Decomposition and Meta-DRL based Multi-Objective Optimization for Asynchronous Federated Learning in 6G-Satellite Systems

Yu Zhou, Lei Lei, *Member, IEEE,* Xiaohui Zhao, Lei You, *Member, IEEE,* Yaohua Sun, Symeon Chatzinotas, *Fellow, IEEE*

***Abstract*—Wireless-based federated learning (FL), as an emerging distributed learning approach, has been widely studied for 6G systems. When the paradigm shifts from terrestrial to non-terrestrial networks (NTN), FL may need to address several open challenges, e.g., the limited service time of low earth orbit (LEO) satellites, the straggler issue in synchronous FL, and time-efficient uploading and aggregation for massive devices. In this work, we exploit the synergy of LEO and FL for future integrated 6G-satellite systems by taking advantage of ubiquitous wireless access provided by LEO and appealing characteristics of collaborative training and data privacy preservation in FL. The studied LEO-FL framework may need to improve multi-metric performance in practice. Different from most FL works, we simultaneously improve the communication-training efficiency and local training accuracy from a multi-objective optimization (MOO) perspective. To solve the problem, we propose a decomposition and meta-deep reinforcement learning based MOO algorithm for FL (DMMA-FL), aiming at adapting to the dynamic satellite-terrestrial environments, achieving efficient uploading and aggregation, and approaching Pareto optimal sets. Compared to single-objective optimization, heuristics-based, and learning-based MOO algorithms, the effectiveness and advantages of the proposed LEO-FL framework and DMMA-FL algorithm are assessed on MNIST and CIFAR-10 datasets.**

***Index Terms*—LEO satellite, asynchronous federated learning, multi-objective optimization, meta-reinforcement learning**

I. Introduction

In 5G and 6G systems, emerging intelligent devices and applications generate unprecedented amounts of data at the edge network every day [1]. This, in turn, stimulated the need for on-device data processing and accurate-efficient inference at the edge. Due to limited computing capabilities, it is impractical for edge devices to train a powerful learning model by collecting a huge amount of training data [2]. In addition, the concerns of data privacy prevent data sharing among devices and third-party institutions. To address these issues, federated learning (FL) is introduced to complete training and inference without sharing sensitive raw data among edge devices [3]. By leveraging on-device training and local model aggregation, FL has exhibited great potential to train a common high-quality learning model and received considerable attention from both academia and industry [4]–[6]. However, previous wireless-based FL studies have primarily focused on terrestrial systems within a limited coverage area, where the potential of FL in integrating fragmented computing resources from edge devices have not been fully demonstrated.

When the 6G paradigm shifts from terrestrial to non-terrestrial networks (NTN), low earth orbit (LEO) satellites have become an important component in 6G NTN [7]. Relying on wide coverage and seamless connectivity, LEO can act as a central server in FL to aggregate parameters of local models and update the global model. The LEO-FL system has recently received early-stage studies. For instance, Razmi *et al.* [8] proposed a ground-assisted LEO-FL system and considered a global model convergence problem. Chen *et al.* [9] studied a satellite FL system and analyzed four methods of combining machine learning with satellite networks. Matthiesen *et al.* [10] investigated a classification of satellite FL systems based on the communication capabilities of the satellites, the constellation design, and the location of the parameter server. From another line of studies, general terrestrial-based wireless FL approaches are widely investigated. Device selection and resource allocation are crucial for accelerating FL convergence and improving system performance. Several studies have focused on these aspects. Chen *et al.* [11] proposed a probabilistic device selection scheme to accelerate the global model convergence in FL systems. Chen *et al.* [12] designed an FL scheme called SDEFL to reduce the communication-computing time via efficiently allocating computing resources in the FL training process across heterogeneous devices. Additionally, researchers have also addressed the challenge of the straggler issue, where the central server needs to wait for the slowest device's training before aggregating the local models. For instance, Chen *et al.* [13] proposed a fully asynchronous

This work was supported in part by the Open Research Fund of National Mobile Communications Research Laboratory of Southeast University under Grant 2023D02; in part by the National Key Laboratory Foundation under Grant 2023-JCJQ-LB-007; in part by the Qin Chuangyuan Innovation and Talent Project, under Grant QCYRCXM-2023-049; in part by the Natural Science Foundation of Sichuan Province under Grant 2023NSFSC0455; in part by the National Natural Science Foundation of China under Grant 62371071; and in part by the Young Elite Scientists Sponsorship Program by CAST under Grant 2021QNRC001. (Corresponding author: Lei Lei)

Yu Zhou, Lei Lei, and Xiaohui Zhao are with the School of Information and Communications Engineering, Xi'an Jiaotong University, 710049 Xi'an, China (email: zy-yu.zhou@connect.polyu.hk, lei.lei@xjtu.edu.cn, xhzhao@stu.xjtu.edu.cn). Yu Zhou is also with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR 999077, China. Lei Lei is also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China.

Lei You is with Department of Engineering Technology, Technical University of Denmark, Ballerup, Denmark (e-mail: leiyo@dtu.dk).

Yaohua Sun is with the State Key Laboratory of Networking and Switching Technology (SKL-NST), Beijing University of Posts and Telecommunications, Beijing 510300, China (e-mail: sunyaohua@bupt.edu.cn).

Symeon Chatzinotas is with Interdisciplinary Center for Security, Reliability and Trust, University of Luxembourg, 1855 Luxembourg (email: symeon.chatzinotas@uni.lu).

FL framework in which devices perform online model training with continuous streaming of local data while a parameter server aggregates global model parameters from the devices. Hu *et al.* [14] introduced an asynchronous strategy with periodic aggregation, aiming to achieve fast convergence while avoiding excessive model updating typically encountered in fully asynchronous FL settings. Nguyen *et al.* [15] proposed FedBuffer, a novel buffered asynchronous aggregation method that addresses scalability and privacy concerns through secure aggregation and differential privacy techniques.

For LEO-FL systems, the previous works were studied to a limited extent. Some key issues need to be addressed. Firstly, tailored schemes of device selection and resource optimization for LEO-FL systems need to be investigated. Specifically, LEOs' high mobility results in time-varying environments and limited training time. Only a part of the devices in the service area can participate in FL local training. Different participating devices may affect the performance of the global model. In addition, edge devices typically have limited computing resources and energy. Thus, efficient optimization of computation-communication resources is of importance.

Secondly, the LEO-FL system needs to address the straggler issue caused by conventional synchronous FL. Since the mobility of LEOs could lead to the heterogeneity of communication channel conditions and the computing resources of devices are limited, the studied system could suffer from the straggler issue if utilizing traditional synchronous global model aggregation, as it would be difficult for the LEO to wait for all the participated devices to complete the model training before executing global model aggregation. To address this challenge, efficient schemes for asynchronous uploading and aggregation are needed.

Thirdly, since the service time of the LEO is limited, it is necessary to reduce the communication-computation time throughout the whole FL process while keeping good accuracy. In practice, it often occurs that multiple performance metrics are in conflict with each other. Most studies simplify the optimization process by summarizing various metrics with predefined weights and solving a single-objective optimization problem (SOP). However, SOP in this context may have several issues, e.g., non-unified units among different metrics, inaccuracies of weight allocation due to ill-conditioned matrixes, and difficulties in approaching the Pareto front in large-scale optimization. Therefore, multi-objective optimization (MOO) is worth considering. The aim of this paper is to overcome these challenges in LEO-FL systems. The main contributions are summarized as follows:

- We study an LEO-FL system and formulate a multi-objective optimization problem (MOP) to minimize the global model convergence and the communication-computing time in the FL training process by jointly determining device selection, transmit power, and computing resource allocation. To our best knowledge, this is the first attempt to investigate high-quality and efficient MOO solutions for LEO-FL systems.
- We propose a decomposition and meta-deep reinforcement learning (meta-DRL) based MOO algorithm (DMMA-FL) to solve the MOP in LEO-FL systems. Compared to conventional MOO methods, the combined optimization and learning design in DMMA-FL accelerates the decision-making process in each iteration and improves the capabilities of approaching the Pareto optimum and enhances solutions' dynamic adaptation. Additionally, meta-DRL can efficiently handle decomposed sequence optimization subproblems with Markov properties compared with conventional and heuristic methods.
- To overcome the effect of the straggler issue in LEO-FL systems, we design an asynchronous uploading and weighted aggregation (AUWA) scheme for time-efficient FL training and transmission.
- The effectiveness and performance gains of the considered LEO-FL system and the proposed DMMA-FL are validated on MNIST and CIFAR-10 datasets in both independent and identically distributed (IID) and Non-IID settings.

The paper is structured as follows: Section II introduces the system model. Section III formulates the MOP. In Section IV, we provide a detailed explanation of our proposed method. The experimental results are discussed in Section V. Finally, Section VI concludes the paper. Additionally, for the ease of illustration, all main mathematical notations are summarized in Table I.

TABLE I
THE LIST OF MAIN MATHEMATICAL NOTATIONS

Notation	Description
$\boldsymbol{x}_{m,n}$	The input feature output label for the n-th data sample in $\mathcal{D}_m$
$y_{m,n}$	The output label for the n-th data sample in $\mathcal{D}_m$
D_m	The number of data samples in $\mathcal{D}_m$
$\boldsymbol{w}^t$	The parameters of the global model in the t-th round
$\boldsymbol{w}_m^t$	The parameters of the local model of device m in the t-th round
$F_m(\boldsymbol{w}_m^t)$	The loss function of the local model of device m
$f_n(\boldsymbol{x}_{m,n}, y_{m,n}; \boldsymbol{w}_m^t)$	The loss function on the n-th sample data of device m
τ_l	The learning rate of the local model
$F(\boldsymbol{w}^{t*})$	The loss function of the global model
τ_g	The learning rate of the global model
h_m^t	The channel state between the LEO and device m in the t round
G_T	The transmitting antenna gain
G_R	The receiving antenna gain
G_C	The channel loss
$r_{m,u}^t$	The achievable transmission rate in the uplink for device m in the t-th round
b_m^t	The bandwidth and transmit power in the uplink for device m in the t-th round
p_m^t	The transmit power in the uplink for device m in the t-th round
$L_{m,u}^{t,com}$	The communication time in the uplink for device m in the t-th round
X	The size of the global model
$E_{m,u}^{t,com}$	The energy consumption in the uplink for device m in the t-th round
$r_{m,d}^t$	the achievable transmission rate in the downlink for device m in the t-th round
$L_{m,d}^{t,com}$	The communication time in the downlink for device m in the t-th round
$E_{m,d}^{t,com}$	The communication energy consumption in the downlink for device m in the t-th round
$L_m^{t,cmp}$	The computing time for device m in the t-th round
κ	The number of local rounds
g_m^t	The CPU frequency of device m in the t-th round
β_g	The number of frequency levels
α_m	The capacitance constant of device m
$E_m^{t,cmp}$	The computing energy consumption for device m in the t-th round
L^t	The communication-computing time in the t-th round
s_m^t	The device selection variable for device m in the t-th round
E^t	The communication-computing energy consumption in the t-th round
$\mathcal{E}(\boldsymbol{w}_m^t)$	The packet error rate of each device m in the t-th round
$\mathcal{U}(\boldsymbol{w}_m^t)$	The determination for whether to perform retransmission of device m
β_p	The number of power levels
C^t	The global model convergence reference function
λ^i	The i-th weight vector
$\boldsymbol{z}^*$	The Pareto optimal solution for each subproblem
$\boldsymbol{\theta}_h$	The parameters of the actor network for the sampled weight vector h
$\boldsymbol{\phi}_h$	The parameters of critic networks for the sampled weight vector h
ρ	The learning rate of the actor and critic networks
$\boldsymbol{\theta}_{meta}$	The parameters of the meta-actor network
$\boldsymbol{\phi}_{meta}$	The parameters of the meta-critic network
η	The learning rate of meta-learning
T_{ft}	The number of fine-tuning episodes

II. System Model

A. LEO-FL Systems

As depicted in Fig. 1, we consider an LEO-FL system, which includes an LEO equipped with multiple antennas for global model sharing and aggregation, and a set of M single-antenna devices with limited communication and computing resources for local model update and uploading. We denote the set of all devices as $\mathcal{M} = \{1, \cdots, M\}$. For each device m, it updates its local model by utilizing its own unique local dataset $\mathcal{D}_m$, which consists of D_m data samples $(\boldsymbol{x}_{m,n}, y_{m,n})$, $n = 1, \cdots, D_m$. Here, $\boldsymbol{x}_{m,n}$ and $y_{m,n}$ represent the input feature and the corresponding output label for the n-th data sample, respectively, and D_m denotes the number of data samples. Own to the LEO's dynamic position and limited bandwidth, only a portion of devices are selected to participate in the FL training process. We denote the set of the selected devices as $\mathcal{M}'$.

