SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2025, Vol. 68, Iss. 9, 190305:1–190305:15
https://doi.org/10.1007/s11432-024-4523-1

Special Topic: Cohesive Clustered Satellites System for 5GA and 6G Networks

Split-LEO: efficient AI model training over LEO satellite networks

Wen WU[1] & Xinyu HUANG[2*]

[1]*Pengcheng Laboratory, Shenzhen 518055, China*
[2]*Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, Canada*

Received 19 December 2024/Revised 8 May 2025/Accepted 24 July 2025/Published online 15 August 2025

Abstract In this paper, we propose a novel split federated learning scheme for accelerating the artificial intelligence (AI) model training in low Earth orbit (LEO) satellite networks, named Split-LEO. Specifically, the proposed scheme splits the entire AI model into multiple satellite-side models deployed at LEO satellites and multiple ground-side models deployed at the ground station. Each satellite parallelly performs model training via the collaboration with its corresponding ground-side model and then aggregates satellite-side and ground-side models into a global model, thereby significantly reducing training delay. Furthermore, we formulate an optimization problem with the objective of minimizing training delay via optimizing split point selection and computing resource allocation of satellites and the ground station. To solve the non-convex optimization problem, we transform the problem and then decompose it into two subproblems. The former split point selection subproblem is solved by using an exhaustive search method, while the latter computing resource allocation subproblem is solved by a Lagrange multiplier update algorithm. Extensive simulation results demonstrate that the proposed scheme can significantly reduce training delay while preserving model accuracy as compared with the state-of-the-art benchmarks.

Keywords split learning, AI model training, LEO satellite network, split point selection, resource allocation

Citation Wu W, Huang X Y. Split-LEO: efficient AI model training over LEO satellite networks. Sci China Inf Sci, 2025, 68(9): 190305, https://doi.org/10.1007/s11432-024-4523-1

1 Introduction

With the recent rapid advancement of low Earth orbit (LEO) satellite launch technology, the satellite network segment is expected to be a key component in the future sixth-generation (6G) network, which integrates satellite and terrestrial networks to provide network services anytime and anywhere [1–3]. The success of LEO satellite networks has stimulated major commercial companies, such as SpaceX, Amazon, and OneWeb [4]. For instance, SpaceX plans to launch approximately 42000 LEO satellites to construct the meta-constellation for communications [5]. According to a recent report by March 2025, more than 7000 SpaceX satellites have been launched and are operational in orbit.

With the development of advanced sensing technology, LEO satellites can acquire large-scale Earth imagery and conduct complex Earth observation tasks, such as real-time earthquake warning, remote sensing, climate change, and cooperative monitoring [6, 7]. In these Earth observation tasks, massive Earth imagery and sensor data are collected by these LEO satellites and processed by advanced artificial intelligence (AI) models. The data volume of an image of the Earth imagery, particularly high-resolution satellite imagery, can be easily up to several gigabytes. Transmitting large volumes of raw data from LEO satellites to ground stations for model training renders significant communication overhead. In addition, significant transmission delay will be incurred due to bandwidth-limited satellite-ground communication links. Therefore, a distributed learning approach that trains the AI model at LEO satellites without transmitting massive raw data is a potential solution.

In the literature, several pioneering studies have been devoted to investigating distributed learning schemes in LEO satellite networks [8–14]. Elmahallawy et al. [8, 9] proposed a federated learning (FL) framework for satellite networks, in which high-altitude balloon platforms are adopted as parameter servers to enhance satellite visibility. Zhai et al. [10] proposed a novel decentralized FL framework, in which AI models are trained and then aggregated in a decentralized manner, thereby avoiding significant

* Corresponding author (email: xinyu.huang1@uwaterloo.ca)

downlink communication overhead between satellites and the ground station. Taking satellite heterogeneity into account, Lin et al. [11] proposed an FL substructure scheme to speed up the training process. Han et al. [12] proposed a satellite-assisted FL scheme that enables ground users in different remote areas to collaboratively train the AI model. An extended work proposed a novel FL scheme tailored to space-air-ground integrated networks, leveraging the nodes within space and air network segments as edge computing units and model aggregators [13]. Considering the impact of satellite mobility, Razmi et al. [14] designed an enhanced FL scheme for LEO satellite constellations, which utilized intra-orbit inter-satellite links to mitigate the impact of intermittent connectivity. The existing studies focus on designing different FL schemes for enabling distributed model training over satellite networks. Although feasible, they still incur significant communication overhead, especially when the AI model is large. Moreover, due to the constraints of satellite hardware and energy supply, the computing capabilities of satellites are much lower than those of ground stations. The above FL schemes leverage limited satellite computing resources to perform local model training, which suffers from long training delay. Therefore, designing an efficient distributed learning scheme to reduce training delay in resource-limited LEO satellite networks is paramount.

Split learning (SL) has emerged as a potential solution to address the above issues, which can facilitate computation-efficient and communication-efficient distributed AI model training [15, 16]. The basic idea behind SL is to partition the entire AI model into a front-end part (i.e., the first few layers) that runs on the mobile device and a back-end part (i.e., the last few layers) that operates on the edge server. The partition point is referred to as the split point or cut layer in the literature. The front-end part is called the device-side model, while the back-end part is named as the server-side model [17–19]. Inspired by the concept of SL, we aim to introduce it into the LEO satellite networks. In this way, an AI model can be divided into two components between the satellite and the ground station, i.e., the satellite-side model running on the LEO satellites and the ground-side model operating on the ground station. During the SL process, only a small amount of smashed data (i.e., the output at the split point) and its corresponding gradient are exchanged between the satellites and the ground station to enable model training. Once a proper split point is selected, a small portion of the computational workload is handled by the resource-limited satellites, while a large portion of the computational workload is offloaded to the resource-abundant ground station. Therefore, an SL-based solution is deemed a potential distributed learning approach for resource-limited LEO satellite networks.

Designing an efficient SL scheme for resource-limited satellite networks encounters several challenges. Firstly, the vanilla SL scheme trains the AI model sequentially, which results in significant training delay. Such training delay is the product of the number of satellites and the individual training time of each satellite. Given the thousands of satellites in the LEO constellation, the training delay can be substantial. Therefore, a parallel training mechanism should be integrated into the SL scheme to accelerate the training process. Secondly, different satellites possess heterogeneous computing capabilities, and the channel conditions between satellites and ground base stations are also dynamic. Device heterogeneity and network dynamics result in a pronounced straggler effect for a parallel training scheme. Thus, judiciously selecting split points considering satellite heterogeneity and network dynamics is essential to alleviate the straggler effect.

