensing location in wireless sensor networks. In [20], meta-learning was applied to find a common initialization vector that enables fast training of an autoencoder for the fading channels. Most of the meta-learning methods were applied in the areas of pattern recognition [16], robotics [17], [18], and physical layer communications [20]. The considered learning tasks in these works are simple and the action space is small, e.g., [19], [20]. However, when the learning techniques, e.g., DRL, AC-MAML, or meta-critic learning, are applied to address combinatorial optimization problems in a dynamic LEO-terrestrial network, the action space can be huge and the input-output relationships can become more complex. These may degrade the efficiency of the adopted learning methods.

B. Motivations and Contributions

Beyond state-of-the-art, we intend to address the following questions in this paper:

- Which learning technique can lead to more performance gain in addressing resource management problems for dynamic LEO-terrestrial networks?
- How to deal with the huge action space and improve the learning efficiency?
- How to make the learning solutions more adaptive to dynamic environments?

In this study, we design an enhanced meta-critic learning algorithm (EMCL) for dynamic LEO-terrestrial systems. To the best of our knowledge, this is arguably the first work to present meta-critic learning to address resource scheduling problems and emphasize the adaptation to non-ideal dynamic environments. The major contributions are summarized as follows:

- We design a tailored metric for over-loaded LEO systems with dense user distribution, aiming at serving more users and delivering more requested data.
- We formulate the resource scheduling problem as a quadratic integer programming (QIP) and provides two offline optimization-based benchmarks, i.e., optimal branch and bound (B&B) algorithm and suboptimal alternating direction method of multipliers-based heuristic algorithm (ADMM-HEU).
- Due to the combinatorial nature and the high complexity of the offline solutions, we solve the problem from the perspective of DRL by reformulating the problem to a Markov decision process (MDP) to make online decisions intelligently.
- To enhance the adaptation to dynamic environments, we propose an EMCL algorithm based on the meta-critic framework, where the critic has good generalization ability to evaluate the actors for new tasks such that the learning agent can adjust the policy timely when the environment changes. Compared to previous learning methods, the novelty stems from: 1) The tailored design of a hybrid neural network to extract the features from the current and historical samples; 2) The integrated Wolpertinger policy to allow the actor to make decisions more efficiently in an exponentially increased action space.
- To identify a promising solution, we evaluate the proposed EMCL with other benchmarks in three practical dynamic scenarios, i.e., bursty user demands, dramatically fluctuated channel states, and user departure/arrival. The numerical results verify EMCL's effectiveness and fast-response capabilities in adapting to dynamic environments.

The rest of the paper is organized as follows. The system model is presented in Section II. We formulate a resource scheduling problem and develop optimal and suboptimal solutions for performance benchmarks in Section III. In Section IV, we model the problem as an MDP and develop an EMCL algorithm. Numerical results are demonstrated and analyzed in Section V. Finally, Section VI concludes the paper.

II. System Model

A. LEO-Terrestrial Network

In practice, terrestrial base stations can become over-loaded and congested. This common issue has been received considerable attention from the academia, industry, and standardization bodies, e.g., 3GPP Release 17 [21]. In this work, we address this challenging issue via developing satellite-aided solutions. As shown in Fig. 1, the BSs with limited resources might not be able to serve all the users and deliver all the requested data demands within a required transmission or queuing delay. To relieve the burden of the terrestrial BSs, LEO satellites are introduced to offload traffic from BSs or provide backhauling services. The LEO employs a transparent payload. The terrestrial communication uses C-band, e.g., sub-6GHz in 5G applications [22], whereas the LEO satellite adopts Ka-band to access wider bandwidths since the radio frequency

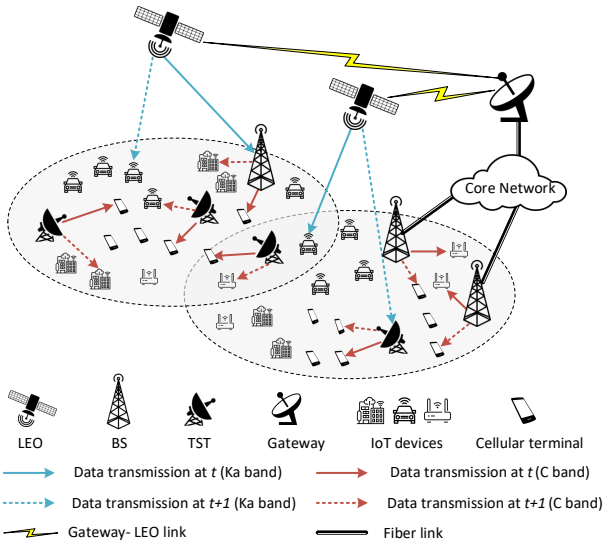

Fig. 1. An illustrative LEO-terrestrial communication system

congestion has become a serious issue in the lower bands due to the satellites' increased usage [3]. We consider two types of mobile terminals (MTs) that coexist in the system. The first type is the normal cellular terminals, e.g., cell phones, that can be served by BSs or TSTs, typically with large traffic demands. The other is the machine-type communication terminals, e.g., massive IoT devices requesting lower data demands, which are equipped with a 3GPP terrestrial-non-terrestrial network (TN-NTN) compliant dual-mode that can be either served by LEO via Ka-band or by BS/TST through C-band [23]. Compared to conventional cellular BS, TST is a small-size terminal that acts as a flexible and cost-saving access point. A TST can either receive backhauling services from LEO over Ka-band or transmit data to MTs over C-band [5]. The terrestrial BSs can request data from the core network through optical fiber links or from the LEO satellites through the BS-LEO link.

We denote $\mathcal{K} = \{1, ..., k, ..., K\}$ as the union of ground devices (GDs) served as the receivers, including MTs, BSs and TSTs, and denote $\mathcal{N} = \{1, ..., n, ..., N\}$ as the union of transmitters. The set $\mathcal{N}$ can be further divided into two subsets $\mathcal{N}_1$ and $\mathcal{N}_2$, where $\mathcal{N}_1$ consists of the transmitters using Ka-band, i.e., LEO, while $\mathcal{N}_2$ contains the transmitters occupying C-band, i.e., BSs or TSTs. We assume that the transmission tasks are delay-sensitive and to be completed within T time slots. The time domain is divided by time slots, i.e., $\mathcal{T} = \{1, ..., t, ..., T\}$. In data transmission, each transmitter n serves GD in unicast mode, i.e., no joint transmission and no multi-cast transmission. Within a time slot, multiple links can be activated, forming a link group.

B. Channel Modeling

We consider time-varying channels. At time slot t, the channel state between receiver k and transmitter n can be modeled as:

$$h_{k,n,t} = G_T \cdot G_C \cdot G_R, \tag{1}$$

where G_T and G_R are the transmit and receive antenna gain, respectively, and G_C represents the channel fading. For LEO-to-GD channel, a widely used channel fading model in [5], [7], [24] is adopted, which includes free-space path loss, pitch angle fading, atmosphere fading, and Rician small-scale fading.

$$G_C = \left(\frac{c}{4\pi d f_C}\right)^2 \cdot G_P \cdot A(d) \cdot \varphi, \tag{2}$$

where c is the speed of light, d is the propagation distance between LEO and the terminals, f_C is the carrier frequency, G_P is the pitch angle fading factor, and φ is the Rician fading factor. $A(d)$ is the atmospheric loss which is expressed as:

$$A(d) = 10^{\left(\frac{3\chi d}{10H}\right)}, \tag{3}$$

where χ, in dB/km, is the attenuation through the clouds and rain, and H is the altitude of LEO. In downlink transmission, we assume that Doppler shift caused by the high mobility of LEO can be perfectly pre(post)-compensated in the gateway based on the predictable satellite motion and speed [25]. For terrestrial channels, i.e., TST/BS-to-MT, G_C consists of the path loss and Rayleigh small-scale fading [26], which is given by:

$$G_C = \left(\frac{c}{4\pi d f_C}\right)^2 \cdot \phi, \tag{4}$$

where ϕ is the Rayleigh fading factor.

Based on the adopted channel fading models (2) and (4), we further model the time-varying channel as the first state Markov channel (FSMC) to capture the time-correlation characteristics and conduct mathematically tractable analysis [27]. We discretize each channel state $h_{k,n,t}$ into L levels, $\mathcal{H} = \{h_1, ..., h_L\}$, and the transition probability matrix is defined as:

$$\mathbf{P} = \begin{bmatrix} P_{1,1} & \cdots & P_{1,L} \\ \vdots & \ddots & \vdots \\ P_{L,1} & \cdots & P_{L,L} \end{bmatrix}, \tag{5}$$

where an element $P_{l,l'}$ can be written as:

$$P_{l,l'} = \text{Prob}\left[h_{k,n,t+1}{=}h_{l'} | h_{k,n,t}{=}h_l\right], \ h_l, h_{l'} \in \mathcal{H}. \tag{6}$$

That is, at a given time slot t, if $h_{k,n,t} = h_l$, $P_{l,l'}$ refers to the probability of channel state at the next time slot $h_{k,n,t+1}$ transiting from h_l to $h_{l'}$.