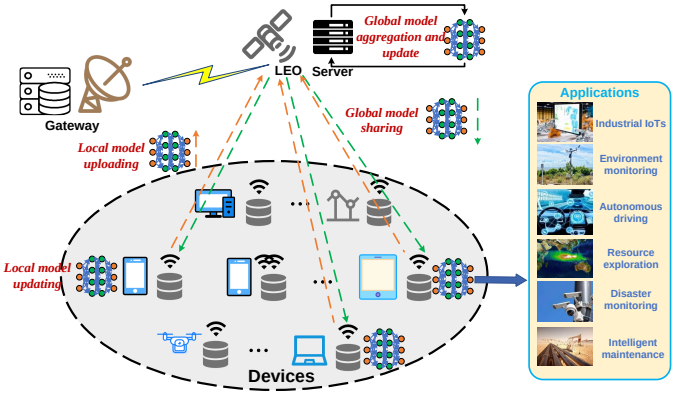


Fig. 1. An LEO-FL system.

The detailed FL training process, consisting of T rounds, is provided as follows:

Step 1: Global model initialization and sharing. In the first round, the LEO determines the training task and initializes the parameters of the global model denoted as $\boldsymbol{w}^t$. In each round t ($t \in \mathcal{T} = \{1, \cdots, T\}$), the LEO selects devices within its service area and shares the global model with them.

Step 2: Local model update and uploading. Upon receiving the global model, each selected device m ($m \in \mathcal{M}'$) uses its own $\mathcal{D}_m$ to update its local model, whose parameters are denoted as $\boldsymbol{w}_m^t = \boldsymbol{w}^t$. The objective of updating the local model is to find the optimal $\boldsymbol{w}_m^{t*}$ by minimizing the loss function of the local model of device m, denoted as $F_m(\boldsymbol{w}_m^t) := \frac{1}{D_m}\sum_{n=1}^{D_m} f_n(\boldsymbol{x}_{m,n}, y_{m,n}; \boldsymbol{w}_m^t)$, where $f_n(\boldsymbol{x}_{m,n}, y_{m,n}; \boldsymbol{w}_m^t)$ denotes the loss function on the n-th sample data of device m. In this paper, since FL is used for performing classification tasks, $f_n(\boldsymbol{x}_{m,n}, y_{m,n}; \boldsymbol{w}_m^t)$ is defined as the standard cross-entropy loss function, which is given as $f_n(\boldsymbol{x}_{m,n}, y_{m,n}; \boldsymbol{w}_m^t) = \sum_{cl=0}^{Class} P(y_{m,n} = i)\mathbb{E}_{\boldsymbol{x}_{m,n}|y_{m,n}=i}[\log \hat{y}_{m,n}]$ [6]. Here, $Class$ is the number of classes in datasets (i.e., MNIST and CIFAR-10), $P(y_{m,n} = i)$ is the probability of $y_{m,n}$ belonging to i-th class and $\hat{y}_{m,n}$ is the predicted label for the n-th data sample. Subsequently, the stochastic gradient descent (SGD) algorithm is employed to minimize $F_m(\boldsymbol{w}_m^t)$, and $\boldsymbol{w}_m^t$ is updated as

$$\boldsymbol{w}_m^{t*} = \boldsymbol{w}_m^t - \tau_l \nabla F_m(\boldsymbol{w}_m^t), \tag{1}$$

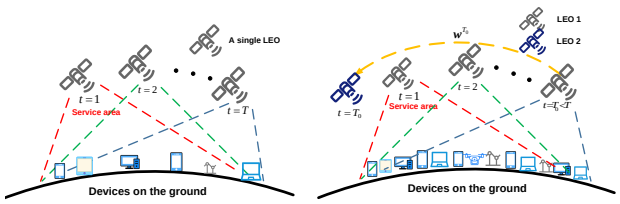
where τ_l is the learning rate of the local model and $\nabla F_m(\boldsymbol{w}_m^t)$ represents the gradient of the local model of device m. Finally, each device m can upload $\boldsymbol{w}_m^{t*}$ to the LEO.

Step 3: Global model aggregation and update. After receiving local models, the LEO performs global model aggregation based on $\boldsymbol{w}_m^{t*}$ using $F(\boldsymbol{w}^{t*}) = \sum_{m\in\mathcal{M}'} \frac{D_m}{D} F_m(\boldsymbol{w}_m^{t*})$, where $F(\boldsymbol{w}^{t*})$ is the loss function of the global model and $D = \sum_{m\in\mathcal{M}'} D_m$ denotes the total number of the data samples of the selected devices. Afterwards, the LEO updates $\boldsymbol{w}^t$ by minimizing $F(\boldsymbol{w}^{t*})$ using the SGD algorithm:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \tau_g \nabla F(\boldsymbol{w}^{t*}), \tag{2}$$

where τ_g is the learning rate of the global model and $\nabla F(\boldsymbol{w}^{t*})$ represents the gradient of the global model. Then, the LEO proceeds to devices in the next round and shares the updated global model with them.

Step 2 and Step 3 repeat until the maximum number of rounds is reached or the global model achieves convergence.



(a) First scenario (b) Second scenario

Fig. 2. Two scenarios in the LEO-FL system

Remark: We consider two scenarios in the studied system. In Fig. 2(a), the FL training process can be completed within the service time of a single LEO. In Fig. 2(b), the FL training process cannot be completed in time, where the global model may not converge or have lower accuracy when the LEO leaves the service area. Moreover, the incoming LEO would need to restart the FL training process from scratch. In the system, we consider collaborations among LEOs. The current global model can be transferred from one LEO to the next, either via inter-satellite links or the gateway.

B. Communication Model

1) Uplink Communication: In the uplink, the parameters of local models are transmitted from the participating devices to the LEO. We assume that the Doppler shift effect caused by high-mobility LEOs can be effectively compensated at the gateway by using information about device locations, satellite orbits, and speeds [16]. In the t-th round, the channel state h_m^t between the LEO and device m can be modeled as:

$$h_m^t = G_T \cdot G_C \cdot G_R, \tag{3}$$

where G_T and G_R represent the transmitting and receiving antenna gain, respectively, and G_C represents the channel loss. In the paper, G_C is assumed a Rician fading channel [17],

denoted as $G_C = \left(\frac{c}{4\pi d f_c}\right)^2 \cdot G_H \cdot A(d) \cdot \delta$, where c is the speed of light, $d = d_m^t$ represents the distance between the LEO and device m in the t-th round, f_c is the carrier frequency, G_H represents pitch angle fading, and δ is the Rician fading factor. Here, $A(d)$ represents atmospheric loss, denoted as $A(d) = 10^{\left(\frac{3\psi d}{10\nu}\right)}$, where ψ in dB/km represents attenuation caused by clouds and rain, and ν is the altitude of the LEO.

In each round, selected device m is assigned one subchannel with h_m^t. Then, the achievable transmission rate in the uplink for device m in the t-th round can be denoted as

$$r_{m,u}^t = b_m^t \log_2\left(1 + \frac{p_m^t \left(h_m^t\right)^2}{\sigma^2}\right), \tag{4}$$

where b_m^t and p_m^t represent the bandwidth and transmit power in the uplink for device m in the t-th round, respectively, and σ^2 represents Gaussian noise.

The communication time in the uplink for device m in the t-th round, which is expressed as

$$L_{m,u}^{t,com} = \frac{X}{r_{m,u}^t} = \frac{X}{b_m^t \log_2\left(1 + \frac{p_m^t (h_m^t)^2}{\sigma^2}\right)}, \tag{5}$$

where X is the size of the global model (i.e., $\boldsymbol{w}^t$) in bits and it can be premeasured [18] after the training task is determined and the parameters of the global model are initialized in Step 1. Then, the energy consumption in the uplink for device m in the t-th round is given by $E_{m,u}^{t,com} = L_{m,u}^{t,com} p_m^t$.

2) Downlink Communication: In the downlink, the global model is transmitted from LEO to edge devices. Similar to [17], [19], [20], we consider a block fading channel and the channel state h_m^t remains within a round. Thus, the achievable transmission rate in the downlink for device m in the t-th round can be defined as $r_{m,d}^t = b_{leo}^t \log_2\left(1 + \frac{p_{leo}\left(h_m^t\right)^2}{\sigma^2}\right)$, where b_{leo}^t and p_{leo} are the bandwidth and transmit power of the LEO in the t-th round, respectively.

Hence, the communication time in the downlink for device m in the t-th round can be given as

$$L_{m,d}^{t,com} = \frac{X}{r_{m,d}^t} = \frac{X}{b_{leo}^t \log_2\left(1 + \frac{p_{leo}(h_m^t)^2}{\sigma^2}\right)}. \tag{6}$$

Similarly, the communication energy consumption in the downlink for device m in the t-th round can be expressed as $E_{m,d}^{t,com} = L_{m,d}^{t,com} p_m^t$.

C. Computation Model

In the system, each device has limited computing ability to update its local model. Specifically, the CPU cycles required to process one data sample $(\boldsymbol{x}_{m,n}, y_{m,n})$ of device m is denoted as c_m [21]. Consequently, the total number of CPU cycles required for device m to perform one local round is $c_m D_m$. Thus, the computing time for device m in the t-th round can be expressed as

$$L_m^{t,cmp} = \frac{\kappa c_m D_m}{g_m^t}, \tag{7}$$

where κ is the number of local rounds and g_m^t denotes the CPU frequency of device m in the t-th round. Here, $g_m^t \in \{g_{m,1}, \cdots, g_{m,\beta_g}\}$ is divided into β_g frequency levels based on their magnitudes. Furthermore, the computing energy consumption for device m in the t-th round can be expressed as $E_m^{t,cmp} = \kappa \alpha_m c_m D_m (g_m^t)^2 = Q_m D_m (g_m^t)^2$, where α_m is the capacitance constant of device m.