To address the above challenges, in this paper, we first propose a novel split FL scheme tailored for LEO satellite networks, named Split-LEO. Specifically, each satellite possesses a satellite-side model, while the resource-abundant ground station possesses multiple ground-side models corresponding to different satellites. The split points of satellites can be selected differently, thereby alleviating the straggler effect caused by computing resource heterogeneity among satellites. In the training process, each satellite performs model training in parallel with the assistance of its corresponding ground-side model, and then aggregates both satellite-side and ground-side models into a global model. In this way, the proposed scheme can reduce training delay through a parallel training mechanism while accommodating satellite heterogeneity. Secondly, we formulate a training delay minimization problem, which jointly optimizes split point selection and computing resource allocation. To solve the problem, we transform the original non-convex problem into a convex optimization problem, and then decompose it into two subproblems. The former split point selection subproblem is tackled using an exhaustive search algorithm, since the split point selection decision variable is combinatorial. The latter computing resource allocation subproblem is solved by deriving the Karush-Kuhn-Tucker (KKT) conditions and employing an iterative Lagrange multiplier update algorithm. Extensive simulation results based on real-world Starlink Phase 1 satellite constellation demonstrate that the proposed scheme can achieve superior performance in terms of training

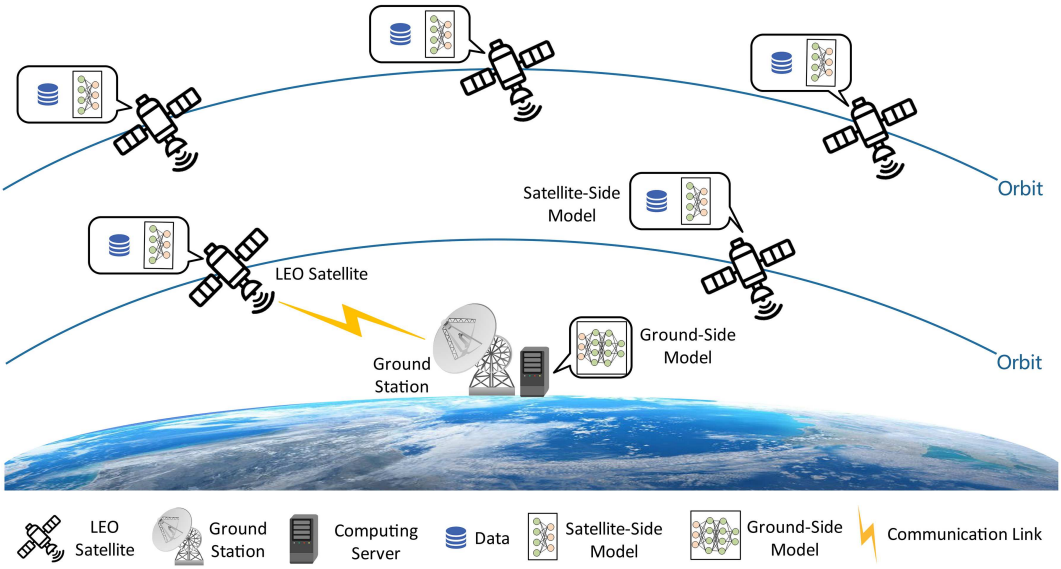

Figure 1 (Color online) Considered scenario.

delay reduction as compared with the state-of-the-art benchmarks.

The main contributions of this paper are summarized as follows:

- We propose a split-LEO scheme to accelerate model training in LEO satellite networks;
- We formulate a joint optimization problem with the objective of minimizing the training delay;
- We transform and decompose the problem into a split point selection subproblem and a computing resource allocation subproblem;
- We design an exhaustive search algorithm and a Lagrange multiplier update algorithm to determine split point selection and computing resource allocation.

The remainder of this paper is organized as follows. The proposed scheme and the system model are presented in Sections 2 and 3, respectively. Problem formulation and the proposed solution are presented in Section 4. Simulation results are provided in Section 5. Finally, Section 6 concludes this paper.

2 Proposed scheme

2.1 Considered scenario

As shown in Figure 1, a typical LEO satellite network is considered, consisting of a number of moving LEO satellites and a ground station, which is detailed as follows.

- LEO satellites: Let $\mathcal{S}$ denote the set of LEO satellites that move along the predetermined orbits. Each satellite $s \in \mathcal{S}$ is equipped with an on-board computing server for AI model training. Due to hardware heterogeneity, computation workload dynamics, and energy supply constraints, different satellites have different amounts of computing capabilities at different time slots.
- Dataset: Each satellite has a local dataset $\mathcal{D}_s, \forall s \in \mathcal{S}$, denoted by

$$\mathcal{D}_s = \{\mathbf{x}_i, y_i\}_{i=1}^{D_s}, \forall s \in \mathcal{S}, \tag{1}$$

where D_s represents the number of data samples at satellite s. Here, $\mathbf{x}_i \in \mathbb{R}^{Q\times 1}$ and $y_i \in \mathbb{R}^{1\times 1}$ denote the input data sample and the corresponding label, respectively, where Q denotes the dimension of the input data sample. Taking all the satellites into account, the whole dataset is denoted by $\mathcal{D} = \cup_{\forall s\in\mathcal{S}}\mathcal{D}_s$. Note that the data possessed by each satellite are acquired using advanced on-board sensors. For instance, high-resolution satellite images are obtained via synthetic aperture radars [20].

- Ground station: The ground station is placed at a fixed location, which can communicate with satellites within its coverage. The ground station is equipped with a powerful computing server with sufficient energy supply, whose computing capability is much higher than that of satellites. Additionally, the network controller at the ground station is in charge of collecting real-time network information, including satellite computing capabilities and channel conditions between satellites and the ground station. Based

Table 1 Summary of notations.

Notation	Description
$\mathbf{v}^t$	Split point selection decision at training round t
$\mathbf{c}^t$	Ground station's computing resource allocation decision at training round t
$\mathbf{f}^t$	Satellite computing resource allocation decision at training round t
B	Batch size
$\mathcal{S}^t$	LEO satellite constellation at training round t
t	Training round index
K	Participating number of satellites in each training round
$\mathcal{K}^t$	Available set of satellites at training round t
$\mathcal{V}$	Available set of split points
$\mathcal{T}$	Set of all training rounds
$\mathbf{w}^s$	Satellite-side AI model
$\mathbf{w}^g$	Ground-side AI model
$\mathbf{w}$	The whole AI model
W_g, W_s	Spectrum resource of the uplink and downlink transmission
P_g, P_s	Transmission power of the ground station and satellite
$\mathcal{D}_s$	Local dataset at satellite s
$\mathcal{D}$	Entire dataset across all satellites
$L(\mathbf{w})$	Average loss function given model parameter $\mathbf{w}$
η	Learning rate of model training
$D\left(v_k^t, f_k^t, c_k^t\right)$	Training delay at training round t
$\beta(v)$	Data size of smashed data at split point v
$\gamma(v)$	Data size of smashed data's gradient at split point v
$\xi\left(v\right)$	Data size of satellite-side model at split point v
$\eta_s\left(v\right)$	Computational workload of satellite-side model at split point v

on this, the network controller adopts an AI-based algorithm to coordinate the entire model training process. Note that main notations in this paper are summarized in Table 1.

2.2 Model training process

In the considered scenario, all satellites and the ground station cooperate to train an AI model. Let $l\left(\mathbf{x}_i, y_i; \mathbf{w}\right)$ denote the loss function (e.g., cross-entropy, mean squared error, and log likelihood) associated with data sample i given the model parameter $\mathbf{w}$. Taking all data samples at one satellite into account, the average loss function at satellite s is given by

$$L_s(\mathbf{w}) = \frac{1}{D_s} \sum_{i=1}^{D_s} l\left(\mathbf{x}_i, y_i; \mathbf{w}\right), \forall s \in \mathcal{S}. \tag{2}$$

Then, taking all the satellites into account, the average loss function across the whole dataset can be written as

$$L(\mathbf{w}) = \frac{\sum_{s\in\mathcal{S}} D_s L_s(\mathbf{w})}{\sum_{s\in\mathcal{S}} D_s}, \tag{3}$$

which is obtained via weighting local loss functions among satellites.