III. Optimization Problem and ADMM-based Heuristic Approach

In this section, we formulate a resource scheduling problem for the considered over-loaded LEO-5G systems. Due to the high complexity to obtain the global optimum, we propose an ADMM-based heuristic approach as an offline benchmark.

A. Optimization Problem

We denote $\mathcal{G} = \{1, ..., g, ..., G\}$ as a set by enumerating all the valid link groups, Each group consists of multiple transmitter-GD links. We use binary indicators $\alpha_{k,n,g}$ to represent the activated links in group g, where $\alpha_{k,n,g} = 1$ if the transmitter-GD link (n, k) is included in group g and will be activated when group g is scheduled, otherwise, 0.

We remark that only valid links, i.e., satisfying the following unicast constraints, can form a link group.

$$\sum\nolimits_{n\in\mathcal{N}} \alpha_{k,n,g} \leq 1,\ \forall\ k\in\mathcal{K},\ g\in\mathcal{G}, \tag{7}$$

$$\sum\nolimits_{k\in\mathcal{K}} \alpha_{k,n,g} \leq 1,\ \forall\ n\in\mathcal{N},\ g\in\mathcal{G}. \tag{8}$$

(7) means that each GD k in group g receives data from at most one transmitter, and (8) represents each transmitter n in group g serves no more than one GD. Confined by (7) and (8), the SINR and the rate of GD k in group g at time slot t are expressed in (9) and (10), respectively.

$$\begin{aligned}\gamma_{k,g,t} =& \frac{\sum_{n\in\mathcal{N}_1} h_{k,n,t}\alpha_{k,n,g}p_k}{\sum_{j\in\mathcal{K}\setminus k}\sum_{n\in\mathcal{N}_1} h_{j,n,t}\alpha_{j,n,g}p_k+\sigma^2}\\ &+\frac{\sum_{n\in\mathcal{N}_2} h_{k,n,t}\alpha_{k,n,g}p_k}{\sum_{j\in\mathcal{K}\setminus k}\sum_{n\in\mathcal{N}_2} h_{j,n,t}\alpha_{j,n,g}p_k+\sigma^2},\end{aligned} \tag{9}$$

and

$$R_{k,g,t} = \Phi B\log_2(1+\gamma_{k,g,t}), \tag{10}$$

where p_k is the transmit power to GD k and Φ is the duration of each time slot. We define the decision variables as $\mathbf{x} = [x_{1,1},...,x_{g,t},...,x_{G,T}]$ where

$$x_{g,t} = \begin{cases} 1, & \text{if group } g \text{ is scheduled at time slot } t,\\ 0, & \text{otherwise.}\end{cases}$$

In the objective design, considering the over-loaded scenario with densely deployed users, the system with limited resources may not be able to satisfy every user's data demand D_k. We then consider a composite utility function in (11), and define that GD k is served, i.e., $f_k(\mathbf{x})=1$, when a threshold $D'_k (D'_k < D_k)$ is satisfied.

$$f_k(\mathbf{x}) = \mathbb{1}\left(\sum_{t\in\mathcal{T}}\sum_{g\in\mathcal{G}} R_{k,g,t}x_{g,t} - D'_k\right), \tag{11}$$

where $\mathbb{1}(\cdot)$ is an indicator function such that $\mathbb{1}(\beta) = \begin{cases} 1, & \text{if } \beta>0\\ 0, & \text{if } \beta\leq 0\end{cases}$. We convert the non-linear function $f_k(\mathbf{x})$ to be linear function by introducing auxiliary variables $\mathbf{y} = [y_1,...,y_k,...,y_K]$ and linear constrains (12d), where $y_k = f_k(\mathbf{x})$. The optimization problem is formulated as:

$$\begin{aligned}\mathbf{P1}: \min_{x_{g,t},y_k}\ & f(\mathbf{x},\mathbf{y}) = \eta_0\left(\sum_{k\in\mathcal{K}} y_k - K\right)^2 +\\ & \sum_{k\in\mathcal{K}}\eta_k\left(\sum_{t\in\mathcal{T}}\sum_{g\in\mathcal{G}} R_{k,g,t}x_{g,t} - D_k\right)^2 \qquad (12a)\\ s.t.\ & \bar{\gamma}_k - \gamma_{k,g,t} \leq V\left(1 - x_{g,t}\sum_{n\in\mathcal{N}}\alpha_{k,n,g}\right),\\ & \forall k\in\mathcal{K}, g\in\mathcal{G}, t\in\mathcal{T}, \qquad (12b)\\ & \sum_{g\in\mathcal{G}} x_{g,t}\leq 1,\ \forall t\in\mathcal{T}, \qquad (12c)\\ & D'_k y_k \leq \sum_{t\in\mathcal{T}}\sum_{g\in\mathcal{G}} R_{k,g,t}x_{g,t},\ \forall k\in\mathcal{K}, \qquad (12d)\\ & x_{g,t}\in\{0,1\},\ \forall g\in\mathcal{G},\ t\in\mathcal{T}, \qquad (12e)\\ & y_k\in\{0,1\},\ \forall\ k\in\mathcal{K}, \qquad (12f)\end{aligned}$$

where $\bar{\gamma}_k$ is the SINR threshold of GD k, V is a positive sufficiently large value, and $\eta_0,...,\eta_K$ are the weight factors. The GDs with higher priority will be assigned with larger η_k, e.g., VIP clients or the users requesting emergency communications. The objective (12a) aims at serving as many GDs as possible and delivering more data to GDs such that the two gaps in the overloaded system can be minimized. The first term in the objective is used to keep the user fairness. The optimal scheduler is encouraged to meet the minimum requirement D'_k to serve more users and gain utilities from the first term. In the second term, when some users' weights η_k are considerable, consuming some resources and satisfying higher demand D_k can be preferred.

- The constraints (12b) mean that if GD k in group g is scheduled at time slot t, i.e., $x_{g,t}\sum_{n\in\mathcal{N}}\alpha_{k,n,g}=1$, the SINR of GD k should be higher than the threshold $\bar{\gamma}_k$.
- The constraints (12c) represent no more than one group can be scheduled to a time slot.
- In constraints (12d), we define that if GD k is served, i.e., $y_k=1$, the received data should be larger than D'_k.

Next, we identify the convexity of $\mathbf{P1}$ when the binary variables are relaxed.

Lemma 1. *The relaxation problem of* $\mathbf{P1}$ *is convex.*

Proof. See Appendix A. □

Based on Lemma 1, we conclude that $\mathbf{P1}$ is an integer convex optimization problem. The optimum can be obtained by B&B that solves a convex relaxation problem at each node [29]. Although the complexity increases exponentially, the B&B based approach can provide a performance benchmark at least for small-medium instances.

B. ADMM-based Heuristic Algorithm

To reduce the complexity in solving the large-scale problems, we develop a suboptimal algorithm. We observe that $\mathbf{P1}$ has a variable-splitting structure, which motivates the development of ADMM based approaches [30]. The algorithm is summarized in Alg. 1, first solving the convex relaxation problem of $\mathbf{P1}$ based on ADMM (in lines 2-8), following by a rounding operation (in lines 9-13). In ADMM, we divide the relaxed variables into $T+1$ blocks $\hat{\mathbf{x}}_1,...,\hat{\mathbf{x}}_T,\hat{\mathbf{y}}$, where $\hat{\mathbf{x}}_t = [\hat{x}_{1,t},...,\hat{x}_{G,t}]$, and introduce auxiliary variables $\mathbf{z} = [z_1,...,z_K]$ such that the inequality constraints (12d) are replaced by:

$$\begin{cases} z_k = D'_k\hat{y}_k - \sum_{g\in\mathcal{G}}\sum_{t\in\mathcal{T}} R_{k,g,t}\hat{x}_{g,t}, & \forall k\in\mathcal{K},\\ z_k\leq 0, & \forall k\in\mathcal{K}.\end{cases} \tag{13}$$

The augmented Lagrangian function is expressed as:

$$\begin{aligned}&L(\hat{\mathbf{x}}_1,...,\hat{\mathbf{x}}_T,\hat{\mathbf{y}},\mathbf{z},\boldsymbol{\lambda})\\ =&f(\hat{\mathbf{x}},\hat{\mathbf{y}}) + \sum_{k\in\mathcal{K}}\lambda_k\left(z_k - D'_k\hat{y}_k + \sum_{g\in\mathcal{G}}\sum_{t\in\mathcal{T}} R_{k,g,t}\hat{x}_{g,t}\right)\end{aligned}$$