D. Communication-Computation Time Model

Since there are two manners (i.e., synchronous and asynchronous) [22] for devices to upload local models in the uplink, the communication-computing time in the t-th round for two manners can be calculated as:

$$\bar{L}^t = \max\{s_m^t L_{m,d}^{t,com}\} + \max\{s_m^t L_m^{t,cmp}\} + \max\{s_m^t L_{m,u}^{t,com}\}, \tag{8}$$

$$L^t = \max\{s_m^t (L_{m,d}^{t,com} + L_m^{t,cmp} + L_{m,u}^{t,com})\}, \tag{9}$$

where s_m^t represents the device selection variable for device m in the t-th round. Specifically, $s_m^t = 1$ indicates that device m is selected and assigned a subchannel, while $s_m^t = 0$ means device m is not selected.

Similarly, the communication-computing energy consumption in the t-th round can be expressed as $E^t = \sum_{m=1}^{M} s_m^t \left(E_{m,d}^{t,com} + E_m^{t,cmp} + E_{m,u}^{t,com}\right)$.

E. Packet Error and Retransmision Model

Since this is an early attempt for FL in 6G-satellite system, we utilize the mainly used error model in FL systems [23]–[25]. Specifically, we make the assumption that the local model of each device is transmitted as a single packet in the uplink, and then employ a cyclic redundancy check mechanism to verify data integrity in the received local models. Then, the packet error rate of the local model of each device m in the t-th round is calculated as

$$\mathcal{E}(\boldsymbol{w}_m^t) = 1 - e^{-\frac{\varepsilon(I + b_m^t \sigma^2)}{p_m^t h_m^t}} \tag{10}$$

where ε is a waterfall threshold and I is the interference. Based on (10), we design the retransmission model, where $\mathcal{U}(\boldsymbol{w}_m^t)$ is defined to determine whether perform retransmission of device m:

$$\mathcal{U}(\boldsymbol{w}_m^t) = \begin{cases} 1, \text{with probability} & 1 - \mathcal{E}(\boldsymbol{w}_m^t), \\ 0, \text{with probability} & \mathcal{E}(\boldsymbol{w}_m^t). \end{cases} \tag{11}$$

Thus, when receiving $\boldsymbol{w}_m^t$, the LEO could check the value of $\mathcal{U}(\boldsymbol{w}_m^t)$ from (11). If $\mathcal{U}(\boldsymbol{w}_m^t) = 1$, there is no error in the local model of device m, and the LEO stores $\boldsymbol{w}_m^t$ for the following global model update. If $\mathcal{U}(\boldsymbol{w}_m^t) = 0$, it means an error is detected in $\boldsymbol{w}_m^t$, and the LEO would send a retransmission request to device m. Then, device m will resend $\boldsymbol{w}_m^t$, and the LEO will check the local model again. This process could continue until the LEO stores $\boldsymbol{w}_m^t$.

III. Problem Formulation

A. MOP Formulation

In the optimization problem, we aim to simultaneously optimize device selection and communication-computation resource allocation to achieve global model training convergence

and reduce the time consumed in FL. MOP is formulated in $\mathcal{P}1$. $\mathcal{C}1$ specifies if device m is selected to participate in the FL training process, $s_m^t = 1$; otherwise, $s_m^t = 0$; $\mathcal{C}2$ defines the bounds of frequency for device m; $\mathcal{C}3$ indicates the lower and upper bounds of p_m^t; $\mathcal{C}4$ ensures that no more than K devices could participate in each round; $\mathcal{C}5$ illustrates that the total bandwidth in the t-th round cannot exceed the maximal bandwidth B_{tot}; $\mathcal{C}6$ indicates the total transmission rate in the uplink should not exceed the uplink capacity R_{cap}; $\mathcal{C}7$ states the computing energy consumption for all the participated devices should be less than a limit value E_0^c; $\mathcal{C}8$ illustrates the final test accuracy of the global model cannot be smaller than ϵ_g; and $\mathcal{C}9$ states the communication-computing energy consumption in each round needs to adhere to a predefined limitation E_0^t.

$$\mathcal{P}_1: \min_{s_m^t, g_m^t, p_m^t} \mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2) = \left(F\left(\boldsymbol{w}^t\right), L^t\right), \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12a}$$

$$\text{s.t.} \quad \mathcal{C}1: s_m^t \in \{0,1\}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12b}$$

$$\mathcal{C}2: g_m^t \in \{g_{m,1}, \cdots, g_{m,\beta_g}\}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12c}$$

$$\mathcal{C}3: p_{min} \le p_m^t \le p_{max}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12d}$$

$$\mathcal{C}4: \sum_{m=1}^{M} s_m^t \le K, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12e}$$

$$\mathcal{C}5: \sum_{m=1}^{M} s_m^t b_m^t \le B_{tot}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12f}$$

$$\mathcal{C}6: \sum_{m=1}^{M} s_m^t r_{m,u}^t \le R_{cal}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12g}$$

$$\mathcal{C}7: \sum_{m=1}^{M} s_m^t Q_m D_m (g_m^t)^2 \le E_0^c, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{12h}$$

$$\mathcal{C}8: Acc_{global} \ge \epsilon_g, \tag{12i}$$

$$\mathcal{C}9: E^t \le E_0^t. \tag{12j}$$

B. Problem Transformation

In $\mathcal{P}1$, calculating and optimizing the value of $F(\boldsymbol{w}^t)$ is a challenging task [18]. Furthermore, establishing the relationship between s_m^t and $\boldsymbol{w}^t$ is complex due to the dynamic changes of $\boldsymbol{w}^t$ during the FL training process. To overcome these challenges, we adopt a combination of age of update and training loss methods to measure global model convergence.

Age of update method utilizes the number of times the model has been trained to describe convergence, represented as $O_m^{t+1} = (O_m^t + 1)(1 - s_m^t)$ [26]. Additionally, the training loss method tracks the loss for all devices, with the training loss for device m in the t-th round denoted as $\mathcal{L}_m^t = F_m(\boldsymbol{w}^t)$. By incorporating these methods, $F(\boldsymbol{w}^t)$ is transformed into a global model convergence reference function, which can be expressed as:

$$C^t = -\frac{\left(\sum_{m=1}^{M} s_m^t D_m \mathcal{L}_m^t O_m^t\right)^{1-\varrho}}{1-\varrho}, \tag{13}$$

where $\varrho \in (0,1)$ is a constant to adjust the sensitivity to the value change of $\sum_{m=1}^{M} s_m^t D_m \mathcal{L}_m^t O_m^t$.

Additionally, in practical scenarios, the transmit power of devices are usually categorized. Therefore, p_m^t can be discretized into an integer variable represented as $p_m^t \in \{p_{m,1}, \cdots, p_{m,\beta_p}\}$, where β_p denotes the number of power levels.

Consequently, $\mathcal{P}_1$ can be transformed as follows:

$$\mathcal{P}_2: \min_{s_m^t, g_m^t, p_m^t} \mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2) = (C^t, L^t), \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{14a}$$

$$\text{s.t.} \quad \mathcal{C}1 - \mathcal{C}2, \tag{14b}$$

$$\mathcal{C}3: p_m^t \in \{p_{m,1}, \cdots, p_{m,\beta_p}\}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \tag{14c}$$

$$\mathcal{C}4 - \mathcal{C}9. \tag{14d}$$

IV. The Proposed Algorithm

A. Motivations and the Core Idea of DMMA-FL

To solve $\mathcal{P}_2$ and obtain an appropriate FL solution, the following issues need to be addressed. Firstly, conventional approaches for handling $\mathcal{P}_2$ typically transform $\mathcal{P}_2$ into SOP by assigning weights to each objective. However, the predefined weights may not be able to adapt to practical problems when the complex environment varies, and then redesigned weights are required, which is inefficient. To this end, $\mathcal{P}_2$ needs to be directly solved to obtain a Pareto optimal set. Most iterative solutions for MOP rely on population-based MOO methods, whose performance heavily depends on the population size. If the population size is large, this can lead to excessive iteration times. Secondly, s_m^t, g_m^t and p_m^t are discrete variables, leading to $\mathcal{P}_2$ classified as a combinatorial optimization problem (COP). Additionally, $\mathcal{P}2$ is also a sequential optimization problem with Markov properties, in which we need to determine the device selection, transmit power, and computing resource allocation in each round. Thus, when the problem scale increases, obtaining optimal or near-optimal solutions can become computationally intensive, which is inefficient and time-consuming. Thirdly, both synchronous and asynchronous schemes need to be assessed for LEO-FL. In the synchronous scheme, the time of one round is significantly constrained by the slowest device, which could lead to the straggler issue. To address this challenge, one approach is to transform the synchronous procedure into an asynchronous one. In asynchronous methods, LEO no longer needs to wait for all devices to finish the local model training before conducting update aggregation. However, the fully asynchronous strategy may result in substantial communication costs due to frequent model parameter exchanges.

To overcome these challenges, we propose a novel decomposition and meta-DRL based MOO algorithm for the LEO-FL system, i.e., DMMA-FL. In Fig. 3, the core idea of DMMA-FL consists of three phases: decomposition, meta-DRL, and FL execution. In the decomposition phase, DMMA-FL divides the MOP into single-objective subproblems using a decomposition method based on weight vectors and aims to optimize these subproblems simultaneously. In the meta-DRL phase, each subproblem is modeled as a Markov decision

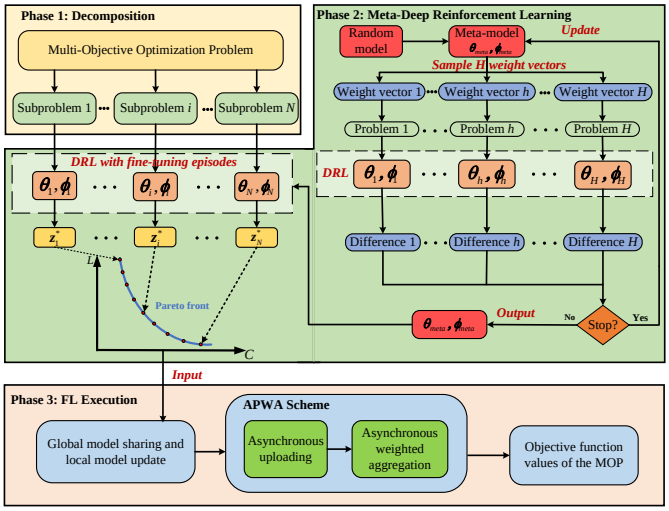


Fig. 3. The three core components of DMMA-FL: decomposition, meta-DRL and FL execution.

process (MDP) and encoded using an LSTM-CNN-based actor-critic framework. DMMA-FL utilizes meta-DRL to learn meta-actor and meta-critic networks by solving subproblems corresponding to all sampled weight vectors. With the well-trained meta-actor and meta-critic networks, we perform small fine-tuning episodes for each subproblem to obtain its Pareto optimal solution. Consequently, by alternating weight vectors, we obtain the Pareto optimal solution set. Based on this set, we proceed to the FL execution phase, where AUWA scheme is introduced to address the straggler issue and reduce communication costs. Finally, DMMA-FL obtains the Pareto optimal solution set and the corresponding objective function values of the MOP (i.e., C and L).

