# Communication-and-Computation Efficient Split Federated Learning in Wireless Networks: Gradient Aggregation and Resource Management

Yipeng Liang, *Graduate Student Member, IEEE*, Qimei Chen, *Member, IEEE*, Rongpeng Li, *Member, IEEE*, Guangxu Zhu, *Member, IEEE*, Muhammad Kaleem Awan, and Hao Jiang, *Member, IEEE*

***Abstract*—With the prevalence of emerging artificial intelligence services in next-generation wireless edge networks, Split Federated Learning (SFL), which divides a learning model into server-side and client-side models, has emerged as an appealing technology to deal with the heavy computational burden for network edge clients. However, existing SFL frameworks would frequently upload smashed data and download gradients between the server and each client, leading to severe communication overheads. To address this issue, this work proposes a novel communication-and-computation efficient SFL framework, which allows dynamic model splitting (server- and client-side model cutting point selection) and broadcasting of aggregated smashed data gradients. We theoretically analyze the impact of the cutting point selection on the convergence rate, revealing that model splitting with a smaller client-side model size leads to a better convergence performance and vise versa. Based on the above insights, we formulate an optimization problem to minimize the model convergence rate and latency under the consideration of data privacy via a joint Cutting point selection, Communication and Computation resource allocation (CCC) strategy. To deal with the proposed mixed integer nonlinear programming optimization problem, we develop an algorithm by integrating the Double Deep Q-learning Network (DDQN) with convex optimization methods. Extensive experiments validate our theoretical analyses across various datasets, and the numerical results demonstrate the effectiveness and superiority of the proposed communication-efficient SFL compared with existing schemes, including parallel split learning and traditional SFL mechanisms.**

***Index Terms*—Communication-and-computation efficient, distributed training, edge AI, federated split learning, resource allocation.**

## I. Introduction

WITH significant advancements in Artificial Intelligence (AI), future 6G networks are envisioned to transition from "connected things" to "connected intelligence" by decentralizing AI from the central cloud to edge networks [1]. This evolution aims to support edge AI vision in the 6G networks, enabling pervasive intelligence to support emerging intelligent applications such as, eXtended Reality (XR), intelligent transportation systems, and Internet of Things (IoT) [2], [3], [4].

In this context, Distributed Collaborative Machine Learning (DCML) has emerged as a pivotal technology, particularly due to its inherent data privacy advantages [5]. Among various DCML approaches, Federated Learning (FL) and Split Learning (SL) have become particularly attractive in recent years. FL facilitates the training of a complete Machine Learning (ML) model through collaboration between a central server and distributed clients, without requiring clients to share their local data [6], [7], [8], [9]. However, while FL supports parallel model training across multiple clients, resource-constrained devices in edge networks, such as those in IoT environments, often struggle to handle the computational demands of training complete ML models [10], especially Large Language Models (LLMs). To deal with this issue, SL offers a promising solution by offloading a portion of the computational burden to the server. Specifically, SL divides a complete ML model into smaller network portions, deploying one portion on the client and the other on the server. The vanilla SL conducts model training sequentially, with the server interacting with clients one by one to update the model [11]. However, this sequential approach introduces significant latency, particularly when managing a large number of clients. Moreover, it can lead to catastrophic forgetting, which severely impacts learning performance [12]. To overcome these limitations, Split Federated Learning (SFL) combines the strengths of FL and SL, enabling parallel model training while alleviating the computational burden on clients [13]. With the recent prevalence of LLMs, SFL presents considerable potential for facilitating their training and inference at the edge of 6G networks.

Received 2 January 2025; revised 27 April 2025 and 29 June 2025; accepted 24 July 2025. Date of publication 7 August 2025; date of current version 22 December 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2024YFE0115900 and Grant 2024YFB2908001, in part by Hubei Province's Sharp Knife Breakthrough Project under Grant 2025BEA001, in part by the National Natural Science Foundation of China under Grant 62371313, in part by Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005, and in part by Guangdong Young Talent Research Project under Grant 2023TQ07A708. The associate editor coordinating the review of this article and approving it for publication was N. Shlezinger. *(Corresponding author: Qimei Chen.)*

Yipeng Liang, Qimei Chen, and Hao Jiang are with the School of Electronic Information, Wuhan University, Wuhan 430072, China (e-mail: liangyipeng@whu.edu.cn; chenqimei@whu.edu.cn; jh@whu.edu.cn).

Rongpeng Li is with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: lirongpeng@zju.edu.cn).

Guangxu Zhu is with Shenzhen Research Institute of Big Data, Shenzhen 518172, China (e-mail: gxzhu@sribd.cn).

Muhammad Kaleem Awan is with Oman Telecommunications Company Ltd., Ruwi 112, Oman (e-mail: Muhammad.Awan@omantel.om).

Digital Object Identifier 10.1109/TWC.2025.3594006

Despite the advantageous integration of SL and FL, SFL necessitates frequent exchanges of information, such as smashed data and corresponding gradients, between the server and clients to update both server-side and client-side models. Additionally, the synchronous aggregation of client-side models at the server introduces additional communication overhead. Consequently, communication overhead has become a significant challenge in SFL. Although SFL has garnered increasing attention in recent years, efforts to mitigate this communication overhead remain limited. To address this gap, this work proposes a co-design of Cutting point selection, Communication and Computation resource allocation (CCC) strategy for SFL.

### *A. Related Work*

Different from existing DCML approach such as FL, the research on SFL is still in its early stages. The SFL framework was proposed in [13], enabling parallel training and synchronous aggregation of both client-side and server-side models, which has recently garnered significant interest in various fields, including medical image segmentation [14], wireless networks [15], and emotion detection [16]. To improve the training efficiency of SFL, existing research has addressed several key challenges, including training latency, data privacy and security, non-independent and identically distributed (Non-IID) data, and communication overhead.

To reduce training latency, prior work has proposed client selection strategies and cluster-based training schemes based on the sequential training process and client heterogeneity in SFL. For example, the authors in [17] introduced a cluster-based approach that partitions clients into multiple clusters. Within each cluster, client-side models are trained and aggregated in parallel, followed by sequential training of server-side and client-side models across clusters. A resource management algorithm was proposed in [18] to minimize training latency of SFL by jointly optimizing the selection of the cutting point and the allocation of computational resources. In [19], an Fed-Pairing scheme was proposed to enhance training efficiency by pairing clients with varying computational resources.

In terms of data privacy and security, existing studies analyzed the influence of the model cutting point on information leakage and designed cutting strategies to balance performance and privacy. The authors in [21] investigated a privacy-aware SFL, where a client-based privacy approach was introduced to enhance resilience against attacks. In [22], the authors analyzed the tradeoff between privacy and energy consumption in SFL, with a particular focus on the impact of the cutting point selection.

To address the Non-IID issue, existing work introduced regularization techniques to mitigate divergence among local models. For instance, the MergeSFL framework was proposed in [23] addressed Non-IID challenges by employing feature merging and batch size regulation across different clients.

To alleviate communication overhead, the Parallel Split Learning (PSL) framework has been proposed to eliminate synchronous aggregation, thereby improving communication efficiency. For instance, the authors in [24] presented a PSL method to prevent overfitting through minibatch size selection and client layer synchronization at each client. In [25], a last-layer gradient aggregation scheme was proposed for PSL to reduce training and communication latency at the server. A joint subchannel allocation, power control, and cutting point selection strategy was further proposed, considering heterogeneous channel conditions and computing capabilities among clients. In [26], the authors explored a personalized PSL framework to address Non-IID issues, employing a bisection method with a feasibility test to optimize the tradeoff between energy consumption for computation and wireless transmission, training time, and data privacy. A local-loss-based training method was proposed in [27] to expedite the PSL training process by incorporating an auxiliary network into the client-side model, serving as the local loss function for model updates. Similarly, a communication-and-storage efficient SFL framework was explored in [20], where an auxiliary network was integrated into the client-side model to facilitate local updates. This approach maintained a single server-side model at the server, thereby eliminating the need to transmit gradients from the server.

Despite these efforts, previous approaches for SFL still suffer from communication overhead. While PSL eliminates client-side model aggregation, clients are still required to individually upload smashed data to the server and download the corresponding gradients to update both client-side and server-side models, which leads to substantial communication burden, particularly in environments with limited communication resources. Furthermore, the absence of global model aggregation in PSL may lead to model inconsistency across clients.

### *B. Motivation and Contribution*

Motivated by the above critical issue, we explore a novel communication-and-computation efficient SFL with Gradient Aggregation (SFL-GA) framework in this work. Specifically, the SFL-GA framework enables dynamic model cutting point selection based on the wireless communication environment, privacy requirements, and computation abilities of edge devices. According to the model splitting, gradients of the smashed data at the server are aggregated, and then broadcast to all the devices to effectively reduce communication overhead. For further communication-and-computation efficiency enhancement, we introduce a joint CCC strategy as shown in Fig. 1. In detail, the communication and computation resource allocation schemes influence the communication rate and computation speed, respectively. Meanwhile, the model cutting point affects communication overhead, computation burden, convergence rate, and privacy leakage. These elements collectively influence both the communication and computation latency per round, as well as the number of communication rounds required for model convergence. Ultimately, the above elements collectively determine the overall communication and computation latency.

Overall, the main contributions of this work are summarized as follows.

- **Communication-and-computation efficient SFL-GA framework:** We propose a novel SFL-GA framework,

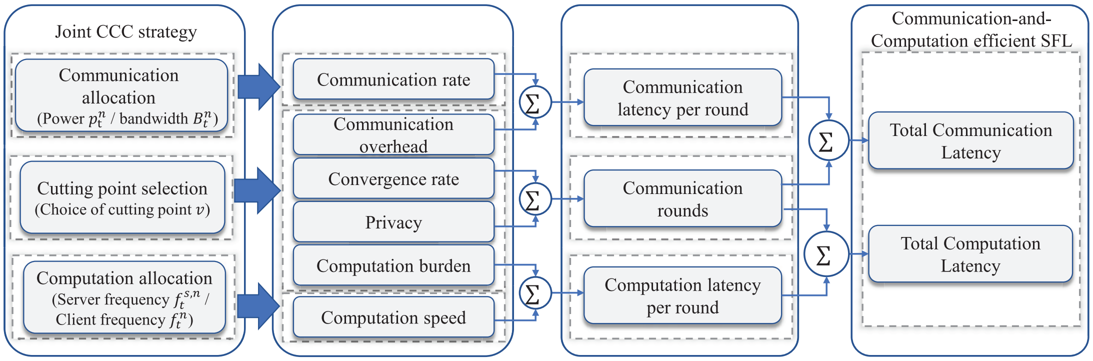

Joint CCC strategy
Communication allocation (Power $p_t^n$ / bandwidth $B_t^n$)
Cutting point selection (Choice of cutting point $v$)
Computation allocation (Server frequency $f_t^{s,n}$ / Client frequency $f_t^n$)
Communication rate
Communication overhead
Convergence rate
Privacy
Computation burden
Computation speed
Σ
Communication latency per round
Communication rounds
Computation latency per round
Communication-and-Computation efficient SFL
Total Communication Latency
Total Computation Latency

Fig. 1. Illustration of the proposed joint CCC strategy.

which enables dynamic model cutting point selection and aggregated gradient broadcasting. Specifically, the cutting point is selected to improve the communication and computation efficiency. On the other hand, we aggregate all the smashed data gradients before broadcasting instead of traditional individual gradients feedback to each client, thus alleviates the communication overhead.