The AI model training process aims to minimize the average loss function $L(\mathbf{w})$ in an iterative manner, thereby identifying the optimal model parameter $\mathbf{w}^\star$, which can be expressed by

$$\mathbf{w}^\star = \arg\min_{\mathbf{w}} L(\mathbf{w}). \tag{4}$$

Given the limited computing capabilities of satellites, traditional FL schemes in satellite networks will experience significant training delay. The reason is that computation-intensive AI models are only trained on resource-limited satellites. Moreover, heterogeneous computing capabilities and time-varying channel conditions of satellites lead to a pronounced straggler effect, further slowing down the overall training process. To address the above challenges, in the following, we propose an efficient SL scheme that accelerates model training by utilizing the abundant computing resources of the ground station while accommodating satellite heterogeneity.

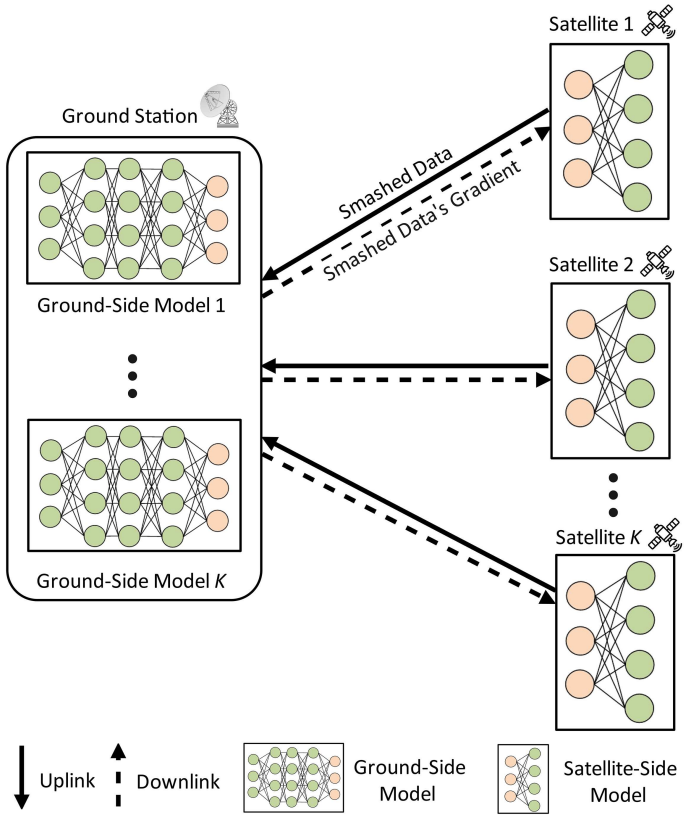

Figure 2 (Color online) Illustration of the proposed split-LEO scheme.

2.3 Proposed split-LEO scheme

As shown in Figure 2, a novel distributed learning scheme, named Split-LEO, is depicted. The basic idea behind this scheme are to establish several ground-side models with the same model architecture at the ground station for different participating satellites. This allows each satellite to conduct model training in parallel, thus significantly reducing the training delay, as compared with the vanilla SL scheme that performs model training in order.

Let $t \in \mathcal{T}$ represent the index of training rounds. Each satellite can hold a small portion of the entire AI model, referred to as satellite-side model, represented by $\mathbf{w}^s$. The remaining part of the AI model is placed at the ground station, referred to as ground-side model, denoted by $\mathbf{w}^g$. Note that $\mathbf{w} = \{\mathbf{w}^s, \mathbf{w}^g\}$. The procedure of the Split-LEO scheme operates in an iterative manner, which is outlined as follows.

(1) System configuration stage. In each training round, the ground station randomly selects K satellites from the set of accessible satellites to attend the training process. Due to the mobility of satellites, the set of accessible satellites in each training round t is time-varying, denoted by $\mathcal{S}^t \subseteq \mathcal{S}$. Note that we have $|\mathcal{S}^t| \geqslant K$. The set of selected satellites is defined as $\mathcal{K}^t \subseteq \mathcal{S}^t$, where $|\mathcal{K}^t| = K$. Then, the network controller collects the computing capabilities and channel conditions of the selected satellites. Based on this, the network controller makes the following decisions.

• Split point selection: The selected split point for each satellite should satisfy the following constraint:

$$v_k^t \in \mathcal{V}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}, \tag{5}$$

where $\mathcal{V}$ represents the feasible split points of the considered AI model. Once the split point is determined, the satellite-side model is defined as $\mathbf{w}^s(v_k^t)$, and the ground-side model is defined as $\mathbf{w}^g(v_k^t)$, i.e., $\mathbf{w} = \{\mathbf{w}^s(v_k^t), \mathbf{w}^g(v_k^t)\}$. Note that K ground-side models, i.e., $\mathbf{w}^g(v_k^t), \forall k \in \mathcal{K}^t$ are deployed at the ground station.

• Computing resource allocation: The computing resource of the ground station is allocated to different ground-side models. Let c_k^t denote the computing resource allocation decision, satisfying the following constraints:

$$0 \leqslant c_k^t \leqslant 1, \forall k \in \mathcal{K}^t, t \in \mathcal{T}, \tag{6}$$

Algorithm 1 Satellite-side model training algorithm.

for each training round t **do**
  for each satellite $k \in \mathcal{K}^t$ in parallel **do**
    Configure satellite computing frequency f_k^t;
    Perform the FP of the satellite-side model based on its local dataset and generate the smashed data;
    Transmit the smashed data to the ground station using satellite-ground communication link;
    Wait for the corresponding smashed data's gradient;
    Perform BP and update the satellite-side model;
    Transmit the updated satellite-side model to the ground station for model aggregation.
  end for
end for

$$\sum_{k\in\mathcal{K}^t} c_k^t \leqslant 1, \forall t \in \mathcal{T}. \tag{7}$$

Eq. (7) indicates that the total amount of allocated computing resources should not exceed the computing capacity of the ground-side computing server.

Once the above decisions are optimized via the optimization algorithm running at the ground station, which will be elaborated in Subsection 4.3. Once decisions are made, the ground station will configure the AI model split and computing resources at both selected satellites and the ground station.

(2) Ground-side model training stage. This stage consists of the following discontinued steps. First, the ground station sends the configured satellite-side model $\mathbf{w}^s(v_k^t)$ to each selected satellite via ground-satellite communication links. For simplicity, we assume that the radio spectrum resource between satellites and the ground station is equally allocated to each satellite for data transmission in this paper. Second, the ground station waits until the reception of the smashed data from the satellite. The smashed data are used for the forward propagation (FP) of the ground-side model. Third, the ground station calculates the loss by comparing the label and predicted results. The ground station executes the backward propagation (BP) until the split point, such that the smashed data's gradient is generated, which is given by

$$\mathbf{w}^g(v_k^t) \leftarrow \mathbf{w}^g(v_k^t) - \zeta \sum_i \frac{\nabla l\left(\mathbf{x}_i, y_i; \mathbf{w}^g(v_k^t)\right)}{B}, \forall k \in \mathcal{K}^t, \tag{8}$$

where $\zeta > 0$ represents the learning rate for AI model training and B indicates the batch size. The ground station sends the gradient of aggregated data to the corresponding satellite. Finally, the ground station waits for the updated models from the satellites and combines these with the ground station models to construct a new AI model, i.e.,

$$\bar{\mathbf{w}}_m^d(t) = \frac{\sum_{k\in\mathcal{K}^t} D_u^t \mathbf{w}_k^t}{\sum_{k\in\mathcal{K}^t} D_k^t}, \tag{9}$$

where $\mathbf{w}_k^t = \{\mathbf{w}^s(v_k^t), \mathbf{w}^g(v_k^t)\}$ denotes the concatenated satellite-side model and ground-side model for satellite k.