We provide a general framework of DMMA-FL in Algorithm. 1. First, the MOP is divided into single-objective subproblems via *Decomposition Phase* given in IV-B (line 1). For each subproblem, its Pareto optimal solution $\boldsymbol{z}^* = \{s_m^{t*}, g_m^{t*}, p_m^{t*} | \forall m \in \mathcal{M}, t \in \mathcal{T}\}$ is obtained via *Meta-DRL Phase* provided in IV-C (line 3). Then, $\boldsymbol{w}$ is initialized by the LEO (line 4). Then, we perform *FL execution phase* in the main loop (lines 5-17). Within the main loop, for each round t, the LEO sends $\boldsymbol{w}^t$ to the selected devices based on s_m^{t*} and then $L_{m,d}^{t,com}$ is calculated via (6) (line 6). When receiving $\boldsymbol{w}^t$, the selected device m trains and uploads its local model, and calculates $L_m^{t,cmp}$ and $L_{m,u}^{t,com}$ based on $\boldsymbol{z}^*$ via (7) and (5) (lines 9-10). Afterwards, *AUWA Scheme*, introduced by IV-D, is performed to achieve asynchronous uploading and weighted aggregation. Consequently, C^t and L^t are calculated by (13) and (9). Then, the LEO could perform *Collaboration Scheme*, given in II-A, to ensure achieving global model convergence when a large number of devices are involved (line 15). When the main loop ends, C and L are obtained (line 17). Then, DMMA-FL alternates the weight vector to solve the next subproblem. Finally, $\boldsymbol{z}^*$ and its corresponding C and L for each subproblem are output.

Algorithm 1 General framework of DMMA-FL

1: Decompose the MOP into several single-objective subproblems via *Decomposition Phase*;
2: **for** each subproblem corresponding to λ **do**
3: Obtain $\boldsymbol{z}^*$ for λ via *Meta-DRL Phase*;
4: Initialize $\boldsymbol{w}$ for the LEO;
5: **for** $t = 1 : T$ **do**
6: The LEO sends the global model to devices based on $s_m^{t*}|_{m \in \mathcal{M}}$ and calculates $L_{m,d}^{t,com}$ via (6);
7: **for** $m = 1 : M$ **do**
8: **if** $s_m^t = 1$ **then**
9: Device m trains its local model based on $\boldsymbol{w}^t$ and calculates $L_m^{t,cmp}$ based on $\boldsymbol{z}^*$ via (7);
10: Device m uploads its local model and calculates $L_{m,u}^{t,com}$ based on $\boldsymbol{z}^*$ via (5);
11: **end if**
12: **end for**
13: *AUWA Scheme*
14: Calculate C^t and L^t via (13) and (9);
15: *Collaboration Scheme*;
16: **end for**
17: Calculate C and L by $C = \sum_{t=1}^{T} C^t$ and $L = \sum_{t=1}^{T} L^t$;
18: **end for**
Output: $\boldsymbol{z}^*$ and corresponding C and L for each subproblem.

B. Phase 1: Decomposition

DMMA-FL first utilizes the decomposition strategy to divide $\mathcal{P}_2$ into scalar single-objective subproblems based on weight vectors, where each weight vector represents a single-objective subproblem. The detail of generating weight vectors is given as follows. A set of N uniformly distributed weight vectors denoted as $\lambda^i, \forall i \in \{1, \cdots, N\}$ is generated by the simplex method. Here, the number of weight vectors N is expressed as $N = C_{Q+l-1}^{l-1}$, where l is the number of objectives ($l = 2$ in this paper), Q is a pre-defined parameter to control the number of weight vectors, and for each weight vector λ^i, it satisfies the constraint $\sum_{q=1}^{l} \lambda_q^i = 1$. Then, these weight vectors correspond to N single-objective subproblems.

In this paper, a commonly used decomposition approach named the weighted sum approach is considered, where the i-th subproblem is defined as:

$$\min g^{ws}(\boldsymbol{z}|\lambda^i) = \sum_{q=1}^{l} \lambda_q^i \mathcal{F}_q(\boldsymbol{z}) = \lambda_1^i \mathcal{F}_1(\boldsymbol{z}) + \lambda_2^i \mathcal{F}_2(\boldsymbol{z}), \quad (15)$$

where $\boldsymbol{z} = \{s_m^t, g_m^t, p_m^t | \forall m \in \mathcal{M}, t \in \mathcal{T}\}$.

C. Phase 2: Meta-DRL

After performing *Phase 1*, we only need to optimize these decomposed subproblems of (15) simultaneously to obtain the Pareto optimal solution set. Since s_m^t, g_m^t, and p_m^t are discrete variables, each subproblem can be considered a COP, known as NP-hard. Additionally, each subproblem like (15) is a sequential decision-making problem which aims to minimize the objective function in all the FL rounds instead of a single round. To solve this COP with sequential characteristics, current approaches, such as deterministic and heuristic methods,

are inefficient and time-consuming. Employing a heuristic search approach needs to create a population for all rounds of FL, which substantially increases the dimensionality and poses a large-scale optimization problem. DRL is well-suited for sequential problems, as it makes decisions for the current round to minimize a long-term objective function and limits the search space. Recently, DRL has been utilized to deal with COPs with sequential characteristics such as TSP [27] and VRP [28] efficiently. To this end, we covert each subproblem into a MDP and employ DRL to solve it. The key components of the MDP for each subproblem i are defined as follows:

1) State: The state in the t-th step is defined as

$$\mathcal{S}^t = \{b_{leo}^t, b_m^t, h_m^t, O_m^t, \mathcal{L}_m^t, D_m, E^t\}, \forall m \in \mathcal{M}. \quad (16)$$

2) Action: We define s_m^t, g_m^t and p_m^t in the t-th round as the action in the t-th step:

$$\mathcal{A}^t = \{s_m^t, g_m^t, p_m^t\}, \forall m \in \mathcal{M}. \quad (17)$$

3) Reward: The reward in the t-th step is defined as:

$$\mathcal{R}^t = -g^{ws}(\boldsymbol{z}|\lambda^i) = -\lambda_1^i \mathcal{F}_1(\boldsymbol{z}) - \lambda_2^i \mathcal{F}_2(\boldsymbol{z}). \quad (18)$$

To solve the MDP converted by a subproblem, DRL is employed to acquire a policy and, based on the observed states, select a sequence of optimal actions (i.e., s_m^{t*}, g_m^{t*} and p_m^{t*}), which construct the Pareto optimal solution $\boldsymbol{z}^*$ of the subproblem. Among various DRL approaches, we utilize the actor-critic method, which combines the advantages of both value function and policy gradient methods. In the actor-critic method, an actor network with $\pi(\mathcal{A}^t|\mathcal{S}^t)$, generates stochastic policies and selects $\mathcal{A}^t$ based on $\mathcal{S}^t$, and a critic network with $Q(\mathcal{S}^t, \mathcal{A}^t)$, evaluates the performance of $\mathcal{A}^t$. Specifically, the policy gradient of the actor network can be denoted as:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &\approx \mathbb{E}[\nabla_{\boldsymbol{\theta}} Q(s, a|\boldsymbol{\phi})|s = \mathcal{S}^t, a = \pi(\mathcal{S}^t|\boldsymbol{\theta})] \\ &= \mathbb{E}[\nabla a Q(s, a|\boldsymbol{\phi})|s = \mathcal{S}^t, a = \pi(\mathcal{S}^t) \nabla_{\boldsymbol{\theta}} \pi(s|\boldsymbol{\theta})|_{s=\mathcal{S}^t}],\end{aligned} \quad (19)$$

where $\boldsymbol{\theta}$ is the parameters of the actor network. Similarly, the loss function of the critic network can be provided as follows:

$$L(\boldsymbol{\phi}) = \mathbb{E}[\left(\mathcal{R}^t + \gamma Q(\mathcal{S}^{t+1}, \mathcal{A}^{t+1}|\boldsymbol{\phi}) - Q(\mathcal{S}^t, \mathcal{A}^t|\boldsymbol{\phi})\right)^2], \quad (20)$$

where $\boldsymbol{\phi}$ is the parameters of the critic network and γ is a discount factor.

To enhance dynamic adaptation in the changing environment of different subproblems, similar to [29], we introduce a hybrid neural network structure for each subproblem, as depicted in Fig. 4. This structure combines a CNN, an LSTM, and an MLP to replace the original critic network. By learning features from the current environment and historical trajectories, this hybrid network enables effective adaptation to dynamic conditions. Specifically, the CNN is used to evaluate the decisions made by the actor network based on $\mathcal{S}^t$ and $\mathcal{A}^t$. It assesses the quality of the actor's choices by capturing spatial patterns and dependencies in the input data. The LSTM is employed to identify the weight vector based on time-series transitions $\mathcal{B}^{[t-\bar{t},t-1]}$, where $\bar{t}$ denotes the number of time-series transitions. By considering temporal dependencies, the LSTM assists in understanding the evolving dynamics of the environment and helps the critic network accurately assess the actor's performance. Then, the features output from LSTM-CNN are $\zeta_1 = f_{CNN}(\mathcal{S}^t, \mathcal{A}^t)$ and $\zeta_2 = f_{LSTM}(\mathcal{S}^t, \mathcal{A}^t)$, where ζ_1 is the general Q-value, ζ_2 is the weight vector identification embedding, Then, we take ζ_1 and ζ_2 as inputs and pass them through an MLP to obtain the Q-value:

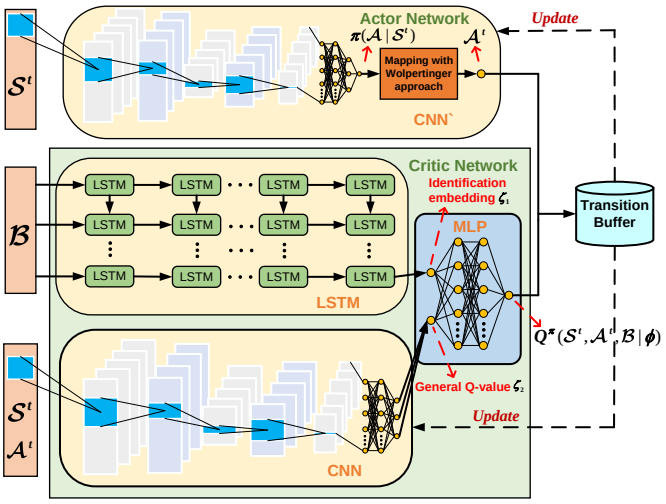


Fig. 4. Illustration of the network structure for each subproblem.

$$Q^\pi(\mathcal{S}^t, \mathcal{A}^t, \mathcal{B}^{[t-\bar{t},t-1]}|\boldsymbol{\phi}) = f_{MLP}(\zeta_1, \zeta_2). \quad (21)$$

Additionally, another CNN is used as the actor network, which generates the mean μ and variance ξ^2 of the stochastic policy based on $\mathcal{S}^t$: $[\mu, \xi^2] = f_{CNN'}(\mathcal{S}^t; \boldsymbol{\theta})$. Then, the stochastic policy is obtained by following Gaussian distribution $\mathcal{N}(\mu, \xi^2)$:

$$\pi(\mathcal{A}^t|\mathcal{S}^t; \boldsymbol{\theta}) \sim \mathcal{N}(\mu, \xi^2). \quad (22)$$

Then, the actor network generates an continuous action $\bar{\mathcal{A}}$ from the stochastic policy following the Gaussian distribution, which is given as:

$$\bar{\mathcal{A}} = f_{\pi(\mathcal{A}|\mathcal{S};\boldsymbol{\theta})}(\mathcal{S}) \quad (23)$$

where $f_{\pi(\mathcal{A}^t|\mathcal{S}^t;\boldsymbol{\theta})}$ is a mapping from the state space to the continuous action space under $\pi(\mathcal{A}^t|\mathcal{S}^t;\boldsymbol{\theta})$. However, since the decision variables in $\mathcal{P}_2$ are discrete, we need to convert $\bar{\mathcal{A}}$ into the discrete action $\mathcal{A}$ from a discrete action space $\mathcal{W}$. There are two mainly used discretization approaches, the simple approach and greedy approach. The simple approach is to choose the nearest integer value to $\bar{\mathcal{A}}$, which is denoted as $\mathcal{A}^* = \arg\min_{\mathcal{A}\in\mathcal{W}} |\mathcal{A} - \bar{\mathcal{A}}|$. It can explore neighboring regions and expand the exploration space. However, this method can lead to a significant likelihood of deviating from the optimum, and may subsequently result in slow convergence. The greedy approach is to optimize Q-value at each step, which given as $\mathcal{A}^* = \arg\min_{\mathcal{A}\in\mathcal{W}} Q(\mathcal{S}, \mathcal{A})$. However, this method potentially leads to getting trapped in local optima and come with higher complexity. Thus, similar to [29], [30], Wolpertinger approach is adopted to balance exploration and exploitation, which is given as

$$\mathcal{A}^* = \arg\min_{\mathcal{A}\in\mathcal{W}} |\mathcal{A} - \bar{\mathcal{A}}| \quad (24)$$

where $\mathcal{W}^*$ is a subset of $\mathcal{W}$ and contains W nearest neighbors of $\mathcal{A}$. Additionally, the solution of simple approach is included in $\mathcal{W}^*$. Wolpertinger approach selects the highest Q-value action from $\mathcal{W}^*$ and it can become the greedy approach and simple approach when $W = |\mathcal{W}^*|$ and $W = 1$, respectively. Note that a larger value of W aids in expanding exploration, whereas a lower value of W promotes exploitation. Thus, an appropriate value for W ($W = 10$ in this paper) is to balance between exploration and exploitation.

