SFedXL: Semi-synchronous Federated Learning with Cross-Sharpness and Layer-Freezing

Mingxiong Zhao, *IEEE Member*, Shihao Zhao, Chenyuan Feng, *IEEE Member*, Howard H. Yang, *IEEE Member*, Dusit Niyato, *IEEE Fellow*, and Tony Q. S. Quek, *IEEE Fellow*

***Abstract*—Federated learning (FL) emerges as a potential solution for enabling multiple terminal devices to collaboratively accomplish computational tasks within a Unmanned Aerial Vehicle (UAV) swarm. However, traditional FL approaches, predicated on synchronous data aggregation, are not feasible for a UAV swarm owing to the inherently variable and dynamic nature of their communication networks compared with terrestrial systems. Furthermore, the data procured by UAVs is often highly heterogeneous, attributable to disparities in deployment environments and device attributes. Considering the distinct flight paths and unique operational conditions encountered by different UAVs, a considerable amount of data remains unlabeled. To tackle the challenges associated with asynchronous operations and the prevalence of unlabeled data, we introduce a novel framework termed Semi-synchronous FL with Cross-Sharpness and Layer-Freezing (SFedXL), tailored for a UAV swarm. In particular, we devise a cross-sharpness model training strategy aimed at optimizing the utilization of both labeled and unlabeled datasets. Additionally, we propose an innovative semi-synchronous model aggregation protocol, complemented by client-specific layer-freezing and client cluster scheduling, designed to expedite the training process. Our simulation results indicate that the proposed algorithm surpasses current FL methods in terms of object recognition accuracy and communication efficiency, albeit with a trade-off of increased local computation latency.**

***Index Terms*—Semi-supervised federated learning, semi-synchronous federated learning, data heterogeneity, device asynchrony, lightweight model training.**

I. Introduction

The advent of agile Unmanned Aerial Vehicles (UAVs) has broadened their application in dynamic and distributed settings, notably in aerial data acquisition for Internet of Things (IoT) and smart city initiatives [2]–[4]. Given their onboard processing prowess, UAVs can locally process collected data and train machine learning (ML) models, which is invaluable for real-time decision-making scenarios such as disaster management and urban surveillance [5], [6]. Meanwihle, Federated learning (FL) stands as a pioneering paradigm for distributed ML that maintains data privacy across local devices, unifying them under a globally shared statistical model coordinated by a central server. This approach adeptly mitigates concerns over centralized data management, especially where data privacy and security are paramount [7]–[9]. Within the FL ecosystem, a diverse array of client devices pools their resources to collaboratively refine ML models, with the central server orchestrating the training while upholding data privacy [10], [11]. This framework capitalizes on the collective intelligence and capabilities of decentralized devices. Consequently, FL is particularly well-suited for deployment in a UAV swarm. Numerous studies are currently pivoting towards integrating FL within UAV operations [12].

However, the integration of FL within a UAV swarm grapples with substantial challenges, given the decentralized essence of an FL manner and the distinctive traits and limitations of aerial networks. These include unpredictable transmission links and heterogeneous data patterns among UAVs [13]–[16]. While FL provides a cost-effective framework for distributed model training without raw data sharing, its adaptation to a UAV swarm is intricate. Factors such as the abundance of unlabeled data, scarce communication bandwidth, and the finite computational resources on UAVs complicate the scenario [17]–[19]. UAVs are often deployed in applications such as environmental monitoring and disaster response, necessitating rapid decision-making based on collected data. However, the prevalence of unlabeled data complicates model training, necessitating manual labeling or the incorporation of unsupervised or semi-supervised techniques into FL. UAVs operate in dynamic environments where communication links are sporadic, bandwidth-restricted, or disrupted by mobility or interference. These challenges hinder frequent model updates between UAVs and a central server without incurring latency or energy drain. Moreover, onboard computational resources of each UAV are often limited, requiring resource-efficient algorithms and model compression to align with their operational demands.

A. Related Work

Several investigations have delved into the incorporation of FL within a UAV swarm, with a primary focus on overcoming the data and resource constraints of UAVs and enhancing learning efficacy.

1) FL with Unlabeled Data: To surmount the FL challenge with unlabeled data, several innovative methods have emerged.

M. Zhao, and S. Zhao are with the Engineering Research Center of Cyberspace, National Pilot School of Software, Yunnan University, Kunming, China. (E-mail: jimmyzmx@gmail.com; shihaozhao@mail.ynu.edu.cn).

C. Feng is with EURECOM, Sophia Antipolis 06410, France (E-mail: Chenyuan.Feng@eurecom.fr).

H. H. Yang is with Zhejiang University/University of Illinois Urbana-Champaign Institute, Zhejiang University, Haining 314400, China (E-mail: haoyang@intl.zju.edu.cn).

D. Niyato is with School of Computer Science and Engineering, Nanyang Technological University, Singapore (E-mail: dniyato@ntu.edu.sg)

T. Q. S. Quek is with the Information Systems Technology and Design Pillar, Singapore University of Technology and Design, 487372, Singapore (E-mail: tonyquek@sutd.edu.sg).

Part of the work was presented at 2024 IEEE 24th International Conference on Communication Technology [1].

Corresponding author: Chenyuan Feng.

One approach is to apply semi-supervised learning techniques, which could harness a modest subset of labeled data to shepherd the learning process for an expansive corpus of unlabeled data. Within an FL framework, this methodology could entail the amalgamation of labeled data from a cohort of clients with unlabeled data contributed by others. An alternative prevalent strategy is the utilization of pseudo-labeling, also known as self-training. This technique involves an initialization of a model with labeled data, which then proceeds to generate pseudo-labels upon the unlabeled data. Thereafter, this model can be subjected to iterative refinement as an increasing volume of pseudo-labeled data accrues, progressively enhancing the model's predictive acumen. For instance, FedMatch [20] harnesses a federated semi-supervised learning approach, capitalizing on disjoint learning and inter-client consistency. Alongside this, FedCD [21], a class-aware balanced dual teacher model, ensures the efficient exploitation of unlabeled data in federated semi-supervised learning. SemiPFL [22] proposes a personalized semi-supervised FL framework, tailored for edge intelligence, to optimize learning in environments with scarce labeled data. However, these methods often overlook the statistical variance between clients' unlabeled and labeled datasets when formulating the loss function during optimization. Recent centralized learning advancements suggest that semi-supervised learning can significantly benefit from sharpness-aware optimization techniques.

2) FL with Constrained Resources: Due to the limited computational power and wireless communication resources, some clients, often referred to as *stragglers*, experience considerable delays when engaging in FL. These delays can severely impair the efficacy of traditional synchronous FL models [23]. To alleviate the impact of stragglers and diminish communication costs, various solutions have been proposed. Model compression techniques, including model pruning [24], sparsification [25], and quantization [26], are extensively utilized to alleviate communication overhead. Polaris [27] accelerates asynchronous FL by employing an adaptive client selection strategy, while SITUA-CQ [26] improves FL convergence speed and efficiency by incorporating client-specific criteria in a situation-aware clustering and quantization level selection technique. Despite these methods to reduce communication overhead, model quantization necessitates additional support in the deployment environment, and model pruning and sparsification often require specialized model architectures, complicating their deployment in resource-limited settings. To overcome these limitations, ALF [28], an automatic layer freezing approach, was introduced to enhance communication efficiency in cross-device FL. However, a reliable method for determining which layers to freeze is yet to be established.

Despite these advancements, a substantial gap remains in addressing the combined challenges of managing large volumes of unlabeled data, ensuring robust communication under network constraints, and efficiently utilizing computational resources within a UAV swarm. Our work aims to bridge these gaps by proposing a framework that integrates semi-supervised learning techniques into FL, optimizing both communication and computational efficiency within UAV swarms.

B. Contributions

In this study, we introduce a novel framework called Semi-synchronous Federated Learning with Cross-Sharpness and Layer-Freezing (SFedXL), tailored for a UAV swarm. We design the framework to optimize data utilization, expedite the training process, and ultimately enhance model accuracy. Our main contributions are summarized as follows:

- To effectively harness unlabeled data amassed by each UAV, we devise a novel semi-supervised FL method based on cross-sharpness learning (XSL) scheme [29]. While semi-supervised FL is adept at leveraging a combination of extensive unlabeled datasets alongside limited labeled data, it is heavily contingent on the similarity and transferability between these datasets, which can lead to suboptimal local minima detrimental to the unlabeled data. Our method maintains consistent learning performance across both datasets by diminishing the cross-sharpness in predictive functions, thereby amplifying the efficacy.
- To counter the straggler issue stemming from asynchronous UAV operations, we developed a theoretical model to assess the influence of dropouts or stragglers on training latency in FL. Based on our analysis, we propose a Client Clustering Selection (CCS) algorithm. We design the algorithm to reduce latency and verify its effectiveness theoretically.
- To address the limitations imposed by limited resources, we devise a Client-wise Layer Freezing (CWLF) scheme. This scheme is adept at reducing communication overhead while also mitigating the impact of training noise and data heterogeneity across various client UAVs.
- Extensive simulation results demonstrate that our proposed SFedXL method surpasses classical FL methods in terms of object recognition accuracy and communication cost. These findings underscore the significant practical benefits and potential of our framework in enhancing UAV swarm operations within an FL context.

The rest of the paper is organized as follows: Section II delineates the system model and underscores the primary challenges encountered. Section III presents a detailed description of the proposed algorithms. Section IV showcases the simulation outcomes and offers an in-depth analysis. Ultimately, Section V concludes this work.

II. System Model & Problem Formulation

As depicted in Fig. 1, we aim to implement an FL manner in a UAV swarm comprising $C+1$ devices, which includes one server UAV and C client UAVs, denoted by the set $\mathcal{C} \triangleq \{0, 1, 2, \cdots, C\}$. Within this configuration, each client UAV is equipped for data acquisition and computational tasks, while the server UAV is regarded as a central server, responsible for orchestrating the collaborative efforts of the entire swarm. Throughout their operation, the client UAVs are tasked with gathering substantial volumes of unlabeled data alongside a limited set of labeled data. We define the unlabeled and labeled datasets on the c-th client UAV as $D_{c,u} = \{z_{c_i}\}_{i=1}^{N_{c,u}}$ and $D_{c,l} = \{(x_{c_j}, y_{c_j})\}_{j=1}^{N_{c,l}}$, respectively. Here, z_{c_i} signifies the

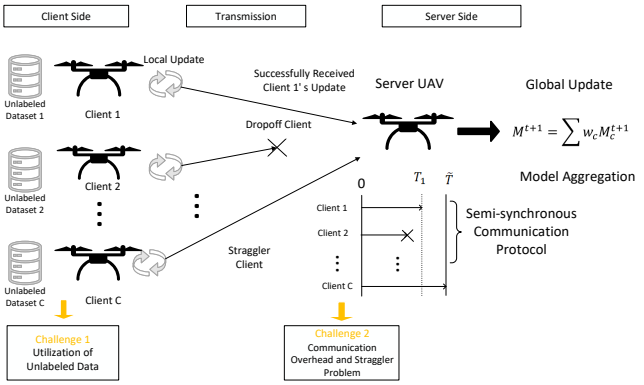


Fig. 1: Overall semi-synchronous FL framework for a UAV swarm and the main challenges.

i-th unlabeled sample, x_{c_j} represents the j-th data feature within the input space $\mathcal{X}$, and y_{c_j} corresponds to its label within the label space $\mathcal{Y}$. $N_{c,u}$ and $N_{c,l}$ represent the sizes of the unlabeled and labeled datasets, respectively, with the understanding that $N_{c,u} \gg N_{c,l}$. Furthermore, we denote the aggregate of all labeled datasets from the client UAVs as $D_l = \bigcup_{c=1}^{C} D_{c,l}$, which constitutes the global labeled dataset. Similarly, $D_u = \bigcup_{c=1}^{C} D_{c,u}$ aggregates the unlabeled datasets from all client UAVs, forming the global unlabeled dataset.