- **Theoretical convergence analysis and problem formulation:** A theoretical convergence analysis of our proposed framework is conducted, which reveals that cutting a smaller client-side model leads to better convergence performance. Meanwhile, the cutting point selection also influences the privacy leakage, communication overhead, and computation latency, significantly. Based on these insights, we formulate a convergence rate and latency optimization problem under the limitation of communication and computation resources as well as the privacy leakage requirement, which is a Mixed-Integer Non-Linear Programming (MINLP).
- **Joint CCC strategy:** To address the MINLP issues, a joint CCC strategy that integrates double-deep Q-learning (DDQN) algorithm and convex optimization techniques is developed. Specifically, the problem is decomposed into two subproblems: resource allocation and cutting point selection. The resource allocation subproblem is resolved using existing convex optimization methods, while the cutting point selection subproblem is tackled with the DDQN algorithm.
- **Performance evaluation:** Numerical results are conducted to validate the theoretical analyses, and evaluate the superior performance of the proposed SFL-GA mechanism compared with benchmarks, including SFL, PSL, and FL.

The rest of this paper is organized as follows. Section II introduces the SFL-GA framework and corresponding system models. In Section III, we theoretically analyze the convergence performance for the proposed framework. In Section IV, we formulate the optimization problem and design the resource allocation strategy. Numerical results are presented in Section V followed by a conclusion in Section VI.

Throughout the paper, we use the following notation: We use $a$ to denote a scalar, $\mathbf{a}$ is a column vector, $\mathbf{A}$ is a matrix, and $|\cdot|$ represents the modulus operator. The Euclidean norm is written as $\|\cdot\|$, $\langle \mathbf{a}, \mathbf{a}' \rangle$ is the inner product of $\mathbf{a}$ and $\mathbf{a}'$, and $\mathbb{E}$ represents mathematical expectation.

TABLE I
MAIN NOTATIONS IN THIS WORK

| Symbol | Description |
|---|---|
| $\mathbf{w}^s$ | server-side model |
| $\mathbf{w}^c$ | client-side model |
| $\mathbf{s}_t^n$ | gradients of smashed data |
| $\phi\left(v\right)$ | client-side model size |
| $\mathbf{g}_t^n$ | SFL-GA's minibatch gradients |
| $\mathbf{h}_t^n$ | SFL-GA's fullbatch gradients |
| $\nabla F(\mathbf{w}_t)$ | SFL's fullbatch gradients |
| $\gamma_F^s\left(v\right)$, $\gamma_F^c\left(v\right)$ | forward propagation workload |
| $\gamma_B^s\left(v\right)$, $\gamma_B^c\left(v\right)$ | back propagation workload |
| $\mathbf{S}_t^n$ | smashed data |
| $h_t^n$ | channel gain |
| $N_0$ | Gaussian noise |
| $V$ | model layers |
| $\nabla F\left(\cdot\right)$ | gradient of $F\left(\cdot\right)$ |
| $\eta$ | learning rate |
| $\tau$ | local update |
| $X_t(v)$ | communication bit |
| $p_t^n$, $P$ | transmit power |

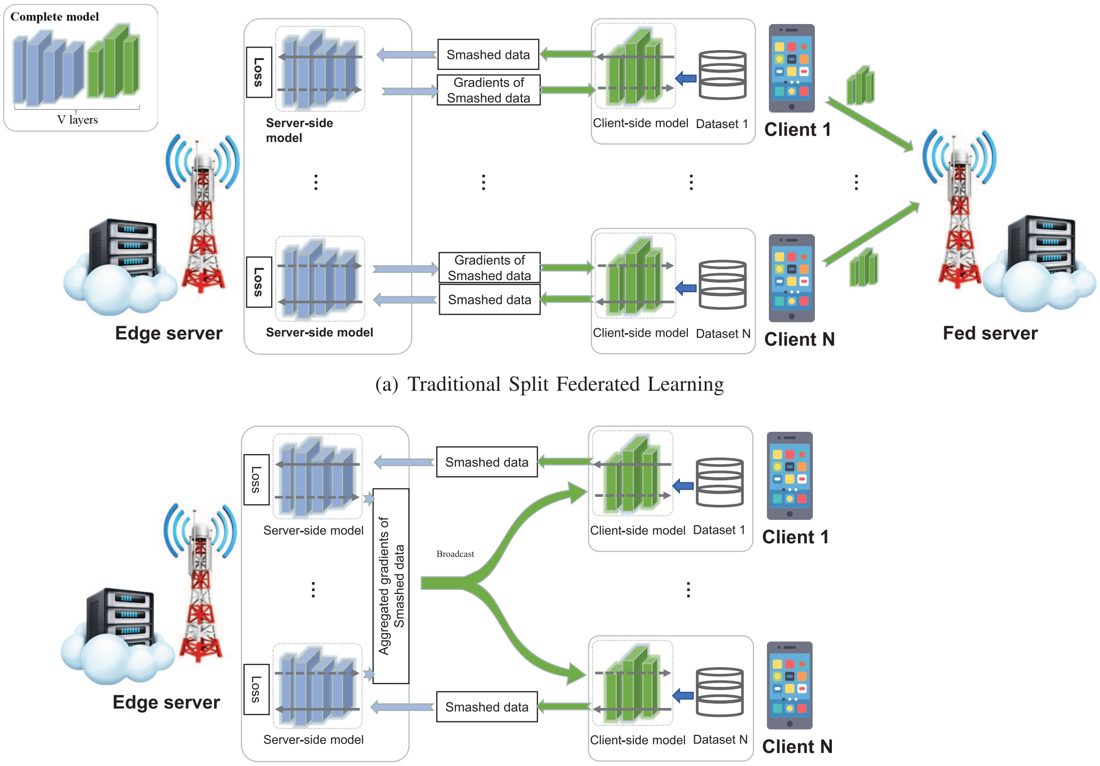
## II. SYSTEM MODEL

In this section, we first introduce a novel SFL-GA framework, and then discuss the related system model. The main notations of this work is summarized in Table I.

### A. SFL-GA Framework

As shown in Fig.2, we consider an SFL wireless network with one server and a set of clients denoted by $\mathcal{N} \triangleq \{1, 2, \ldots, N\}$. All of the clients are collaboratively training a shared ML model $\mathbf{w} \in \mathbb{R}^q$ of size $q$ and layer $V$ for a specific data analysis task, such as classification and recognition. In the SFL framework, each client splits/cuts its learning model at the $v \in \mathcal{V} \triangleq \{1, 2, 3, \ldots, V-1\}$-th layer. Thus, the entire model is divided into the client-side model $\mathbf{w}^c \in \mathbb{R}^{\phi(v)}$ and the server-side model $\mathbf{w}^s \in \mathbb{R}^{q-\phi(v)}$, deployed on the client and server for model training, respectively. Here, $\phi\left(v\right)$ denotes the client-side model size, which depends on the cutting point $v$.

Therefore, the desired ML model is collaboratively trained by the server and clients over $T$ communication rounds.



(a) Traditional Split Federated Learning

(b) Split Federated Learning with Gradient Aggregation framework

Fig. 2. Proposed SFL-GA framework and traditional SFL.

During each communication round $t \in \mathcal{T} \triangleq \{1, 2, \ldots, T\}$, the training process unfolds as follows:

1) **Smashed data generation**: All clients conduct forward propagation (FP) based on their local datasets and generate smashed data.
2) **Server-side model update**: All clients transmit the smashed data, along with the corresponding labels, to the server. Thereafter, the server performs FP and calculates the loss function in parallel based on the received smashed data and their corresponding labels. Then, the server executes back propagation (BP) to obtain the updated server-side models and the smashed data gradients.
3) **Server-side gradients aggregation**: The server aggregates these updated server-side models into a new global server-side model, as well as aggregates the gradients of the smashed data.
4) **Gradient broadcast**: The server broadcasts the aggregated gradients of the smashed data to all the clients. Unlike the traditional SFL, where the gradients of smashed data are transmitted to corresponding clients separately, our proposed framework effectively reduces communication overhead.
5) **Client-side model BP**: The clients perform client-side model BP and update their client-side models based on the received gradients. Since the client-side models have identical weight parameters and are updated based on the same gradients, synchronous client-side model aggregation is eliminated, leading to a further communication overhead reduction.

The core concept of the proposed framework is to aggregate the gradients of smashed data from all the clients and then broadcast the aggregated gradients to these clients. Compared with the traditional SFL mechanism, it can significantly reduced the communication overhead. Our SFL-GA framework fundamentally differs from over-the-air FL [9] in two key aspects: (1) Communication paradigm: Instead of requiring simultaneous parameter aggregation over shared uplink bandwidth as in [9], SFL-GA employs dedicated frequency bands for each client to transmit their distinct intermediate representations (smashed data). This orthogonal resource allocation eliminates inter-client interference and relaxes strict synchronization requirements. (2) Model updating mechanism: The server-side sub-models undergo gradient-based update using client-specific smashed data prior to global aggregation, whereas conventional over-the-air FL directly aggregates global model with aggregated parameters. This architectural innovation allows our framework to achieve finer-grained training while maintaining spectral efficiency.

### B. SFL-GA Model

Denote that the complete ML model $\mathbf{w}$ is partitioned into the server-side model $\mathbf{w}^s$ and the client-side model $\mathbf{w}^c$, denoted as $\mathbf{w} = [\mathbf{w}^s; \mathbf{w}^c]$. Moreover, each client $n$ is assumed to possess a local dataset $\mathcal{D}^n$ with size of $D^n$.

For communication round $t$, all clients perform client-side model FP in parallel based on their own client-side models $\mathbf{w}^c_{t-1}$. Specifically, each client $n$ takes a mini-batch of samples

$\xi^n$, randomly chosen from $\mathcal{D}^n$, as the input for $\mathbf{w}_{t-1}^c$ to obtain the smashed data, which can be expressed as

$$\mathbf{S}_t^n = \ell\left(\mathbf{w}_{t-1}^c; \xi^n\right), \tag{1}$$

where $\ell$ is the the client-side function mapping from the input data to the smashed data. Then, the server collects the smashed data along with their corresponding labels from all the clients for the server-side model update and aggregation. Specifically, the server first performs FP to calculate the loss for each device by inputting the corresponding smashed data into the server-side model $\mathbf{w}_{t-1}^s$. Therefore, the local loss with respect to the complete model of device $n$ can be expressed as

$$\begin{aligned} F\left(\mathbf{w}_{t-1}^n\right) &= F\left(\mathbf{w}_{t-1}^{s,n}, \mathbf{w}_{t-1}^c; \xi^n\right) \\ &= \frac{1}{D^n} \sum_{(x_j, y_j)\in\xi^n} f\left(\mathbf{w}_{t-1}^{s,n}, \mathbf{w}_{t-1}^c; (x_j, y_j)\right), \end{aligned} \tag{2}$$

where $f$ is the loss function. $(x_j, y_j)$ is the $j$-th sample of $\xi^n$ with data $x_j$ and label $y_j$. Subsequently, after server-side model BP, the server obtains the gradients of the loss function for client $n$'s model update, which is given by

$$\mathbf{g}_{t-1}^{s,n} = \nabla_{\mathbf{w}^s} F\left(\mathbf{w}_{t-1}^{s,n}, \mathbf{w}_{t-1}^c; \xi^n\right). \tag{3}$$

Meanwhile, the gradients of the smashed data of each device $n$ are also computed as

$$\mathbf{s}_t^n = \nabla \mathbf{S}_t^n. \tag{4}$$

Different from the existing SFL in [13] where the server sends $\mathbf{s}_t^n$ to the corresponding client $n$, the server in this work broadcasts the aggregated gradients of smashed data to all clients, which is given by

$$\mathbf{s}_t = \sum_{n=1}^{N} \rho^n \mathbf{s}_t^n, \tag{5}$$

where $\rho^n = \frac{D^n}{D}$ with $D = \sum_{n=1}^N D^n$ being the total size of datasets across clients.