Algorithm 1 ADMM-HEU

1: **input:** D_k, D'_k and $R_{k,n,t}$.
2: Relax $\mathbf{P1}$ to a continuous problem $\mathbf{P1}'$.
3: Initialize $\hat{\mathbf{x}}_t^0$, $\hat{\mathbf{y}}^0$, $\mathbf{z}^0$, $\boldsymbol{\lambda}^0$ and $i = 0$.
4: **for** $i = 0, ..., I_{iter}$ **do**
5: Update $\hat{\mathbf{x}}_t, \hat{\mathbf{y}}$ and $\mathbf{z}$ by Eq. (15), (16) and (17).
6: $\lambda_k^{i+1} = \lambda_k^i + \rho \left(z_k^i - D'_k \hat{y}_k^i + \sum_{g \in \mathcal{G}} \sum_{t \in \mathcal{T}} R_{k,g,t} \hat{x}_{g,t}^i \right)$.
7: **end for**
8: Obtain relaxed solution $\hat{x}_{g,t}$.
9: **for** $t \in \mathcal{T}$ **do**
10: Find $g^\dagger = \underset{g \in \mathcal{G}}{\operatorname{argmax}} \{\hat{x}_{1,t}, ..., \hat{x}_{G,t}\}$.
11: Set $x^*_{g^\dagger,t} = 1$ and $x^*_{g,t} = 0, \ \forall g \neq g^\dagger$.
12: **end for**
13: Calculate y_k^* based on Eq. (11).
14: **output:** $x^*_{g,t}$ and y_k^*

$$+\frac{\rho}{2} \sum_{k \in \mathcal{K}} \| z_k - D'_k \hat{y}_k + \sum_{g \in \mathcal{G}} \sum_{t \in \mathcal{T}} R_{k,g,t} \hat{x}_{g,t} \|^2, \tag{14}$$

where $\rho > 0$ is the penalty parameter and $\boldsymbol{\lambda} = [\lambda_1, ..., \lambda_K]$ are the lagrangian multipliers. We define I_{iter} as the total number of iterations of the algorithm. In each iteration i, ADMM update each variable block as follows (in line 5) and update multipliers (in line 6):

$$\hat{\mathbf{x}}_t^{i+1} = \underset{\hat{\mathbf{x}}_t \in \mathcal{X}_t}{\operatorname{argmin}} L(\hat{\mathbf{x}}_1^i, ..., \hat{\mathbf{x}}_T^i, \hat{\mathbf{y}}^i, \mathbf{z}^i, \boldsymbol{\lambda}^i), \ \forall t \in \mathcal{T}, \tag{15}$$

$$\hat{\mathbf{y}}^{i+1} = \underset{\hat{\mathbf{y}} \in \mathcal{Y}}{\operatorname{argmin}} L(\hat{\mathbf{x}}_1^i, ..., \hat{\mathbf{x}}_T^i, \hat{\mathbf{y}}^i, \mathbf{z}^i, \boldsymbol{\lambda}^i), \tag{16}$$

$$\mathbf{z}^{i+1} = \underset{\mathbf{z} \in \mathcal{Z}}{\operatorname{argmin}} L(\hat{\mathbf{x}}_1^i, ..., \hat{\mathbf{x}}_T^i, \hat{\mathbf{y}}^i, \mathbf{z}^i, \boldsymbol{\lambda}^i), \tag{17}$$

where $\mathcal{X}_t = \{\mathbf{x}_t | (12b), (12c), (12d)\}$, $\mathcal{Y} = \{\mathbf{y} | 0 \leq y_k \leq 1\}$ and $\mathcal{Z} = \{\mathbf{z} | z_k \leq 0\}$. When ADMM terminates, the continuous solution $\hat{x}_{g,t}$ is obtained in line 8. The rounding process is then carried out in lines 10-13 to convert the largest $\hat{x}_{g,t}$ in each time slot to 1 (selecting the most promising group g for each t) and keep others 0.

The developed ADMM-HEU can provide sub-optimal benchmarks within an acceptable time span, since the subproblems in (15)-(17) can be solved in a parallel manner and with a smaller size than the original problem. However, ADMM-HEU is still an offline algorithm because iteratively solving the subproblems in (15)-(17) has to consume a considerable amount of time, which might not be able to adapt to fast network variations timely.

IV. The Proposed EMCL Algorithm

To enable an intelligent and online solution, we address the problem from an RL perspective. In algorithm design, we integrate meta-learning into an actor-critic framework to achieve good adaptability to dynamic environments.

A. Actor-Critic and Meta-Critic

Firstly, we briefly introduce actor-critic and meta-critic learning approaches as a basis to present the proposed EMCL. AC is a type of RL algorithm that takes advantage of both value-based methods, e.g., Q-learning, and policy-based methods, e.g., REINFORCE, with fast convergent properties and the capability to deal with continuous action spaces [31]. The learning agent in AC contains two components, where the actor is responsible for making decisions while the critic is used for evaluating the decisions by the value functions. Specifically, at each learning step t[1], the actor takes action based on a stochastic policy, i.e., $a_t \sim \pi(a|s_t)$, where $\pi(a|s_t)$ is the probability of taking an action under state s_t, typically following the Gaussian distribution [32]. The critic is to generate a Q-value function $Q(s_t, a_t) = \mathbb{E}_\pi[\bar{r}_t | s_t, a_t]$, where $\bar{r}_t$ is the accumulated reward at step t, and $\mathbb{E}_\pi[\beta]$ is the expected value of β over the policy π. The goal of the learning agent is to find a policy to maximize the expected accumulated reward (or Q-value).

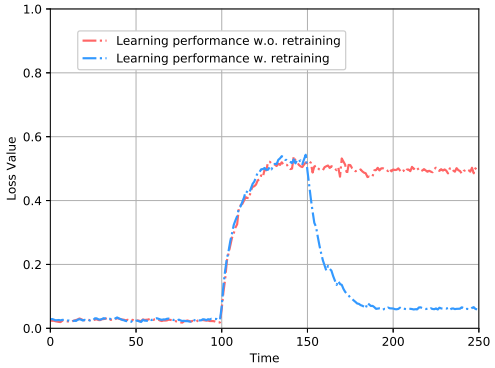

Fig. 2. Evolution of loss over time-varying demands.

A critical issue in conventional learning approaches, including AC, is that the performance of a learning model largely depends on the adopted training or observed data sets. The practical LEO-5G systems are highly complex and dynamic, such as fast and dramatic variations in channel states, demands, user arrival/departure, and network topologies. This typically causes the new inputs to become no longer relevant to the statistical properties of the historical data [33]. As a consequence, the previous learning model can become invalid and the model may need to be re-trained to adapt. To illustrate this impact, we use Fig. 2 to depict a typical evolution of AC's loss value over time-varying demands. From 0 to 100 time slots, the demand is time-varying but follows the historical statistical property, leading to a well-adapted AC with low and stable loss values. When a surged demand arrives at the 100-th time slot, the AC model is no longer applicable to this new input due to the rapidly deteriorated loss values. When the agent in AC consumes a considerable amount of time in new data collection and re-training, the performance can return to the previous level.

To address this issue, meta-critic-based approaches become an emerging technique that takes advantage of a variety of previously observed tasks to infer the meta-knowledge, such that a new learning task can be quickly trained with few ob-

[1]In this paper, a learning step corresponds to a time slot.

servations [16]. Meta-critic learning combines meta-learning with an AC framework to enhance the generalization ability. However, conventional meta-critic learning is not effective in dealing with the large discrete space in **P1**. In addition, there is no uniform standard to parameterize the learning model and extract meta-knowledge in dynamic environments. Thus, we propose an EMCL algorithm to enable an efficient dynamic-adaptive solution.

B. EMCL Framework

1) MDP Reformulation: First, we reformulate the original problem **P1** as an MDP by defining action, state and reward.

- As the actor is to select a group from set $\mathcal{G}$ at each time slot t, the action is defined as an assigned link group,

$$a_t = g \in \mathcal{G}. \tag{18}$$

- The state consists of the channel coefficients $h_{k,n,t}$, modeled as FSMC with the transition probability defined in (6), and the delivered data for user k up to time slot t, where $b_{k,t} = b_{k,t-1} + R_{k,a_t,t}$.

$$s_t = \{h_{1,1,t}, ..., h_{K,N,t}, b_{1,t}, ..., b_{K,t}\}. \tag{19}$$

  All possible states are included in the state space $\mathcal{S}$. The next state only depends on the current state and action but is unrelated to the past, which means the state transition from s_t to s_{s+1} follows the Markov property [31].
- The reward is closely related to the objective of **P1**. We define the reward as (20).

$$r_t = \sum_{k=0}^{K} \eta_k (\Delta_{k,t-1}^2 - \Delta_{k,t}^2), \tag{20}$$

  where $\Delta_{k,t} = \begin{cases} \sum_{k=1}^{K} \mathbb{1}\left(b_{k,t} - D'_k\right) - K, & k = 0, \\ b_{k,t} - D_k, & k \neq 0. \end{cases}$
  Then, the accumulated reward at step t is given by $\bar{r}_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where $\gamma \in [0, 1]$ is a discounted factor.

Under the designed MDP, we verify the consistency between the goals of the RL algorithm and the original optimization problem such that the policy provided by the learning agent can minimize the objective in **P1**.

Lemma 2. *When $\gamma = 1$, the objective of the learning agent is equivalent to that of the optimization problem* **P1**.