(3) Satellite-side model training stage. This stage consists of the following steps to train the AI model. Once the satellite-side model is received, satellite $k \in \mathcal{K}^t$ trains the satellite-side model based on its local dataset given the determined satellite computing frequency f_k^t, i.e., the FP of the satellite-side model. Until the split point is reached, the smashed data are generated. The smashed data are transmitted to the ground station using the LEO-ground communication link. Then, the satellite k waits for the smashed data's gradient to continue the BP of the satellite-side model, thereby updating the satellite-side model, which is defined as

$$\mathbf{w}^s(v_k^t) \leftarrow \mathbf{w}^s(v_k^t) - \zeta \sum_i \frac{\nabla l\left(\mathbf{x}_i, y_i; \mathbf{w}^s(v_k^t)\right)}{B}, \forall k \in \mathcal{K}^t. \tag{10}$$

Once completed, the updated satellite-side model is transmitted to the ground station for model aggregation.

One training round includes the above stages. Multiple training rounds are performed in an iterative manner until satisfactory model accuracy is reached. The detailed training procedure of satellite-side and ground-side model training is detailed in Algorithms 1 and 2, respectively.

Algorithm 2 Ground-side model training algorithm.

for each training round t **do**
  Send the configured satellite-side model to each selected satellite;
  Wait for the smashed data from each satellite;
  for each ground-side model $k \in \mathcal{K}^t$ in parallel **do**
    Perform FP of the ground-side model;
    Calculate the loss by comparing the label and predicted results;
    Execute the BP of the ground-side model $\mathbf{w}^g(v_k^t)$ and generate smashed data's gradient;
  end for
  Send the smashed data's gradient to the corresponding satellite;
  Wait for the updated satellite-side model $\mathbf{w}^s(v_k^t)$;
  Concatenate the satellite-side model and ground-side model;
  Aggregate all the satellite-side and ground-side models into an updated whole AI model;
end for

3 System model

In the proposed scheme, the model training is performed at both satellites and the ground station, which incurs computation delay and transmission delay.

3.1 Computation delay

In the proposed scheme, the satellite-side and ground-side models are trained at the satellites and the ground station, respectively. As such, the corresponding delays are analyzed as follows.

Satellite-side model training delay: Let $\eta_s\left(v_k^t\right)$ represent the computation workload in terms of floating point operations (FLOPs) of satellite-side model training given split point v_k^t, taking both FP and BP into consideration. The computation workload of FP can be accurately calculated based on the FLOPs of model inference. The computation workload of BP is assumed to be twice that of FP, since BP takes approximately twice as long as FP during neural network training [21]. As such, it is assumed that $\eta_s\left(v_k^t\right) = 3\eta_s^{\mathrm{FP}}\left(v_k^t\right)$ where $\eta_s^{\mathrm{FP}}\left(v_k^t\right)$ denotes the FP computation workload at split point v_k^t. The computation delay at satellite k is given by

$$d_{s,k}^t = \frac{\eta_s\left(v_k^t\right)}{F_k\delta}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}, \tag{11}$$

where δ represents the number of FLOPs that can be processed in a processing unit cycle, depending on the architecture of the processor [22,23], such as central processing units (CPUs) and graphics processing units (GPUs). F_k denotes the computing frequency of satellite k.

Ground-side model computation delay: Let η represent the computation workload of the entire AI model. As such, the computation workload of the ground-side model is given by $\eta - \eta_s\left(v_k^t\right)$. Let F_g represent the overall frequency of the computing server at the ground station. Multiple ground-side models are deployed on the ground station, and the allocated computing resource for the k-th ground-side model is $c_k^t F_g$, as determined by the controller at the ground station. Similar to that in (11), the computation delay of the ground-side model k is given by

$$d_{g,k}^t = \frac{\eta - \eta_s\left(v_k^t\right)}{c_k^t F_g\delta}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}. \tag{12}$$

3.2 Transmission delay

During the training process of the proposed scheme, some data are exchanged between the satellite and ground station, including satellite-side model, smashed data, and smashed data's gradients. The transmission delay includes four components, i.e., satellite-side model distribution delay, smashed data transmission delay, smashed data's gradient transmission delay, and updated satellite-side model transmission delay. The detailed analysis is as follows.

First, the ground station sends the satellite-side models to each satellite. Let

$$R_{g,k}^t = W_g \log_2\left(1 + \frac{P_g G_g h_{g,k}^t}{N_o}\right), \forall k \in \mathcal{K}^t, t \in \mathcal{T} \tag{13}$$

represent the average transmission rate from the ground station to the k-th participating satellite [24–26]. Here, W_g, G_g, P_g, N_o, and $h_{g,k}^t$ represent the spectrum bandwidth, directional antenna gain of ground

station, transmission power of the ground station, white noise power, and channel gain between ground station and satellite k, respectively. Since the satellite transmission system adopts the directional antenna, interference from other communication links is assumed to be negligible. As such, interference is not considered in the communication model [27].

Let $\xi\left(v_k^t\right)$ denote the data size of the satellite-side model given the split point v_k^t. The satellite-side model distribution delay is given by

$$d_{1,k}^t = \frac{\xi\left(v_k^t\right)}{R_{g,k}^t}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}. \tag{14}$$

Second, the satellite sends the smashed data to the ground station for performing the FP of the ground-side model. Let W_s denote the amount of spectrum resource from satellites to the ground station, which is equally allocated to each satellite. Similar to (13), the average transmission rate from satellite k to the ground station is represented by

$$R_{s,k}^t = \frac{W_s}{K} \log_2 \left(1 + \frac{P_s G_s h_{s,k}^t}{N_o}\right), \forall k \in \mathcal{K}^t, t \in \mathcal{T}, \tag{15}$$

where G_s and P_s represent the directional antenna gain and the transmission power of each satellite, respectively. Here, $h_{s,k}^t$ denotes the average channel gain from satellite k to the ground station. In addition, the data size of smashed data given split point v_k^t is defined as $\beta(v_k^t)$. As such, the corresponding smashed data transmission delay is given by

$$d_{2,k}^t = \frac{\beta(v_k^t)}{R_{s,k}^t}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}. \tag{16}$$

Third, the ground station needs to send the smashed data's gradient back to each satellite for performing the BP of the satellite-side model. Let $\gamma(v_k^t)$ denote the data size of the smashed data's gradient given split point v_k^t. The corresponding delay is calculated as follows:

$$d_{3,k}^t = \frac{K\gamma(v_k^t)}{R_{g,k}^t}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}. \tag{17}$$

It is worth noting that all the satellites equally share the uplink spectrum resource to transmit the smashed data's gradient.

Fourth, the updated satellite-side model is sent to the ground station for model concatenation and model aggregation. The corresponding delay is given by

$$d_{4,k}^t = \frac{\xi\left(v_k^t\right)}{R_{s,k}^t}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}. \tag{18}$$

The overall transmission delay is calculated by taking the above four components into account.

3.3 Model training delay

The per-round training delay should take all the above computation and communication delay components into account. In addition, the model aggregation can only be performed until the last satellite-side model is successfully received. As such, the per-round model training delay is given by

$$D\left(v_k^t, c_k^t\right) = \max_{k \in \mathcal{K}^t} \{d_{s,k}^t + d_{g,k}^t + d_{1,k}^t + d_{2,k}^t + d_{3,k}^t + d_{4,k}^t\}, \forall t \in \mathcal{T}. \tag{19}$$

The above equation indicates that the model training process exhibits the straggler effect.