A. Traditional FL with Labeled Data

In traditional FL, the objective function for local training at the c-th UAV over its own labeled dataset, is formulated as follows:

$$\min_{\mathbf{M}_c} J(f_{\mathbf{M}_c}, D_{c,l}) := \frac{1}{N_{c,l}} \sum_{i=1}^{N_{c,l}} \ell(f_{\mathbf{M}_c}(x_{c_i}), y_{c_i}), \tag{1}$$

where $f_{\mathbf{M}_c} : \mathcal{X} \to \mathcal{Y}$ is an r-layer neural network (NN) model, parameterized by $\mathbf{M}_c$, at the c-th UAV, and $\ell\left(f_{\mathbf{M}_c}(x_{c_i}), y_{c_i}\right)$ denotes the loss function to measure the discrepancy between the predictions of the local model $f_{\mathbf{M}_c}$ and the actual labels y_{c_i} for the training instance x_{c_i}. For assessing the local model's accuracy, we employ the cross-entropy loss, a prevalent choice for evaluating the divergence between the model's predictions and the true dataset labels. For assessing the local model's accuracy, we employ the cross-entropy loss, a prevalent choice for evaluating the divergence between the model's predictions and the true dataset labels. Under the orchestration of the server UAV, which facilitates collaboration among all client UAVs, the collective ambition of the FL system is to converge towards a global model f_M. This model aims to minimize the global loss function $J(f_M, D_l)$ across the entire aggregated dataset D_l. The global loss function is formulated as follows:

$$\min_{M} J(f_M, D_l) := \sum_{c=1}^{C} \frac{1}{N_{c,l}} \sum_{i=1}^{N_{c,l}} \ell(f_M(x_{c_i}), y_{c_i}), \tag{2}$$

where $\mathbf{M}$ denotes the global model parameters.

A conventional approach entails iterative communication between several client UAVs, denoted as U_c, and the server UAV, denoted as U_s, complemented by gradient descent techniques to update the global model. For the sake of clarity and without loss of generality, we concentrate on the t-th communication round. At the beginning of the t-th round, the server UAV selects a portion of client UAVs from entire UAV swarm for participation in the training process, denoted by Γ_t. Let K denote the size of Γ_t. Subsequently, the server UAV distributes the global model f_{M^t} to all selected client UAVs. On the client-side, each participant engages in E_l epochs of local model enhancement, where $E_l \geq 1$. Let e denote the index of local updates. For each client UAV $c \in \Gamma_t$ and for $e = 0, \cdots, E_l - 1$, the evolution of local model parameters during the t-th round at the c-th UAV is given as follows:

$$\begin{aligned} M_{c,0}^t &\leftarrow M^t, \\ M_{c,e+1}^t &\leftarrow M_{c,e}^t - \eta \nabla_{M_{c,e}^t} J(f_{M_{c,e}^t}, D_{c,l}), \\ M_c^{t+1} &\leftarrow M_{c,E_l}^t, \end{aligned} \tag{3}$$

where M^t represents the global model parameters downloaded during the t-th round, M_c^{t+1} denotes the local model parameters updated by the c-th UAV updated local model parameters during the t-th round, $M_{c,e}^t$ denotes the local model parameters updated by the c-th UAV at the e-th epoch of local training during the t-th round, and η represents the learning rate.

Let $\Gamma_t^{'}$ epresent the set of client UAVs whose model updates are successfully received by the server UAV. Due to unreliable transmissions, we have $\Gamma_t^{'} \subseteq \Gamma_t$. Let n denote the number of client UAVs the server UAV expects to receive updates from successfully, with $n \leq K$. In this work, we adopt a semi-synchronous communication protocol to coordinate the FL training process [27]. Specifically, the server UAV will stop waiting and update the global model when either of the following conditions is met: receiving model updates from n client UAVs or reaching the timeout threshold $\tilde{T}_t$. In accordance with the FedAvg framework [7], the global model parameters at the t-th round evolve as follows:

$$M^{t+1} = \frac{\sum_{c \in \Gamma_t^{'}} N_{c,l} {\mathbf{M}_c}^{t+1}}{\sum_{c \in \Gamma_t^{'}} N_{c,l}}. \tag{4}$$

B. Unlabeled and Heterogeneous Data Issue

In real-world scenarios, each UAV often collects a substantial amount of unlabeled data, particularly in unfamiliar environments [30], [31]. To effectively utilize this unlabeled data, semi-supervised learning has emerged as a promising approach [32]. The core idea of semi-supervised learning is to leverage a pre-trained model with sufficient classification capabilities to generate surrogate labels for the unlabeled data. The aim of semi-supervised learning with FL is to train a shared model that performs well on a global dataset. To integrate both labeled and unlabeled data, the global loss function of semi-supervised FL can be reformulated as follows:

$$\begin{aligned} \min_{M} J_{\mathrm{SSFL}} &= \min_{M} (J_l(M, D_l) + J_u(M, D_u)) \\ &= \min_{M} \sum_{c=1}^{C} \sum_{i=0}^{N_{c,l}} \left(\frac{1}{N_{c,l}} \ell\left(f_M(x_{c_i}), y_{c_i}\right) \right. \\ &\quad \left. + \sum_{c=1}^{C} \sum_{j=0}^{N_{c,u}} \frac{\mathbb{I}}{N_{c,u}} \ell\left(f_M\left(\mathcal{A}\left(z_{c_j}\right)\right), \widehat{y}_{c_j}\right) \right), \end{aligned} \tag{5}$$

where $J_l(M, D_l)$ represents the loss function for supervised learning over the labeled dataset D_l, $J_u(M, D_u)$ denotes the semi-supervised regularization term for the unlabeled dataset D_u; $\mathcal{A}(\cdot)$ is a data augmentation function that transforms a raw unlabeled data point z_{c_j} into an augmented variant, from which a pseudo-label $\widehat{y}_{c_j}$ is obtained; $\mathbb{I}$ is a boolean variable that selects high-confidence pseudo-labels; $\mathbb{I} = 1$ if $\widehat{y}_{c_j} > \tau$, and $\mathbb{I} = 0$ otherwise, where τ is a predefined confidence threshold.

Recent studies have shown that training exclusively on labeled data and using augmentation techniques to generate pseudo-labels for unlabeled data can lead to significant generalization mismatches. However, introducing a cross-sharpness term can mitigate this issue [29]. In Fig. 2, we plot the loss landscapes of a model trained solely on labeled data and a model trained with both labeled and unlabeled data, incorporating a cross-sharpness term. The results demonstrate that the loss landscape becomes smoother, with the model achieving lower losses and higher accuracy. The contour values for labeled and unlabeled data in the first row are lower than those in the second row, indicating that training techniques based on minimizing cross-sharpness may converge to a better model with a lower training loss. Focusing on the two subplots in the third column, we observe that the accuracy curves for both labeled and unlabeled data follow a consistent trend in semi-supervised learning with cross-sharpness, whereas a noticeable mismatch appears in the purely supervised learning scenario. Despite its great potential, few studies have extended this method to the FL context, particularly in addressing challenges such asdevice's asynchronous operation characteristics and limited resources. To ensure consistency between labeled and unlabeled data learning in a UAV swarm, we propose a novel cross-sharpness regularization-based FL algorithm in Section III-A.

C. Limited Resources Issue

For ML, the parameters of models, particularly deep neural networks (DNNs), are conventionally refined in a sequential, layer-wise manner. Within the context of FL paradigm, recent studies have highlighted that it is not imperative to update all DNN layers with each training iteration. By pinpointing the layers that exhibit minimal parameter fluctuations during the training phase and subsequently "freezing" their parameters, the redundancy of updates and the associated data transmissions can be reduced, thereby alleviating the challenges posed by limited resources [28]. The main idea of the layer-freezing methodology lies in an optimal selection of which layers to render static, that is, which parameters to omit from transmitting.

1) Communication Overhead: Let a boolean variable $\mathbb{I}_l^{t,c}$ to indicate the layer-freezing decision. If the parameters of layer l are updated and transmitted during round t on client c, then $\mathbb{I}_l^t = 1$; otherwise, $\mathbb{I}_l^t = 0$. The communication overhead for client UAV c to transmit model parameters to the server UAV is expressed as:

$$C_s^{t,c} = \sum_{l=1}^{L} \mathbb{I}_l^{t,c} P_l b, \tag{6}$$

where P_l denotes the number of parameters in the l-th layer, and b is the number of bits used to quantify each parameter.

The cumulative communication overhead for the subset of $|\Gamma^t|$ client UAVs in downloading the global model parameters from the server UAV (denoted as C_{dl}^t), and in uploading their updated local model parameters (denoted as C_{ul}^t) can be articulated as follows:

$$\begin{aligned} C_{dl}^t &= \sum_{c\in\Gamma^t} \sum_{l=1}^{L} P_l b, \\ C_{ul}^t &= \sum_{c\in\Gamma^t} C_s^{t,c}. \end{aligned} \tag{7}$$

2) Learning Latency: Within this framework, the learning latency per round is constituted by two primary components: i) on the client side, there is client model training and client-side uploading latency; ii) on the server side, there are the latencies associated with global model aggregation, client selection, and global model distribution. Among these factors, the former is dominant and will be analyzed in greater detail in the forthcoming sections.

a) Client UAV Updating Latency: Due to device heterogeneity, different client UAVs will experience divergent levels of computation and communication latency. The communication latency for the c-th client UAV is characterized by:

$$T_{c,t}^{\text{cm}} = \frac{C_s^{t,c}}{B\log_2\left(1+\frac{P_{\text{c,tx}}h_c}{N_0}\right)}, \tag{8}$$

with

$$h_c = 10^{-\frac{\text{PL}(d_c)}{10}}, \quad \text{PL}(d_c) = \text{PL}_0 + 10\gamma\log_{10}\left(\frac{d_c}{d_0}\right), \tag{9}$$

where B denotes the bandwidth allocated to each UAV, $P_{\text{c,tx}}$ denotes the transmission power of the c-th UAV, h_c and N_0 denote the channel gain and noise power between the server UAV and the c-th client UAV, respectively, $\text{PL}(d_c)$ denotes the path loss for the c-th client UAV, located at a distance d_c from the server UAV, PL_0 represents the reference path loss at a predefined reference distance $d_0 = 1$ meter, γ is the path loss exponent, and d_c denotes the separation between the c-th client UAV and the server UAV.