Proof. See Appendix B □

2) Meta Critic and Task-Specific Actor: As shown in Fig. 3, we design a hierarchical structure in EMCL containing a meta critic[2] and multiple actors. The learning model is trained over multiple tasks, $\mathcal{J}^{(1)}, ..., \mathcal{J}^{(I)}$. The meta critic can evaluate the actions for any task with a Q-value while each actor provides a stochastic policy dedicated to a specific task. At time step t, $s_t^{(i)}$, $a_t^{(i)}$, and $r_t^{(i)}$ represent the state, action, and reward for task i, respectively. An

[2]In this paper, "meta-critic learning" refers to an algorithm that combines AC and meta-learning while "meta critic" refers to the critic in the framework.

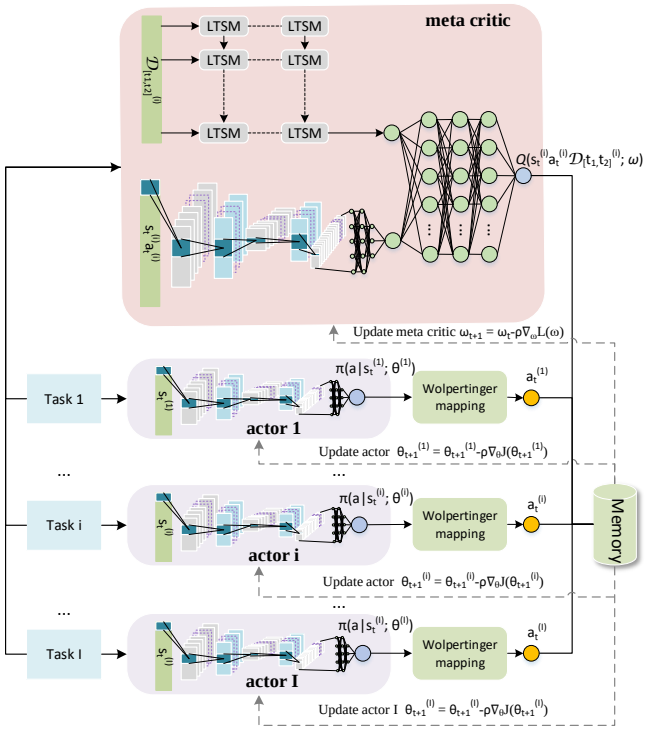

Fig. 3. The proposed EMCL framework.

episode $\mathcal{D}^{(i)} = \{s_1^{(i)}, a_1^{(i)}, r_1^{(i)} ..., s_T^{(i)}, a_T^{(i)}, r_T^{(i)}\}$ can be sampled from the first step to the terminal step T. We denote $\mathcal{D}_{[u,w]}^{(i)}$ as a segment of $\mathcal{D}^{(i)}$ from step u to w, i.e., $\mathcal{D}_{[u,w]}^{(i)} = \{s_u^{(i)}, a_u^{(i)}, r_u^{(i)}, ..., s_w^{(i)}, a_w^{(i)}, r_w^{(i)}\}$. Since the explicit meta critic and actors are difficult to obtain, we adopt the function approximation method. The meta critic is parameterized as a neural network (NN) with the weights $\boldsymbol{\omega}$, i.e., $Q(s_t^{(i)}, a_t^{(i)}, \mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega})$. We note that, in addition to $s_t^{(i)}$ and $a_t^{(i)}$, the input includes the most recent $\bar{t}$ samples $\mathcal{D}_{[t-\bar{t},t-1]}^{(i)}$. Each task-specific actor is modeled as an NN $\pi(a|s_t^{(i)}; \boldsymbol{\theta}^{(i)})$ with the weights $\boldsymbol{\theta}^{(i)}$.

To optimize the weights, we minimize the loss functions by gradient descend. The loss function of the meta critic $L(\boldsymbol{\omega})$ is defined as the average temporal difference (TD) error over all tasks:

$$L(\boldsymbol{\omega}) = \frac{1}{I} \sum_{i=1}^{I} \mathbb{E}_{\pi(\boldsymbol{\theta}^{(i)})} \Big[\big(Q(s_{t+1}^{(i)}, a_{t+1}^{(i)}, \mathcal{D}_{[t-\bar{t}+1,t]}^{(i)}; \boldsymbol{\omega}) - r_t - \gamma Q(s_t^{(i)}, a_t^{(i)}, \mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega}) \big)^2 \Big], \tag{21}$$

where the TD error reflects the similarity between the estimated Q-value and actual Q-value. For the task-specific actor, the loss function $J(\boldsymbol{\theta}^{(i)})$ is the negative Q-value:

$$J(\boldsymbol{\theta}^{(i)}) = \mathbb{E}_{\pi(\boldsymbol{\theta}^{(i)})} \Big[-Q(s_t^{(i)}, a_t^{(i)}, \mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega}) \Big], \tag{22}$$

such that minimizing $J(\boldsymbol{\theta}^{(i)})$ is equivalent to maximizing the expected accumulated reward. The update rules are given by:

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \rho \nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}), \tag{23}$$

$$\boldsymbol{\theta}_{t+1}^{(i)} = \boldsymbol{\theta}_{t}^{(i)} - \rho \nabla_{\boldsymbol{\theta}^{(i)}} J(\boldsymbol{\theta}^{(i)}). \tag{24}$$

Based on the fundamental results of the policy gradient theorem [31], the gradients of $L(\boldsymbol{\omega})$ and $J(\boldsymbol{\theta}^{(i)})$ are:

$$\nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}) = \frac{1}{I} \sum_{i=1}^{I} \Big[2L(\boldsymbol{\omega}) \nabla_{\boldsymbol{\omega}} (Q(s_{t+1}^{(i)}, a_{t+1}^{(i)}, \mathcal{D}_{[t-\bar{t}+1,t]}^{(i)}; \boldsymbol{\omega}) - Q(s_t^{(i)}, a_t^{(i)}, \mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega})) \Big], \tag{25}$$

$$\nabla_{\boldsymbol{\theta}^{(i)}} J(\boldsymbol{\theta}^{(i)}) = -Q(s_t^{(i)}, a_t^{(i)}, \mathcal{D}; \boldsymbol{\omega}) \nabla_{\boldsymbol{\theta}^{(i)}} \log \pi(a | s_t^{(i)}; \boldsymbol{\theta}^{(i)}). \tag{26}$$

3) Algorithm Summary: We summarize the proposed EMCL in Alg. 2, which includes two phases: the meta training phase and the online learning phase. For the former, the learning model is trained over different learning tasks. At each learning episode, we sample I learning tasks. We obtain the approximated Q-value (in line 6) and stochastic policy (in line 7) by the approximation functions. The final actions are determined by the Wolpertinger approach in line 8, which will be elaborated in the following subsection. In line 9, the memory is used to store the experienced learning tuples $\{s_t^{(i)}, s_{t+1}^{(i)}, a_t^{(i)}, r_t^{(i)}\}$. At each step, we extract a batch of tuples from the memory as the training data for updating $\boldsymbol{\omega}$ and $\boldsymbol{\theta}^{(i)}$ by (23) and (24) in line 10 and 12, respectively. In the online learning phase, given a new task, the well-trained meta critic $\boldsymbol{\omega}^*$ can be directly used to estimate the Q-value and only the actor needs to be re-trained.

Algorithm 2 EMCL

Meta training phase:

1: **input**: Multiple task samples; initial $\boldsymbol{\omega}_0$.
2: **for** each learning episode **do**
3: Sample I tasks and initialize $\boldsymbol{\omega}_0, \boldsymbol{\theta}_0^{(1)}, ..., \boldsymbol{\theta}_0^{(I)}$.
4: **for** each learning step t **do**
5: **for** each task i **do**
6: Obtain Q-value by the meta critic in (29).
7: Obtain stochastic policy by the actor in (31).
8: Take actions $a_t^{(i)}$ by the Wolpertinger approach.
9: Store tuples $\{s_t^{(i)}, s_{t+1}^{(i)}, a_t^{(i)}, r_t^{(i)}\}$ in the memory.
10: Take a batch of data and update $\boldsymbol{\theta}^{(i)}$ by (24).
11: **end for**
12: Update $\boldsymbol{\omega}$ by (23).
13: **end for**
14: **end for**
15: **output**: The well-trained meta critic $\boldsymbol{\omega}^*$.

Online learning phase:

16: **input**: A new task; initial $\boldsymbol{\theta}_0$; well-trained meta critic $\boldsymbol{\omega}^*$.
17: **for** each learning episode **do**
18: **for** each learning step t **do**
19: Obtain Q-value by the meta critic in (29).
20: Obtain stochastic policy by the actor in (31).
21: Take an action a_t by the Wolpertinger approach.
22: Store tuples $\{s_t, s_{t+1}, a_t, r_t\}$ in the memory.
23: Take a batch of data and update $\boldsymbol{\theta}$ by (24).
24: **end for**
25: **end for**
26: **output**: The optimal actor $\boldsymbol{\theta}^*$.

C. Tailored Design in EMCL

1) Parameterization with Hybrid Neural Networks: There is no uniform standard for parameterization in conventional meta-critic learning. Considering dynamic environments, the distribution of the new input data and the previous observations may deviate. Towards fast adaptation to the dynamic environment, the critic should be able to identify different tasks, where the information for task identification can be refined from the experienced data, which usually forms time-related series [18]. The widely used DNN might have limitations in efficiency and in mining features from time-series data due to massive number of weights and feed-forward structure. In the proposed EMCL, we design tailored neural networks to enable the meta critic and the actors to fit the complex nonlinear relationships and extract the meta-knowledge from historical data.