The training process of DRL for each subproblem described above requires a significant number of episodes to achieve well-trained actor and critic networks, which can be time-consuming. To tackle this challenge, a neighborhood-based parameter transfer learning strategy (NPTLS) has recently been proposed, where the subproblem corresponding to a weight vector can utilize the transfer knowledge of its close weight vector to accelerate training. NPTLS has been preliminarily applied to the latest research of vehicle networks [31] and control systems [32] to solve related MOPs. However, NPTLS has limited flexibility. It is only suitable for training the actor and critic networks of the subproblems with two closely related weight vectors. Moreover, NPTLS requires saving the parameters of all actor and critic networks, which is inefficient when dealing with a large number of objectives or subproblems.

To overcome these challenges, we introduce meta-DRL to train meta-actor and meta-critic networks with better initial parameters. By leveraging the well-trained meta-actor and meta-critic networks, the actor and critic networks are obtained via small fine-tuning episodes for any newly generated weight vector. Compared with NPTLS, meta-DRL does not need to meet the relationship of two closely weight vectors and only stores the parameters of the meta-actor and meta-critic networks instead of saving the parameters of all actor and critic networks. This approach enhances efficiency and alleviates the burden of parameter storage.

Similar to meta-learning, meta-DRL consists of a learning process and a fine-tuning process. The procedure of meta-DRL is outlined in Algorithm 2. Firstly, the parameters of the meta-actor and meta-critic networks, denoted as $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$, are initialized randomly (line 1). Within the main loop, a set of H weight vectors are sampled from a given distribution Λ, which includes all potential weight vectors (line 3). Afterwards, for each sampled weight vector h, the parameters of the actor and critic networks, denoted as $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$, are set the same as $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$ (line 5). Then, the transition buffer B is initialized as $\emptyset$ (line 6). Next, DRL is employed to train $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$. Specifically, in each episode eps, the initial state $\mathcal{S}^t|_{t=1}$ is randomly generated (line 8). Within each learning step t, $\mathcal{A}^t$ is selected according to $\mathcal{S}^t$ using the actor network via (24) (line 10). Subsequently, $\mathcal{R}^t$ is obtained and $\mathcal{S}^t$ is transformed into a new state $\mathcal{S}^{t+1}$ (line 11). Then, Q-value and the stochastic policy are calculated by (21) and (22), respectively (lines 12-13). Afterward, the transition $\{\mathcal{S}^t, \mathcal{A}^t, \mathcal{R}^t, \mathcal{S}^{t+1}\}$ is saved into B. After storing enough transitions (i.e., $eps > eps_0$), a mini-batch of υ transitions is sampled from B to update $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$ (lines 17-18). The update rules are as follows:

Algorithm 2 Meta-DRL Phase

Learning Process

1: Randomly initialize $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$;
2: **for** $iter = 1 : T_{meta}$ **do**
3: Sample $\mathcal{H} = \{\lambda^1, \cdots, \lambda^H\}$ from Λ;
4: **for** $h = 1 : H$ **do**
5: Set $\boldsymbol{\theta}_h = \boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_h = \boldsymbol{\phi}_{meta}$;
6: Set $B = \emptyset$;
7: **for** eps=1:$Maxeps$ **do**
8: Randomly initialize $\mathcal{S}^t|_{t=1}$;
9: **for** $t = 1 : T$ **do**
10: Select $\mathcal{A}^t$ through (24) based on $\mathcal{S}^t|_{t=1}$;
11: Obtain $\mathcal{R}^t$ and $\mathcal{S}^{t+1}$;
12: Obtain Q-value via $\boldsymbol{\theta}_h$ based on (21);
13: Obtain stochastic policy by $\boldsymbol{\phi}_h$ based on (22);
14: Store $\{\mathcal{S}^t, \mathcal{A}^t, \mathcal{R}^t, \mathcal{S}^{t+1}\}$ into B;
15: **end for**
16: **while** $eps > eps_0$ **do**
17: Sample a mini-batch of υ transitions from B;
18: Update $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$ based on (25) and (26);
19: **end while**
20: **end for**
21: Obtain the well-trained $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$;
22: **end for**
23: Update $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$ based on (27) and (28);
24: **end for**
25: Obtain the well-trained $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$.

Fine-tuning Process

26: Generate N weight vectors via *Decomposition Phase*;
27: **for** $j = 1 : N$ **do**
28: Set $\boldsymbol{\theta}_j = \boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_j = \boldsymbol{\phi}_{meta}$ for λ^j;
29: Perform T_{ft} episodes and obtain the well-trained $\boldsymbol{\theta}_j$ and $\boldsymbol{\phi}_j$;
30: Obtain $\boldsymbol{z}_j^*$ of the j-th subproblem;
31: **end for**

Output: $PF = \{\boldsymbol{z}_1^*, \cdots, \boldsymbol{z}_N^*\}$

$$\boldsymbol{\theta}_h = \boldsymbol{\theta}_h - \rho \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_h), \tag{25}$$

$$\boldsymbol{\phi}_h = \boldsymbol{\phi}_h - \rho \nabla_{\boldsymbol{\phi}} L(\boldsymbol{\phi}_h), \tag{26}$$

where ρ is the learning rate of the actor and critic networks.

The training process of DRL iterates until the termination condition is met (i.e., $eps > Maxeps$). Subsequently, the well-trained $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$ for λ^h are obtained (line 21). Next, the differences between $\boldsymbol{\theta}_h$ and $\boldsymbol{\theta}_{meta}$, and between $\boldsymbol{\phi}_h$ and $\boldsymbol{\phi}_{meta}$ are calculated. Then, $\boldsymbol{\theta}_h$ and $\boldsymbol{\phi}_h$ are updated as follows (line 23):

$$\boldsymbol{\theta}_{meta} = \boldsymbol{\theta}_{meta} + \frac{\eta}{h} \sum_{h=1}^{H} (\boldsymbol{\theta}_h - \boldsymbol{\theta}_{meta}), \tag{27}$$

$$\boldsymbol{\phi}_{meta} = \boldsymbol{\phi}_{meta} + \frac{\eta}{h} \sum_{h=1}^{H} (\boldsymbol{\phi}_h - \boldsymbol{\phi}_{meta}), \tag{28}$$

where η represents the learning rate of meta-learning. The main loop continues to run until $iter > T_{meta}$, where T_{meta}

is the maximal number of iterations. Finally, the well-trained $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$ are obtained (line 25).

Based on the well-trained $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$, we perform the fine-tuning process to obtain the Pareto optimal solution set. Specifically, N weight vectors are generated through *Decomposition Phase* (line 26). Note that N is personally determined based on the specific situation or practical problem. Then, for each λ^j, the parameters of its actor and critic networks, denoted as $\boldsymbol{\theta}_j$ and $\boldsymbol{\phi}_j$ are set the same as $\boldsymbol{\theta}_{meta}$ and $\boldsymbol{\phi}_{meta}$ (line 28). Afterwards, we execute a few fine-tuning episodes, denoted as T_{ft} (i.e., $T_{ft} \ll T_{meta}$), and then obtain the well-trained $\boldsymbol{\theta}_j$ and $\boldsymbol{\phi}_j$ (line 29). According to the well-trained $\boldsymbol{\theta}_j$ and $\boldsymbol{\phi}_j$, the Pareto optimal solution $\boldsymbol{z}_j^* = \{s_{m,j}^{t*}, f_{m,j}^{t*}, p_{m,j}^{t*} | \forall m \in \mathcal{M}, t \in \mathcal{T}\}$ of the j-th subproblem is obtained (line 30). Subsequently, the Pareto optimal solution set $PF = \{\boldsymbol{z}_1^*, \cdots, \boldsymbol{z}_N^*\}$ is established by storing N Pareto optimal solutions corresponding to N subproblems by alternating the weight vectors. Finally, PF is output.

D. Phase 3: FL Execution

Based on $\boldsymbol{z}^*$ for each subproblem, we run the FL execution phase and obtain the objective function value for each subproblem. During the FL execution phase, due to the LEO's high-speed movement and potential resource limitations of the devices, devices may be located outside the service area of the LEO. This can cause devices to prevent them from participating in the FL training process and causing the straggler issue in synchronous methods (i.e., FedAvg [3]). The fully asynchronous method (i.e., FedAsyn [13]) can solve the straggler issue, but it requires frequent model interactions, resulting in high communication costs. Recently, an asynchronous strategy based on periodic aggregation (short for FedAsynPA [14]) has been proposed to address the straggler issue while reducing frequent model interactions. However, this approach suffers from inefficiency, as devices that upload their local models earlier have to wait for the entire fixed period to finish before receiving the updated global model from the LEO. Consequently, device resources remain idle for an extended period, leading to decreased system efficiency. To overcome these challenges, we propose AUWA scheme, which aims to mitigate the straggler issue and reduce communication costs.

AUWA scheme consists of two asynchronous schemes: asynchronous uploading and asynchronous weighted aggregation. In the asynchronous uploading scheme, the selected devices update and upload their local models asynchronously, allowing them to enter and complete their training independently. In the asynchronous weighted aggregation scheme, the LEO does not immediately perform global model aggregation upon receiving each local model. Instead, the uploaded local models are stored in a buffer. Similar to FedBuffer in [15], the buffer can be implemented through a cryptographic algorithm. When the buffer contains a certain number of the uploaded local models, the LEO perform global model aggregation. Different from FedBuffer, the weight assigned to each local model during the global model aggregation in AUWA scheme is based on the performance of the local model, rather than solely relying on the traditional data size.