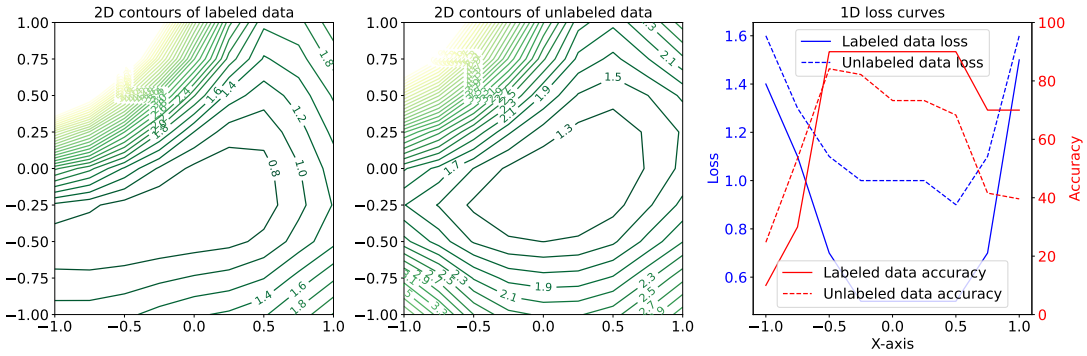
Similar to [33], the computation latency for the c-th client UAV with layer-freezing strategy can be expressed as

$$T_{c,t}^{cp} = \frac{N_{c,l}(1-\gamma_c^t)\sigma_c}{\varphi_c}, \text{ with } \gamma_c^t = \sum_{l=1}^{L} \mathbb{I}_l^{t,c}, \tag{10}$$

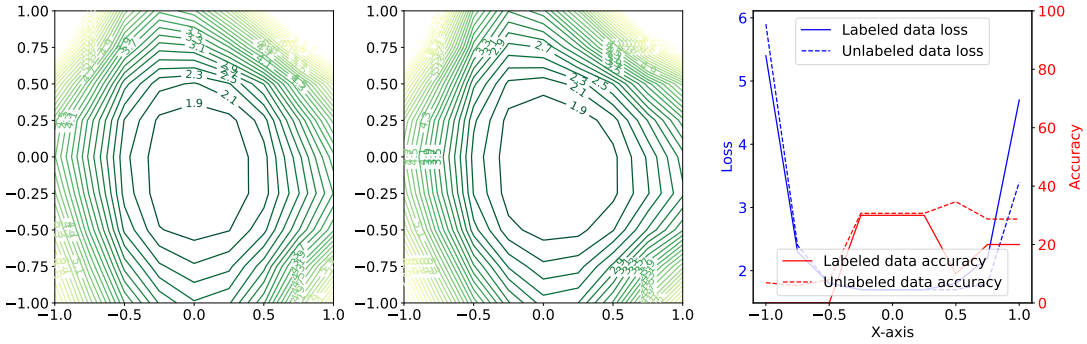
where $T_{c,t}^{cp}$ is the local training latency of the c-th UAV during round t, γ_c^t is the percentage of frozen parameters, $N_{c,l}$ is the size of the local labeled dataset on the c-th UAV, σ_c and φ_c correspond to the total clock cycles expended on local training and the CPU frequency of the c-th UAV, respectively.

Considering the dynamic nature of real-world environments where communication links may falter, UAVs could potentially disconnect. In such instances, the update latency for the c-th client UAV is characterized as follows:

$$T_{c,t} = \begin{cases} +\infty, \text{ if the } c\text{-th UAV is dropping out,} \\ T_{c,t}^{\text{tr}} = T_{c,t}^{\text{cm}} + T_{c,t}^{\text{cp}}, \text{ otherwise.} \end{cases} \tag{11}$$



(a) loss landscapes of FedXSL (with cross-sharpness)



(b) loss landscapes with FedAvg (without cross-sharpness)

Fig. 2: The loss landscapes for both labeled and unlabeled data, derived from training on a single UAV utilizing the SAT6 dataset with an allocation of 100 labels per category. (i) The first and the second rows correspond to the outcomes post the 60th epoch of localized stochastic gradient descent (SGD) training, with and without the incorporation of cross-sharpness, respectively. (ii) The first two columns show the two-dimensional loss contours for the labeled and unlabeled datasets, respectively, while the third column illustrates the one-dimensional loss curves.

b) Server UAV Updating Latency: Considering the poor communication conditions and inadequate computation capability of certain client UAVs, the server UAV may suffer from the straggler problem. To prevent the server UAV from indefinitely postponing the aggregation of the global model waiting for a disconnected participant, we adopt a semi-synchronous FL approach in this study. Specifically, the server UAV will wait for at most $\tilde{T}_t$ time period before generating an aggregated global model, anticipating at least n local model updates from K selected client UAVs [34]. The server UAV waiting latency during round t and its expectation duration are given as:

$$\begin{aligned} T_t^{\text{sw}} &= \min\left\{\max_{c_i\in\Gamma_t'}\{T_{c_i,t}^{\text{tr}}\}, \tilde{T}_t\right\}, \\ \mathbb{E}\{T_t^{\text{sw}}\} &= (1-P_{stp})\max_{c_i\in\Gamma_t'}\{T_{c_i}^{\text{tr}}\} + P_{stp}\tilde{T}_t, \end{aligned} \tag{12}$$

where ρ_c denotes the dropout probability for the c-th client UAV at each round, and P_{stp} signifies the likelihood of the straggler issue arising.

c) Straggle Problem: Let C_m denote the set of straggler UAVs whose update latency, even with reliable transmission, exceeds the maximum server UAV waiting time, defined as $C_m = \{c|T_c^{\text{tr}} \geq \tilde{T}_t \text{ and } c \in \mathcal{C}\}$. In this work, the straggler problem is defined as the event $T_t^{\text{sw}} = \tilde{T}_t$, which arises under two specific conditions: 1) the dropout of at least $(K-n)$ UAVs within a single training round, signified by the event A_{drop} ; 2) the selection of at least one straggler UAV from C_m, indicated by the event A_{last}. Given that events A_{last} and A_{drop} are independent of each other, the probability of the server UAV having to wait for the duration of $\tilde{T}_t$ before performing model aggregation is formulated as follows:

$$\begin{aligned} P_{stp} &= P(T_t^{sw} = \tilde{T}_t) = P(A_{\text{drop}} \cup A_{\text{last}}) \\ &= P(A_{\text{drop}}) + P(A_{\text{last}}) - P(A_{\text{drop}})P(A_{\text{last}}), \end{aligned} \tag{13}$$

which is monotonically increasing with respect to $P(A_{\text{last}})$ and $P(A_{\text{drop}})$, $P(A_{\text{drop}})$ is determined by the transmission circumstance, and $P(A_{\text{last}})$ is determined by user selection strategy. Our objective is to minimize the expected server waiting latency by refining the client scheduling process, which will be described in Section III-C.

III. Algorithm Description and Analysis

A. Cross-Sharpness Learning (XSL) on the User Side

To harmonize the learning performance between labeled and unlabeled data, a pioneering approach centered on cross-sharpness minimization has been introduced in a centralized learning context [29]. The essence of this technique is twofold: First, adversarial perturbations are imposed on the parameters

of the model trained with labeled data to pinpoint the worst-case model. Second, the worst-case model is realigned with the original model by diminishing their predictive disparities—referred to as cross-sharpness—on the unlabeled data. Consequently, the abundant reservoir of unlabeled data is optimally leveraged to refine the learning direction by minimizing the cross-sharpness. In the context of FL, to efficiently utilize unlabeled samples, each UAV integrates a cross-sharpness regularization as an additional penalty within its local loss function. This adaptation enables a better utilization of the available data, enhancing the overall learning efficacy.

In the proposed Cross-Sharpness Learning (XSL) manner, the local loss function of the c-th client UAV is formulated as

$$\min_{\mathbf{M}_c} J_{c,\mathrm{SSFL}} = \min_{\mathbf{M}_c} (J_{c,l} + J_{c,\mathrm{XS}}), \tag{14}$$

where $J_{c,\mathrm{XS}}$ is the cross-sharpness regularization of the c-th client UAV, its minimization can be expressed as

$$\begin{aligned} \min_{\mathbf{M}_c} J_{c,\mathrm{XS}} &:= \min_{\mathbf{M}_c} \sum_{z_{c_i} \in D_{c,u}} \ell\left(f_{\widetilde{M}_c}(z_{c_i}), f_{\mathbf{M}_c}(z_{c_i})\right), \\ \tilde{M}_c &= \mathbf{M}_c + \epsilon_c^*, \end{aligned} \tag{15}$$

where $\tilde{M}_c$ denotes the worst-case model over the c-th client UAV's labeled data, and ϵ_c^* denotes the optimal model parameter perturbation at the c-th UAV. Similar to [35], the optimal model parameter perturbation can be expressed as

$$\begin{aligned} \epsilon_c^* &= \arg \max_{\|\epsilon_c\|_2 \le \beta} \ell(f_{\mathbf{M}_c+\epsilon_c}(x_{c_j}), y_{c_j}) \\ &\approx \beta \frac{\nabla_{\mathbf{M}_c} \ell\left(f_{\mathbf{M}_c}(x_{c_j}), y_{c_j}\right)}{\|\nabla_{\mathbf{M}_c} \ell\left(f_{\mathbf{M}_c}(x_{c_j}), y_{c_j}\right)\|_2}, \end{aligned} \tag{16}$$

where $\beta > 0$ is a constant to limit the magnitude of perturbation over ϵ_c inside a l_2-sphere[1].

For $e = 0, ..., E_l - 1$, the local update of the c-th client UAV at the e-th epoch during the t-th round is given as:

$$\begin{aligned} M_{c,0}^t &\leftarrow M^t, \\ M_{c,e+1}^t &\leftarrow M_{c,e} - \eta \nabla_{M_{c,e}} J_{c,SSFL}, \\ M_c^{t+1} &\leftarrow M_{c,E_l}^t. \end{aligned} \tag{17}$$

The details of XSL algorithm are listed in Algorithm 1.

Algorithm 1 Cross-Sharpness Learning (XSL) Function

Require: $D_{c,l}, D_{c,u}, M_t, \beta, E_l$
Ensure: ${\mathbf{M}_c}^{t+1}$
1: Initialize the local model parameters ${\mathbf{M}_c}^0 \leftarrow M^t$
2: **for** $e = 0$ to $E_l - 1$ **do**
3: Calculate local labeled loss $J_{c,l}$ according to (1)
4: Calculate optimal model parameter perturbation ϵ_c^* according to (16)
5: Calculate worst-case model $\tilde{M}_c$ and cross-sharpness regularization $J_{c,XS}$ according to (15)
6: Update local model parameters according to (17)
7: **end for**
8: **Return** ${\mathbf{M}_c}^{t+1}$

B. Client-Wise Layer Freezing (CWLF) on the User Side

To improve resource utilization efficiency, we design a Client-Wise Layer Freezing (CWLF) mechanism that dynamically freezes a certain percentage of client model parameters during the local training. A substantial body of literature has demonstrated that a significant portion of model parameters often reach a stable state well before full convergence, with the stable parameter count increasing progressively over the training duration [28].Thus, it is unnecessary to keep updating and sending these parameters once they have reached their best values, as doing so only wastes extra computing power and bandwidth without improving performance. Therefore, it is prudent to freeze these stable parameters, exempting them from further updates and uploads, thereby incurring savings in terms of resources.