As shown in Fig. 3, for the meta critic, a hybrid neural network (HNN) combing convolutional neural network (CNN), long-short term memory (LSTM), and DNN is applied to learn the features from the current state-action pairs and historical trajectories [34]. Thereinto, CNN is computation-efficient via adopting the parameter sharing and pooling operations, and is effective to extract spatial features from the input data. LSMT, as a type of recurrent neural network, has advantages in extracting features from time-related sequential data. Thus, in the designed meta critic, the CNN is used to extract a feature from the current action-state pair $s_t^{(i)}, a_t^{(i)}$ to evaluate the decisions made by the actor based on the current environment. The LSTM is adopted for task-identification based on the time-series data $\mathcal{D}_{[t-\bar{t},t-1]}^{(i)}$. We denote $f_{cnn}(\mathbf{x}; \mathbf{w})$, $f_{lstm}(\mathbf{x}; \mathbf{w})$ and $f_{dnn}(\mathbf{x}; \mathbf{w})$ as the outputs of CNN, LSTM, and DNN, respectively, which are the functions of input $\mathbf{x}$ and weight $\mathbf{w}$. The features output from CNN and LSTM are:

$$\xi_1 = f_{cnn}(s_t^{(i)}, a_t^{(i)}; \boldsymbol{\omega}_{cnn}), \tag{27}$$

$$\xi_2 = f_{lstm}(\mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega}_{lstm}). \tag{28}$$

Then, we take the features as input and pass them through a fully-connected DNN to obtain the approximated Q-value:

$$Q^{\pi}(s_t^{(i)}, a_t^{(i)}, \mathcal{D}_{[t-\bar{t},t-1]}^{(i)}; \boldsymbol{\omega}) = f_{dnn}(\xi_1, \xi_2; \boldsymbol{\omega}_{dnn}). \tag{29}$$

For the task-specific actors, we adopt CNN as the approximator which takes the current state as the input and outputs the mean μ and variance ϑ^2 of the stochastic policy. We assume the stochastic policy follows Gaussian distribution $N(\mu, \vartheta^2)$, such that

$$[\mu, \vartheta^2] = f_{cnn}(s_t^{(i)}; \boldsymbol{\theta}^{(i)}), \tag{30}$$

$$\pi(a | s_t^{(i)}; \boldsymbol{\theta}^{(i)}) = N(\mu, \vartheta^2). \tag{31}$$

2) Action Mapping with the Wolpertinger Policy: The decision variables in $\mathbf{P1}$ are discrete such that we need to map the action from the stochastic policy to a discrete action space. However, the previous action mapping policies in meta-critic learning are not efficient since the action space is large for $\mathbf{P1}$. Thus, in EMCL, the Wolpertinger policy is adopted for faster convergence [35].

Following the stochastic policy π, the actor first produces an action $\hat{a}$ with continuous value, i.e.,

$$f_\pi : \mathcal{S} \to \hat{\mathcal{A}}, \quad f_\pi(s) = \hat{a}, \tag{32}$$

where f_π is a mapping from the state space $\mathcal{S}$ to a continuous action space $\hat{\mathcal{A}}$ under the policy π. As the real action space $\mathcal{G}$ is discrete in $\mathbf{P1}$, the following two conventional approaches can be used for discretization [31]:

- Simple approach: $a_s^* = \underset{a \in \mathcal{G}}{\operatorname{argmin}} |a - \hat{a}|^2$.
- Greedy approach: $a_g^* = \underset{a \in \mathcal{G}}{\operatorname{argmax}} Q(s, a)$.

The simple approach is to select the closest integer value to $\hat{a}$. This approach may result in a high probability of deviating from the optimum, especially at the beginning of learning, and further lead to slow convergence [31]. The greedy approach optimizes Q-value at each step but the complexity is proportional to the exponentially increased space $\mathcal{G}$ [31]. To achieve a trade-off between the complexity and learning performance, the Wolpertinger mapping approach is considered.

- Wolpertinger approach: $a_w^* = \underset{a \in \mathcal{M}^*}{\operatorname{argmax}} Q(s, a)$,

where $\mathcal{M}^*$ is a subset of $\mathcal{G}$ and contains M nearest neighbors of $\hat{a}$. In the Wolpertinger approach, the final action is determined by selecting the highest-scoring action from $\mathcal{M}^*$. The Wolpertinger mapping becomes the greedy approach and simple approach when $M = |\mathcal{G}|$ and $M = 1$, respectively, and the solution of simple approach a_s^* is included in $\mathcal{M}^*$.

Lemma 3. *We assume* $\mathcal{M}^* = \{a_1, ..., a_M\}$ *and* $\begin{cases} Q(s, a_m) \sim U(Q(s, a_s^*) - \kappa, Q(s, a_s^*) + \kappa), & m \neq m' \\ Q(s, a_m) = Q(s, a_s^*), & m = m', \end{cases}$ *where* $U(a, b)$ *refers to uniform distribution and* κ *is a constant, then*

$$\mathbb{E}\left[Q(s, a_w^*)\right] = Q(s, a_s^*) + \kappa \left(1 - \frac{2(2^M - 1)}{M \cdot 2^M}\right). \tag{33}$$

Proof. See Appendix C □

From Lemma 3, when $M > 1$, $\mathbb{E}\left[Q(s, a_w^*)\right] > Q(s, a_s^*)$, which means that the Wolpertinger approach finds the actions with higher Q-values than the simple approach at each learning step, and a larger M leads to a higher expected Q-value. In addition, the complexity of the Wolpertinger approach is lower than the greedy approach as the size of searching space decreases from $|\mathcal{G}|$ to $|\mathcal{M}^*|$. Thus, for the problems with huge discrete spaces, the Wolpertinger approach enables fast convergence to the maximum Q-value with a proper M.

D. Complexity Analysis for EMCL

For the meta critic, an HNN, composed of CNN, LSTM, and DNN, is employed to estimate the Q-value. We assume CNN includes V_1 convolutional layers. We denote $o_{c,v}$, $o_{k,v}$, $o_{f,v}$ are the number of convolutional kernels, the spatial size of the kernel, and the spatial size of the output feature map in the v-th layer, respectively. The stripe of kernel is 1, and the input size is $o_{c,0} = K(N + 1) + 1$. The time complexity of CNN is $\mathcal{O}\left(\sum_{v=1}^{V_1} o_{c,v-1} \varrho_v\right)$, where $\varrho_v = o_{k,v}^2 o_{c,v} o_{f,v}^2$ [36]. For the LSTM, we consider V_2 layers, and denote $o_{l,v}$ and $o_{e,v}$ are the input size and number of memory cells for layer v, respectively, where $o_{l,0} = m(K(N + 1) + 1)$. The time complexity is given by $\mathcal{O}\left(\sum_{v=1}^{V_2} o_{e,v}(4o_{l,v-1} + \varsigma_v)\right)$, where $\varsigma_v = 4o_{e,v} + o_{l,v} + 3$ [37]. For the fully-connected DNN, the time complexity is $\mathcal{O}\left(2o_{d,1} + \sum_{v=2}^{V_3} o_{d,v-1} o_{d,v}\right)$, where V_3 is the number of layers of DNN, $o_{d,v}$ is the input size for layer v [38]. For the actor, as the stochastic policy is approximated by a CNN, the time complexity is identical with that of CNN in the meta critic. Overall, the total time complexity of EMCL is calculated by $\mathcal{O}\left(K(N + 1)L_1 + L_2\right)$, where $L_1 = \varrho_1 + 4mo_{e,1}$ and $L_2 = \varrho_1 + o_{e,1}(4m + \varsigma_1) + 2o_{d,1} + \sum_{v=2}^{V_1} o_{c,v-1}\varrho_v + \sum_{v=2}^{V_2} o_{e,v}(4o_{l,v-1} + \varsigma_v) + \sum_{v=2}^{V_3} o_{d,v-1} o_{d,v}$. When the parameters of the learning model are determined, the complexity increases linearly with $\mathbf{P1}$'s input size, i.e., K and N.

V. Numerical Results

In this section, we compare the performance of the proposed EMCL algorithm with the following five benchmark algorithms:

- OPT: optimal solution (B&B).
- ADMM-HEU: suboptimal solution (Alg. 1).
- GRD: a greedy suboptimal algorithm proposed in [39].
- AC-DDPG: a classic AC algorithm with deep deterministic policy gradient proposed in [40].
- AC-MAML: AC with model-agnostic meta-learning proposed in [17].

The first three provide benchmarks from an optimization perspective, while the last two compare with EMCL from a learning perspective. In the simulation, the parameter settings are similar as in [5], [41]. The adopted parameters for implementing EMCL are summarized in Table I.