After receiving $\mathbf{s}_t$ from the server, both the server and the clients can update the server- and client-side models, respectively. Specifically, each client first computes the gradient of its client-side model, denoted by $\mathbf{g}_t^c$, based on $\mathbf{s}_t$. The client-side model is then updated via gradient descent. Consequently, the complete ML model $\mathbf{w}_t^n$ of the client $n$ in the $t$-th communication round can be updated through multiple epochs, described as follows

$$\mathbf{w}_t^n = \begin{bmatrix} \mathbf{w}_t^{s,n} \\ \mathbf{w}_t^c \end{bmatrix} = \begin{bmatrix} \mathbf{w}_{t-1}^s \\ \mathbf{w}_{t-1}^c \end{bmatrix} - \eta \sum_{i=1}^{\tau} \begin{bmatrix} \mathbf{g}_{t-1,i}^{s,n} \\ \mathbf{g}_{t-1,i}^c \end{bmatrix}, \tag{6}$$

where $\eta$ is the learning rate. $\tau$ is the number of local epochs. Since the model $\mathbf{w}_t^c$ at each client is updated based on the same gradient $\mathbf{g}_{t-1}^c$ and the same parameters $\mathbf{w}_{t-1}^c$, each client attains the same model parameters after updating. Therefore, the proposed SFL-GA eliminates the necessity for client-side model aggregation as [13]. Consequently, we only need to aggregate the server-side model at the server, which is given by

$$\mathbf{w}_t^s = \sum_{n=1}^{N} \rho^n \mathbf{w}_t^{s,n}. \tag{7}$$

As a result, with the complete global model $\mathbf{w}_t = [\mathbf{w}_t^s, \mathbf{w}_t^c]$ at the $t$-th communication round, the global loss function can be represented as

$$F\left(\mathbf{w}_t\right) = \sum_{n=1}^{N} \rho^n F\left(\mathbf{w}_t^n\right). \tag{8}$$

Assuming the global model converges after $T$ communication rounds, the training objective of Eq. (8) is to find a minimal global model $\mathbf{w}^* = [\mathbf{w}^{s*}; \mathbf{w}^{c*}]$ that satisfies

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} F\left(\mathbf{w}_T\right). \tag{9}$$

### C. Communication Model

In this subsection, we introduce the communication model in this work. We assume that the channel remains constant during a given communication round but may vary across different rounds. For an arbitrary communication round $t$, the communication process of each client includes an uplink phase for uploading smashed data and a downlink phase for broadcasting gradients.

Regarding the uplink communication, the total available bandwidth $B$ is divided into multiple orthogonal subchannels to transmit the smashed data and labels from each client to the server. Therefore, the achievable data rate of device $n$ can be expressed as

$$r_t^{n,U} = B_t^n \log_2\left(1 + \frac{p_t^n g_t^n}{B_t^n N_0}\right), \tag{10}$$

where $B_t^n$ is the bandwidth allocated to client $n$. $g_t^n$ is the channel gain between the server and device $n$. $p_t^n$ and $N_0$ denote the transmit power of device $n$ and the thermal noise spectrum density, respectively.

For the downlink communication, the server broadcasts the aggregated gradients (Eq. (5)) through downlink multicast transmission, a widely adopted approach in wireless networks such as Wi-Fi and 5G systems. Since this transmission occupies the entire bandwidth, the achievable rate for device $n$ can be represented as

$$r_t^{n,D} = B \log_2\left(1 + \frac{P g_t^n}{B N_0}\right), \tag{11}$$

where $P$ denotes the transmit power of server.

Note that the size of smashed data and corresponding gradients depend on the cutting point $v$. Therefore, the latency of uplink and downlink transmission can be respectively written as

$$l_t^{n,U} = \frac{X_t\left(v\right)}{r_t^{n,U}}, \tag{12}$$

$$l_t^{n,D} = \frac{X_t\left(v\right)}{r_t^{n,D}}, \tag{13}$$

where $X_t\left(v\right)$ is the communication bit size of smashed data (and its gradient) related to cutting point $v$.

### D. Computation Model

In this subsection, we present the computation model of the proposed SFL-GA. During communication round $t$, SFL initiates with client-side model FP. Let $\gamma_F^n(v)$ denote the computation workload (in FLOPs) of client $n$ for performing FP with one data sample [28], [29]. Therefore, the latency for client-side model FP is given by

$$l_t^{n,F} = \frac{D^n \gamma_F^n(v)}{f_t^n}, \tag{14}$$

where $f_t^n$ denotes the central processing unit (CPU) resource of device $n$.

Subsequently, SFL performs FP and BP to update the server-side model at the server. Let $\gamma_F^s(v)$ and $\gamma_B^s(v)$ denote the computation workload of the server for performing FP and BP with one data sample, respectively. Therefore, the latency for both server-side model FP and BP is given by

$$l_t^{n,s} = \frac{D^n \left(\gamma_F^s(v) + \gamma_B^s(v)\right)}{f_t^{s,n}}, \tag{15}$$

where $f_t^{s,n}$ is the CPU computation resource of server that allocated to server-side model of client $n$.

Finally, SFL-GA conducts client-side model BP to update the model at each client. Let $\gamma_B^n(v)$ denote the computation workload of client $n$ for performing BP with one data sample. Then, we have

$$l_t^{n,B} = \frac{D^n \gamma_B^n(v)}{f_t^n}. \tag{16}$$

### E. Privacy Model

In the proposed SFL framework, the privacy concerns arise from the transmission of smashed data between clients and the server. Existing research has demonstrated the significant impact of cutting point on privacy leakage [21], [31]. Specifically, private input data can be reconstructed from smashed data via inversion attacks. Prior studies have demonstrated that increasing the number of layers on the client side can significantly reduce the effectiveness of such attacks by limiting the amount of information exposed to the server [22], [30]. However, accurately characterizing the relationship between cutting point selection and the extent of data reconstruction remains a challenge. To quantify this relationship, this work adopts the privacy model proposed in [26], which defines the following privacy constraint

$$\log\left(1 + \frac{\phi_t(v)}{q}\right) \geq \epsilon, \forall t, \tag{17}$$

where $q$ is the full model size and $\epsilon$ serves as the privacy protection threshold. Eq. (17) guarantees the chosen cutting point in each communication round maintains the required privacy preservation level. Note that larger value of $\epsilon$ corresponds to stronger privacy protection.

## III. Theoretical Analysis and Performance Evaluation

In this section, we present a theoretical analysis of the proposed SFL-GA framework. We start by analyzing the convergence of SFL-GA, followed by a discussion of its complexity and scalability.

### A. Convergence Analysis

To facilitate analysis, we denote the SFL-GA's mini-batch (stochastic) gradient and full-batch gradient of the loss function, respectively, as

$$\mathbf{g}_t^n = \left[\mathbf{g}_t^{s,n}; \mathbf{g}_t^c\right], \tag{18}$$

and

$$\mathbf{h}_t^n = \left[\nabla_{\mathbf{w}^s} F\left(\mathbf{w}_t^n\right); \nabla_{\mathbf{w}^c} \tilde{F}\left(\mathbf{w}_t\right)\right], \tag{19}$$

where $\nabla_{\mathbf{w}^c} \tilde{F}\left(\mathbf{w}_t\right)$ is the aggregated full-batch gradient of the client-side models.

Recall that in traditional SFL, each client $n$ generates the gradient of the client-side model $\nabla_{\mathbf{w}^c} F\left(\mathbf{w}_t^n\right)$ based on its own gradient of smashed data $\mathbf{s}_t^n$, instead of the aggregated one in Eq. (6), to update $\mathbf{w}_t^c$. This discrepancy may affect the convergence performance. Therefore, we denote the SFL' full-batch gradients as

$$\nabla F(\mathbf{w}_t^n) = \left[\nabla_{\mathbf{w}^s} F\left(\mathbf{w}_t^n\right); \nabla_{\mathbf{w}^c} F\left(\mathbf{w}_t^n\right)\right], \tag{20}$$

Accordingly, we further define the the SFL's global gradient as

$$\begin{aligned} \nabla F(\mathbf{w}_t) &= \sum_{n=1}^{N} \nabla F(\mathbf{w}_t^n) \\ &= \left[\sum_{n=1}^{N} \nabla_{\mathbf{w}^s} F\left(\mathbf{w}_t^n\right); \sum_{n=1}^{N} \nabla_{\mathbf{w}^c} F\left(\mathbf{w}_t^n\right)\right]. \end{aligned} \tag{21}$$

To begin with, we introduce the following assumptions, which are commonly adopted in existing works, such as [15] and [27].

*Assumption 1 (L-smoothness):* For any $\mathbf{w}, \mathbf{v}$, the loss function is either continuously differentiable or Lipschitz continuous with a non-negative Lipschitz constant $L \geq 0$, which can be formulated as

$$F(\mathbf{v}) - F(\mathbf{w}) \leq (\mathbf{v} - \mathbf{w})^\top \nabla F(\mathbf{w}) + \frac{L}{2}\|\mathbf{v} - \mathbf{w}\|^2. \tag{22}$$

*Assumption 2 (Unbiased Gradient and Bounded Variance):* For each client, the stochastic gradient is unbiased, i.e., $\mathbb{E}\left(\mathbf{g}_t^{s,n}\right) = \nabla_{\mathbf{w}^s} F\left(\mathbf{w}_t^n\right)$ and $\mathbb{E}\left(\mathbf{g}_t^c\right) = \nabla_{\mathbf{w}^c} \tilde{F}\left(\mathbf{w}_t^n\right)$. Moreover, the variance of stochastic gradients of each client is bounded by

$$\mathbb{E}\left(\|\mathbf{g}_t^n - \nabla F(\mathbf{w}_t^n)\|^2\right) \leq \sigma^2. \tag{23}$$

Note that in both the SFL-GA framework and the traditional SFL framework, the server-side model is updated using the same smashed data. The divergence between the two frameworks arises only in the updating of the client-side model: SFL-GA employs aggregated gradients of the smashed data, while the traditional SFL framework uses the gradients from each client's individual smashed data. This divergence influences the convergence behavior, and accurately characterizing this discrepancy is challenging. Nevertheless, we can observe that this discrepancy is significantly related to the size of the client-side model. Specifically, as the size of the client-side model increases, the differences between the client-side models in the SFL-GA framework and the traditional SFL framework become more pronounced, thereby exerting a greater impact on the convergence of both frameworks. Considering the convergence discrepancy related to the size of the client-side model, we introduce the following assumption.

*Assumption 4. (Bounding the Difference Between SFL Gradient and SFL-GA Gradient):* When the client-side model holds a size of $\phi_t(v)$ in $t$-th round, the expected gradient variance between SFL and SFL-GA is bounded by

$$\begin{aligned}
&\mathbb{E}\left(\|\mathbf{h}_t^n - \nabla F(\mathbf{w}_t^n)\|^2\right)\\
&= \mathbb{E}\left(\|\nabla_{\mathbf{w}^s} F(\mathbf{w}_t^n) - \nabla_{\mathbf{w}^s} F(\mathbf{w}_t^n)\|^2\right.\\
&\quad \left. + \left\|\nabla_{\mathbf{w}^s} \tilde{F}(\mathbf{w}_t) - \nabla_{\mathbf{w}^c} F(\mathbf{w}_t^n)\right\|^2\right)\\
&= \mathbb{E}\left\|\nabla_{\mathbf{w}^c} \tilde{F}(\mathbf{w}_t) - \nabla_{\mathbf{w}^c} F(\mathbf{w}_t^n)\right\|^2\\
&\leq \Gamma(\phi_t(v)), \qquad (24)
\end{aligned}$$

where $\Gamma(\phi_t(v)) = \frac{k\phi_t(v)}{q}$ is a monotonic non-decreasing function of $\phi_t(v)$ with $k$ being a positive constant. Assumption 4 indicates that the gradient difference between vanilla SFL and SFL-GA is related to the client-side model size $\phi_t(v)$ and is bounded. Specifically, a smaller client-side model size $\phi_t(v)$ results in a smaller gradient difference.