The CWLF scheme necessitates users to monitor and review the model's historical parameter updates for each layer. Given the non-independent and identically distributed (non-IID) nature of user data in FL and the potential for training noise, it is expected that gradient changes will not only be non-zero but also exhibit fluctuations. To address these issues, we adopt a moving average strategy, which helps to smooth out fluctuations and provides a more stable basis for assessment. Furthermore, recognizing the substantial variation in gradient update magnitudes across the different layers of the client model, we implement a normalization strategy to facilitate a unified freezing threshold τ. It is worth mentioning that we assume that all layers are involved in training in the first round, that is, all of them are unfrozen layers.

For round $t \ge 1$, the Euclidean norm of the discrepancy between the local model parameters $M_{c,l}^t$ of client c and the global model parameters, M^t serves as a metric for measuring the extent of parameter updates at layer l. This can be expressed as:

$$\mathcal{D}_{c,l}[t] = \|M_{l:}^t - M_{c,l:}^t\|, \tag{18}$$

where $\mathcal{D}_{c,l}$ and $\mathcal{D}_{c,l}[t]$ denote the list containing all n_w magnitudes of historical parameter updates for layer l at client UAV c and its t-th element, respectively; $M_{l:}^t$ and $M_{c,l:}^t$ the parameters of the l-th layer of the local model of client UAV c and the global model, respectively. The normalized moving average of updates for layer l at client UAV c is expressed as

$$\bar{\mathcal{D}}_{c,l}[t] = \frac{1}{2} \sum_{i=t}^{t+1} \frac{\mathcal{D}_{c,l}[i] - \min\{\mathcal{D}_{c,l}[t]\}}{\max\{\mathcal{D}_{c,l}\} - \min\{\mathcal{D}_{c,l}\}}. \tag{19}$$

Let n_w denote the length of historical update to be assessed. Then, the gradient of parameter updates can be calculated as

$$\beta_{c,l} = \frac{n_w \sum_{i=1}^{n_w} i \mathcal{D}_{c,l}[i] - \sum_{i=1}^{n_w} i \sum_{i=1}^{n_w} \mathcal{D}_{c,l}[i]}{n_w \sum_{i=1}^{n_w} i^2 - \left(\sum_{i=1}^{n_w} i\right)^2}. \tag{20}$$

Let τ denote a predefined threshold for freezing, the l-th layer will be designated as frozen if $\beta_{c,l} < \tau$. Upon completion of

[1]The approximation is derived using a first-order Taylor expansion and the properties of dual norms [29], [35]. Its accuracy depends on the smoothness of the loss function and the validity of the Taylor expansion. When the loss function exhibits high nonlinearity, the approximation may become less precise. However, in many practical applications, particularly in deep learning, this approach has been shown to be effective, as gradient information generally provides sufficient directional guidance for optimization.

local training, client UAV c will transmit to the server UAV only the parameters of unfrozen layers. In this work, we use $M_{c,LF}^t$ to denote the updated local model trained with the CWLF algorithm by client UAV c during round t. The detailed CWLF scheme is outlined in Algorithm 2.

Algorithm 2 Client-Wise Layer Freezing (CWLF) Function

Require: $L, n_w, M^t, M_c^t, D_{c,l}, t$
Ensure: $M_{c,LF}^t$
for each layer l =1,...,L **do**
  Initialize $\mathcal{D}_{c,l} = \{\}, \forall c$
  if $t = 1$ **then**
    $\mathcal{D}_{c,l} \leftarrow \mathcal{D}_{c,l} \cup \{\|M_{l:}^t\|\}$
  else if $t > 1$ **then**
    $\mathcal{D}_{c,l} \leftarrow \mathcal{D}_{c,l} \cup \left\{\|M_{l:}^t - M_{c,l:}^t\|\right\}$
  end if
  if $|\mathcal{D}_{c,l}| = n_w$ **then**
    Obtain $\beta_{l,c}$ based on (19) and (20)
  end if
  if $\beta_{c,l} < \tau$ **then**
    Freeze the parameters of layer l in $M_{c,LF}^{t+1}$
  else
    Update the parameters of layer l in $M_{c,LF}^{t+1}$
  end if
end for
Return $M_{c,LF}^{t+1}$

C. Client Clustering Selection (CCS) on the Server Side

To minimize server waiting latency, we design a Client Clustering Selection (CCS) scheme. Initially, the server UAV calculates the utility of each client UAV, arranging them in descending order of utility to establish a sequence that reflects their utility. Subsequently, the server UAV divides all client UAVs into G clusters, with each cluster's size being equal to K. For the t-th round, the server randomly selects a cluster from the set of available client clusters, denoted as $\mathcal{G} = \{g_1, \ldots, g_G\}$. In this framework, the utility function is defined as $u_c = \frac{1}{\mathbb{E}\{T_c^{\text{tr}}\}}$, where u_c signifies the utility of the c-th client UAV. Considering that C is not necessarily an integer multiple of K, meaning the number of client UAVs in the G-th cluster may be less than K. In such cases, we designate the G-th cluster as consisting of K client UAVs with the lowest utility. The details of the CCS scheme are presented in Algorithm 3.

To evaluate the efficacy of our CCS algorithm, we provide the following theorem:

Theorem *1:* In the context of semi-synchronous FL, let $\mathcal{C}$ denote the complete set of client UAVs, and $\mathcal{C}_m$ denote the subset of straggler client UAVs. Denote the cardinality of $\mathcal{C}$ by C and that of $\mathcal{C}_m$ by m. The probabilities $P_{\text{RS}}(A_{\text{last}})$ and $P_{\text{CCS}}(A_{\text{last}})$ represent the likelihood of selecting at least one straggler client from $\mathcal{C}_m$ when the server randomly selects K clients for each round under a standard random scheduling strategy and our proposed CCS algorithm, respectively. Then,

Algorithm 3 Client Clustering Selection (CCS) Function

Require: $\mathcal{C}, C, K$
Ensure: Γ_t
Calculate the utility of all clients, $\mathbf{U} = \{u_1, u_2, \ldots, u_c\}$
Re-arrange the utility in descending order as $\tilde{\mathbf{U}}$
if $C \bmod K = 0$ **then**
  Arrange all client UAVs into G clusters based on $\tilde{\mathbf{U}}$, and obtain the sets of client clusters $\mathcal{G} = \{g_1, \ldots, g_G\}, G = \lceil \frac{C}{K} \rceil$
else
  Arrange all clients into G clusters based on $\tilde{\mathbf{U}}$, designate the G-th cluster as consisting of K client UAVs with the lowest utility, and obtain the sets of client clusters $\mathcal{G} = \{g_1, \ldots, g_G\}, G = \lceil \frac{C}{K} \rceil$
end if
Randomly select one client cluster and assign it to Γ_t
Return Γ_t

we have

$$\begin{aligned} P_{\text{RS}}(A_{\text{last}}) &= 1 - \frac{\binom{C-m}{K}}{\binom{C}{K}} \\ P_{\text{CCS}}(A_{\text{last}}) &= \frac{a}{\lceil \frac{C}{K} \rceil} \leq \frac{aK}{C}, \\ P_{\text{CCS}}(A_{\text{last}}) &\leq P_{\text{RS}}(A_{\text{last}}), \forall C, K, m \geq 1, \end{aligned} \tag{21}$$

where a denotes the number of client UAV clusters that contain at least one straggler client UAV.

Proof: Please refer to Appendix A. ■

Theorem 1 posits that the proposed CCS method effectively reduces the probability of selecting at least one straggler UAV from the set of straggler client UAVs, thereby leading to a reduction in server waiting latency within the framework of semi-synchronous FL scenarios.

D. Overall Algorithm

The overall design of the proposed the Semi-synchronous Federated Learning with Cross-Sharpness and Layer-Freezing (SFedXL) algorithm is illustrated in Algorithm 4.

Computational Complexity Analysis: The computational complexity of XSL is determined by three key steps. i) Adversarial Perturbation Generation: This involves computing the gradient of the loss function w.r.t. model parameters, whose computational complexity is $O(N_{c,l}|M_c|)$, with $N_{c,l}$ and $|M_c|$ denoting the number of labeled data points and the amount of model parameters at the c-th client UAV, respectively; ii) Cross-Sharpness Regularization: This requires forward propagation on unlabeled data, whose computational complexity is $O(N_{c,u}|M_c|)$, with $N_{c,u}$ denoting the number of unlabeled data points at the c-th client UAV. iii) Local Model Updates: This involve gradient computation and parameter updates, whose computational complexity is $O(N_{c,l}|M_c|)$.The computational complexity of CWLF is determined by four key steps. i) Historical Parameter Update Monitoring: This involves computing the Euclidean norm for each layer's parameter updates, whose complexity is $O(L|M_{c,l}|)$, with L denoting the total

Algorithm 4 Semi-synchronous Federated Leaning with Cross-Sharpness and Layer-Freezing (SFedXL) Algorithm

Require: $\mathcal{C}, D_{c,l}, \forall c, D_{c,u}, \forall c, C, K, n, T, \tau, M^0$
Ensure: M^T
for $t = 0$ to $T-1$ **do**
  Server UAV calls **CCS** Function to select the client set Γ_t;
  Server UAV sends M^t to all selected clients in Γ_t
  Server UAV set the maximum waiting time $\tilde{T}_t$
  for client UAV $c \in \Gamma_t$ in parallel **do**
    Call **XSL** Function to perform local training
    Call **CWLF** Function to freezing layers
    Upload $M^t_{c,LF}$ to the server UAV
  end for
  while $\Gamma'_t < n$ OR $T_t < \tilde{T}_t$ **do**
    Server UAV records client's latency
    Server UAV calculates client's utility
  end while
  Server UAV updates the global model based on (4)
end for
Return M^T

number of layers and $|M_{c,l}|$ denoting the amount of parameters in the l-th layer at the c-th client UAV. ii) Normalized Moving Average Calculation: This smooths fluctuations in historical updates over a window of length n_w, whose complexity is $O(Ln_w)$. iii) Gradient Calculation and Freezing Decision: This involves computing the gradient of historical updates over the moving window n_w, whose complexity is $O(Ln_w)$. iv) Parameter Upload: Only unfrozen layers' parameters are uploaded. The complexity is $O((1-\tau)|M_{c,l}|)$, where τ is the freezing threshold. The computation complexity of the CCS strategy is extremely low, e.g., $O(C)$, where C is the number of client UAVs. Therefore, the total complexity of SFedXL is $O(TN|M| + TLn_w + CT))$, where $N = \sum_{c=1}^{C} N_{c,u} + N_{c,l}$ denoting the total data points, and $|M|$ denotes the amount of model parameters.