The overall model training delay of the proposed scheme until satisfactory model performance is achieved can be represented by $\sum_{t \in \mathcal{T}} D\left(v_k^t, c_k^t\right)$. Given the above analytical results, we will optimize the scheme via mitigating the straggler effect, thereby reducing the overall training delay in dynamic satellite networks.

4 Problem formulation and proposed solution

4.1 Problem formulation

In the following, we aim to minimize the average model training delay by optimizing split point selection and the ground station's computing resource allocation. Hence, the long-term training delay minimization problem is formulated as follows:

$$\begin{aligned}
&\mathcal{P}_0: \min_{\{\mathbf{v}^t,\mathbf{c}^t\}_{t\in\mathcal{T}}} \sum_{t\in\mathcal{T}} D\left(v_k^t, c_k^t\right), \\
&\text{s.t.} \quad v_k^t \in \mathcal{V}, \forall k \in \mathcal{K}^t, t \in \mathcal{T}, && \text{(20a)} \\
&0 \leqslant c_k^t \leqslant 1, \forall k \in \mathcal{K}^t, t \in \mathcal{T}, && \text{(20b)} \\
&\sum_{k\in\mathcal{K}^t} c_k^t \leqslant 1, \forall t \in \mathcal{T}, && \text{(20c)}
\end{aligned}$$

where $\mathbf{v}^t = [v_1^t, v_2^t, \ldots, v_K^t] \in \mathbb{N}^{K\times 1}$ and $\mathbf{c}^t = [c_1^t, c_2^t, \ldots, c_K^t] \in \mathbb{R}^{K\times 1}$ denote the split point selection decision and the ground station's computing resource allocation decision, respectively. Constraints (20a), (20b), and (20c) ensure that the optimization variables are within the feasible regions.

We can find that all constraints are one-shot, and the overall training delay can be decomposed into the sum of the training delay in each training round. Hence, the optimization decisions in different training rounds can be deemed as independent. As such, problem $\mathcal{P}_0$ can be decoupled into a set of one-shot optimization problems, i.e.,

$$\begin{aligned}
&\mathcal{P}_1: \min_{\mathbf{v}^t,\mathbf{c}^t} D\left(v_k^t, c_k^t\right), \\
&\text{s.t.} \quad v_k^t \in \mathcal{V}, \forall k \in \mathcal{K}^t, && \text{(21a)} \\
&0 \leqslant c_k^t \leqslant 1, \forall k \in \mathcal{K}^t, && \text{(21b)} \\
&\sum_{k\in\mathcal{K}^t} c_k^t \leqslant 1. && \text{(21c)}
\end{aligned}$$

The above optimization problem is difficult to solve for the following reasons. First, the objective function $D\left(v_k^t, c_k^t\right)$ is non-convex due to the maximization operation of training delay among all the satellites. As such, the problem is a non-convex optimization problem. Second, the split point selection variable is combinatorial, such that the problem is a combinatorial optimization problem, while the computing resource allocation variable is continuous. As such, the problem is a mixed-integer non-convex optimization problem, which is challenging to solve. Lastly, the coupling relationship exists between optimization variables, rendering difficulty in solving the joint optimization problem.

4.2 Problem transformation

In this subsection, we first transform the non-convex problem into a convex optimization problem, and then decompose the problem into two subproblems, i.e., a split point selection subproblem and a computing resource allocation subproblem. Note that we omit t in the following analysis for simplicity, since the following optimization problem is solved in each training round.

Let $A_k = d_{1,k}^t + d_{2,k}^t + d_{3,k}^t + d_{4,k}^t, \forall k \in \mathcal{K}^t$ denote the transmission delay in each training round. The objective function can be rewritten as

$$D\left(v_k, c_k\right) = \max_{k\in\mathcal{K}} \left\{ \frac{\eta_s\left(v_k\right)}{F_k\delta} + \frac{\eta - \eta_s\left(v_k\right)}{c_k F_g \delta} + A_k \right\}. \tag{22}$$

In addition, we introduce a new slack variable τ in problem $\mathcal{P}_1$ [28], which defines the maximum training delay among all satellites, i.e.,

$$\frac{\eta_s\left(v_k\right)}{F_k\delta} + \frac{\eta - \eta_s\left(v_k\right)}{c_k F_g \delta} + A_k \leqslant \tau,$$

which guarantees the training delay of each satellite is less than τ.

In this way, the above problem $\mathcal{P}_1$ is equivalent to the following problem:

$$\begin{aligned}
&\mathcal{P}_2: \min_{\mathbf{v},\mathbf{c},\tau} \tau, \\
&\text{s.t.} \quad \frac{\eta_s(v_k)}{F_k\delta} + \frac{\eta - \eta_s(v_k)}{c_k F_g \delta} + A_k \leqslant \tau, \forall k \in \mathcal{K}, && \text{(23a)} \\
&v_k \in \mathcal{V}, \forall k \in \mathcal{K}, && \text{(23b)} \\
&0 \leqslant c_k \leqslant 1, \forall k \in \mathcal{K}, && \text{(23c)} \\
&\sum_{k\in\mathcal{K}} c_k \leqslant 1. && \text{(23d)}
\end{aligned}$$

According to the properties of the optimization problem, once the split point selection decision is determined, computing resource allocation decisions can be determined by solving a convex optimization problem. Hence, the problem $\mathcal{P}_2$ is further decomposed into the computing resource allocation subproblem $\mathcal{P}_{3\text{-}1}$ and cut layer selection subproblem $\mathcal{P}_{3\text{-}2}$, which is given as follows:

$$\begin{aligned}
&\mathcal{P}_{3\text{-}1}: \min_{\mathbf{c},\tau} \tau, \\
&\text{s.t.} \quad \frac{\eta_s(v_k)}{F_k\delta} + \frac{\eta - \eta_s(v_k)}{c_k F_g \delta} + A_k \leqslant \tau, \forall k \in \mathcal{K}, && \text{(24a)} \\
&0 \leqslant c_k \leqslant 1, \forall k \in \mathcal{K}, && \text{(24b)} \\
&\sum_{k\in\mathcal{K}} c_k \leqslant 1, && \text{(24c)}
\end{aligned}$$

$$\begin{aligned}
&\mathcal{P}_{3\text{-}2}: \min_{\mathbf{v}} \tau, \\
&\text{s.t.} \quad v_k \in \mathcal{V}, \forall k \in \mathcal{K}. && \text{(25a)}
\end{aligned}$$

Subproblem $\mathcal{P}_{3\text{-}1}$ is to minimize the training delay via optimizing computing resource allocation and slack variable τ. Subproblem $\mathcal{P}_{3\text{-}2}$ is to minimize the training delay via optimizing the split point selection. In the following subsections, we will solve these two subproblems, respectively. Specifically, a two-layer optimization algorithm is proposed to make the split point selection decision in the outer layer and the computing resource allocation decision in the inner layer.