With above assumptions, we introduce Lemma 1 to demonstrate the upper bound of the improvement of the global loss function in each round.

*Lemma 1:* When the learning rate $\eta$ satisfies $0 \leq 2L^2\eta^2\tau(\tau-1) \leq \frac{1}{5}$ in the $t$-th communication round, the improvement of the global loss function is bounded by

$$\begin{aligned}
\mathbb{E}\left(F(\mathbf{w}_{t+1}) - F(\mathbf{w}_t)\right) &\leq -\frac{\eta\tau}{4}\|\nabla F(\mathbf{w}_t)\|^2 + \eta\tau\Gamma(\phi_t(v))\\
&+ L\eta^2\tau\sigma^2\sum_{n=1}^{N}(\rho^n)^2 + \frac{5L^2\eta^3\sigma^2\tau(\tau-1)}{4}. \qquad (25)
\end{aligned}$$

*Proof:* Please refer to Appendix. □

Based on Lemma 1, we further introduce the following Theorem to show the upper bound of the average squared gradient norm, which illustrates the convergence performance.

*Theorem 1:* Under the condition of $0 \leq 2L^2\eta^2\tau(\tau-1) \leq \frac{1}{5}$, the average squared gradient norm after T communication rounds is bounded by

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\mathbf{w}_t)\|^2 &\leq \underbrace{\frac{4(F(\mathbf{w}_T) - F^*)}{\eta\tau T}}_{\text{Effects of initialization}} + \underbrace{\frac{4}{T}\sum_{t=1}^{T}\Gamma(\phi_t(v))}_{\text{Effects of cutting point}}\\
&\underbrace{+ 4L\eta\sigma^2\sum_{n=1}^{N}(\rho^n)^2 + 5L^2\eta^2\sigma^2(\tau-1)}_{\text{Effects of gradient variance}}. \qquad (26)
\end{aligned}$$

*Remark 1:* It is observed that Eq. (26) is influenced by the initialization, gradient variance, as well as the cutting point. While the proposed framework effectively reduces communication overhead, the cutting point selection may adversely impact convergence performance, thereby requiring more communication rounds to achieve convergence. In particular, reducing the impact of the cutting point during the training process can lower the bound of Eq. (26), suggesting that a smaller client-side model size enhances convergence performance. At the same time, the cutting point also affects communication overhead, computational burden, and privacy leakage. This observation motivates the co-design of CCC for achieving a communication-and-computation efficient SFL.

### B. Convergence Rate and Scalability

To facilitate the analysis of convergence rate, we let $\rho^n = \frac{1}{N}$. Therefore, Eq. (26) can be reformulated as

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\mathbf{w}_t)\|^2 &\leq \frac{4(F(\mathbf{w}_T) - F^*)}{\eta\tau T} + \frac{4L\eta\sigma^2}{N}\\
&+ 5L^2\eta^2\sigma^2(\tau-1) + \frac{4}{T}\sum_{t=1}^{T}\Gamma(\phi_t(v)). \qquad (27)
\end{aligned}$$

If the learning rate satisfies $\eta = \sqrt{\frac{N}{\tau T}}$ [7], the convergence rate of SFL-GA is given by

$$\begin{aligned}
&\mathbb{E}\left(\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(\mathbf{w}_t)\|^2\right)\\
&\leq \mathcal{O}\left(\frac{1}{\sqrt{\tau NT}} + \frac{\sigma^2}{\sqrt{\tau NT}} + \frac{N(\tau-1)\sigma^2}{\tau T} + M\right), \qquad (28)
\end{aligned}$$

where $M = \frac{4}{T}\sum_{t=1}^{T}\Gamma(\phi_t(v))$. If $M$ is bounded, the convergence rate of the proposed SGL-GA is given by $\mathcal{O}\left(\frac{1}{\sqrt{\tau NT}}\right) + \mathcal{O}\left(\frac{\sigma^2 N(\tau-1)}{\tau T}\right)$.

To evaluate the scalability of our proposed SFL-GA, we focus on the convergence behavior in Eq. (28) with respect to the number of clients $N$. From Eq. (28), the first two terms decrease with the increment of $N$. It indicates that the convergence rate benefits from a larger number of clients through improving the average updates performance. On the other aspect, the third term of Eq. (28) increases linearly with $N$, which suggests a potential risk in convergence rate. Consequently, the convergence behavior initially improves but eventually deteriorates as $N$ continues to grow, which is consistent with traditional FL due to the effect of local gradient variance from SGD.

## IV. Problem Formulation and Resource Optimization

In this section, we formulate a convergence rate and latency optimization problem based on the system models and convergence results. Subsequently, a joint CCC strategy that solved by the DDQN algorithm and convex optimization is designed.

### A. Problem Formulation

According to the latency analysis in Section II, the gradients of smashed data are aggregated before broadcasting to all clients. Therefore, the total latency for communication round $t$ is derived by

$$l_t = \max_n\{l_t^{n,U} + l_t^{n,F} + l_t^{n,s}\} + \max_n\{l_t^{n,D} + l_t^{n,B}\}. \qquad (29)$$

Apparently, the wireless channel conditions and heterogeneous computing capabilities of clients may significantly affect the latency of SFL, while the cutting point influences the convergence rate. In this work, we aim to realize communication-and-computation efficient SFL. To this end, we formulate a convergence rate and latency optimization problem while considering resource and privacy constraints.

Let $\boldsymbol{f}^s = \left[f_1^{1,s}, \ldots, f_T^{N,s}\right]$, $\boldsymbol{f}^c = \left[f_1^{1,c}, \ldots, f_T^{N,c}\right]$, $\boldsymbol{p} = \left[p_1^1, \ldots, p_T^N\right]$ and $\boldsymbol{B} = \left[B_1^1, \ldots, B_T^N\right]$. The communication-and-computation efficient problem can be formulated as

$$\mathcal{P}1: \min_{\{v, \boldsymbol{f}^s, \boldsymbol{f}^c, \boldsymbol{p}, \boldsymbol{B}\}} \sum_{t=1}^{T} \left(w\Gamma\left(\phi_t\left(v\right)\right) + l_t\left(v\right)\right), \tag{30}$$

$$\text{s.t.} \quad v \in \mathcal{V}, \tag{30a}$$

$$0 \le f_t^n \le f_{\max}^{n,c}, \ \forall n, t, \tag{30b}$$

$$0 \le p_t^n \le p_{\max}^n, \forall n, t, \tag{30c}$$

$$\sum_{n=1}^{N} f_t^{n,s} \le f_{\max}^s, \ \forall t, \tag{30d}$$

$$\log\left(1 + \frac{\phi_t\left(v\right)}{q}\right) \ge \epsilon, \forall t, \tag{30e}$$

$$\sum_{n=1}^{N} B_t^n \le B, \ \forall t, \tag{30f}$$

where $w$ is a weighted factor to balance the convergence rate and latency. (30a) is the layer constraint of ML model, $f_{\max}^{n,c}$ and $p_{\max}^s$ in (30b) and (30c) are the the maximum computation resource and transmit power of each client, respectively, $f_{\max}^{n,s}$ in (30d) denotes the maximum computation resource constraint for model update at the server. (30e) is the constraint for privacy protection, (30f) ensures that the total bandwidth for all clients doesn't exceed the available bandwidth $B$.

$\mathcal{P}1$ is a min-max MINLP problem, which is coupling of cutting point selection and resource allocation. The major difficulty of solving $\mathcal{P}1$ lies in the efficient determination of the cutting point selection under dynamic fading channels and heterogeneous capabilities of clients. Conventional optimization methods often rely on iterative adjustments or exhaustive search, leading to high computational complexity. In contrast, DDQN is particularly appealing in handling discrete decision-making problems with finite action spaces. Therefore, a joint CCC strategy is developed based on DDQN algorithm.

### B. Joint CCC Strategy

To address the min-max issue of $\mathcal{P}1$, we introduce auxiliary variables $\chi_t$ and $\psi_t$. Then, $\mathcal{P}1$ can be equivalently reformulated as

$$\mathcal{P}2: \min_{\{\boldsymbol{v}, \boldsymbol{f}^s, \boldsymbol{f}^c, \boldsymbol{p}, \boldsymbol{B}\}} \sum_{t=1}^{T} \left(w\Gamma\left(\phi_t\left(v\right)\right) + \chi_t + \psi_t\right), \tag{31}$$

$$\text{s.t.} \quad (30a), (30b), (30c), (30d), (30e), (30f), \tag{31a}$$

$$l_t^{n,U} + l_t^{n,F} + l_t^{n,s} \le \chi_t, \tag{31b}$$

$$l_t^{n,D} + l_t^{n,B} \le \psi_t \ \forall t. \tag{31c}$$

Nonetheless, $\mathcal{P}2$ remains an NP-hard problem. Intuitively, it can be divided into two subproblems: resource allocation and cutting point selection. Therefore, we can address $\mathcal{P}2$ by jointly solving these subproblems. In what follows, we present the optimization methods for the resource allocation and cutting point selection subproblems, respectively.

*1) Resource Allocation:* Given the optimal cutting point selection variables $\boldsymbol{v}^*$, the resource allocation subproblem is independent to the communication rounds. Therefore, it can be decomposed into $T$ separate subproblems, each addressed independently. Without loss of generality, the resource allocation subproblem for communication roud $t$ is formulated as

$$\mathcal{P}2.1: \min_{\{\boldsymbol{f}^s, \boldsymbol{f}^c, \boldsymbol{p}, \boldsymbol{B}\}} \chi_t + \psi_t, \tag{32}$$

subject to (30b), (30c), (30d), (30f), (31b), (31c).

It can be easily shown that $\mathcal{P}2.1$ is a convex optimization problem. Therefore, it can be resolved by existing optimization technique (e.g., CVX).

*2) Cutting Point Selection:* Given the optimal resource allocation $\boldsymbol{f}^*$, $\boldsymbol{p}^*$ and $\boldsymbol{B}^*$. The cutting point selection subproblem can be expressed as

$$\mathcal{P}2.2: \min_{\{\boldsymbol{v}\}} \sum_{t=1}^{T} \left(w\Gamma\left(\phi_t\left(v\right)\right) + \chi_t + \psi_t\right), \tag{33}$$

subject to (30a), (30e), (31b), (31c).

Due to the integer variables, $\mathcal{P}2.2$ is an integer programming. The DDQN algorithm has been regarded as an efficient method to tackle the integer programming problem [32], [33], [34], which thus is adopted to solve $\mathcal{P}2.2$. Before using the DDQN algorithm, the subproblem $\mathcal{P}2.2$ needs to be transformed into a Markov Decision Process (MDP) problem with a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where $\mathcal{S}$, $\mathcal{A}$, $\mathcal{P}$ and $\mathcal{R}$ are the state space, action space, state transition probability, and reward, respectively. Specifically, the corresponding elements in the tuple are presented as follows.