Resource Efficiency Analysis: As for computational efficiency, while XSL introduces additional computational overhead by utilizing unlabeled data to optimize model performance, the CWLF mechanism effectively minimizes unnecessary parameter updates, thereby reducing overall computational requirements. As for communication Efficiency, the CWLF mechanism reduces communication overhead by limiting the number of parameters that need to be uploaded. In FL environments, where communication overhead often becomes a bottleneck, the CWLF mechanism mitigates this issue by dynamically freezing parameters. As for storage efficiency, the CWLF mechanism retains only the update history of unfrozen layers, thus reducing storage requirements and enhancing storage efficiency. The computation overhead of the CCS strategy is extremely low and does not consume communication or storage resources. In summary, the integration of XSL, CWLF and CCS mechanisms significantly optimizes resource utilization across computation, communication, and storage. This advantage is especially pronounced in scenarios involving large-scale datasets and complex models.

IV. Simulation Results

In this work, we focus on the image classification task for a fleet of UAVs and utilize two prominent open-source datasets: SAT6 [36] and FLAME [37]. We consider 1 server UAV and 10 client UAVs for data collection and participation in the training process, spanning a total of 100 global training rounds. The number of client UAVs selected per round is fixed at 50% of the total client population, denoted as $K = 0.5 \times C$. The CPU frequency ϕ_c of each client UAV is distributed according to a Gaussian distribution with a mean of 2.5 GHz and a standard deviation of 0.25 GHz.

All UAVs follow a random waypoint model [38] with a speed of 10 m/s and pause times uniformly distributed between 0 and 10 seconds, effectively simulating dynamic trajectories and fluctuating communication distances. Each client UAV is allocated a bandwidth B of 50 MHz for communication. The channel is allocated to each user by Time Division Multiple Access (TDMA). The transmission power $P_{\text{c, tx}}$ of the users is modeled by a normal distribution $\mathcal{N}(0.1, 0.01)$. The dropout probability ρ_c for each client UAV adheres to a uniform distribution $U(0, 0.5)$. In our experiments, the path loss exponent γ is configured to be 2.5.

The hyperparameters β, η, and τ are crucial in determining the performance of SFedXL. Their effects can be summarized as follows: i) Robustness control parameter β: Regulates the stability of the worst-case model. A smaller β enhances stability, while a larger β improves robustness but may introduce instability; ii) Learning rate η: Balances convergence speed and stability. A smaller η ensures steady convergence, whereas a larger η accelerates training but increases the risk of divergence; iii) Layer freezing threshold τ: Determines the extent of layer freezing. A smaller τ conserves resources but may lead to premature freezing, while a larger τ ensures thorough training at the expense of higher resource consumption. By carefully tuning these hyper-parameters, we achieve an optimal balance between model performance and resource efficiency, enabling effective and scalable FL. Specifically, β, η, and τ are set as 0.01, 0.001 and 0.1, respectively.

A. Experiment Setup

1) Dataset: The SAT6 dataset comprises 405,000 satellite imagery samples spanning six distinct categories—barren land, trees, grassland, roads, buildings, and water bodies—which are prevalent in remote sensing and classification tasks. These 28x28 pixel images, inclusive of RGB and infrared channels, serve as a robust platform for evaluating FL algorithms, particularly those grappling with challenges such as scarce labeling and data imbalance. The FLAME dataset, developed to assess FL algorithms under adversarial conditions, encompasses labeled data suitable for both binary and multi-class classification tasks. It is a crucial benchmark for assessing model resilience in environments characterized by data heterogeneity.

2) Data Split: We randomly and uniformly select ρ percent of the entire labeled dataset as labeled data. In this study, we set $\rho = 5\%, 10\%, 100\%$. To model data heterogeneity, we use the commonly-adopted Dirichlet distribution. The variable $\epsilon_{q,c}$

represents the proportion of data instances with the label q at the c-th client UAV, relative to the total number of instances labeled q. The probability vector $\epsilon_q = (\epsilon_{q,1}, \epsilon_{q,2}, \ldots, \epsilon_{q,C})$ follows a Dirichlet distribution, i.e., $\epsilon_q \sim \text{Dir}_C(\alpha)$, where $\text{Dir}_C(\alpha)$ denotes the Dirichlet distribution across C client UAVs, and α is the concentration parameter. A smaller α reflects higher data heterogeneity, while $\alpha = \infty$ represents a scenario with homogeneous data. In this study, we set $\alpha = 0.1$ for all $q \in \{1, \ldots, Q\}$, where Q is the total number of label types.

3) Performance Metrics: The final l earning accuracy, total learning latency, and total communication cost are used as performance metrics. Among these, the final learning accuracy is defined as:

$$\text{Acc} = \frac{N_{tp} + N_{tn}}{N_{tp} + N_{tn} + N_{fp} + N_{fn}}, \tag{22}$$

where N_{tp}, N_{tn}, N_{fp}, and N_{fn} represent the number of true positives, true negatives, false positives, and false negatives, respectively.

4) Comparison Algorithms: All experiments were implemented using PyTorch, and each result is reported as the average and standard deviation of five simulations. To evaluate the effectiveness of the proposed solution, we compare it with the following methods:

- **FedAvg [7]**: Serving as a baseline, the server UAV randomly selects different client UAVs at each round for model updates.
- **FedProx [16]**: A personalized variant of FedAvg, FedProx introduces a regularization term to control the deviation between local updates and the global model.
- **FedMatch [20]**: This method combines inter-client consistency with disjoint learning schemes. The former enables each client to focus on the unique features of its local data, avoiding a uniform feature set, while the latter ensures alignment in model updates across clients.
- **Polaris [27]**: This approach allows clients to update their local models independently and asynchronously, reducing latency by prioritizing clients with more reliable communication links.
- **ALF [28]**: This method optimizes communication efficiency by calculating the stability index of each layer's parameters in the DNN model. Layers are frozen when their stability index falls below a predefined threshold. Following the settings in [28], the stability threshold for layer freezing is set at 0.13.
- **Independent Training**: Each client UAV performs local training independently using its own labeled data. The overall learning accuracy is characterized by the average accuracy across all clients, while the learning latency for each round is determined by the slowest client.

B. Learning Performance Comparison

Table I and Table II summarize the primary experimental results on the FLAME and SAT6 datasets, comparing various FL strategies at different labeled data ratios. The performance metrics include the final learning accuracy, total learning latency (in minutes), and total communication cost (in MB), along with their deviations from the baseline, i.e., the classical FedAvg algorithm.

1) Learning Accuracy Comparison: At a labeled data ratio of 5%, the FedAvg baseline achieves an accuracy of 74.97%. FedProx shows a marginal improvement of 0.14%, while other algorithms, such as FedMatch, Polaris, and ALF, exhibit slight decreases in accuracy. In contrast, our proposed SFedXL algorithm significantly enhances accuracy by 4.91%, a result of the XSL strategy, which effectively leverages the unlabeled data. Independent training performs the worst, highlighting the benefits of FL-based collaboration across devices. Similar trends are observed for labeled data ratios of 10% and 100%. Furthermore, as the proportion of labeled data increases, all algorithms improve in accuracy. However, the performance gaps between our algorithm and the baseline narrow, indicating that our XSL strategy excels when a substantial amount of unlabeled data is available.

2) Training Latency Comparison: In terms of latency, FedAvg almost always experiences the least total learning latency due to its simplicity. In contrast, FedProx incurs additional local training latency due to the computation of a regularization term to mitigate local-global model divergence. Similarly, FedMatch incurs extra latency for computing inter-client consistency. Polaris and ALF calculate the differences between the updated and previous models, and upload only the parameters that have changed significantly to reduce communication overhead. This reduction offsets any additional computation time, so the overall training delay is essentially unaffected. In contrast, independent training increases the total learning latency by approximately 15%-20% compared to FedAvg, primarily due to the impact of straggler users. Our approach, which capitalizes on unlabeled data, adds local computation latency but remains competitive in overall performance.

3) Communication Costs Comparison: When comparing communication costs, FedMatch increases costs by 173.76% over the baseline FedAvg, due to the transmission of additional auxiliary models. FedProx, Polaris, and ALF have a minimal impact on communication costs. The first two algorithms were not designed to reduce communication overhead, while ALF specifically targets communication efficiency in supervised learning scenarios. However, our experiments indicate that in contexts with large volumes of unlabeled and heterogeneous client data, a certain degree of discrepancy often exists between a client local model and the newly received global model. This discrepancy complicates efforts to maintain the parameter stability index below a predefined stability threshold before the algorithm finally converges. In contrast, our proposed SFedXL significantly reduces latency. This improvement can be attributed to two key factors. First, our proposed CWLF strategy effectively decreases both local computation and communication latencies. Additionally, our time utility-based client clustering and CCS strategies substantially reduce waiting times among users within the same communication round compared to random selection.

Similar trends are observed with the SAT6 dataset, where our algorithm demonstrates substantial improvements in learn-

TABLE I: Performance Comparison on FLAME Dataset under Different Strategies

Strategy	Accuracy(%)	ΔAccuracy	Latency (Mins)	ΔLatency	Commun. Cost (MB)	ΔCommun. Cost
	Labeled Data Ratio = 5.0%					
FedAvg [7] (baseline)	74.97±0.08	–	**72.57**±13.88	–	2845.66±0.0	–
FedProx [16]	75.11±0.11	0.14%	98.8±23.02	36.15%	2845.66±0.0	0.0%
FedMatch [20]	74.64±0.11	-0.33%	286.62±63.26	294.96%	7790.34±733.09	173.76%
Polaris [27]	74.72±0.07	-0.25%	73.78±12.44	1.67%	2845.66±0.0	0.0%
ALF [28]	74.13±0.17	-0.84%	74.78±14.05	3.05%	2845.65±0.02	-0.0%
Independent Training	35.96±0.74	-39.01%	82.85±19.24	14.17%	–	–
SFedXL (ours)	**79.88**±0.28	**4.91**%	222.51±63.54	206.62%	**1739.43**±470.95	**-38.87** %
	Labeled Data Ratio = 10.0%					
FedAvg [7] (baseline)	79.09±0.25	–	**126.22**±13.02	–	2845.66±0.0	–
FedProx [16]	78.99±0.22	-0.1%	187.97±27.82	48.92%	2845.66±0.0	0.0%
FedMatch [20]	78.47±0.32	-0.62%	573.19±51.11	354.11%	7654.16±849.95	168.98%
Polaris [27]	77.9±0.2	-1.19%	131.4±10.45	4.1%	2845.66±0.0	0.0%
ALF [28]	78.02±0.26	-1.07%	130.5±11.41	3.39%	2845.64±0.02	-0.0%
Independent Training	39.66±0.74	-39.43%	150.56±18.41	19.28%	–	–
SFedXL (ours)	**82.35**±0.2	**3.26**%	403.0±98.07	219.28%	**1824.18**±511.97	**-35.9** %
	Labeled Data Ratio = 100.0%					
FedAvg [7] (baseline)	82.95±0.33	–	880.54±130.24	–	2845.66±0.0	–
FedProx [16]	83.61±0.7	0.66%	1267.27±161.35	43.92%	2845.66±0.0	0.0%
FedMatch [20]	82.77±0.45	-0.18%	4111.23±380.65	366.9%	7550.85±845.6	165.35%
Polaris [27]	84.12±0.36	1.17%	**872.74**±90.93	**-0.89**%	2845.66±0.0	0.0%
ALF [28]	82.87±0.65	-0.08%	884.08±140.69	0.4%	2845.64±0.01	-0.0%
Independent Training	42.99±1.02	-39.96%	1065.61±154.39	21.02%	–	–
SFedXL (ours)	**84.31**±0.48	**1.36**%	2935.77±398.09	233.4%	**1964.38**±571.47	**-30.97** %