4.3 Optimization algorithm

4.3.1 *Optimal computing resource allocation*

For subproblem $\mathcal{P}_{3\text{-}1}$, it can be easily seen that the objective function and constraints (24b) and (24c) are convex. For constraint (24a), its second-order derivative is given by $2(\eta - \eta_s(v_k))/c_k^3 F_g \delta$. Since $c_k > 0$, the second-order derivative is positive. As such, the above optimization problem is a convex optimization problem. To solve the problem, the Lagrange function is derived for problem $\mathcal{P}_{3-1}$, which is given by

$$\mathcal{L}(\mathbf{c}, \tau, \lambda, \alpha) = \tau + \sum_{k\in\mathcal{K}} \lambda_k \left(\frac{\chi_k}{c_k} + \phi_k - \tau \right) + \alpha \left(\sum_{k\in\mathcal{K}} c_k - 1 \right). \tag{26}$$

Here, $\lambda_k, \forall k \in \mathcal{K}$ and α denote the Lagrange multipliers for constraints (24a) and (24b), respectively. $\phi_k = \eta_s(v_k)/F_k^\delta + A_k$ and $\chi_k = (\eta - \eta_s(v_k))/F_g\delta$. The optimization problem is solved by the following Theorem 1.

Theorem 1. The optimal value $\tau^\star$ is given by

$$\tau^\star = \max_{k\in\mathcal{K}} \left\{ \frac{\chi_k}{c_k^\star} + \phi_k \right\}, \tag{27}$$

where

$$c_k^\star = \frac{\sqrt{\lambda_k \chi_k}}{\sum_{k\in\mathcal{K}} \sqrt{\lambda_k \chi_k}}, \forall k \in \mathcal{K} \tag{28}$$

represents the optimal computing resource allocation decision.

Algorithm 3 Computing resource allocation algorithm.

1: Initialize Lagrange multiplier $\lambda_k(0)$, and $p \leftarrow 0$;
2: **while** accuracy ϵ is not achieved **do**
3: Obtain $\alpha^\star$, $\{c_k^\star\}_{\forall k \in \mathcal{K}}$, and $\tau^\star$ via (A3), (A4), and (A5), respectively;
4: Obtain the subgradient of the Lagrange multiplier $\nabla\lambda_k$ via (29);
5: Update the Lagrange multiplier $\lambda_k(p)$ via (30);
6: $p \leftarrow p + 1$;
7: **end while**

Proof. Proof is provided in Appendix A.

Given the optimal value $\tau^\star$, the optimal value of the Lagrange multiplier $\lambda_k^\star, \forall k \in \mathcal{K}$ can be obtained by solving the Lagrange dual function. Since the Lagrange dual problem is a convex optimization problem, the corresponding subgradient is given as follows:

$$\nabla\lambda_k = \frac{\chi_k}{c_k^\star} + \phi_k - \tau^\star, \forall k \in \mathcal{K}. \tag{29}$$

Given the above subgradient, the Lagrange multiplier can be updated in each step via

$$\lambda_k(p+1) = \lambda_k(p) + \zeta\nabla\lambda_k(p), \forall k \in \mathcal{K}, \tag{30}$$

where $\zeta > 0$ is the step size and p is the number of iterations. The above optimization is given in Algorithm 3. It is worth noting that the algorithm terminates when the target accuracy ϵ is achieved.

4.3.2 *Optimal split point selection*

In the following, we aim to obtain the optimal split point selection decision $v_k^\star$ via solving subproblem $\mathcal{P}_{3\text{-}2}$. It can be seen that the split point selection subproblem is a combinatorial optimization problem. The reason is that the split point selection decisions belong to the set of available split points for an AI model. Hence, the problem can be solved by an exhaustive search algorithm. In this way, the number of possible combinations of split points is $V!/(V-K)!$. Since the number of split points is limited, the computation complexity is constrained when the number of attending satellites is small.

5 Performance evaluation

5.1 Simulation setup

We adopt the Starlink Phase 1 as the satellite constellation parameters in the simulation, consisting of 72 orbits with 22 satellites on each orbit. The inclination angle is 53°, the satellite's altitude is 550 km, and the satellite's velocity is 7.56 km/s. By default, the elevation angle is set to 40°. The carrier frequency is set to 28 GHz, and the spectrum resources of the uplink and downlink channels are both set to 100 MHz, respectively. The transmit powers of the satellite and ground station are set to 40 and 50 dBm, respectively. To compensate for the huge path loss due to long transmission distance, we adopt a directional antenna with an antenna gain of 50 dBi, and the path loss factor of the satellite-ground channel is set to 2.5 [26]. The latitude and longitude of the ground station are set to (45, −80). As shown in Figure 3, it can be seen that the number of available satellites for the ground station varies over different time slots, ranging from 1 to 5 satellites available, which indicates that the dynamics of available satellites will participate in the model training process. In the simulation, at most three satellites are randomly selected to participate in the model training unless specified. Moreover, due to satellite velocity, the transmission rate between satellites and the ground station also varies across time.

In addition, we consider a typical image classification task, and the CIFAR-10 dataset is adopted. In the CIFAR-10 dataset, each data sample is a color image with a label associated with ten classes, such as "Airplane" and "Automobile" [29]. Regarding the data distribution, we consider both independent and identically distributed (IID) and non-IID scenarios. The non-IID dataset is generated based on the Dirichlet distribution with a concentration parameter of 0.5 for the dataset partition. Three AI models are adopted to perform image classification tasks, including (1) LeNet for MNIST-10 dataset, which is a 12-layer chain-topology deep neural network (DNN) model, consisting of six 3×3 convolution layers (32, 32, 64, 64, 128, 128 channels, each of which is activated by ReLU and batch normalized, and every two of which are followed by 2×2 max pooling) followed by three fully-connected layers (382 and 192 units with

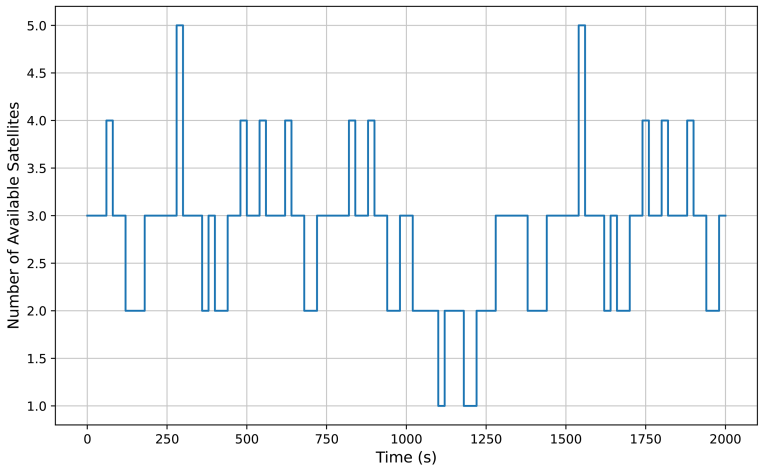

Figure 3 (Color online) Number of satellites with respect to different time slots.

ReLU activation and another 10 units activated by soft-max). The data size of the entire LeNet model is 18.93 MB; (2) GoogleNet for the CIFAR-10 dataset, which is a 22-layer model with 9 Inception blocks; and (3) ResNet-18 for the CIFAR-10 dataset, which is an 18-layer model with eight residual blocks.

We compare the proposed scheme with the following benchmarks. (1) Vanilla SL scheme: In the vanilla SL scheme, the AI model is trained across the satellites in a sequential manner. (2) FL scheme: In the FL scheme, all the satellites train the shared AI model locally and then send the updated local models to the ground station, thereby aggregating into a new model for the next round model training [30]. For fairness considerations, all the schemes adopt the same model initialization.