- State space $\mathcal{S}$. It is observed the channel gain at the beginning of communication round $t$. Therefore, we define the state space in communication round $t$ as
$$\mathbf{s}_t = \left\{ g_t^n, \sum_{i=1}^{t-1} \left(\Gamma\left(\phi_i\left(v\right)\right) + \chi_i + \psi_i\right) \right\}_{\forall n}. \tag{34}$$
- Action space $\mathcal{A}$. Since the cutting point $v$ is selected from $\mathcal{V}$ that composed of $V-1$ layers in each communication round, we define the state space in communication round $t$ as $\mathbf{a}_t = \{1, 2, \ldots, V-1\}$.
- State transition probability $\mathcal{P}$. Let $\mathcal{P}\left(s_{t-1}|s_t, a_t\right)$ be the probability of transitioning from state $s_{t-1}$ to state $s_t$ under action $a_t$.
- Reward $\mathcal{R}$. Reward $r_t$ is designed to evaluate the quality of a learning policy under state-action pair $(\mathbf{s}_t, \mathbf{a}_t)$, which is defined as
$$r_t\left(\mathbf{s}_t, \mathbf{a}_t\right) = \begin{cases} \Gamma\left(\phi_t\left(v\right)\right) + \chi_t + \psi_t, & log\left(1 + \frac{\phi_t(v)}{q}\right) \ge \epsilon, \\ C, & \text{otherwise.} \end{cases} \tag{35}$$
where $C$ is a sufficiently large value used as a penalty.

Based on the tuple above, we further define the cumulative discounted reward for $t$-th communication round as

$$U_t = \lim_{T \to +\infty} \sum_{i=t}^{T} \gamma^{i-t} r_i \left(\mathbf{s}_i, \boldsymbol{a}_i\right), \tag{36}$$

where $\gamma \in (0, 1]$ is the discount factor for weighting future rewards. Then the MDP problem is formulated, aiming to find an optimal policy $\pi^*$ that maximizes the expected long-term discounted rewards, i.e.,

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[U_t\right]. \tag{37}$$

To tackle the MDP problem, the DDQN algorithm defines an agent that interacts with the environment to choose better actions based on a certain policy $\pi$ for maximizing long-term discounted rewards. To this end, the DDQN introduces a state-action function $Q^{\pi}\left(\mathbf{s}_t, \mathbf{a}_t; \boldsymbol{\theta}\right)$ for a certain policy $\pi$ as the expected future long-term reward for a state-action pair $(\mathbf{s}_t, \mathbf{a}_t)$, which is presented by

$$Q^{\pi}\left(\mathbf{s}_t, \boldsymbol{a}_t; \boldsymbol{\theta}\right) = \mathbb{E}_{\pi}\left[U_t | \mathbf{s}_t, \boldsymbol{a}_t\right], \tag{38}$$

where $\boldsymbol{\theta}$ is the parameter vector of the Q-network.

To find the optimal policy $\pi^*$, it is equivalent to obtaining the optimal action-value function $Q^*\left(\mathbf{s}_t, \boldsymbol{a}_t; \boldsymbol{\theta}\right)$, which can be achieved through the Bellman equation as

$$Q^{^*}\left(\mathbf{s}_t, \boldsymbol{a}_t; \boldsymbol{\theta}\right) = r_t + \gamma \max_{\boldsymbol{a}_{t+1}} Q^{^*}\left(\mathbf{s}_{t+1}, \boldsymbol{a}_{t+1}; \boldsymbol{\theta}\right). \tag{39}$$

The optimal action-value function $Q^{^*}$ can be obtained by optimizing the parameter vector $\boldsymbol{\theta}$ of the Q-network. To this end, the DDQN algorithm optimizes the parameter $\boldsymbol{\theta}$ by minimizing the following loss function

$$\mathcal{L}(\boldsymbol{\theta}) = \left(r_t + \gamma \max_{\boldsymbol{a}_{t+1}} Q\left(\mathbf{s}_{t+1}, \arg\max_{\boldsymbol{a}_{t+1}} Q\left(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}; \boldsymbol{\theta}\right); \hat{\boldsymbol{\theta}}\right)\right.$$
$$\left. - Q(\mathbf{s}_t, \mathbf{a}_t; \boldsymbol{\theta})\right)^2. \tag{40}$$

Then, a gradient descent method is employed to minimize the loss function $\mathcal{L}(\boldsymbol{\theta})$. As a result, the optimal policy is achieved by obtaining the optimal parameter vector $\boldsymbol{\theta}^*$.

Following the proposed optimization methods above, we now introduce a joint cutting point control and resource allocation strategy for $\mathcal{P}1$. Specifically, we employ the DDQN to optimize the cutting point selection subproblem by reformulating $\mathcal{P}2.2$ as a MDP. During each exploration in the DDQN algorithm, the agent takes an action to acquire rewards as defined in Eq. (34) where $\chi_t$ and $\psi_t$ are obtained through resolving $\mathcal{P}2.1$ by convex optimization technique.

The detailed procedure is shown in Algorithm 1.

### C. Complexity Analysis of Algorithm 1

As described previously, the proposed Algorithm 1 integrates convex optimization method and DDQN algorithm, where the agent needs to resolve $\mathcal{P}2.1$ before obtaining a reward. Therefore, we first analyze the complexity of solving $\mathcal{P}2.1$. Then, the complexity of Algorithm 1 is further presented. Note that $\mathcal{P}2.1$ is a convex optimization problem, which can be resolved with a polynomial complexity, e.g., $\mathcal{O}(N^{3.5})$ [36]. The DDQN network is represented by an fully-connected NN (FCNN) in this work. Generally, the computational complexity of an FCNN is $\mathcal{O}\left(\sum_k \left(2I_k - 1\right) I_{k+1}\right)$ [37], where $k \in [0, K]$ denotes the layer index and $I_k$ represents the neuron number of hidden layers. Let $M$ be the total number of episodes and $T$ be the number of steps per episode. Then, the overall computational complexity of Algorithm 1 is $\mathcal{O}\left(TM\left(\sum_k \left(2I_k - 1\right) I_{k+1}\right) N^{3.5}\right)$. The convergence of Algorithm 1 is contingent upon Q-network training. Given that Q-network updates exhibit approximately constant latency per episode, the algorithm's overall convergence latency is fundamentally governed by the convergence characteristics of the Q-network.

**Algorithm 1** The Joint CCC Strategy for $\mathcal{P}1$

```
Input: Initialize parameter vector of Q-networks $\boldsymbol{\theta}^1$;
       Initialize the experience buffer; Maximum episode
       number $L_{\max}$.
for episode $\ell = 1$ to $L_{\max}$ do
    Reset the initial state $\mathbf{s}_1$;
    for communication round $t = 1$ to $T$ do
        DDQN agent selects discrete action $\boldsymbol{a}_t$ based on
          the observed state $\mathbf{s}_t$;
        Obtain the optimal $f_t^{n,c*}$, $f_t^{n,s*}$, $p_t^{n*}$ and $B_t^{n*}$ by
          resolving $\mathcal{P}2.1$;
        Calculate the reward $r_t$ with $f_t^{n,c*}$, $f_t^{n,s*}$, $p_t^{n*}$
          and $B_t^{n*}$;
        Observe the next $\mathbf{s}_{t+1}$;
        Add transition $(\mathbf{s}_t, \boldsymbol{a}_t, r_t, \mathbf{s}_{t+1})$ to the replay
          buffer;
        Sample a minibatch from the replay buffer;
        Update DQN network by the gradient descent
          method: $\boldsymbol{\theta}^{t+1} \leftarrow \boldsymbol{\theta}^t$;
    end
end
```

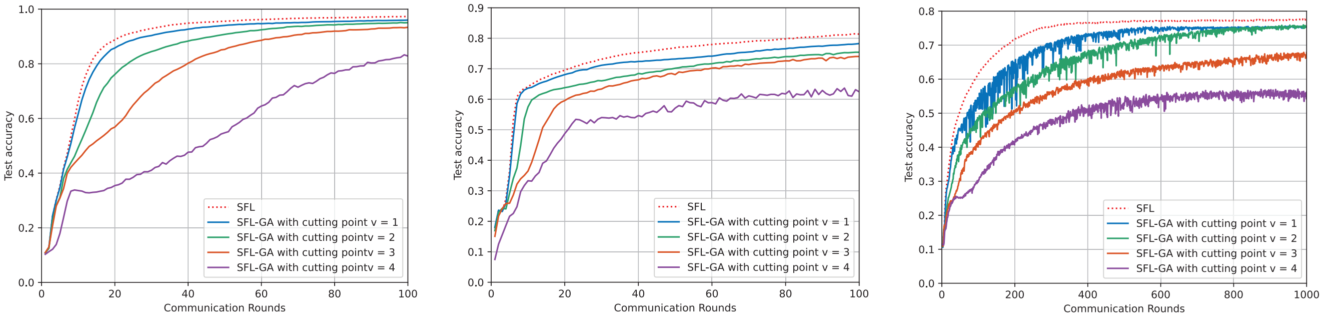
## V. Simulation Results

In this section, we provide simulation results to validate the effectiveness of proposed SFL-GA and the efficiency of developed algorithm design.

### A. Experiment Setup

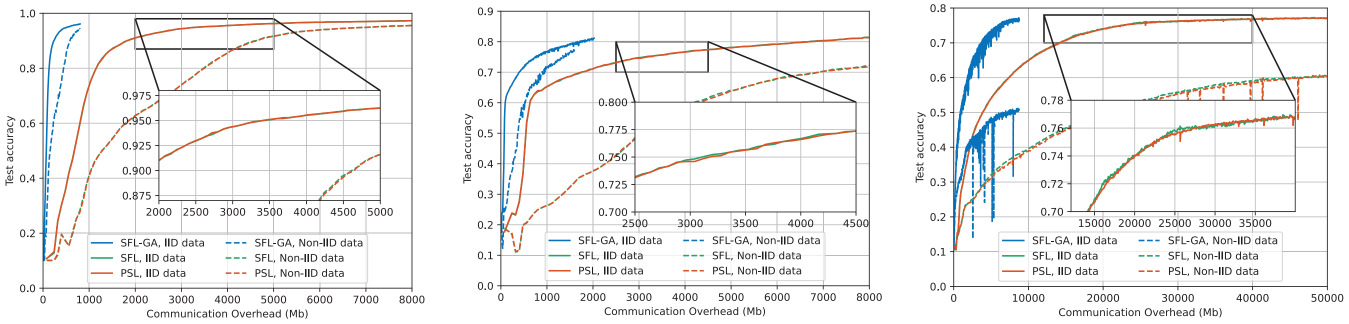
*1) Proposed Training Setting:* The experiments are conducted on a environment with a server and $N = 10$ devices. The learning task is to train a Convolutional Neural Networks (CNN) model for different classification tasks. To evaluate our proposed scheme, we conduct the experiments over three different datasets: MNIST, fashion MNIST and CIFAR-10 datasets. We use similar model architectures as adopted in [35] for model training. The max CPU-cycle frequency $f_t^{n,\max}$ for each client is 0.1 GHz, and the total CPU-cycle frequency for server is 100 GHz. We assume the computation workloads for each client and server are set to $\gamma_F^n = \gamma_B^n = 5.6$ MFlops and $\gamma_F^n = \gamma_B^n = 86.01$ MFlops, respectively [15].

*2) Wireless Communication Setting:* We assume that the path loss of wireless channels between devices and the edge server is given by $128.1 + 37.6 log10(d)$ (in dB), where $d$



(a) MINIST (b) FMINIST (c) CIFAR-10

Fig. 3. Convergence performance evaluation over different cutting layer.



(a) MNIST (b) FMNIST (c) CIFAR-10

Fig. 4. Communication overheads over different schemes.

represents the distance in kilometer (km) [38]. We assume that the thermal noise spectrum density $N_0 = -174$ dBm. The maximum transmit power budgets for each client and server are $p^n_{\max} = 25$ dBm and $P = 33$ dBm, respectively. The total bandwidth $B = 20$ MHz.