Table I: Parameter setting

Number of GDs K	5-9
Number of transmitters N	1 LEO, 1 BS and 2 TSTs
Time limitation T	10 time slots
Duration of time slot Φ	0.1 s
Altitude of LEO	780 km
Transmit power of LEO	100 W
Transmit power of BS	40 W
Transmit power of TST	2 W
Bandwidth for C-band	20 MHz
Bandwidth for Ka-band	400 MHz
Carrier frequency of C-Band	4 GHz
Carrier frequency of Ka-Band	30 GHz
Noise power spectral density	-174 dBm/Hz
Parameterized meta critic	HNN
Parameterized actor	CNN
Distribution of stochastic policy	Gaussian
Learning rate	0.001
Batch size	128
Memory size	10,000
Discount factor	0.9
Size of search space in Wolpertinger policy	10
Environment update interval	200 time slots
Software platform	Python 3.6 with TensorFlow 1.12.0

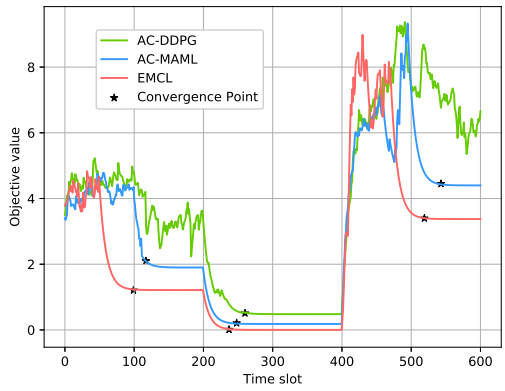

Fig. 4. Performance in adapting to dynamic scenario 1: user entry and leave.

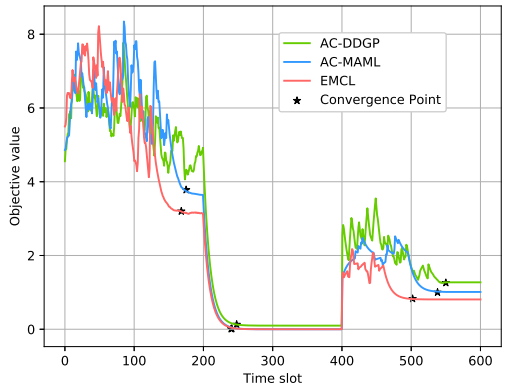

Fig. 5. Performance in adapting to dynamic scenario 2: bursty demands.

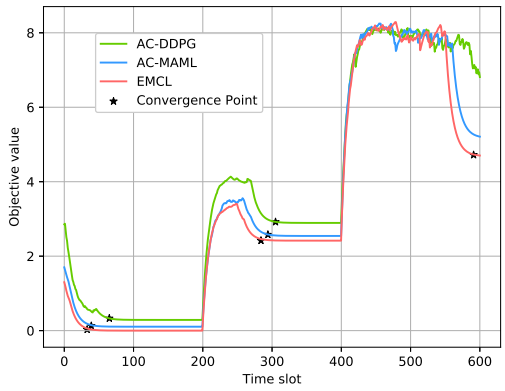

Fig. 6. Performance in adapting to dynamic scenario 3: unforeseen channel variations.

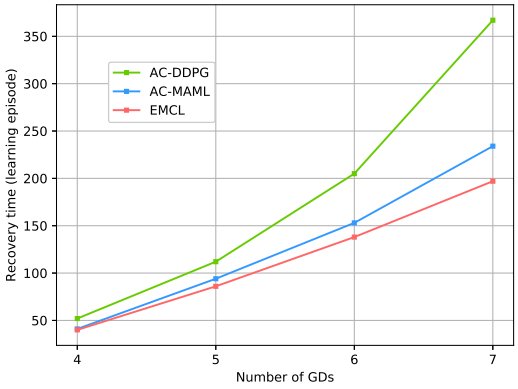

Fig. 7. Recovery time vs. number of users

A. *Capability in Dealing with Dynamic Environments*

To verify the capability of the proposed EMCL in dealing with dynamic environments, Fig. 4-6 compare EMCL with AC-MAML and AC-DDPG in three dynamic scenarios. In Fig. 4, we consider the first scenario with users' irregular access and departure, which can be disruptive to the typical statistical properties, e.g., deviating from Poisson distribution. The learning model needs to be adjusted to re-adapt to the new environment. We update the environment information every 200 time slots. From Fig. 4, both EMCL and AC-MAML are able to converge before each update, but EMCL saves 28.66% recovery time and reduces 45.42% objective value than AC-MAML, where we define a recovery time counting from the moment of dramatic performance degradation until the performance recoveries to the normal level. For AC-DDPG, the convergence performance is inferior to the others, and fails to converge when updating occurs at the 200-th and 600-th time slot. We remark that the case of user departure is easier to be adapted. Fewer users in the system reduce the problem dimension, and thus simplify the learning task, leading to a halved recovery time and flat curves between the 200-th and the 400-th slots in three algorithms. In contrast, it is more difficult to deal with the case of user arrival, referring to the large fluctuation after the 400-th slot, mainly due to lacking relevant new-user data and the exponentially increased dimension. We can observe that EMCL has strong capabilities in adapting to this difficult case and achieves more performance gains than the other two algorithms.

In Fig. 5, we evaluate the algorithms' capabilities in adapting to unforeseen dynamic demands. In the simulation, we assume that users can randomly request any volume of data, e.g., switching from a low-speed voice call to a data-hungry HD video service, or vice versa. In Fig. 6, we consider the channel states can undergo non-ideal large fluctuations, e.g., sharply deteriorated channel conditions due to the large obstacles or the rain/cloud blocks appearing in the transmission path. Analogous to Fig. 4, we collect the updated environment information every 200 time slots. From Fig. 5 and Fig. 6, AC-DDPG has poor convergence performance, since AC-DDPG needs to re-train the learning model from scratch when the environment changes, leading to a slow adaptation, while EMCL and AC-MAML extract the meta-knowledge from multiple tasks to accelerate the convergence speed. EMCL re-fits the learning model in a timely manner than AC-MAML. This is because EMCL uses meta critic to

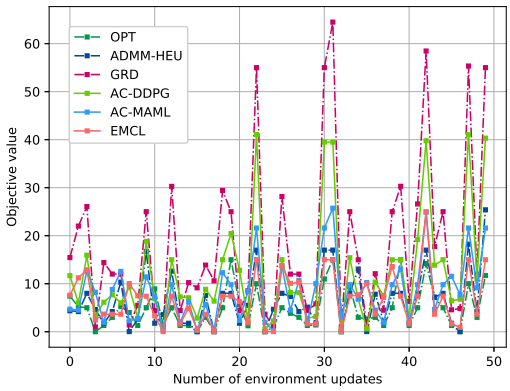

Fig. 8. Objective value vs. number of updates

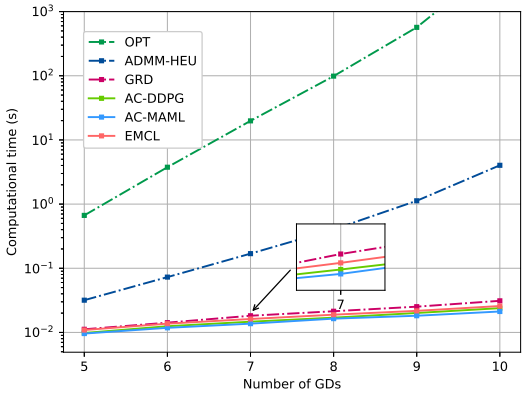

Fig. 9. Computational time vs. number of users

guide the actor to adjust scheduling schemes more effectively in a dynamic environment, and the designed HNN and Wolpertinger mapping approach can improve the learning accuracy and efficiency in large discrete action spaces.

Fig. 7 further summarizes the average recovery time based on Fig. 4-6 for the three algorithms with respect to the numbers of GDs. In general, the more GDs in the system, the longer the recovery time required to adapt to the new environment. On average, EMCL saves 29.83% and 10.21% recovery time compared to AC-DDPG and AC-MAML, respectively, and the time-saving gain of EMCL becomes even larger when more GDs in the system.

B. Trade-Offs between Computational Time and Optimality

To demonstrate EMCL's trade-off performance between approaching the optimum (Fig. 8) and computational time (Fig. 9), we compare EMCL with five benchmarks. In Fig. 8, we observe 50 environmental information updates and record the average objective values within each update cycle. For AC-MAML and AC-DDPG, the average gaps to the optimum are 45.26% and 57.23%, respectively, while for EMCL, the average gap drops to 27.58%. The performance of EMCL is slightly better than ADMM-HEU, around 3.54%. For GRD, the average gap to the optimum is 74.15%, which is inferior to the AC-based algorithms.

Fig. 9 compares the computational time with respect to the number of GDs. OPT is the most time-consuming algorithm, as expected. Compared to OPT, ADMM-HEU saves 98.14% computational time by decomposing variables into multiple blocks and performing parallel computations. The computational time in ECML, two AC algorithms, and GRD keep at the millisecond level, but the proposed EMCL achieves smaller gaps to the optimum, hence concludes the better trade-off performance of EMCL than other benchmarks.