5.2 Simulation results

As shown in Figure 4(a), the training performance of GoogleNet with respect to the number of training rounds is evaluated over the IID CIFAR-10 dataset. Several important observations can be obtained. First, it can be seen that both schemes achieve nearly the same accuracy performance after convergence, i.e., around 80%, indicating that the proposed scheme can preserve model accuracy. Second, the proposed scheme takes a larger number of training rounds than the vanilla SL benchmark scheme. Specifically, the proposed scheme takes around 1400 training rounds, while the vanilla SL takes around 1700 training rounds until convergence. The overall training delay is the product of the per-round training delay and the number of training rounds. As shown in Figure 4(b), the training delay performance is presented. It is evident that the proposed scheme has a shorter training latency to reach convergence, as compared with the vanilla SL and FL schemes. Specifically, the proposed scheme converges within about 7500 min, while the benchmark vanilla SL and FL schemes converge within approximately 20000 and 30000 min, respectively. In other words, the proposed scheme speeds up AI model training by about 2.6× as compared with the vanilla SL scheme. The reason is that the proposed scheme can enable parallel model training among different satellites, thereby reducing per-round training delay as compared with the vanilla SL scheme that operates sequentially.

As shown in Figure 5, we evaluate the impact of different AI models. The training performance of the ResNet-18 model over the IID CIFAR-10 dataset is presented in Figure 5(a). Similar to that in Figure 4, it can be seen that the proposed scheme and vanilla SL scheme still achieve nearly the same accuracy. In addition, the proposed scheme takes more training rounds than the vanilla SL scheme. The reason is that the proposed scheme obtains the global model via model aggregation, thereby slowing down the training process. In Figure 5(b), the training delay performance is evaluated. We can observe that the proposed scheme has a much shorter training delay as compared with the benchmark schemes. Specifically, the proposed scheme takes about 630 min, which speeds up model convergence by about 2.8×, as compared with that of the vanilla SL scheme (about 1780 min). The model size of ResNet-18 is smaller than that of GoogleNet, such that the training delay is shorter than that of GoogleNet.

As shown in Figure 6, we evaluate the training performance over non-IID scenarios. First, similar to the observations found in Figures 4(a) and 5(a), the proposed scheme takes more training rounds while preserving nearly the same accuracy performance, as compared with benchmarks. Second, the

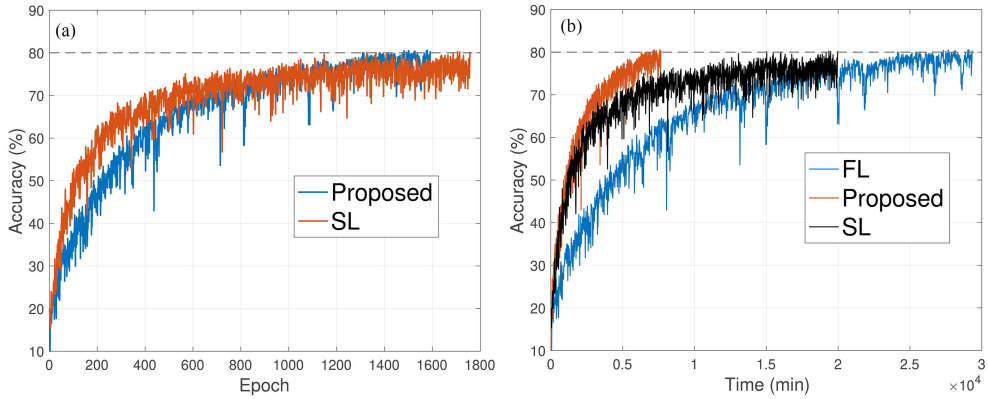

Figure 4 (Color online) Training performance of the GoogleNet over the IID CIFAR-10 dataset. (a) Model accuracy in terms of the number of training rounds; (b) model accuracy in terms of training delay.

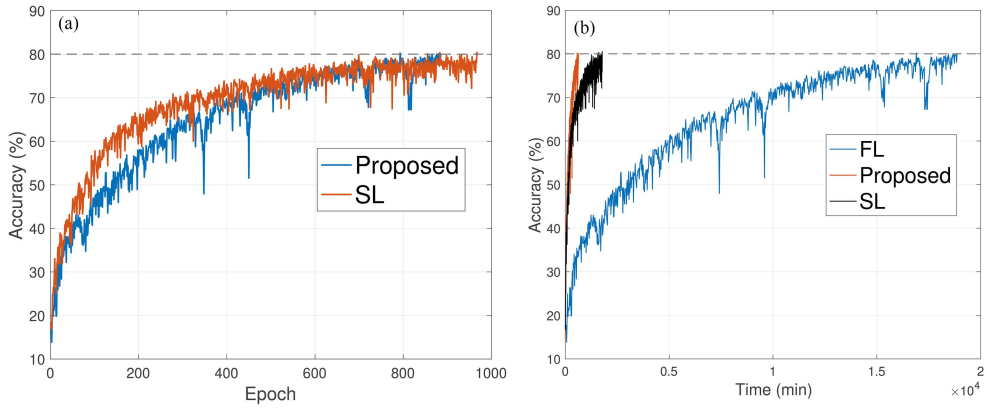

Figure 5 (Color online) Training performance of the ResNet-18 over the IID CIFAR10 dataset. (a) Model accuracy in terms of the number of training rounds; (b) model accuracy in terms of training delay.

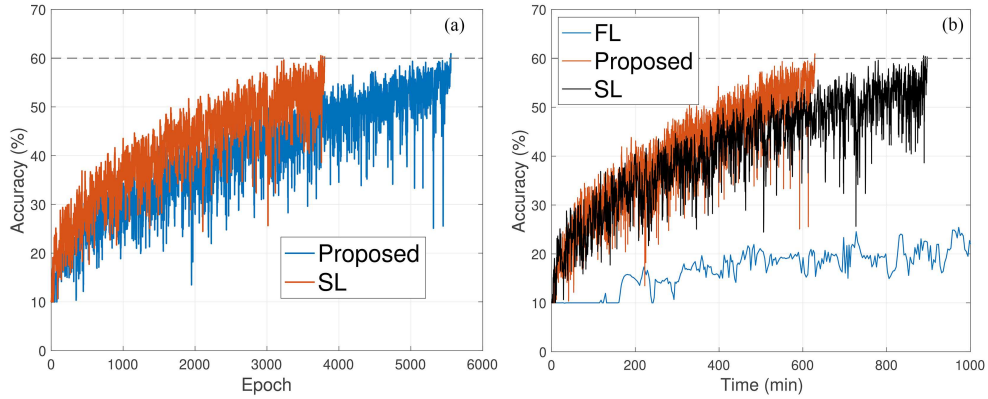

Figure 6 (Color online) Training performance of the ResNet-18 over the non-IID CIFAR10 dataset. (a) Model accuracy in terms of the number of training rounds; (b) model accuracy in terms of training delay.

proposed scheme and vanilla SL schemes can converge in non-IID scenarios, while the FL scheme may not, indicating the robustness of the SL-based schemes in model convergence. In addition, training delay performance in Figure 6(b) shows that the proposed scheme outperforms the existing benchmarks in terms of training delay. Specifically, the proposed scheme reduces the training delay by 1.4× as compared with the vanilla SL scheme.

6 Conclusion

In this paper, we have proposed a novel split federated learning scheme named split-LEO, designed to accelerate the model training process in resource-limited satellite networks. To reduce communication overhead and the computational workload on the satellite, the proposed scheme deploys only a small portion of the AI model on the satellite, with the remainder located at the ground station. We formulate the training delay minimization problem by jointly optimizing the selection of the split point and computing resource allocation of satellites. An efficient optimization algorithm, based on convex optimization theory, is proposed to solve the problem. The proposed scheme effectively reduces training delay in dynamic satellite networks. In future work, we aim to explore an appropriate satellite selection strategy for the proposed framework to further expedite model convergence.