To evaluate our proposed SFL-GA scheme, we consider the following existing schemes as baselines:

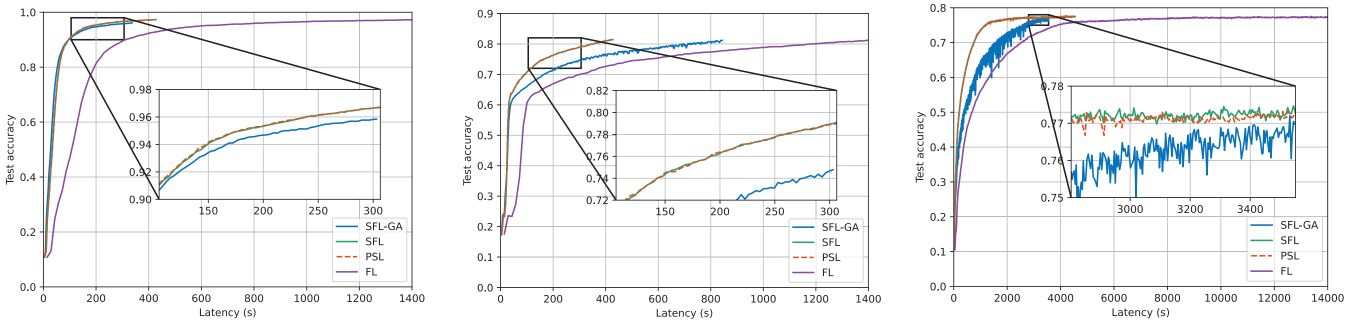
- FL. Each client trains a complete ML model locally and transmits the updated model parameters to the server in each communication round for global model aggregation
- Traditional SFL. The complete model is partitioned between the client and server. In each communication round, clients send smashed data to the server for server-side updates, receive the corresponding gradients for local updates, and then upload their updated client-side models for global aggregation, resulting in high communication overhead.
- PSL. PSL follows a similar training procedure to traditional SFL but omits the global aggregation of client-side models. Each client maintains its own client-side model, reducing communication overhead at the cost of potential model divergence.

## B. Theoretical Analyses

Fig. 3 illustrates the convergence behavior across different cutting points under various datasets. Here, the SFL scheme serves as a benchmark for validating our theoretical analyses. It is evident that SFL outperforms the proposed SFL-GA in terms of communication rounds. Moreover, the convergence performance of SFL-GA deteriorates with an increasing cutting point. For example, while SFL-GA with cutting point $v = 1$ achieves approximately 95% test accuracy over the MINIST dataset, SFL-GA with cutting point $v = 4$ only achieves about 82% after 100 communication rounds. This observation aligns with our theoretical analysis, indicating that a smaller client-side model size leads to better convergence performance for SFL-GA.

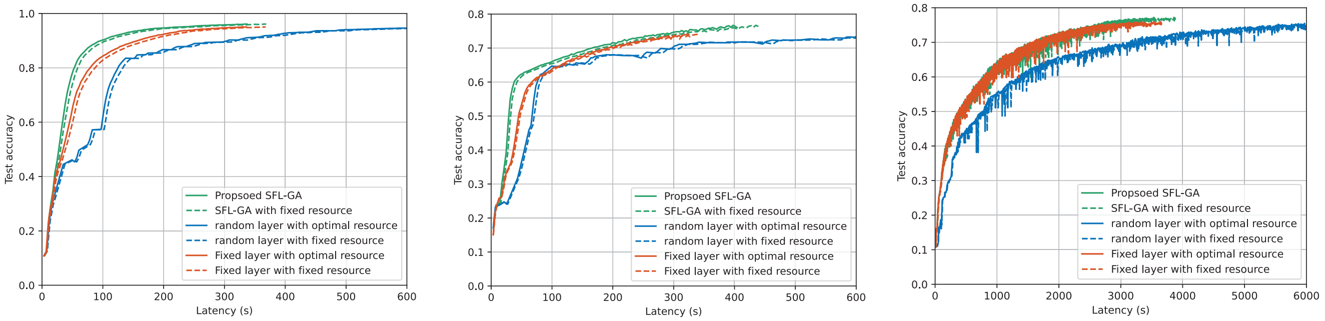
Fig. 4 presents the communication overhead versus convergence performance for both IID and non-IID data partitioning settings, comparing the proposed SFL-GA with existing schemes including traditional SFL and PSL. It is observed that our proposed SFL-GA demonstrates greater communication efficiency compared to traditional SFL and PSL, as it achieves comparable test accuracy with significantly lower communication overhead. For example, the communication overhead for SFL-GA to achieve approximately 94% test accuracy is below 1000 Mb on the MNIST dataset, whereas it exceeds 4000 Mb for traditional SFL. This underscores the effectiveness of the gradient aggregation scheme in SFL-GA for mitigating communication overheads in SFL.

Fig. 5 illustrates the accuracy versus latency across different datasets for different schemes. As shown in Fig. 5, FL exhibits the highest latency to achieve convergence, thereby demonstrating the poorest performance. This is due to the fact that FL updates the entire model on clients with limited computational resources. In contrast, SFL-GA, SFL, and PSL offload portions of the ML model to a server with greater



(a) MNIST (b) FMNIST (c) CIFAR-10

Fig. 5. Accuracy under latency of different schemes.



(a) MNIST (b) FMNIST (c) CIFAR-10

Fig. 6. Accuracy vs latency over different resource strategies.

computational power for model training, thus reducing the latency required for convergence. In particular, our proposed SFL-GA achieves comparable test accuracy to SFL and PSL while significantly reducing communication overhead as shown in Fig. 4. However, it may require more communication rounds, leading to increased convergence latency. This is because the cutting point, optimized for communication efficiency and privacy preservation, can adversely affect convergence behavior, as demonstrated in our theoretical analysis and validated in Fig. 3.

### C. Effectiveness of the Proposed Algorithms

Fig. 6 evaluates accuracy against latency under various resource allocation strategies. We consider both fixed cutting layer and random cutting layer strategies, each assessed using optimal and fixed computation and communication resource allocations as benchmarks. It is seen that that our proposed Algorithm 1 for SFL-GA achieves the shortest latency for convergence among the benchmarks. Additionally, the selection of the cutting layer significantly impacts latency. For instance, with identical computation and communication resource allocations, Algorithm 1 substantially reduces latency compared to the random strategy across different datasets.

Fig. 7 illustrates the convergence of the proposed Algorithm 1 under various privacy constraints corresponding to different cutting points. The results clearly show that the rewards converge within 500 episodes across different constraints, highlighting the effectiveness of Algorithm 1. Meanwhile, the convergence points of the rewards vary depending on the $\epsilon$ value, underscoring the effectiveness of our designed algorithm. For instance, with $\epsilon = 0.001$, the reward converges to approximately $-45$, whereas with $\epsilon = 0.0001$, it converges to about $-250$, demonstrating the significant impact of privacy constraints on model training.

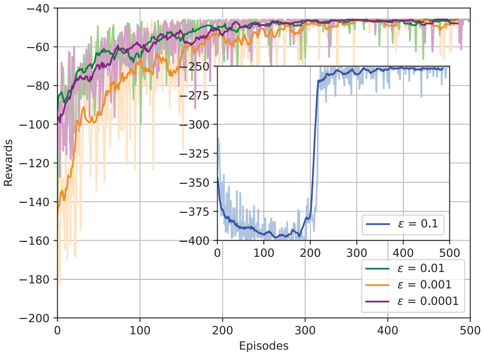


Fig. 7. Convergence performance of Algorithm 1.

Fig. 8 presents the latency under different bandwidth allocations in MNIST datasets. It is evident that latency decreases for all schemes as the available bandwidth increases. This is reasonable since more bandwidth leads to a higher transmission rate, thereby reducing communication latency. Additionally, the proposed SFL-GA achieves the lowest latency for given bandwidth budgets compared to the benchmarks, including FL, traditional SFL, and PSL. In particular, the latency of our framework is significantly lower than that of both SFL and PSL. This improvement mainly results from the proposed

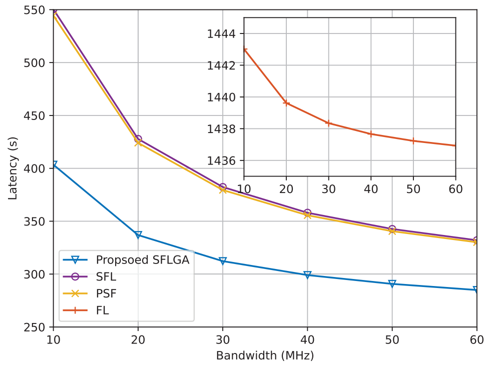


Fig. 8. Latency under different bandwidth allocation.

gradient aggregation mechanism, in which the aggregated gradients of the smashed data are broadcast to all clients. Moreover, it is worth noting that the latency of SFL is slightly higher than that of PSL. This is due to the additional communication overhead in SFL, as it requires client-side model aggregation updates in each communication round.

## VI. Conclusion

In this work, we proposed an SFL-GA framework to improve both communication and computation efficiency for traditional SFL. Specifically, the framework enabled dynamic model cutting point selection by considering wireless network environment, privacy constraints, and computational burden. Given the cutting point, the gradients of smashed data were aggregated before broadcasting, effectively reducing communication overhead. We theoretically analyzed the impact of the cutting point selection on convergence performance of the proposed SFL-GA framework. Furthermore, we formulated an optimization problem to further enhance the communication-and-computation efficiency of the framework, considering the model convergence rate, latency, as well as privacy leakage. To deal with the problem, an efficient joint CCC strategy was designed by integrating the DDQN algorithm and optimization method. Extensive simulation results were provided to verify the effectiveness of the proposed framework and demonstrate the efficiency of the proposed algorithm. In future work, we plan to extend the current design by exploring additional factors beyond client-side model size that may significantly affect the training efficiency of the proposed SFL-GA framework.

## Appendix
## Proof of Lemma 1

To prove Lemma 1, we first derive the following auxiliary variables Averaged Mini-batch Gradient Of SFL-GA, as

$$\bar{\mathbf{g}}_t^n = \begin{bmatrix} \bar{\mathbf{g}}_t^{s,n} \\ \bar{\mathbf{g}}_t^c \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau}\sum_{i=1}^{\tau} \mathbf{g}_{t,i}^{s,n} \\ \frac{1}{\tau}\sum_{i=1}^{\tau} \mathbf{g}_{t,i}^{c} \end{bmatrix}. \tag{41}$$

Meanwhile, we can expressed the Averaged Full-batch Gradient of SFL-GA as

$$\bar{\mathbf{h}}_t^n = \begin{bmatrix} \bar{\mathbf{h}}_t^{s,n} \\ \bar{\mathbf{h}}_t^c \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau}\sum_{i=1}^{\tau} \nabla_{\mathbf{w}^s} F\left(\mathbf{w}_{t,i}^n\right) \\ \frac{1}{\tau}\sum_{i=1}^{\tau} \nabla_{\mathbf{w}^c} \tilde{F}\left(\mathbf{w}_{t,i}\right) \end{bmatrix}. \tag{42}$$

Moreover, the Averaged Full-batch Gradient of SFL can be written as

$$\nabla \bar{F}(\mathbf{w}_t^n) = \begin{bmatrix} \nabla \bar{F}(\mathbf{w}_t^{s,n}) \\ \nabla \bar{F}(\mathbf{w}_t^c) \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau}\sum_{i=1}^{\tau} \nabla_{\mathbf{w}^s} F\left(\mathbf{w}_{t,i}^n\right) \\ \frac{1}{\tau}\sum_{i=1}^{\tau} \nabla_{\mathbf{w}^c} F\left(\mathbf{w}_{t,i}^n\right) \end{bmatrix}. \tag{43}$$