TABLE II: Performance Comparison on SAT6 Dataset under Different Strategies

Strategy	Accuracy(%)	ΔAccuracy	Latency (Mins)	ΔLatency	Commun. Cost(MB)	ΔCommun. Cost
	Labeled Data Ratio = 5.0%					
FedAvg [7] (baseline)	66.14±0.21	–	**90.13**±5.06	–	2028.25±0.0	–
FedProx [16]	69.77±0.55	3.63%	136.99±8.98	51.99%	2028.25±0.0	0.0%
FedMatch [20]	70.04±0.9	3.9%	257.2±19.79	185.36%	5412.0±574.77	166.83%
Polaris [27]	61.93±1.13	-4.21%	94.42±6.12	4.76%	2028.25±0.0	0.0%
ALF [28]	67.08±0.41	0.94%	91.81±6.64	1.86%	2028.23±0.01	-0.0%
Independent Training	29.19±0.7	-36.95%	94.69±6.64	5.06%	–	–
SFedXL(ours)	**82.98**±0.52	**16.84**%	190.49±16.17	111.35%	**1542.02**±295.61	**-23.97** %
	Labeled Data Ratio = 10.0%					
FedAvg [7] (baseline)	74.37±0.48	–	166.22±24.3	–	2028.25±0.0	–
FedProx [16]	75.31±0.72	0.94%	257.35±29.81	54.83%	2028.25±0.0	0.0%
FedMatch [20]	74.61±0.42	0.24%	473.02±55.12	184.58%	5468.9±581.62	169.64%
Polaris [27]	77.43±0.51	3.06%	**165.56**±19.28	**-0.39**%	2028.25±0.0	0.0%
ALF [28]	76.04±0.46	1.67%	169.17±21.7	1.78%	2028.24±0.01	-0.0%
Independent Training	34.98±1.14	-39.39%	178.15±23.93	7.18%	–	–
SFedXL(ours)	**84.7**±0.27	**10.33**%	334.73±39.24	101.38%	**1429.38**±257.4	**-29.53** %
	Labeled Data Ratio = 100.0%					
FedAvg [7] (baseline)	82.13±0.37	–	**1436.95**±238.55	–	2028.25±0.0	–
FedProx [16]	79.66±0.81	-2.47%	2428.58±411.08	69.01%	2028.25±0.0	0.0%
FedMatch [20]	80.82±0.65	-1.31%	4173.46±475.54	190.44%	5418.69±477.13	167.16%
Polaris [27]	80.43±0.59	-1.7%	1459.29±240.19	1.55%	2028.25±0.0	0.0%
ALF [28]	77.8±0.86	-4.33%	1476.43±248.8	2.75%	2028.24±0.01	-0.0%
Independent Training	44.07±0.98	-38.06%	1610.59±357.31	12.08%	–	–
SFedXL(ours)	**85.24**±0.71	**3.11%**	2944.22±521.65	104.89%	**1405.78**±371.02	**-30.69** %

ing accuracy and reduced communication costs. Although training latency increases due to the incorporation of unlabeled data, the significant performance gains achieved by utilizing unlabeled data cannot be overlooked. This contribution plays a pivotal role in the overall success of our algorithm, as it contributes significantly to the overall performance improvement

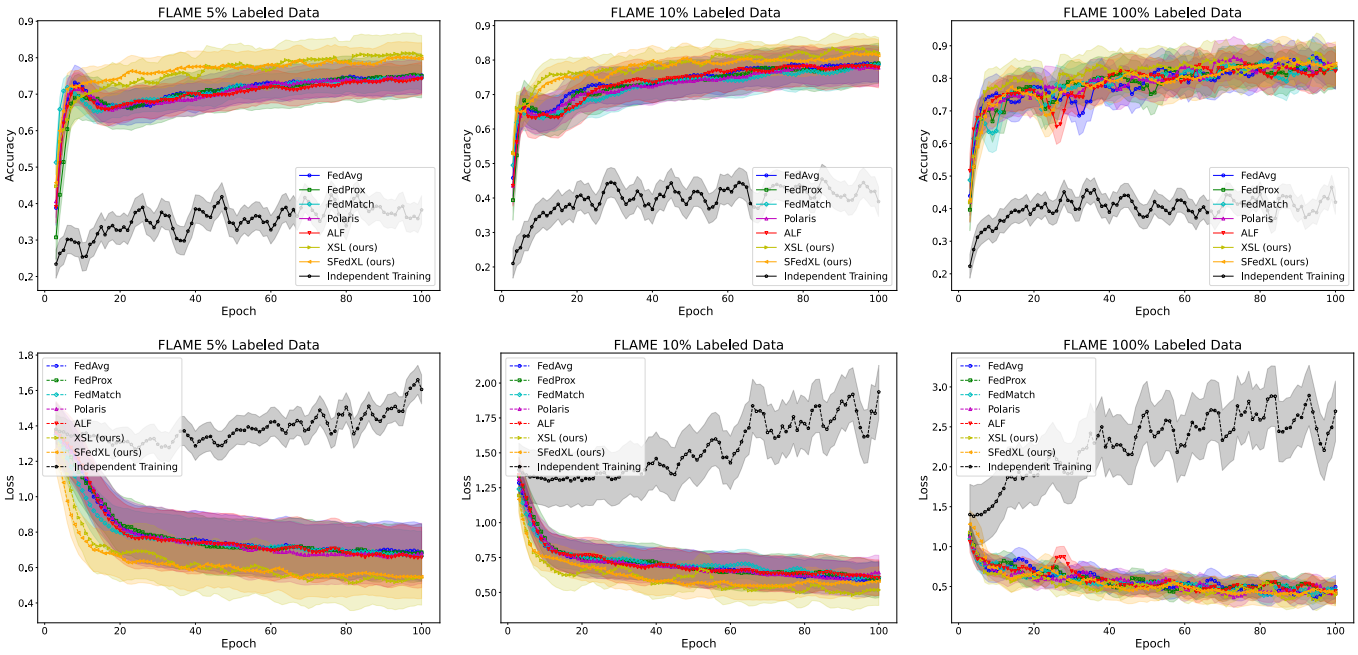


Fig. 3: Convergence performance comparison under different algorithms on FLAME dataset.

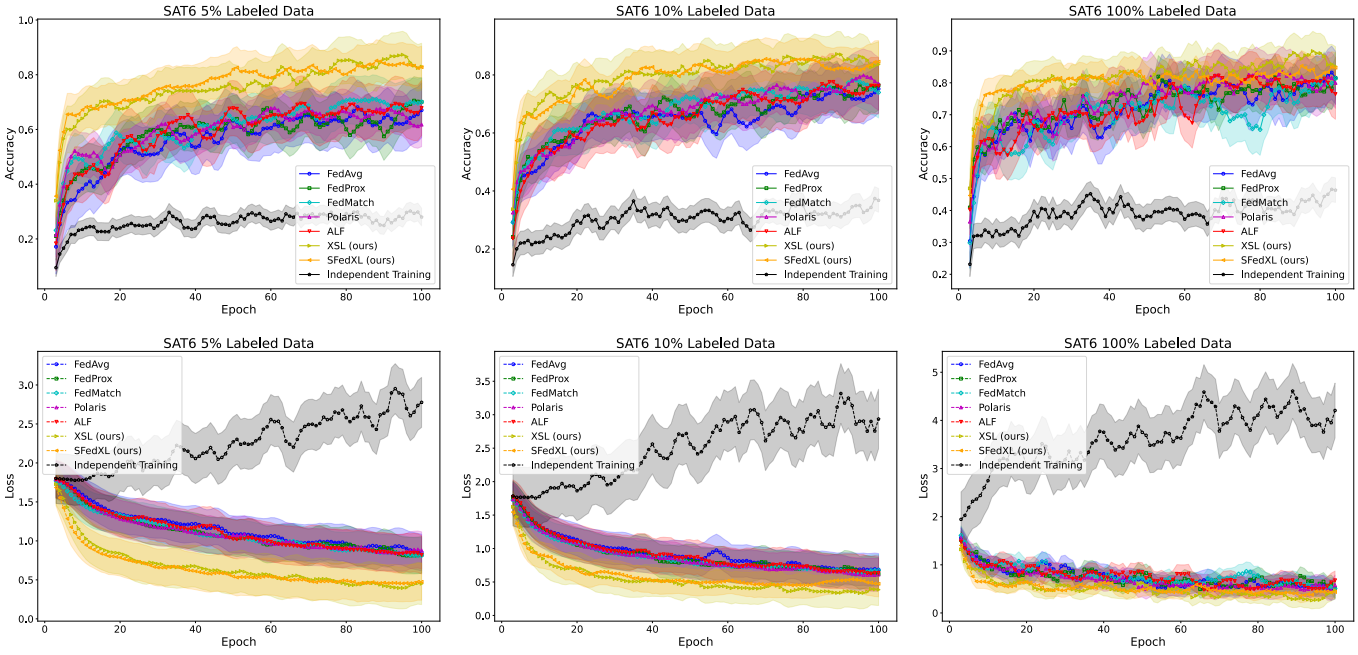


Fig. 4: Convergence performance comparison under different algorithms on SAT6 dataset.

of the algorithm.

C. Algorithm Convergence Comparison

The experimental results shown in Fig. 3 and Fig. 4 highlight the performance of various algorithms on FLAME and SAT6 datasets, tested with different percentages of labeled data. The sub-figures in the top row depict the learning accuracy over 100 communication round, while the charts in the bottom row illustrate loss during each round.