VI. Conclusion

We have investigated a resource scheduling problem in dynamic LEO-terrestrial communication systems to address the mismatch issue in a practical over-loaded scenario. Due to the high computational time of the optimal algorithm and the proposed ADMM-HEU algorithm, we solve the problem from the perspective of DRL to obtain online solutions. To enable the learning model to fast adapt to dynamic environments, we develop an EMCL algorithm that is able to handle the environmental changes in wireless networks, such as bursty demands, users' entry/leave, and abrupt channel change. Numerical results show that, when encountering an environmental variation, EMCL consumes less recovery time to re-fit the learning model, compared to AC-DDPG and AC-MAML. Furthermore, EMCL achieves a good trade-off between solutions quality and computation efficiency compared to offline and AC-based benchmarks.

Appendix A
Proof of Lemma 1

We relax all the binary variables of $\mathbf{P1}$ to continuous variables $\hat{\mathbf{x}} = [\hat{x}_{1,1}, ..., \hat{x}_{g,t}, ..., \hat{x}_{G,T}]^{\mathrm{T}}$ and $\hat{\mathbf{y}} = [\hat{y}_1, ..., \hat{y}_k, ..., \hat{y}_K]^{\mathrm{T}}$, where $\hat{x}_{g,t}, \hat{y}_k \in [0, 1]$. The relaxed objective function is written by:

$$f(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = \eta_0 \left(\mathbf{1}^{\mathrm{T}}\hat{\mathbf{y}} - K\right)^2 + \sum_{k\in\mathcal{K}} \eta_k \left(\mathbf{r}_k^{\mathrm{T}}\hat{\mathbf{x}} - D_k\right)^2, \quad (34)$$

where $\mathbf{1} = [1, ..., 1]^{\mathrm{T}}$ and $\mathbf{r}_k = [R_{k,1,1}, ..., R_{k,g,t}, ..., R_{k,G,T}]^{\mathrm{T}}$. We expand $\left(\mathbf{1}^{\mathrm{T}}\hat{\mathbf{y}} - K\right)^2$ and $\left(\mathbf{r}_k^{\mathrm{T}}\hat{\mathbf{x}} - D_k\right)^2$ as follows:

$$\left(\mathbf{1}^{\mathrm{T}}\hat{\mathbf{y}} - K\right)^2 = \hat{\mathbf{y}}^{\mathrm{T}}\mathbf{E}\hat{\mathbf{y}} - 2D_k\mathbf{1}_k^{\mathrm{T}}\hat{\mathbf{y}} + K, \quad (35)$$

$$\left(\mathbf{r}_k^{\mathrm{T}}\hat{\mathbf{x}} - D_k\right)^2 = \hat{\mathbf{x}}^{\mathrm{T}}\mathbf{R}\hat{\mathbf{x}} - 2D_k\mathbf{r}_k^{\mathrm{T}}\hat{\mathbf{x}} + D_k^2, \quad (36)$$

where $\mathbf{E}$ is an all-ones matrix and

$$\mathbf{R} = \begin{bmatrix} R_{k,1,1}^2 & R_{k,1,1}R_{k,1,2} & \cdots & R_{k,1,1}R_{k,G,T} \\ R_{k,1,2}R_{k,1,1} & R_{k,1,2}^2 & \cdots & R_{k,1,2}R_{k,G,T} \\ \vdots & \vdots & \ddots & \vdots \\ R_{k,G,T}R_{k,1,1} & R_{k,G,T}R_{k,1,2} & \cdots & R_{k,G,T}^2 \end{bmatrix}. \quad (37)$$

Referring to the theorem of quadratic programming, a quadratic function is convex when its corresponding real symmetric matrix is positive semi-definite [28]. According to the definition, $\mathbf{E}$ and $\mathbf{R}$ are positive semi-definite matrices since, given an arbitrary vector $\mathbf{v} = [v_1, ..., v_{G\times T}] \neq \mathbf{0}$, we

can calculate $\mathbf{v}^T\mathbf{E}\mathbf{v} = \left(\mathbf{1}^{\mathrm{T}}\mathbf{v}\right)^2 \geq 0$ and $\mathbf{v}^T\mathbf{R}\mathbf{v} = \left(\mathbf{r}^{\mathrm{T}}\mathbf{v}\right)^2 \geq 0$ [28]. Therefore, $f(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ is convex as it is the summation of $K+1$ convex functions. Besides, the constraints Eq. (12b)-(12d) are linear, hence the conclusion.

Appendix B
Proof of Lemma 2

The objective of the learning agent is to find a policy $\pi(a|s_t)$ that maximizes the expected accumulated reward $\sum_{t=0}^{T}\gamma^t r_t$. With r_t in Eq. (20), we expand $\sum_{t=0}^{T}\gamma^t r_t$ as:

$$\begin{aligned}\sum_{t=0}^{T}\gamma^t r_t =& \sum_{k=0}^{K}\eta_k\left[\gamma\Delta_{k,0}^2 + \sum_{t=1}^{T}(\gamma^{t+1}-\gamma^t)\Delta_{k,t}^2 - \gamma^T\Delta_{k,T}^2\right]\\ \overset{\gamma=1}{=}& \sum_{k=0}^{K}\eta_k\left(\Delta_{k,0}^2 - \Delta_{k,T}^2\right)\\ =& \sum_{k=0}^{K}\eta_k\Delta_{k,0}^2 - \eta_0\left(\sum_{k=1}^{K}\mathbb{1}\left(b_{k,T}-D_k'\right)-K\right)^2\\ &- \sum_{k=1}^{K}\eta_k\left(b_{k,T}-D_k\right)^2, \end{aligned} \tag{38}$$

where $b_{k,T} = \sum_{t=1}^{T} R_{k,a_t,t}$. Thus, we can obtain the optimal policy $a_t^* \sim \pi^*(a|s_t)$ by solving the following problem:

$$\begin{aligned}\max_{\pi(a|s_t)} -\mathbb{E}_{\pi(a|s_t)}&\left[\eta_0\left(\sum_{k=1}^{K}\mathbb{1}\left(\sum_{t=1}^{T}R_{k,a_t,t}-D_k'\right)-K\right)^2\right.\\ &\left.- \sum_{k=1}^{K}\eta_k\left(\sum_{t=1}^{T}R_{k,a_t,t}-D_k\right)^2\right], \end{aligned} \tag{39}$$

which is equivalent to the objective Eq. (12a), thus the conclusion.

Appendix C
Proof of Lemma 3

Denote $Q(s,a_1),...,Q(s,a_M)$ as random variables $X_1,...,X_M$, where $X_{m'} = Q(s,a_s^*)$ and $X_m \sim U(Q(s,a_s^*)-\kappa, Q(s,a_s^*)+\kappa)$, $\forall m \neq m'$. Thus, $Q(s,a_w^*)$ can be expressed as a random variable $\Psi = \max\{X_1,...,X_M\}$. The cumulative distribution function of Ψ is expressed as:

$$\begin{aligned}F_\Psi(\psi) =& \mathbb{P}[\Psi \leq \psi] = \mathbb{P}[\max\{X_1,...,X_M\} \leq \psi]\\ =& \mathbb{P}[X_1 \leq \psi]\mathbb{P}[X_2 \leq \psi]...\mathbb{P}[X_M \leq \psi]\\ =& F_{X_1}(\psi)F_{X_2}(\psi)...F_{X_M}(\psi)\end{aligned} \tag{40}$$

For $m \neq m'$, based on the cumulative distribution function of uniform distribution, we can derive:

$$\begin{aligned}F_{X_m}(\psi) =& \frac{\psi - Q(s,a_s^*)+\kappa}{2\kappa},\\ &\psi \in \left[Q(s,a_s^*)-\kappa, Q(s,a_s^*)+\kappa\right].\end{aligned} \tag{41}$$

For $m = m'$, as $X_{m'} = Q(s,a_s^*)$, the cumulative distribution function is:

$$F_{X_{m'}}(\psi) = \mathbb{P}[X_{m'} \leq \psi] = \begin{cases} 1, & \psi \geq Q(s,a_s^*),\\ 0, & \psi < Q(s,a_s^*).\end{cases} \tag{42}$$

By substituting Eq. (41) and Eq. (42) into Eq. (40),

$$F_\Psi(\psi) = \begin{cases}\left(\frac{y-Q(s,a_s^*)+\kappa}{2\kappa}\right)^{M-1}, & \psi \in [Q(s,a_s^*), Q(s,a_s^*)+\kappa],\\ 0, & \psi \in [Q(s,a_s^*)-\kappa, Q(s,a_s^*)).\end{cases} \tag{43}$$

Then, the probability density function of Ψ can be calculated by solving the first derivative:

$$\begin{aligned}&f_\Psi(\psi) = [F_\Psi(\psi)]'\\ =& \begin{cases}\frac{1}{2^{M-1}}\delta(\psi - Q(s,a_s^*)), & \psi = Q(s,a_s^*),\\ \frac{M-1}{2\kappa}\left(\frac{\psi-Q+\kappa}{2\kappa}\right)^{M-2}, & Q(s,a_s^*) < \psi \leq Q(s,a_s^*)+\kappa,\\ 0, & \text{otherwise},\end{cases}\end{aligned} \tag{44}$$

where $\delta(\cdot)$ is Dirac function. The expectation of Ψ is:

$$\begin{aligned}&\mathbb{E}[\Psi] = \mathbb{E}[Q(s,a_w^*)] = \int_{Q(s,a_s^*)}^{Q(s,a_s^*)+\kappa}\psi f_\Psi(\psi)d\psi\\ =& \frac{1}{2^{M-1}}\int_{Q(s,a_s^*)^-}^{Q(s,a_s^*)^+}\psi\delta(y-Q(s,a_s^*))dy\\ &+ \int_{Q(s,a_s^*)^+}^{Q(s,a_s^*)+\kappa}\psi\frac{M-1}{2\kappa}\left(\frac{\psi-Q(s,a_s^*)+\kappa}{2\kappa}\right)^{M-2}d\psi\\ =& \frac{Q(s,a_s^*)}{2^{M-1}} + Q(s,a_s^*) + \kappa - \frac{2\kappa}{M} - \frac{Q(s,a_s^*)}{2^{M-1}} + \frac{\kappa}{M\cdot 2^{M-1}}\\ =& Q(s,a_s^*) + \kappa\left(1 - \frac{2(2^M-1)}{M\cdot 2^M}\right).\end{aligned} \tag{45}$$

Thus the conclusion.