Then, the update of the global model between two consecutive adjacent rounds is formulated as

$$\mathbf{w}_{t+1} - \mathbf{w}_t = \sum_{n=1}^{N} \rho^n \left[\mathbf{w}_{t+1}^n - \mathbf{w}_t^n\right] = -\eta\tau \sum_{n=1}^{N} \rho^n \bar{\mathbf{g}}_t^n. \tag{44}$$

According to the assumption of **L-smoothness**, the improvement on the global loss can be expressed as

$$\begin{aligned}
&\mathbb{E}\left(F\left(\mathbf{w}_{t+1}\right) - F\left(\mathbf{w}_t\right)\right) \overset{(a)}{\leq} -\eta\tau \left\langle \nabla F(\mathbf{w}_t), \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\rangle \\
&\quad + L\eta^2\tau^2 \mathbb{E}\left[\left\| \sum_{n=1}^{N} \rho^n \left(\bar{\mathbf{g}}_t^n - \bar{\mathbf{h}}_t^n\right)\right\|^2 + \left\| \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\|^2\right] \\
&\overset{(b)}{=} -\frac{\eta\tau}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + L\eta^2\tau^2 \mathbb{E}\left[\left\| \sum_{n=1}^{N} \rho^n \left(\bar{\mathbf{g}}_t^n - \bar{\mathbf{h}}_t^n\right)\right\|^2\right] \\
&\quad + \frac{\eta\tau}{2}\left\| \nabla F(\mathbf{w}_t) - \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\|^2 + \frac{\eta\tau\left(2L\eta\tau - 1\right)}{2}\left\| \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\|^2 \\
&\overset{(c)}{\leq} -\frac{\eta\tau}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + \frac{\eta\tau}{2}\mathbb{E}\left[\left\| \nabla F(\mathbf{w}_t) - \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\|^2\right] \\
&\quad + L\eta^2\tau^2 \mathbb{E}\left[\left\| \sum_{n=1}^{N} \rho^n \left(\bar{\mathbf{g}}_t^n - \bar{\mathbf{h}}_t^n\right)\right\|^2\right] \\
&\overset{(d)}{=} -\frac{\eta\tau}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + \frac{\eta\tau}{2}\mathbb{E}\left[\left\| \nabla F(\mathbf{w}_t) - \sum_{n=1}^{N} \rho^n \bar{\mathbf{h}}_t^n \right\|^2\right] \\
&\quad + L\eta^2\tau^2 \sum_{n=1}^{N} \left(\rho^n\right)^2 \frac{1}{\tau^2} \sum_{i=1}^{\tau} \mathbb{E}\left[\left\|\left(\mathbf{g}_t^n - \mathbf{h}_t^n\right)\right\|^2\right] \\
&\overset{(e)}{\leq} -\frac{\eta\tau}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + L\eta^2\tau\sigma^2 \sum_{n=1}^{N} \left(\rho^n\right)^2 \\
&\quad + \frac{\eta\tau}{2}\sum_{n=1}^{N} \rho^n \underbrace{\mathbb{E}\left[\left\|\nabla F(\mathbf{w}_t) - \bar{\mathbf{h}}_t^n\right\|^2\right]}_{A_1},
\end{aligned} \tag{45}$$

where $(a)$ results from the facts that $\mathbb{E}\left[\bar{\mathbf{g}}_t^n - \bar{\mathbf{h}}_t^n\right] = \frac{1}{\tau}\sum_{i=1}^{\tau} \mathbb{E}\left[\mathbf{g}_t^n - \mathbf{h}_t^n\right] = 0$ and $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$. $(b)$ comes from the fact that $2\langle a, b\rangle = \|a\|^2 + \|b\|^2 - \|a-b\|^2$. $(c)$ is hold when $\eta \leq \frac{1}{2L\tau}$. $(d)$ comes form the fact that $\mathbb{E}\left[\left\langle \bar{\mathbf{g}}_t^i - \bar{\mathbf{h}}_t^i, \bar{\mathbf{g}}_t^j - \bar{\mathbf{h}}_t^j \right\rangle\right] = 0, \forall i \neq j$. $(e)$ is achieved due to the Jensen Inequality.

To find the upper bound of Eq. (45), we applied the inequality of $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ on $A_1$ that

$$A_1 = \mathbb{E}\left[\left\|\left(\nabla F(\mathbf{w}_t) - \nabla \bar{F}(\mathbf{w}_t^n)\right) + \left(\nabla \bar{F}(\mathbf{w}_t^n) - \bar{\mathbf{h}}_t^n\right)\right\|^2\right]$$

$$
\begin{aligned}
&\leq 2\mathbb{E}\left[\left\|\left(\nabla F(\mathbf{w}_t) - \nabla \bar{F}(\mathbf{w}_t^n)\right)\right\|^2 + \left\|\left(\nabla \bar{F}(\mathbf{w}_t^n) - \bar{\mathbf{h}}_t^n\right)\right\|^2\right] \\
&\overset{(e)}{\leq} \frac{2}{\tau}\sum_{i=1}^{\tau}\mathbb{E}\left(\left\|\nabla F(\mathbf{w}_t) - \nabla F(\mathbf{w}_{t,i}^n)\right\|^2 + \left\|\nabla F(\mathbf{w}_{t,i}^n) - \mathbf{h}_{t,i}^n\right\|^2\right) \\
&\overset{(f)}{\leq} \frac{2}{\tau}\sum_{i=1}^{\tau}\mathbb{E}\left\|\left(\nabla F(\mathbf{w}_t) - \nabla F(\mathbf{w}_{t,i}^n)\right)\right\|^2 + 2\Gamma\left(\phi_t\left(v\right)\right) \\
&\overset{(g)}{\leq} \frac{2L^2}{\tau}\sum_{i=1}^{\tau}\underbrace{\mathbb{E}\left\|\mathbf{w}_t - \bar{\mathbf{w}}_{t,i}^n\right\|^2}_{A_2} + 2\Gamma\left(\phi_t\left(v\right)\right), \qquad (46)
\end{aligned}
$$

where $(e)$ comes form the Jensen's inequality, and $(f)$ is achieved due to **Assumption 4**. $(g)$ follows the Lipschitz-smooth property. $\bar{\mathbf{w}}_{t,i}^n$ is the model of client *n* obtained after the *i*-th local update based on the SFL gradient $\nabla F(\mathbf{w}_{t,i}^n;\xi)$. Similar with Eq. (46), we further find the upper bound of $A_2$ in Eq. (46) as

$$
\begin{aligned}
A_2 &= \eta^2\mathbb{E}\left\|\sum_{j=1}^{i}\nabla F\left(\mathbf{w}_{t,i}^n;\xi\right)\right\|^2 \\
&\leq 2\eta^2\mathbb{E}\left\|\sum_{j=1}^{i}\left[\nabla F\left(\mathbf{w}_{0,i}^n;\xi\right) - \nabla F\left(\mathbf{w}_{t,i}^n\right)\right]\right\|^2 \\
&\quad + 2\eta^2\mathbb{E}\left\|\sum_{j=1}^{i}\nabla F\left(\mathbf{w}_{t,i}^n\right)\right\|^2 \\
&\overset{(h)}{\leq} 2\ i\eta^2\mathbb{E}\left[\sum_{j=1}^{i}\left\|\nabla F\left(\mathbf{w}_{t,j}^n;\xi\right) - \nabla F\left(\mathbf{w}_{t,j}^n\right)\right\|^2 + \left\|\nabla F\left(\mathbf{w}_{t,j}^n\right)\right\|^2\right] \\
&\overset{(i)}{\leq} 2\ i\eta^2\sum_{j=1}^{i}\sigma^2 + 2\eta^2 i\sum_{j=1}^{\tau}\mathbb{E}\left\|\nabla F\left(\mathbf{w}_{t,j}^n\right)\right\|^2, \qquad (47)
\end{aligned}
$$

where $(h)$ results from Cauchy-Schwarz inequality. $(i)$ is hold from **Assumption 2**. Applying $\sum_{i=1}^{\tau} i = \frac{\tau(\tau-1)}{2}$, we can obtain

$$
\begin{aligned}
&\sum_{i=1}^{\tau}\mathbb{E}\left\|\mathbf{w}_t^n - \mathbf{w}_{t,i}^n\right\|^2 \leq \eta^2\tau(\tau-1)\left(\sigma^2 + \sum_{j=1}^{\tau}\mathbb{E}\left\|\nabla F\left(\mathbf{w}_{t,j}^n\right)\right\|^2\right) \\
&\leq \eta^2\tau\left(\tau-1\right)\left(\sigma^2 + 2L^2\sum_{j=1}^{\tau}\mathbb{E}\left\|\bar{\mathbf{w}}_{t,j}^n - \mathbf{w}_t\right\|^2 + 2\sum_{j=1}^{\tau}\left\|\nabla F(\mathbf{w}_t)\right\|^2\right) \\
&= \eta^2\tau\left(\tau-1\right)\sigma^2 + 2\eta^2\tau^{(}2)\left(\tau-1\right)\left\|\nabla F(\mathbf{w}_t)\right\|^2 \\
&\quad + 2L^2\eta^2\tau\left(\tau-1\right)\sum_{j=1}^{\tau}\mathbb{E}\left\|\bar{\mathbf{w}}_{t,j}^n - \mathbf{w}_t\right\|^2. \qquad (48)
\end{aligned}
$$

By rearranging Eq. (48), we have

$$
\begin{aligned}
\sum_{i=1}^{\tau}\left\|\mathbf{w}_t - \bar{\mathbf{w}}_{t,i}\right\|^2 &\leq \frac{\eta^2\sigma^2\tau\left(\tau-1\right)}{1 - 2L^2\eta^2\tau\left(\tau-1\right)} \\
&\quad + \frac{2\eta^2\tau^2\left(\tau-1\right)}{1 - 2L^2\eta^2\tau\left(\tau-1\right)}\left\|\nabla F(\mathbf{w}_t)\right\|^2 \\
&= \frac{\eta^2\sigma^2\tau\left(\tau-1\right)}{1-A} + \frac{A}{1-A}\left\|\nabla F(\mathbf{w}_t)\right\|^2 \qquad (49)
\end{aligned}
$$

where $A = 2L^2\eta^2\tau\left(\tau-1\right)$.

As a result, $A_1$ in Eq. (46) is bounded by

$$
\begin{aligned}
A_1 &\leq \frac{2L^2\eta^2\sigma^2\left(\tau-1\right)}{1-A} + \frac{2A}{1-A}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + 2\Gamma\left(\phi_t\left(v\right)\right) \\
&\overset{(j)}{\leq} \frac{5}{2}L^2\eta^2\sigma^2\left(\tau-1\right) + \frac{1}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + 2\Gamma\left(\phi_t\left(v\right)\right), \qquad (50)
\end{aligned}
$$

where $(j)$ results from the fact that $A \leq \frac{1}{5}$.

Plug Eq. (50) back into Eq. (45), we have

$$
\begin{aligned}
&\mathbb{E}\left(F\left(\mathbf{w}_{t+1}\right) - F\left(\mathbf{w}_t\right)\right) \leq -\frac{\eta\tau}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + L\eta^2\tau\sigma^2\sum_{n=1}^{N}\left(\rho^n\right)^2 \\
&+ \frac{\eta\tau}{2}\sum_{n=1}^{N}\rho^n\left(\frac{5}{2}L^2\eta^2\sigma^2\left(\tau-1\right) + \frac{1}{2}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + 2\Gamma\left(\phi_t\left(v\right)\right)\right) \\
&= -\frac{\eta\tau}{4}\left\|\nabla F(\mathbf{w}_t)\right\|^2 + \frac{5L^2\eta^3\sigma^2\tau\left(\tau-1\right)}{4} \\
&\quad + L\eta^2\tau\sigma^2\sum_{n=1}^{N}\left(\rho^n\right)^2 + \eta\tau\Gamma\left(\phi_t\left(v\right)\right). \qquad (51)
\end{aligned}
$$

This completes the proof.