1) Accuracy Convergence Comparison: Regardless of the labeled data proportion, the proposed XSL and SFedXL methods consistently demonstrate superior learning accuracy and faster convergence rates, with minimal variance in fluctuations. Notably, the SFedXL method performs on par with the XSL method, indicating that the CWLF strategy—despite freezing certain parameters—does not significantly affect performance. In contrast, algorithms that do not leverage unlabeled data, such as FedProx, FedMatch, Polaris, and ALF, show performance levels nearly identical to the baseline FedAvg algo-

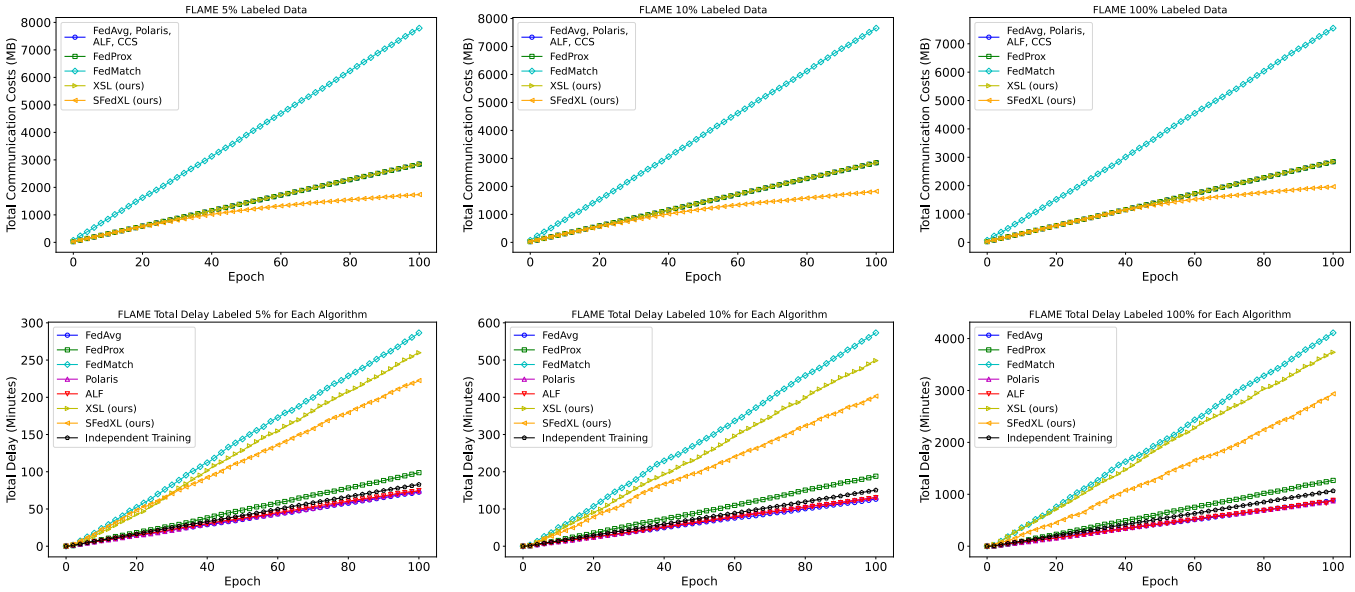


Fig. 5: Learning cost & latency performance comparison under different algorithms on FLAME dataset.

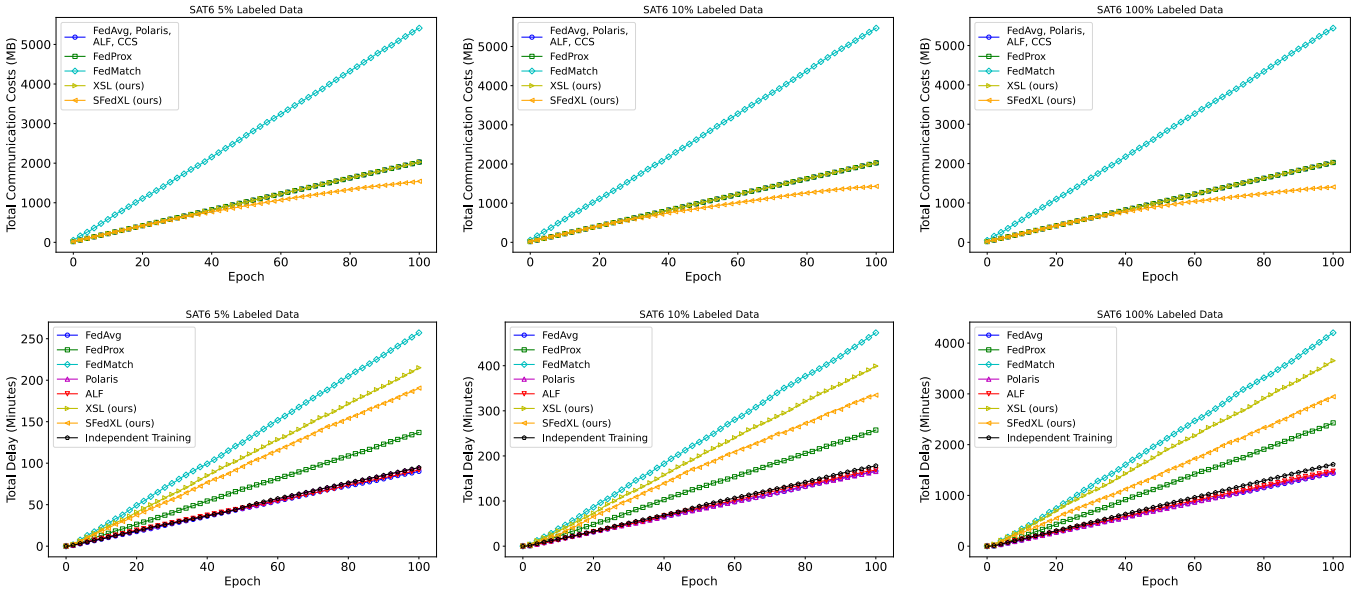


Fig. 6: Learning cost & latency performance comparison under different algorithms on SAT6 dataset.

rithm, with no notable improvements. This suggests that these FL strategies, which are designed for supervised learning, are not well-suited for UAV swarm environments where unlabeled data is abundant. Additionally, the independent training model for each UAV performs the worst, likely due to data heterogeneity and challenges in utilizing unlabeled data effectively.

2) Loss Convergence Comparison: In terms of loss convergence, the XSL and SFedXL methods outperform others by consistently achieving lower loss values across different labeled data ratios, demonstrating learning consistency for both labeled and unlabeled data. Similar to the accuracy trends, other algorithms fail to show significant improvements in loss reduction compared to the FedAvg baseline. While the Independent Training method initially experiences a noticeable decrease in loss, it often plateaus or decreaes, particularly when labeled data is limited. This is likely due to overfitting or poor generalization, a result of the lack of collaborative knowledge sharing typical of FL frameworks. The loss curves for independent training also exhibit signs of overfitting.

In real world, data is collected in a decentralized manner and distributed across multiple UAVs in the swarm, often leading to non-IID data due to variations in distribution and quality. This heterogeneity can cause models to overfit to data from specific UAVs while underfitting to others, negatively impacting both learning convergence and accuracy. Furthermore, discrepancies between the local data of individual UAVs and the test dataset's distribution can be significant. As a result, models trained independently by a single UAV, relying only

TABLE III: Ablation Study on FLAME Dataset under Different Strategies

Strategy			Accuracy(%)	ΔAccuracy	Latency (Mins)	ΔLatency	Commun. Cost(MB)	ΔCommun. Cost
XSL	CWLF	CCS	Labeled Data Ratio = 5.0%					
			74.97±0.08	–	72.57±13.88	–	2845.66±0.0	–
✓			80.77±0.21	5.8%	259.84±67.94	258.06%	2845.66±0.0	0.0%
	✓		73.93±0.14	-1.04%	65.67±14.33	-9.51%	1545.98±475.98	-45.67%
		✓	76.13±0.15	1.16%	63.69±11.04	-12.23%	2845.66±0.0	0.0%
✓	✓		80.18±0.26	5.21%	232.47±75.07	220.35%	1579.71±410.24	-44.49%
✓		✓	**80.98**±0.23	**6.01**%	239.68±69.49	230.28%	2845.66±0.0	0.0%
	✓	✓	74.84±0.24	-0.13%	**53.96**±6.96	**-25.64**%	**1469.03**±710.21	**-48.38** %
	Strategy		Labeled Data Ratio = 10.0%					
			79.09±0.25	–	126.22±13.02	–	2845.66±0.0	–
✓			82.35±0.26	3.26%	471.51±41.69	273.55%	2845.66±0.0	0.0%
	✓		77.96±0.16	-1.13%	122.24±11.61	-3.16%	1953.76±392.81	-31.34%
		✓	79.28±0.2	0.19%	104.79±19.74	-16.98%	2845.66±0.0	0.0%
✓	✓		81.56±0.24	2.47%	446.99±38.77	254.13%	1850.23±526.75	-34.98%
✓		✓	**82.96**±0.17	**3.87**%	421.03±18.35	233.56%	2845.66±0.0	0.0%
	✓	✓	79.1±0.27	0.01%	**97.56**±11.43	**-22.71**%	**1482.15**±617.12	**-47.92** %
	Strategy		Labeled Data Ratio = 100.0%					
			82.95±0.33	–	880.54±130.24	–	2845.66±0.0	–
✓			83.71±0.61	0.76%	3735.7±584.89	324.25%	2845.66±0.0	0.0%
	✓		80.86±0.23	-2.09%	781.46±102.12	-11.25%	1707.33±414.2	-40.0%
		✓	83.7±0.45	0.75%	768.22±108.24	-12.76%	2845.66±0.0	0.0%
✓	✓		83.0±0.42	0.05%	3642.07±508.35	313.62%	1960.79±462.89	-31.1%
✓		✓	**84.59**±0.5	**1.64**%	3117.59±663.62	254.05%	2845.66±0.0	0.0%
	✓	✓	83.57±0.16	0.62%	**657.17**±113.78	**-25.37**%	**1502.77**±440.2	**-47.19** %

TABLE IV: Ablation Study on SAT6 Dataset under Different Strategies

Strategy			Accuracy(%)	ΔAccuracy	Latency (Mins)	ΔLatency	Commun. Cost(MB)	ΔCommun. Cost
XSL	CWLF	CCS	Labeled Data Ratio = 5.0%					
			66.14±0.21	–	90.13±5.06	–	2028.25±0.0	–
✓			83.1±0.48	16.96%	215.03±5.85	138.57%	2028.25±0.0	0.0%
	✓		65.68±0.7	-0.46%	82.87±6.51	-8.06%	1433.78±262.72	-29.31%
		✓	70.45±0.64	4.31%	82.67±1.88	-8.28%	2028.25±0.0	0.0%
✓	✓		82.49±0.47	16.35%	209.34±12.96	132.26%	1511.44±305.1	-25.48%
✓		✓	**85.86**±0.31	**19.72**%	193.15±8.5	114.3%	2028.25±0.0	0.0%
	✓	✓	66.59±0.19	0.45%	**74.17**±4.24	**-17.71**%	**1158.45**±413.97	**-42.88** %
	Strategy		Labeled Data Ratio = 10.0%					
			74.37±0.48	–	166.22±24.3	–	2028.25±0.0	–
✓			84.04±0.65	9.67%	397.87±42.27	139.37%	2028.25±0.0	0.0%
	✓		73.48±0.36	-0.89%	146.99±17.2	-11.57%	1507.95±298.76	-25.65%
		✓	77.77±0.41	3.4%	144.41±20.3	-13.12%	2028.25±0.0	0.0%
✓	✓		83.37±0.21	9.0%	387.33±55.86	133.03%	1433.64±347.74	-29.32%
✓		✓	**87.03**±0.43	**12.66**%	350.53±78.93	110.89%	2028.25±0.0	0.0%
	✓	✓	76.3±0.55	1.93%	**137.22**±18.31	**-17.44**%	**1302.94**±335.32	**-35.76** %
	Strategy		Labeled Data Ratio = 100.0%					
			82.13±0.37	–	1436.95±238.55	–	2028.25±0.0	–
✓			86.26±0.48	4.13%	3653.64±480.84	154.26%	2028.25±0.0	0.0%
	✓		79.7±0.62	-2.43%	1319.89±161.58	-8.15%	1484.49±241.84	-26.81%
		✓	83.24±0.53	1.11%	1188.01±191.3	-17.32%	2028.25±0.0	0.0%
✓	✓		85.14±0.47	3.01%	3468.13±484.99	141.35%	1368.4±269.95	-32.53%
✓		✓	**89.21**±0.4	**7.08**%	3280.69±622.96	128.31%	2028.25±0.0	0.0%
	✓	✓	82.96±0.35	0.83%	**1163.03**±222.64	**-19.06**%	**1272.41**±264.86	**-37.27** %

on local data, tend to perform poorly on the test set. The loss function curve for independent training can also be volatile, sometimes failing to converge.