References

[1] Y. Li, E. Pateromichelakis, N. Vucic, J. Luo, W. Xu, and G. Caire, "Radio resource management considerations for 5G millimeter wave backhaul and access networks," in *IEEE Communication Magazine*, vol. 55, no. 6, pp. 86–92, Jun. 2017.

[2] J. Wu, C. Yuen, N. M. Cheung, and J. Chen, "Delay-constrained high definition video transmission in heterogeneous wireless networks with multi-homed terminals," in *IEEE Transaction on Mobile Computing*, vol. 15, no. 3, pp. 641–655, Mar. 2016.

[3] O. Kodheli et al., "Satellite communications in the new space era: a Survey and future challenges," in *IEEE Communications Surveys and Tutorials*, vol. 23, no. 1, pp. 70-109, 2021.

[4] L. You, K. Li, J. Wang, X. Gao, X. Xia, and B. Ottersten, "Massive MIMO transmission for LEO satellite communications," in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 8, pp.1851-1865, Jun. 2020.

[5] B. Di, H. Zhang, L. Song, Y. Li, and G. Y. Li, "Ultra-dense LEO: integrating terrestrial-satellite networks into 5G and beyond for offloading," in *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 47-62, Jan. 2019.

[6] Y. Li, N. Deng, and W. Zhou, "A hierarchical approach to resource allocation in extensible multi-layer LEO-MSS," in *IEEE Access*, vol. 8, pp.18522-18537, Jan. 2020.

[7] S. Wang, Y. Li, Q. Wang, M. Su, and W. Zhou, "Dynamic downlink resource allocation based on imperfect estimation in LEO-HAP cognitive system," in Proc. *IEEE International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct. 2019.

[8] J. H. Lee, J. Park, M. Bennis, and Y. C. Ko, "Integrating LEO satellite and UAV relaying via reinforcement learning for non-terrestrial networks," in Proc. *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2020.

[9] S. He, T. Wang, and S. Wang, "Load-aware satellite handover strategy based on multi-agent reinforcement learning," in Proc. *IEEE Global Communications Conference (GLOBCOM)*, Dec. 2020.

[10] B. Deng, C. Jiang, H. Yao, S. Guo and S. Zhao, "The next generation heterogeneous satellite communication networks: integration of resource management and deep reinforcement learning," in *IEEE Wireless Communications*, vol. 27, no. 2, pp. 105-111, Apr. 2020.

[11] L. Lei, Y. Yuan, T. X. Vu, S. Chatzinotas, M. Minardi, and J. F. Mendoza Montoya, "Dynamic-adaptive AI solutions for network slicing management in satellite-integrated B5G systems," in *IEEE Network Magazine*, 2021.

[12] Y. Shen, Y. Shi, J. Zhang and K. B. Letaief, "LORM: learning to optimize for resource management in wireless networks with few training samples," in *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 665-679, Jan. 2020.

[13] O. Simeone, S. Park and J. Kang, "From learning to meta-learning: reduced training overhead and complexity for communication systems," in *6G Wireless Summit (6G SUMMIT)*, pp. 1-5, 2020.

[14] H. Sun, W. Pu, M. Zhu, X. Fu, T. H. Chang and M. Hong, "Learning to continuously optimize wireless resource in episodically dynamic environment," in Proc. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4945-4949, 2021.

[15] K. Javed and M. White, " Meta-learning representations for continual learning," in Proc. *Neural Information Processing Systems (NIPS)*, pp. 1820-1830, Dec. 2019.

[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in Proc. *International Conference on Machine Learning (ICML)*, pp. 1126-1135, Jul. 2017.

[17] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," in Proc. *International Conference on Machine Learning (ICML),* pp. 5331–5340, 2019.

[18] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang, "Learning to learn: meta-critic networks for sample efficient learning," in *arXiv preprint arXiv:1706.09529*, Jun. 2017.

[19] H. Wu, Z. Zhang, C. Jiao, C. Li and T. Q. S. Quek, "Learn to sense: a meta-learning-based sensing and fusion framework for wireless sensor networks," in *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8215-8227, Oct. 2019.

[20] S. Park, O. Simeone and J. Kang, "Meta-learning to communicate: fast end-to-end training for fading channels," in Proc. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5075-5079, 2020.

[21] 3GPP TR 23.737, "Study on architecture aspects for using satellite access in 5G (release 17)".

[22] M. Cheng, J. B. Wang, J. Cheng, J. Y. Wang and M. Lin, "Joint scheduling and precoding for mmwave and sub-6ghz dual-mode networks," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13098-13111, Nov. 2020.

[23] X. Fang, W. Feng, T. Wei, Y. Chen, N. Ge, and C. X. Wang, "5G embraces satellites for 6G ubiquitous IoT: Basic models for integrated satellite terrestrial networks,"in *IEEE Internet of Things Journal*, vol. 8, no. 18, pp.14399-14417, Mar. 2021.

[24] A. Alsharoa and M. S. Alouini, "Improvement of the global connectivity using integrated satellite-airborne-terrestrial networks with resource optimization," in *IEEE Transactions on Wireless Communications,* vol. 19, no. 8, pp. 5088-5100, Apr. 2020.

[25] "Doppler compensation, uplink timing advance and random access in NTN," 3GPP TSG RAN WG1 Meeting, no. 97, R1-1906087, May 2019.

[26] K. Guo et al., "Performance analysis of hybrid satellite-terrestrial cooperative networks with relay selection," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 9053-9067, Aug. 2020.

[27] H. Wang and N. Moayeri, "Finite-State Markov Channel-A Useful Model for Radio Communication Channels," in *IEEE Transactions on Vehicular Technology*, vol. 44, no. 1, pp. 163-171, Feb. 1995.

[28] Nocedal, J. and Wright, S., "Numerical optimization," Springer Science & Business Media, 2006.

[29] C. H. Papadimitriou and K. Steiglitz, "Combinatorial optimization: algorithms and complexity," Mineola, NY, USA: Dover, 1998.

[30] S. Boyd, N. Parikh, and E. Chu, "Distributed optimization and statistical learning via the alternating direction method of multipliers," Hanover, MA, USA: Now Publishers Inc., 2011.

[31] R. S. Sutton and A.G. Barto, "Reinforcement Learning: An Introduction", Cambridge, Massachusetts, USA: MIT Press, 2018.

[32] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in HetNets with hybrid energy supply: an actor-critic reinforcement learning approach," in *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 680-692, Nov. 2017.

[33] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: a review," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346-2363, Dec. 2019.

[34] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning". London, MIT press, 2016.

[35] J. Ye, and H. Gharavi, "Deep reinforcement learning-assisted energy harvesting wireless networks," in *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 990-1002, Jun. 2021.

[36] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in Proc. *IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 5353-5360, 2015.

[37] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in Proc. *International Speech Communication Association (ISCA)*, 2014.

[38] Y. Yuan, L. Lei, T. X. Vu, S. Chatzinotas, S. Sun, and B. Ottersten, "Energy minimization in UAV-aided networks: actor-critic learning for constrained scheduling optimization," in *IEEE Transactions on Vehicular Technology*, vol. 70, no. 5, pp. 5028-5042, May 2021.

[39] Z. Jiang, S. Chen, S. Zhou, and Z. Niu, "Joint user scheduling and beam selection optimization for beam-based massive MIMO downlinks," in *IEEE Transactions on Wireless Communications,* vol. 17, no. 4, pp. 2190-2204, Jan. 2018.

[40] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic Policy Gradient Algorithms," in Proc. *Neural Information Processing Systems (NIPS)*, Jun. 2014.

[41] O. Popescu, "Power budgets for cubesat radios to support ground communications and inter-satellite links," in *IEEE Access*, vol. 5, pp. 12618-12625, Jun. 2017.