Acknowledgements This work was supported in part by Pengcheng Laboratory Major Key Project (Grant Nos. 2025QYA002, PCL2024A01) and National Natural Science Foundation of China (Grant No. 62201311).

References

1 Shen X, Gao J, Wu W, et al. Holistic network virtualization and pervasive network intelligence for 6G. IEEE Commun Surv Tutorials, 2022, 24: 1–30
2 Shen X, Gao J, Wu W, et al. AI-assisted network-slicing based next-generation wireless networks. IEEE Open J Veh Technol, 2020, 1: 45–66
3 Xu L, Jiao J, Jiang S Y, et al. Semantic-aware coordinated transmission in cohesive clustered satellites: utility of information perspective. Sci China Inf Sci, 2024, 67: 199301
4 Osoro O B, Oughton E J. A techno-economic framework for satellite networks applied to low earth orbit constellations: assessing Starlink, OneWeb and Kuiper. IEEE Access, 2021, 9: 141611–141625
5 Harris M. Tech giants race to build orbital internet. IEEE Spectr, 2018, 55: 10–11
6 Shukla S, Macharia D, Husak G J, et al. Enhancing access and usage of Earth observations in environmental decision-making in Eastern and Southern Africa through capacity building. Front Sustain Food Syst, 2021, 5: 504063
7 Wu W, Zhou C, Li M, et al. AI-native network slicing for 6G networks. IEEE Wireless Commun, 2022, 29: 96–103
8 Elmahallawy M, Luo T, Ramadan K. Communication-efficient federated learning for LEO constellations integrated with HAPs using hybrid NOMA-OFDM. IEEE J Sel Areas Commun, 2024, 42: 1097–1114
9 Elmahallawy M, Luo T. FedHAP: fast federated learning for LEO constellations using collaborative HAPs. In: Proceedings of IEEE WCSP, Nanjing, 2022. 888–893
10 Zhai Z, Wu Q, Yu S, et al. FedLEO: an offloading-assisted decentralized federated learning framework for low Earth orbit satellite networks. IEEE Trans Mobile Comput, 2024, 23: 5260–5279
11 Lin Z, Chen Z, Fang Z, et al. FedSN: a federated learning framework over heterogeneous LEO satellite networks. IEEE Trans Mobile Comput, 2025, 24: 1293–1307
12 Han D J, Hosseinalipour S, Love D J, et al. Cooperative federated learning over ground-to-satellite integrated networks: joint local computation and data offloading. IEEE J Sel Areas Commun, 2024, 42: 1080–1096
13 Han D J, Fang W, Hosseinalipour S, et al. Orchestrating federated learning in space-air-ground integrated networks: adaptive data offloading and seamless handover. IEEE J Sel Areas Commun, 2024, 42: 3505–3520
14 Razmi N, Matthiesen B, Dekorsy A, et al. On-board federated learning for satellite clusters with inter-satellite links. IEEE Trans Commun, 2024, 72: 3408–3424
15 Singh A, Vepakomma P, Gupta O, et al. Detailed comparison of communication efficiency of split learning and federated learning. 2019. ArXiv:1909.09145
16 Gao Y, Kim M, Abuadbba S, et al. End-to-end evaluation of federated learning and split learning for Internet of Things. 2020. ArXiv:2003.13376
17 Wu W, Li M, Qu K, et al. Split learning over wireless networks: parallel design and resource management. IEEE J Sel Areas Commun, 2023, 41: 1051–1066
18 Zhang S, Wu W, Hu P, et al. Split federated learning: speed up model training in resource-limited wireless networks. In: Proceedings of IEEE ICDCS, Hong Kong, 2023. 985–986
19 Zhang S, Wu W, Song L, et al. Efficient model training in edge networks with hierarchical split learning. IEEE Trans Mobile Comput, 2025. doi: 10.1109/TMC.2025.3569407
20 Moreira A, Prats-Iraola P, Younis M, et al. A tutorial on synthetic aperture radar. IEEE Geosci Remote Sens Mag, 2013, 1: 6–43
21 Katharopoulos A, Fleuret F. Not all samples are created equal: deep learning with importance sampling. In: Proceedings of International Conference on Machine Learning, Stockholm, 2018. 2525–2534
22 Zeng Q, Du Y, Huang K, et al. Energy-efficient resource management for federated edge learning with CPU-GPU heterogeneous computing. IEEE Trans Wireless Commun, 2021, 20: 7947–7962
23 Zhang X, Zhou X, Lin M, et al. Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of IEEE CVPR, Salt Lake City, 2018. 6848–6856
24 Wu H, He M, Shen X, et al. Network performance analysis of satellite-terrestrial vehicular network. IEEE Internet Things J, 2024, 11: 16829–16844
25 Wu W, Chen N, Zhou C, et al. Dynamic RAN slicing for service-oriented vehicular networks via constrained learning. IEEE J Sel Areas Commun, 2021, 39: 2076–2089
26 He M, Wu H, Shen X, et al. Cooperative resource scheduling for environment sensing in satellite-terrestrial vehicular networks. IEEE Internet Things J, 2025, 12: 6734–6748
27 Zhou C, Wu W, He H, et al. Deep reinforcement learning for delay-oriented IoT task scheduling in SAGIN. IEEE Trans Wireless Commun, 2021, 20: 911–925
28 Liu C, Chua T J, Zhao J. Time minimization in hierarchical federated learning. In: Proceedings of IEEE/ACM SEC, Seattle, 2022. 96–106
29 Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images. 2009
30 Bonawitz K, Eichner H, Grieskamp W, et al. Towards federated learning at scale: system design. In: Proceedings of MLSys, Stanford, 2019. 1–15

Appendix A Proof of Theorem 1

First, based on the KKT condition[1)], the optimal solution can be obtained via taking the partial derivatives of the above Lagrange function, i.e.,

$$\frac{\partial \mathcal{L}\left(\mathbf{c}, \tau, \lambda, \alpha\right)}{\partial c_k} = -\frac{\lambda_k \chi_k}{c_k^2} + \alpha = 0, \forall k \in \mathcal{K}. \tag{A1}$$

As such, we can have

$$c_k = \sqrt{\frac{\lambda_k \chi_k}{\alpha}}, \forall k \in \mathcal{K}. \tag{A2}$$

Second, substituting the c_k into the complementary slackness condition $\sum_{k\in\mathcal{K}} c_k - 1 = 0$, we can obtain

$$\alpha^\star = \left(\sum_{k\in\mathcal{K}} \sqrt{\lambda_k \chi_k}\right)^2. \tag{A3}$$

By substituting the above result into (A2), we can obtain the optimal ground station computing resource allocation decision as follows:

$$c_k^\star = \frac{\sqrt{\lambda_k \chi_k}}{\sum_{k\in\mathcal{K}} \sqrt{\lambda_k \chi_k}}, \forall k \in \mathcal{K}. \tag{A4}$$

Lastly, based on the definition of τ, the optimal value $\tau^\star$ is given by

$$\tau^\star = \max_{k\in\mathcal{K}} \left\{ \frac{\chi_k}{c_k^\star} + \phi_k \right\}. \tag{A5}$$

Hence, Theorem 1 is proven.

1) Boyd S, Vandenberghe L. Convex Optimization. Cambridge: Cambridge University Press, 2004.