## REFERENCES

[1] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6G: Vision, enabling technologies, and applications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 5–36, Jan. 2022.

[2] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. A. Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, Aug. 2019.

[3] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May 2020.

[4] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge AI: Algorithms and systems," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2167–2191, 4th Quart., 2020.

[5] D. Jin, N. Kannengießer, S. Rank, and A. Sunyaev, "Collaborative distributed machine learning," *ACM Comput. Surveys*, vol. 57, no. 4, pp. 1–36, Dec. 2024.

[6] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[7] Y. Liang, Q. Chen, G. Zhu, H. Jiang, Y. C. Eldar, and S. Cui, "Communication-and-energy efficient over-the-air federated learning," *IEEE Trans. Wireless Commun.*, vol. 24, no. 1, pp. 767–782, Jan. 2025.

[8] Q. Chen, X. Xu, Z. You, H. Jiang, J. Zhang, and F.-Y. Wang, "Communication-efficient federated edge learning for NR-U-based IIoT networks," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12450–12459, Jul. 2022.

[9] D. Zhang, M. Xiao, Z. Pang, L. Wang, and H. V. Poor, "IRS assisted federated learning: A broadband over-the-air aggregation approach," *IEEE Trans. Wireless Commun.*, vol. 23, no. 5, pp. 4069–4082, May 2024.

[10] Q. Chen, H. Cheng, Y. Liang, G. Zhu, M. Li, and H. Jiang, "TinyFEL: Communication, computation, and memory efficient tiny federated edge learning via model sparse update," *IEEE Internet Things J.*, vol. 12, no. 7, pp. 8247–8260, Apr. 2025.

[11] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," 2018, *arXiv:1812.00564*.

[12] H. Hafi, B. Brik, P. A. Frangoudis, A. Ksentini, and M. Bagaa, "Split federated learning for 6G enabled-networks: Requirements, challenges, and future directions," *IEEE Access*, vol. 12, pp. 9890–9930, 2024.

[13] C. Thapa, M. A. P. Chamikara, S. Camtepe, and L. Sun, "SplitFed: When federated learning meets split learning," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 8, pp. 8485–8493, doi: 10.1609/aaai.v36i8.20825.

[14] Z. H. Kafshgari, C. Shiranthika, P. Saeedi, and I. V. Bajic, "Quality-rated learning for segmenting medical images with inaccurate annotations," in *Proc. IEEE Intern. Symp. Biomed. Imag. (ISBI)*, Cartagena, Colombia, 2023, pp. 1–5.

[15] C. Xu, J. Li, Y. Liu, Y. Ling, and M. Wen, "Accelerating split federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 6, pp. 5587–5599, Jun. 2024.

[16] D. Waref and M. Salem, "Split federated learning for emotion detection," in *Proc. 4th Novel Intell. Lead. Emerg. Sci. Conf. (NILES)*, Giza, Egypt, Oct. 2022, pp. 112–115.

[17] W. Wu et al., "Split learning over wireless networks: Parallel design and resource management," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1051–1066, Apr. 2023.

[18] J. Guo, C. Xu, Y. Ling, Y. Liu, and Q. Yu, "Latency minimization for split federated learning," in *Proc. IEEE Veh. Techn. Conf.*, Hong Kong, Jun. 2023, pp. 1–6.

[19] J. Shen, X. Wang, N. Cheng, L. Ma, C. Zhou, and Y. Zhang, "Effectively heterogeneous federated learning: A pairing and split learning based approach," 2023, *arXiv:2308.13849*.

[20] Y. Mu and C. Shen, "Communication and storage efficient federated split learning," in *Proc. IEEE Int. Conf. Commun.*, Rome, Italy, May 2023, pp. 2976–2981.

[21] Z. Zhang, A. Pinto, V. Turina, F. Esposito, and I. Matta, "Privacy and efficiency of communications in federated split learning," *IEEE Trans. Big Data*, vol. 9, no. 5, pp. 1380–1391, Oct. 2023.

[22] J. Lee, M. Seif, J. Cho, and H. V. Poor, "Exploring the privacy-energy consumption tradeoff for split federated learning," 2023, *arXiv:2311.09441*.

[23] Y. Liao, Y. Xu, H. Xu, L. Wang, Z. Yao, and C. Qiao, "MergeSFL: Split federated learning with feature merging and batch size regulation," in *Proc. IEEE Int. Conf. Data Engineer. (ICDE)*, Utrecht, The Netherlands, May 2024, pp. 2054–2067.

[24] J. Jeon and J. Kim, "Privacy-sensitive parallel split learning," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Barcelona, Spain, Jan. 2020, pp. 7–9.

[25] Z. Lin et al., "Efficient parallel split learning over resource-constrained wireless edge networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 10, pp. 9224–9239, Oct. 2024.

[26] M. Kim, A. DeRieux, and W. Saad, "A bargaining game for personalized, energy efficient split learning over wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Glasgow, U.K., May 2023, pp. 1–6.

[27] D. Han, H. Bhatti, J. Lee, and J. Moon, "Accelerating federated learning with split learning on locally generated losses," in *Proc. Int. Conf. Mach. Learn. Workshop*, 2021, pp. 1–12.

[28] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 6848–6856.

[29] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient resource management for federated edge learning with CPU-GPU heterogeneous computing," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 7947–7962, Dec. 2021.

[30] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Oct. 2015, pp. 1322–1333.

[31] J. Kim, S. Shin, Y. Yu, J. Lee, and K. Lee, "Multiple classification with split learning," in *Proc. 9th Int. Conf. Smart Media Appl.*, New York, NY, USA, Sep. 2020, pp. 358–363.

[32] J. Pei, P. Hong, M. Pan, J. Liu, and J. Zhou, "Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 263–278, Feb. 2020.

[33] L. Huang, S. Bi, and Y. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[34] R. Zhang, K. Xiong, Y. Lu, B. Gao, P. Fan, and K. B. Letaief, "Joint coordinated beamforming and power splitting ratio optimization in MU-MISO SWIPT-enabled HetNets: A multi-agent DDQN-based approach," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 2, pp. 677–693, Feb. 2022.

[35] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, Fort Lauderdale, FL, USA, 2017, pp. 1273–1282.

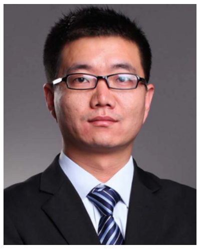
[36] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[37] W. Zhang et al., "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021.

[38] H. Xing, Y. Liu, A. Nallanathan, Z. Ding, and H. V. Poor, "Optimal throughput fairness tradeoffs for downlink non-orthogonal multiple access over fading channels," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 3556–3571, Jun. 2018.

**Yipeng Liang** (Graduate Student Member, IEEE) received the Ph.D. degree in communication and information systems from the School of Electronic Information, Wuhan University, Wuhan, China, in 2025. He is currently with the School of Information Engineering, Nanchang University. His research interests include wireless networks, edge AI, federated learning, and resource management.

**Qimei Chen** (Member, IEEE) received the Ph.D. degree from the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, in 2017. From 2015 to 2016, she was a Visiting Student with the Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA, USA. From 2017 to 2022, she was an Associate Researcher with the School of Electric Information, Wuhan University, Wuhan, China, where she has been an Associate Professor since 2023. Her research interests include intelligent edge communication, unlicensed spectrum, massive MIMO-NOMA, and machine learning in wireless communications. She received an Exemplary Reviewer Certificate of IEEE Wireless Communications Letters in 2020 and 2023. She has served as the Workshop Co-Chair and a TPC Member for IEEE conferences, such as ICC, GLOBECOM, PIMRC, and WCNC. She also served as a Guest Editor for the Special Issues on Heterogeneous Networks of Sensors and NOMA in ISAC (MDPI).



**Rongpeng Li** (Member, IEEE) was a Research Engineer with the Wireless Communication Laboratory, Huawei Technologies Company Ltd., Shanghai, China, from August 2015 to September 2016. From February 2020 to August 2020, he was a Visiting Scholar with the Department of Computer Science and Technology, University of Cambridge, Cambridge, U.K. He is currently an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. His research interests include networked intelligence for science exploration (NICE). He received the Wu Wenjun Artificial Intelligence Excellent Youth Award in 2021. He serves as an Editor for *China Communications*.

**Guangxu Zhu** (Member, IEEE) received the Ph.D. degree in electrical and electronic engineering from The University of Hong Kong in 2019. He is a Senior Research Scientist and the Deputy Director of the Network System Optimization Center, Shenzhen Research Institute of Big Data, and an Adjunct Associate Professor with The Chinese University of Hong Kong, Shenzhen. His recent research interests include edge intelligence, semantic communications, and integrated sensing and communication. He was a recipient of the 2023 IEEE ComSoc Asia–Pacific Best Young Researcher Award and Outstanding Paper Award, the World's Top 2% Scientists by Stanford University, the AI 2000 Most Influential Scholar Award Honorable Mention, the Young Scientist Award from UCOM 2023, and the Best Paper Award from WCSP 2013 and IEEE JSnC 2024. He is the Vice Co-Chair of the IEEE ComSoc Asia–Pacific Board Young Professionals Committee. He serves as an Associate Editor at top-tier journals in IEEE, including IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and IEEE WIRELESS COMMUNICATIONS LETTERS.

**Muhammad Kaleem Awan** received the M.Sc. degree in systems engineering from Pakistan Institute of Engineering and Applied Sciences (PIEAS), Islamabad, Pakistan, and the B.E. degree in electronics engineering from the University of Engineering and Technology (UET), Peshawar, Pakistan. He is currently a Senior Proficient in technology strategy at Omantel, Muscat, Oman, where he leads initiatives in AI/ML-driven network planning, technology strategy, and next-generation broadband connectivity, including 5G/6G, xGSPON, the IoT, Open RAN, and satellite communications. He contributed to the national 5G network architecture design for Omantel. He has more than 20 years of international experience across Oman, Nigeria, and Pakistan in RF planning, broadband networks, wireless access technologies, and performance optimization, having held senior leadership roles with Huawei, Swift Networks, and Wateen Telecom. His expertise spans end-to-end network architecture, AI-based traffic prediction, and data-driven planning frameworks for future network evolution. His current interests include 5G-Advanced, 6G system design, AI-native network planning, non-terrestrial networks (NTN), and data analytics for intelligent broadband systems. He was awarded the Siemens Gold Medal for Best Graduate of Engineering. He was a recipient of the University Gold Medal and the Nazeer Humaira Gold Medal in Mathematics. He completed an Executive Program at the Lagos Business School, where his team ranked in the global top 100 for the Business Strategy Game. He also holds a certification in Data Science and Machine Learning from Massachusetts Institute of Technology (MIT), USA.

**Hao Jiang** (Member, IEEE) received the B.Eng. degree in communication engineering and the M.Eng. and Ph.D. degrees in communication and information systems from Wuhan University, Wuhan, China, in 1999, 2001, and 2004, respectively. He was a Post-Doctoral Research Fellow with LIMOS, Clermont-Ferrand, France, from 2004 to 2005, and was a Visiting Professor with the University of Calgary, Calgary, AB, Canada, and ISIMA, Blaise Pascal University, Clermont-Ferrand. He is currently a Professor with Wuhan University. His research interests include mobile ad hoc networks and mobile big data.