In detail, during each asynchronous round, each selected device m uploads its local model to the LEO once it updates its local model. Then, the LEO utilizes a buffer with a total size of $Buffer_0$ to store the local models, incrementing the buffer size by 1. When the buffer becomes full (i.e., $buffer = Buffer_0$), the LEO performs global model aggregation based on the stored local models. In this paper, a metric u_m^t is introduced to guide the LEO in assigning appropriate weights to the more valuable local models during the global model aggregation, which can be defined as:

$$u_m^t = \frac{\mathbb{E}\{F_m(\boldsymbol{w}_m^0) - F_m(\boldsymbol{w}_m^t)\}}{\sum_{m \in \mathcal{M}'} \mathbb{E}\{F_m(\boldsymbol{w}_m^0) - F_m(\boldsymbol{w}_m^t)\}}, \tag{29}$$

where $\mathbb{E}\{F_m(\boldsymbol{w}_m^0) - F_m(\boldsymbol{w}_m^t)\}$ represents the expected performance of the local model from the selected device m in the t-th round. From (29), it can be observed that devices with better-performing uploaded local models are assigned higher weights in the global model aggregation process.

V. Experimental Studies

A. Global Model Structures and Datasets

In this paper, we utilized three different networks as the FL global models: an MLP and two CNNs named CNN1 and CNN2, whose network structures are the same as [18]. To evaluate the performance of DMMA-FL using these global models, we utilized two datasets named MNIST and CIFAR-10. MNIST dataset [33] comprises 60,000 training samples and 10,000 test samples, consisting of ten classes of handwritten digits. CIFAR-10 dataset [34] contains 60,000 color images with dimensions of 32×32 pixels and is categorized into ten different classes. For the training phase, the data samples were distributed randomly among multiple devices using both the IID and Non-IID settings. In the IID data setting, each device received a local training dataset that contained a uniform sampling of all ten classes. In the Non-IID setting, each device obtained a subset of samples that included different classes, resulting in variations among the devices' datasets.

Overall, we trained six instances, including MLP and CNN1 on MNIST dataset in the IID and Non-IID settings, and CNN2 on CIFAR-10 dataset in the IID and Non-IID settings.

B. Experimental Setup

The parameters for the studied LEO-FL system are configured as follows [16]–[18], [35]. The LEO operates at an altitude of 780 km, covering a service area of 2800 km^2 [36]. Devices are randomly distributed within a rectangular area within the service area of the LEO. The transmission time is limited to a range of 5 to 15 time slots, where each time slot lasts for 0.1 seconds. The bandwidth of the LEO is 400 MHz. The carrier frequency of uplink and downlink is 30 GHz [16]. The noise power spectral density is specified as -174 dBm/Hz. Furthermore, the total dataset size is 47.04 MB. The remaining parameters of the studied system are summarized in Table II. All experiments were conducted using PyTorch 1.15 (Python3.8) and tested on a high-performance workstation equipped with a RTX4060 GPU.

To verify the effectiveness of DMMA-FL, we employed three categories of comparison methods: single-objective,

TABLE II
PARAMETER SETTINGS OF THE STUDIED SYSTEM

Parameter	Value	Parameter	Value	Parameter	Value
T	200	M	100	τ_l	0.01
τ_g	0.01	$\alpha_m, m \in \mathcal{M}$	2e-27	N	30
p_{leo}	100W	$c_m, m \in \mathcal{M}$	20	X	86.6 KB
G_T	60	G_R	20	κ	20
g_1	0	g_{β_g}	10GHz	β_f	40
p_1	0	p_{β_p}	100W	β_p	20
K	30	ϵ_g on MNIST	0.85	R_{cal}	500Kbps
T_{ft}	50	ϵ_g on CIFAR-10	0.6	B_{tot}	100MHz
$Maxeps$	500	T_{meta}	20	eps_0	60
H	50	ϱ	0.01	ρ	0.001
η	0.03	$Buffer_0$	6	υ	256

heuristics-based MOO, and learning-based MOO methods, which are listed as follows:

1) Single-Objective Optimization Methods:

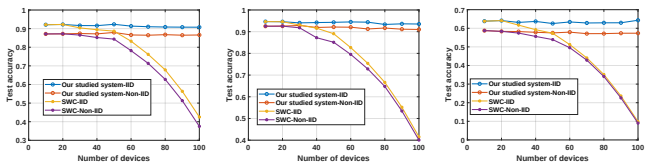
- FedCS: A study considers the impact of device selection to minimize the model training convergence [37], with the objective function as $F = C^t$ $(\forall t \in \mathcal{T})$.
- Sliding DE: A novel scheduling policy based on sliding differential evolution [18] to optimize an SOP denoted as $F = C^t + \epsilon L^t$ $(\forall t \in \mathcal{T})$, where $\epsilon = 0.9$.
- SDEFL: A novel service time efficient FL scheme to minimize the communication-computing time [12] with the objective function as $F = L^t$ $(\forall t \in \mathcal{T})$.

2) Heuristics-Based MOO Methods:

- MOPSO-FL: A multi-objective particle swarm optimization (PSO) approach for FL systems [38].
- NSGA-FL: A fast and elitist multi-objective genetic algorithm applied for FL systems [39].
- MOEA-FL: A multi-objective approach based on decomposition strategy for FL systems [40].

3) Learning-Based MOO Methods:

- ACT-FL: A multi-objective approach based on DRL, where NPTLS and the actor-critic algorithm are proposed to train each subproblem built by a point network [27].
- LRA-FL: A multi-objective approach via resource allocation algorithm for over-the-air FL systems, where it utilizes the decomposition scheme and NPTLS [41].
- DQN-FL: A multi-objective approach based on decomposition scheme and deep Q network (DQN) to train each subproblem [42].



(a) (b) (c)

Fig. 5. Test accuracy obtained by two scenarios. (a) MLP on MNIST dataset. (b) CNN1 on MNIST dataset. (c) CNN2 on CIFAR-10 dataset.

C. Effectiveness of Collaboration Scheme

To demonstrate the effectiveness of the collaboration scheme, we compared the studied system to the system without the collaboration scheme (SWC), referring to the scenarios in Fig. 2. Note that we chose a range of 5 to 100 devices because this range can encompass the majority of satellite communication and other FL scenarios [43]–[47]. For instance, the range of 0 to 30 devices corresponds to sparsely populated areas, such as high mountains or deserts. The range of 30 to 70 devices is suitable for scenes with moderate population density, like town convenience stores and gas stations. Lastly, the range of 70 to 100 devices is intended for densely populated scenarios, such as schools and shopping malls. The experimental results are depicted in Fig. 5. We can observe that when the number of devices is small, there is no significant difference between our studied system and SWC. However, as the number of devices increases, our studied system enables the continuous training of an ideally accurate model on the six instances, benefiting from the ability to transmit the global model to the next LEO. On the other hand, as the number of devices increases, a single LEO cannot complete the FL training process in SWC. Consequently, the next LEO has to start training the global model from scratch, which negatively impacts test accuracy. When the number of devices exceeds a large threshold, the global model of SWC may fail to converge without collaboration. To this end, the collaboration scheme in the studied system is effective.

D. Effectiveness of Multi-Objective Optimization

To illustrate the effectiveness of DMMA-FL, we conducted comparisons with three single-objective optimization methods and six MOO methods. For single-objective methods, since they only generate a single optimal value, we compared single-objective methods with DMMA-FL by the obtained optimal solution(s). As for MOO methods, since the true Pareto optimal solution set is unknown, we employed commonly used metrics named hypervolume (HV) [48] to comprehensively evaluate the diversity and convergence of the generated approximate Pareto optimal solution set. Given a point set $A \subset \mathbb{R}^l$ and a reference point $\boldsymbol{r} \subset \mathbb{R}^l$, where $l = 2$ is the number of optimization objectives. The HV of the set A is calculated as follows:

$$HV(A, \boldsymbol{r}) = \mathcal{L}_e \left(\bigcup_{\boldsymbol{a} \in A} \{\boldsymbol{b} | \boldsymbol{a} \prec \boldsymbol{b} \prec \boldsymbol{r}\} \right) \tag{30}$$

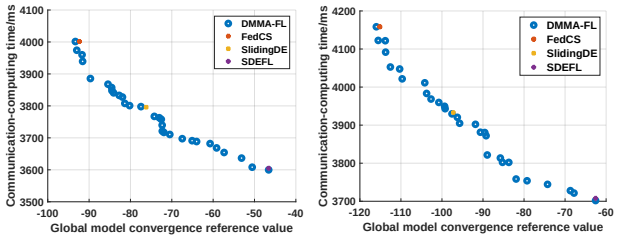
where $\mathcal{L}_e(\cdot)$ denotes the Lebesgue measure of a set: $\mathcal{L}_e(\mathcal{Z}) = \int_{\boldsymbol{z} \in \mathcal{Z}} \mathbf{1}_{\mathcal{Z}}(\boldsymbol{z}) d\boldsymbol{z}$, where $\mathbf{1}_{\mathcal{Z}}$ is the characteristic function of objective space $\mathcal{Z}$. If $\boldsymbol{z} \in \mathcal{Z}$, $\mathbf{1}_{\mathcal{Z}}(\boldsymbol{z}) = 1$; otherwise, $\mathbf{1}_{\mathcal{Z}}(\boldsymbol{z}) = 0$. In the calculation process of the HV, the non-dominated solutions obtained by each algorithm are normalized using the same reference set, and the reference point is commonly fixed at (1, 1). Note that a larger HV illustrates a better approximate Pareto optimal solution set and a better performance of the corresponding MOO method.

We compared DMMA-FL with the three single-objective methods by the obtained Pareto optimal solutions. The experimental results are illustrated in Fig. 6, Fig. 7 and Fig. 8. From them, we can observe that DMMA-FL is able to generate an approximate Pareto optimal solution set in all six instances, which includes approximate Pareto optimal solutions

associated with different weight vectors. In contrast, FedCS, SlidingDE, and SDEFL, being single-objective methods, only have a single optimal solution. In fact, the objective function of FedCS, SlidngDE, and SDEFL represents a single-objective subproblem corresponding to a potential weight vector of the MOP. The optimal solution obtained by these methods represents only a portion of the approximate Pareto optimal solution set generated by DMMA-FL. The inferior performance of FedCS, SlidingDE, and SDEFL compared to DMMA-FL in terms of approximate Pareto optimal solutions can be attributed to their optimization methods, which may become trapped in local optima, resulting in suboptimal solutions.

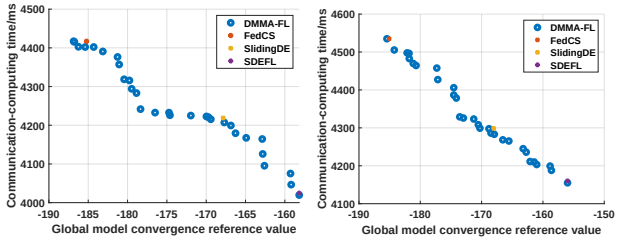
We independently ran all the MOO methods 20 times on each instance and recorded averaged HV and running time. The experimental results are presented in Table.III. It is evident from the table that our method (DMMA-FL) achieves a larger HV across all six instances, which means DMMA-FL can obtain a better approximate Pareto optimal solution set and a better performance compared with other algorithms. ACT-FL follows as the second-best performing method, utilizing parameter transfer learning and operating with only two closely weighted vectors, enabling efficient utilization of local information. In contrast, DMMA-FL leverages meta-actor and meta-critic networks based on global information from all potential weight vectors. LRA-FL performs slightly worse than ACT-FL, primarily because LRA-FL utilizes a DNN framework, whereas ACT-FL employs a point network (i.e., seq2seq), which is better suited for training with DRL. DQN-FL uses DQN based on value function, and its performance is not as good as actor-critic methods like DMMA-FL and LRA-FL. Heuristics-based methods exhibit lower HV compared to learning-based methods, as they lack the ability to store knowledge information and do not benefit from previous knowledge to guide optimization. Among the heuristics-based methods, MOEA-FL is better than NSGA-FL because NSGA-FL may retain the non-dominant solutions that need to be discarded, which has an influence on the search process. MOPSO-FL obtained the worst performance in HV among all the MOO methods. Due to the limitations of PSO, it is easy to get trapped in local optima.

In Table.III, we can observe that DMMA-FL has the shortest running time compared to the other six methods. This can be attributed to the meta-actor and meta-critic networks with better initial parameters. With just a few episodes, DMMA-FL is able to obtain well-trained actor and critic networks that output the Pareto optimal solution set. As for ACT-FL, it employs BPTFS to train the actor and critic networks of the neighboring subproblem with the initial parameters transferred from the actor and critic networks of the previous subproblem. However, the actor and critic networks of the first subproblem are trained from scratch, resulting in higher computing costs. Similar to HV, the network structure affects the running time of LRA-FL. The running time of DQN-FL is behind DMMA-FL, ACT-FL and LRA-FL, because actor-critic methods are faster than DQN, which could use policy gradient to accelerate training. Heuristics-based approaches exhibit slower performance in the running time compared to learning-based methods, as they rely on population-based iterative procedures that often require more than 10,000 iterations to find optimal solutions. NSGA-FL outperforms MOEA-FL in the running time, primarily because MOEA-FL employs a decomposition strategy that may consume additional running time. Among all the MOO methods, MOPSO-FL consumes the most running time.



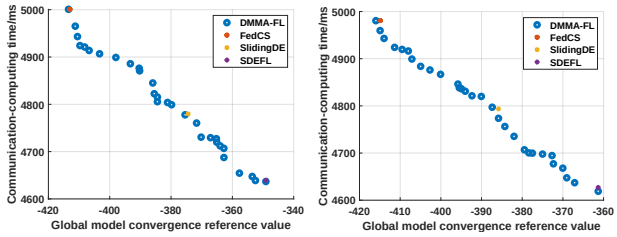
(a) IID (b) Non-IID

Fig. 6. The approximate Pareto optimal solution(s) of DMMA-FL and the three single-objective methods using MLP on MNIST dataset.



(a) IID (b) Non-IID

Fig. 7. The approximate Pareto optimal solution(s) of DMMA-FL and the three single-objective methods using CNN1 on MNIST dataset.



(a) IID (b) Non-IID

Fig. 8. The approximate Pareto optimal solution(s) of DMMA-FL and the three single-objective methods using CNN2 on CIFAR-10 dataset.

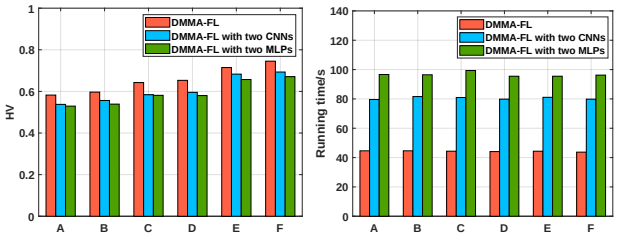
E. Effectiveness of Meta-DRL

1) Effectiveness of Network Structure: To prove the effectiveness of the designed network structure (i.e., LSTM-CNN for critic and one CNN for actor) of Meta-DRL in DMMA-FL, we compared it with two CNNs (one CNN for critic and another CNN for actor) and two MLPs (one MLP for critic and another MLP for actor). Specifically, we conducted experiments on the subproblem corresponding to a randomly sampled weight vector across the six instances. The experimental results are presented in Fig. 9. From Fig. 9(a), we observed that DMMA-FL reaches the larger HV compared to its competitors, followed by DMMA-FL employing two CNNs. This is because the LSTM component

TABLE III
HV AND RUNNING TIME OF DMMA-FL AND THE OTHER SIX MULTI-OBJECTIVE METHODS ON THE SIX INSTANCES.

Instance	DMMA-FL		ACT-FL		LRA-FL		DQN-FL		MOEA-FL		NSGA-FL		MOPSO-FL	
	HV	Running time/s	HV	Running time/s	HV	Running time/s	HV	Running time/s	HV	Running time/s	HV	Running time/s	HV	Running time/s
MLP-MNIST-IID	**0.5824**	**44.62**	0.5610	70.56	0.5501	77.07	0.5329	95.86	0.5150	170.34	0.4599	104.28	0.4356	200.24
MLP-MNIST-Non-IID	**0.5966**	**44.62**	0.5707	71.78	0.5658	79.48	0.5402	90.32	0.5350	167.45	0.4699	105.37	0.4407	203.09
CNN1-MNIST-IID	**0.6426**	**44.36**	0.6096	67.05	0.5940	78.66	0.5832	93.49	0.5640	170.02	0.5025	100.47	0.4708	202.44
CNN1-MNIST-Non-IID	**0.6532**	**44.09**	0.6144	70.17	0.606	75.83	0.5861	97.04	0.5675	171.37	0.5054	99.43	0.4783	204.33
CNN2-CIFAR-10-IID	**0.7147**	**44.35**	0.6410	71.02	0.6315	72.90	0.5914	93.26	0.5798	171.92	0.4990	101.22	0.4875	204.20
CNN2-CIFAR-10-Non-IID	**0.7453**	**43.71**	0.6579	65.68	0.6393	75.32	0.5961	91.89	0.5893	169.08	0.5138	101.44	0.4923	198.64

effectively extracts features from time-related sequential data, while the CNN component efficiently extracts spatial features from the input data. DMMA-FL with two MLPs obtains the lowest HV, indicating poorer performance. In terms of running time, DMMA-FL spends the shortest running time, followed by DMMA-FL with two CNNs, while the performance of DMMA-FL with two MLPs is the worst for running time. To this end, the effectiveness of LSTM-CNN is proven.



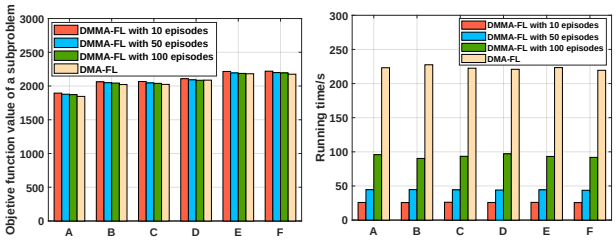
(a) HV (b) Running time

Fig. 9. Effectiveness of the LSTM-CNN network structure. "A" and "B" represent MLP on MNIST dataset in the IID and Non-IID settings, "C" and "D" represent CNN1 on MNIST dataset in the IID and Non-IID settings, "E" and "F" represent CNN2 on CIFAR-10 dataset in the IID and Non-IID settings.

2) Effectiveness of Meta-Learning: To evaluate the effectiveness of meta-learning, we compared DMMA-FL with that without meta-learning, referred to as DMA-FL. Furthermore, we investigated the impact of different fine-tuning episodes (i.e., T_{tf}) by setting them to 10, 50, and 100 for DMMA-FL. Then, we tested the subproblem corresponding to a randomly generated weight vector. The results of these experiments are presented in Fig. 10. From Fig. 10(a), we can observe that DMMA-FL with different fine-tuning episodes achieves similar objective function values as DMA-FL. However, as shown in Fig. 10(b), DMMA-FL with different episodes outperforms DMA-FL regarding running time. These results provide empirical evidence for the effectiveness of meta-learning. Additionally, based on a trade-off between the objective function value and running time, we set the fine-tuning episode as 50 in this paper.

F. Effectiveness of AUWA Scheme

To illustrate the effectiveness of AUWA scheme, we compared it with FedAvg, FedAsyn, FedAsynPA and FedBuffer. The experiment was performed using CNN2 on the CIFAR-10 dataset, as it represents the most complex task among the six instances, and its results are considered representative. The experimental result is shown in Fig. 11. We observed that AUWA scheme outperforms the other four methods in the test accuracy and convergence at a certain time. Specifically,



(a) Objective function value (b) Running time

Fig. 10. Effectiveness of meta-learning. "A" and "B" represent MLP on MNIST dataset in the IID and Non-IID settings, "C" and "D" represent CNN1 on MNIST dataset in the IID and Non-IID settings, "E" and "F" represent CNN2 on CIFAR-10 dataset in the IID and Non-IID settings.

AUWA scheme achieves the fastest convergence and obtains the higher test accuracy, followed by FedBuffer. FedAsynPA achieves a similar test accuracy to FedBuffer. Although the initial test accuracy of FedAsynPA is low, the frequent period aggregations within a short time duration allow for rapid improvement in test accuracy. As a result, the test accuracy can increase rapidly. FedAsyn requires frequent interactions with the devices for parameter exchange, resulting in higher time costs. Therefore, the test accuracy is not as high during certain periods. FedAvg performs the worst because it waits for the slowest device in each round before performing model aggregation.

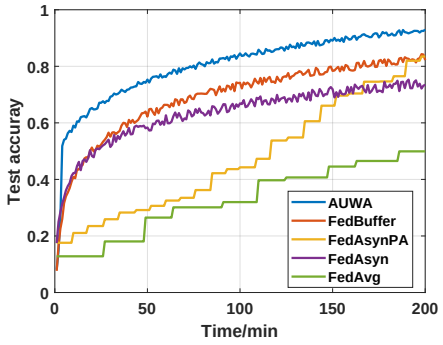


Fig. 11. Test accuracy obtained by the five aggregation strategies using CNN2 on CIFAR-10 dataset in the Non-IID setting

G. Analysis of Overhead

Overhead is an important factor of FL systems, which commonly associated with communication and computation time cost [49], [50]. However, communication and computation time overhead in this paper is one of the objectives of $\mathcal{P}2$, which is influenced by another objective (i.e., the global model convergence reference value C). To this end,

we adopt a simple approach to measure the overhead: maintaining C at approximately the same value and comparing the communication-computing time overhead with MOO methods. In MOO methods, the experiments are conducted using CNN2 on the CIFAR-10 dataset and the reason is same as Fig. 11. The experimental results are presented in Table IV. From it, we can observe that DMML-FL exhibits the lowest overhead when achieving approximately the same global model convergence reference values, followed by ACT-FL, and MOPSO-FL consume most overhead among the MOO methods.

TABLE IV
COMMUNICATION-COMPUTING TIME OVERHEAD OF DMMA-FL AND THE OTHER SIX MULTI-OBJECTIVE METHODS FOR CNN2 ON THE CIFAR-10 DATASET

MOO Methods	Overhead/ms	Global model convergence reference value
DMML-FL	4870.75	-390.25
ACT-FL	4956.21	-391.55
LRA-FL	4968.15	-386.71
DQN-FL	5042.21	-392.06
MOEA-FL	5058.57	-385.32
NSGA-FL	5343.00	-387.77
MOPSO-FL	5477.29	-385.46

VI. CONCLUSION

We studied LEO and FL in future 6G-satellite systems. In the studied LEO-FL system, we investigated several encounter issues in multi-metric performance optimization. These challenges include heterogeneous on-device capabilities at the network edge, the limited coverage time of a satellite, the straggler issue arising from the high mobility of LEO satellites, and the need to handle asynchronous uploading and aggregation from a massive number of devices. To achieve an optimal tradeoff among these challenges, we adopted a novel approach different from most previous FL works. Instead of focusing on individual optimization objectives, we adopted a MOO perspective that simultaneously improves communication-training efficiency and local training accuracy. To address these challenges, we proposed a decomposition and meta-DRL based MOO algorithm for FL called DMMA-FL to enable dynamic adaptation, efficient uploading and aggregation, and Pareto optimal solution sets approaching. Compared to single-objective optimization, heuristics-based and learning-based MOO methods, the effectiveness of the proposed LEO-FL system and DMMA-FL algorithm is illustrated using different FL global models on MNIST and CIFAR-10 datasets in the IID and Non-IID settings.

In the future, we plan to develop the proposed algorithm from the following aspects. Firstly, the combination of advanced techniques, such as continuous learning and behavior regularization, can further improve the sample efficiency and model adaptability of actor and critic networks. Secondly, introducing attention mechanisms to our proposed algorithm enables DRL to better handle long sequences and capture relationships within the sequences. Thirdly, the parameters of our proposed method are set to the same values as references. It is advisable to conduct additional attempts to find the optimal parameters and enhance the performance. Finally, it is expected that researchers will be motivated by the proposed algorithm to devise more advanced methods for FL in 6G-satellite systems.

REFERENCES

[1] J. A. Guerrero-ibanez, S. Zeadally, and J. Contreras-Castillo, "Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and internet of things technologies," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 122–128, 2015.

[2] M. Chen, N. Shlezinger, H. V. Poor, Y. C. Eldar, and S. Cui, "Communication-efficient federated learning," *Proceedings of the National Academy of Sciences*, vol. 118, no. 17, p. e2024789118, 2021.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282.

[4] L. U. Khan, S. R. Pandey, N. H. Tran, W. Saad, Z. Han, M. N. H. Nguyen, and C. S. Hong, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, 2020.

[5] W. Y. B. Lim, J. S. Ng, Z. Xiong, D. Niyato, C. Miao, and D. I. Kim, "Dynamic edge association and resource allocation in self-organizing hierarchical federated learning networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3640–3653, 2021.

[6] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[7] H. Al-Hraishawi, H. Chougrani, S. Kisseleff, E. Lagunas, and S. Chatzinotas, "A survey on nongeostationary satellite systems: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 101–132, 2023.

[8] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "Ground-assisted federated learning in LEO satellite constellations," *IEEE Wireless Communications Letters*, vol. 11, no. 4, pp. 717–721, 2022.

[9] H. Chen, M. Xiao, and Z. Pang, "Satellite-based computing networks with federated learning," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 78–84, 2022.

[10] B. Matthiesen, N. Razmi, I. Leyva-Mayorga, A. Dekorsy, and P. Popovski, "Federated learning in satellite constellations," *IEEE Network*, pp. 1–16, 2023.

[11] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, 2021.

[12] R. Chen, D. Shi, X. Qin, D. Liu, M. Pan, and S. Cui, "Service delay minimization for federated learning over mobile devices," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 990–1006, 2023.

[13] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with Non-IID data," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 15–24.

[14] C.-H. Hu, Z. Chen, and E. G. Larsson, "Scheduling and aggregation design for asynchronous federated learning over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 874–886, 2023.

[15] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba, "Federated learning with buffered asynchronous aggregation," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, vol. 151. PMLR, 28–30 Mar 2022, pp. 3581–3607.

[16] Y. Yuan, L. Lei, T. X. Vu, S. Fowler, and S. Chatzinotas, "Efficient resource scheduling and optimization for over-loaded LEO-terrestrial networks," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 1052–1057.

[17] 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 *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2019, pp. 1–6.

[18] Y. Luo, J. Xu, W. Xu, and K. Wang, "Sliding differential evolution scheduling for federated learning in bandwidth-limited networks," *IEEE Communications Letters*, vol. 25, no. 2, pp. 503–507, 2021.

[19] B. Shen, Y. Wu, J. An, C. Xing, L. Zhao, and W. Zhang, "Random access with massive MIMO-OTFS in LEO satellite communications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 10, pp. 2865–2881, 2022.

[20] W. U. Khan, Z. Ali, E. Lagunas, A. Mahmood, M. Asif, A. Ihsan, S. Chatzinotas, B. Ottersten, and O. A. Dobre, "Rate splitting multiple access for next generation cognitive radio enabled LEO satellite networks," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[21] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[22] V.-D. Nguyen, S. K. Sharma, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Efficient federated learning algorithm for resource allocation in wireless IoT networks," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3394–3409, 2021.

[23] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2021.

[24] B. Li, A. Swami, and S. Segarra, "Power allocation for wireless federated learning using graph neural networks," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 5243–5247.

[25] J. Liu, Z. Chang, G. Min, and Z. Han, "Incentive mechanism design for federated learning in multi-access edge computing," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, 2022, pp. 3454–3459.

[26] M. Ma and V. W. Wong, "Age of information driven cache content update scheduling for dynamic contents in heterogeneous networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 8427–8441, 2020.

[27] K. Li, T. Zhang, and R. Wang, "Deep reinforcement learning for multiobjective optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3103–3114, 2021.

[28] Z. Zhang, Z. Wu, H. Zhang, and J. Wang, "Meta-learning-based deep reinforcement learning for multiobjective optimization problems," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022.

[29] Y. Yuan, L. Lei, T. X. Vu, Z. Chang, S. Chatzinotas, and S. Sun, "Adapting to dynamic LEO-B5G systems: Meta-critic learning based efficient resource scheduling," *IEEE Transactions on Wireless Communications*, vol. 21, no. 11, pp. 9582–9595, 2022.

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

[31] Y. Xu, K. Zhu, H. Xu, and J. Ji, "Deep reinforcement learning for multi-objective resource allocation in multi-platoon cooperative vehicular networks," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[32] S. Luo, L. Zhang, and Y. Fan, "Real-time scheduling for dynamic partial-no-wait multiobjective flexible job shop by deep reinforcement learning," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3020–3038, 2022.

[33] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

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

[36] S. Xia, Q. Jiang, C. Zou, and G. Li, "Beam coverage comparison of LEO satellite systems based on user diversification," *IEEE Access*, vol. 7, pp. 181 656–181 667, 2019.

[37] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[38] H.-G. Han, Z. Liu, W. Lu, Y. Hou, and J.-F. Qiao, "Dynamic mopso-based optimal control for wastewater treatment process," *IEEE Transactions on Cybernetics*, vol. 51, no. 5, pp. 2518–2528, 2019.

[39] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 4, pp. 1310–1322, 2020.

[40] Y. Zhou, X. Liu, and L. Lei, "Multi-objective optimization for bandwidth-limited federated learning in wireless edge systems," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 954–966, 2023.

[41] X. Tu and K. Zhu, "Learning-based multi-objective resource allocation for over-the-air federated learning," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, 2022, pp. 3065–3070.

[42] T. Xu, Y. Liu, Z. Ma, Y. Huang, and P. Liu, "A DQN-based multi-objective participant selection for efficient federated learning," *Future Internet*, vol. 15, no. 6, p. 209, 2023.

[43] Y. Jing, C. Jiang, N. Ge, and L. Kuang, "Resource optimization for signal recognition in satellite MEC with federated learning," in *2021 13th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2021, pp. 1–5.

[44] W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, and X. Shen, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, 2021.

[45] C. Feng, Z. Zhao, Y. Wang, T. Q. S. Quek, and M. Peng, "On the design of federated learning in the mobile edge computing systems," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5902–5916, 2021.

[46] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2020.

[47] 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, 2022.

[48] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 03 2011.

[49] S. Liu, J. Yu, X. Deng, and S. Wan, "FedCPF: An efficient-communication federated learning approach for vehicular edge computing in 6G communication networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1616–1629, 2022.

[50] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Transactions on Computers*, vol. 70, no. 5, pp. 655–668, 2021.

Yu Zhou received the B.S. degree in automation from Southwest Petroleum University, Chengdu, China, in 2020, and the M.S. degree in control science and engineering from Central South University, Changsha, China, in 2023. He worked as a visiting student with the School of Information and Communications Engineering, Xi'an Jiaotong University, Xi'an, China, in 2023. He is currently pursuing the Ph.D. degree with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR, China. His current research interests include federated learning, evolutionary computation and deep reinforcement learning.

Lei Lei (S'12-M'17) received the B.Eng. and M.Eng. degrees from Northwestern Polytechnic University, Xi'an, China. He obtained the Ph.D. degree in 2016 at the Department of Science and Technology, Linköping University, Sweden. He was a research assistant at Institute for Infocomm Research (I2R), A*STAR, Singapore, in 2013. He was with the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg as a research associate and research scientist from 2016 to 2021. He is currently an Associate Professor with Xi'an Jiaotong University, School of Information and Communications Engineering. His current research interests include resource allocation and optimization in terrestrial-satellite networks, energy-efficient communications, and deep learning in wireless communications. He co-authored the IEEE ComSoc Best Readings in Satellite Mega Constellations and Non-Orthogonal Multiple Access. He received the IEEE Sweden Vehicular Technology-Communications-Information Theory (VT-COM-IT) joint chapter best student journal paper award in 2014. He was a co-recipient of the IEEE SigTelCom 2019 Best Paper Award.

Xiaohui Zhao received the M.S. degree in Control Engineering from Xi'an University of Technology, China, in 2021. He is currently pursuing a Ph.D. degree at the School of Information and Communications Engineering, Xi'an Jiaotong University, Xi'an, China. His research interests include wireless communications, orthogonal time frequency space modulation, LEO satellite communications.

Lei You obtained his PhD in Computer Science from Uppsala University's Department of Information Technology in 2019. He was a visiting data scientist in the Boston Consulting Group (BCG) Gamma. He has also served as a data scientist at Bolt and Doordash (Wolt), specializing in on-demand logistics optimization. His earlier research was focused on the application of information theory to optimize resource allocation for data throughput and network reliability in advanced communication systems. Presently, he is an Assistant Professor in Applied Mathematics at Department of Engineering Technology of Technical University of Denmark (DTU). His ongoing research focuses on Optimal Model Refinement, aiming to augment the interpretability and efficiency of machine learning models through mathematical optimization.

Yaohua Sun received the bachelor's degree (Hons.) in telecommunications engineering (with management) and the Ph.D. degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2014 and 2019, respectively. He is currently an Associate Professor with the School of Information and Communication Engineering, BUPT. His research interests include intelligent radio access networks and LEO satellite communication. He has published over 50 papers including 3 ESI highly cited papers. He has been a Reviewer for IEEE Trans. Commun., IEEE Trans. Mob. Comput., etc.

Symeon Chatzinotas (S'06-M'09-SM'13-F'23) is Full Professor and Head of the SIGCOM Research Group at SnT, University of Luxembourg. He is coordinating the research activities on communications and networking across a group of 80 researchers, acting as a PI for more than 40 projects and main representative for 3GPP, ETSI, DVB. He is currently serving in the editorial board of the IEEE Transactions on Communications, IEEE Open Journal of Vehicular Technology and the International Journal of Satellite Communications and Networking. In the past, he has been a Visiting Professor at the University of Parma, Italy and was involved in numerous RD projects for NCSR Demokritos, CERTH Hellas and CCSR, University of Surrey. He was the co-recipient of the 2014 IEEE Distinguished Contributions to Satellite Communications Award and Best Paper Awards at WCNC, 5GWF, EURASIP JWCN, CROWNCOM, ICSSC. He has (co-)authored more than 700 technical papers in refereed international journals, conferences and scientific books.