D. Ablation Study

Table III and Table IV present the results of the ablation study conducted on the FLAME and SAT6 datasets, with

FedAvg serving as the baseline. We performed a comparative analysis of various combinations of the proposed modules, evaluating their effectiveness in terms of achieving the final learning accuracy after 100 communication rounds, as well as their respective training latencies and communication costs.

First, it is evident that employing the XSL strategy alone significantly improves learning accuracy without increasing communication overhead. However, incorporating unlabeled data during local model training extends the local training time. Second, the CWLF strategy, despite freezing certain parameters, results in only a slight and imperceptible decline in accuracy. Moreover, it significantly reduces both training latency and communication costs. Third, applying the CCS strategy independently not only slightly improves learning accuracy but also drastically reduces training latency, without affecting communication overhead. The accuracy improvement can be attributed to the design of the utility function in client cluster scheme, which is a function concerning frozen parameters. In this context, the users with a smaller utility value typically exhibit a larger magnitude of model parameter updates. In the FedAvg algorithm, which employs random scheduling, these low-utility users are often overlooked, as they are seen as potential stragglers in each round. In contrast, the CCS strategy ensures that clients with lower utility values and larger parameter updates are more likely to contribute to model updates, thereby enhancing the accuracy and expediting the convergence.

When evaluating the individual contributions of each algorithm, it becomes clear that the combination of XSL and CCS yields the highest learning accuracy. Meanwhile, the integration of CCS and CWLF consumes the least training time and communication resources. Compared to the baseline FedAvg algorithm, our SFedXL framework effectively balances the strengths of the XSL, CWLF, and CCS strategies, striking a balance between learning performance and resource efficiency, as well as encompassing both training latency and communication overhead.

V. Conclusion

Recently, FL has emerged as a promising solution for coordinating a UAV swarm. However, its practical implementation is hindered by challenges such as device heterogeneity, restricted resources, and the presence of unlabeled data. To address these issues, we have proposed a novel SFedXL algorithm. Specifically, the proposed XSL scheme is designed to effectively utilize unlabeled data, the CWLF scheme mitigates training latencies and reduces communication cost, and the CCS scheme addresses the straggler problem. Simulation results clearly demonstrate that our methods outperform traditional FL approaches in both learning accuracy and learning efficiency, underscoring the potential of our SFedXL framework as a robust solution for handling unlabeled data and asynchronous devices in unstable transmission environments.

Appendix

To prove Theorem 1, we need to show that whenever $m \geq 1$, $K \geq 1$, and $C \geq 1$, the following inequality always holds:

$$\frac{aK}{C} \leq 1 - \frac{\binom{C-m}{K}}{\binom{C}{K}} \tag{23}$$

To prove Theorem 1, we consider the following two cases: **Case 1:** When $1 \leq C < K + m$, the FL server will always select at least one straggler client since we consider client selection without replacement, which means

$$P_{\text{RS}}(A_{\text{last}}) = P_{\text{CCS}}(A_{\text{last}}) = 1, \forall K, m \geq 1.$$

Case 2: When $1 \leq K + m \leq C$, if $m = 1$, we have $a = 1$ and

$$\begin{aligned} P_{\text{CCS}}(A_{\text{last}}) &= \frac{K}{C}, \\ P_{\text{RS}}(A_{\text{last}}) &= 1 - \frac{(C-1)!}{K!(C-1-K)!} \times \frac{K!(C-K)!}{C!} \\ &= 1 - \frac{(C-K)}{C} = \frac{K}{C}, \\ &= P_{\text{CCS}}(A_{\text{last}}); \end{aligned}$$

If $m > 1$, $K \geq 1$, and $C \geq K + m \geq 1$, under our CCS Algorithm, the straggler clients are grouped into the last a clusters, where $a = \lceil \frac{m}{K} \rceil$. Proving the inequality in (23) is equivalent to proving

$$\frac{(C-m)!}{K!(C-m-K)!} \times \frac{K!(C-K)!}{C!} < \frac{C-aK}{C}.$$

Since $C \geq K + m \geq 1$, we have $C - aK > 0$. According to the factorial formula, we have

$$\frac{(C-K)!}{C!} = \prod_{i=0}^{m-1} \frac{C-i-K}{C-i} \times \frac{(C-m-K)!}{(C-m)!}, \tag{24}$$

$$< \prod_{i=0}^{m-1} \frac{C-aK}{C} \times \frac{(C-m-K)!}{(C-m)!} < 1 \tag{25}$$

Thus, for $m > 1$, $K \geq 1$, and $C \geq K + m \geq 1$, the inequality in (23) holds. So far, we have

$$P_{\text{CCS}}(A_{\text{last}}) \leq \frac{aK}{C} \leq 1 - \frac{\binom{C-m}{K}}{\binom{C}{K}} = P_{\text{RS}}(A_{\text{last}}), \forall C, K, m \geq 1,$$

which completes the proof.

References

[1] M. Zhao, S. Zhao, C. Feng, H. H. Yang, and T. Q. S. Quek, "SFedXSL: Semi-Synchronous Federated Cross-Sharpness Learning for UAV Swarm," in *IEEE Int. Conf. Commun. Techno. (ICCT)*, Chengdu, China, 2024.

[2] Z. Jia, Q. Wu, C. Dong, C. Yuen, and Z. Han, "Hierarchical aerial computing for internet of things via cooperation of HAPs and UAVs," *IEEE Internet Things J.*, vol. 10, no. 7, pp. 5676–5688, 2023.

[3] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated Learning for Internet of Things: A Comprehensive Survey," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 3, pp. 1622–1658, 2021.

[4] S. K. Pandya *et al.*, "Federated Learning for Smart Cities: A Comprehensive Survey," *Sustain. Energy Technol. Assess.*, vol. 55, p. 102987, 2023.

[5] E. V. Butilă and R. G. Boboc, "Urban Traffic Monitoring and Analysis Using Unmanned Aerial Vehicles (UAVs): A Systematic Literature Review," *Remote Sens.*, vol. 14, no. 3, p. 620, 2022.

[6] Y. Guan, S. Zou, H. Peng, W. Ni, Y. Sun, and H. Gao, "Cooperative UAV Trajectory Design for Disaster Area Emergency Communications: A Multi-Agent PPO Method," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 8848–8859, 2024.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks From Decentralized Data," in *Proc. Int. Conf. Artif. Intell. Stat. (AISTATS)*, Fort Lauderdale, Florida, USA, 2017, pp. 1273–1282.

[8] S. Wang, M. Chen, C. G. Brinton, C. Yin, W. Saad, and S. Cui, "Performance Optimization for Variable Bitwidth Federated Learning in Wireless Networks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 2340–2356, 2024.

[9] C. Feng, H. H. Yang, S. Wang, Z. Zhao, and T. Q. S. Quek, "Hybrid Learning: When Centralized Learning Meets Federated Learning in the Mobile Edge Computing Systems," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 7008–7022, 2023.

[10] Y. Wang, S. Guo, Y. Deng, H. Zhang, and Y. Fang, "Privacy-Preserving Task-Oriented Semantic Communications Against Model Inversion Attacks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 8, pp. 10 150–10 165, 2024.

[11] C. Feng, H. H. Yang, D. Hu, Z. Zhao, T. Q. S. Quek, and G. Min, "Mobility-Aware Cluster Federated Learning in Hierarchical Wireless Networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 10, pp. 8441–8458, 2022.

[12] Z. Cui, T. Yang, X. Wu, H. Feng, and B. Hu, "The Data Value based Asynchronous Federated Learning for UAV Swarm under Unstable Communication Scenarios," *IEEE Trans. Mob. Comput.*, pp. 1–15, 2023.

[13] T. Wang, X. Huang, Y. Wu, L. Qian, B. Lin, and Z. Su, "UAV swarm-assisted two-tier hierarchical federated learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 1, pp. 943–956, 2024.

[14] M. Dai, Y. Wu, L. Qian, Z. Su, B. Lin, and N. Chen, "UAV-assisted multi-access computation offloading via hybrid noma and fdma in marine networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 1, pp. 113–127, 2023.

[15] Z. Wang, Z. Zhang, Y. Tian, Q. Yang, H. Shan, W. Wang, and T. Q. S. Quek, "Asynchronous Federated Learning Over Wireless Communication Networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 6961–6978, 2022.

[16] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in *Proc. Mach. Learn. Syst. (MLSys)*, vol. 2, Santa Clara, USA, 2020, pp. 429–450.

[17] L. Zhou, S. Leng, Q. Wang, and Q. Liu, "Integrated Sensing and Communication in UAV Swarms for Cooperative Multiple Targets Tracking," *IEEE Trans. Mob. Comput.*, vol. 22, no. 11, pp. 6526–6542, 2023.

[18] C. Feng, D. Feng, G. Huang, Z. Liu, Z. Wang, and X.-G. Xia, "Robust Privacy-Preserving Recommendation Systems Driven by Multimodal Federated Learning," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2024.

[19] S. Li, Y. Wang, S. Guo, and C. Feng, "Task-oriented communication for graph data: A graph information bottleneck approach," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2024.

[20] W. Jeong, J. Yoon, E. Yang, and J. Shin, "Federated Semi-supervised Learning with Inter-client Consistency & Disjoint Learning," in *Proc. Int. Conf. Learn. Representations (ICLR)*, Virtual Conference, 2021.

[21] Y. Liu, H. Wu, and J. Qin, "FedCD: Federated Semi-Supervised Learning with Class Awareness Balance via Dual Teachers," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, vol. 38, no. 4, San Francisco, California, USA, 2024, pp. 3837–3845.

[22] A. Tashakori, W. Zhang, Z. Wang, G. Joshi, Q. Wang, and J. Zhang, "SemiPFL: Personalized semi-supervised federated learning framework for edge intelligence," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 9161–9176, 2023.

[23] C. Feng, A. Arafa, Z. Chen, M. Zhao, T. Q. S. Quek, and H. H. Yang, "Toward understanding federated learning over unreliable networks," *IEEE Trans. Machine Learning Commun. Network.*, vol. 3, pp. 80–97, 2025.

[24] Z. Jiang, Y. Xu, H. Xu, Z. Wang, J. Liu, Q. Chen, and C. Qiao, "Computation and Communication Efficient Federated Learning with Adaptive Model Pruning," *IEEE Trans. Mob. Comput.*, vol. 23, no. 3, pp. 2003–2021, 2024.

[25] Y. Mao, Z. Zhao, M. Yang, L. Liang, Y. Liu, W. Ding, T. Lan, and X.-P. Zhang, "SAFARI: Sparsity-Enabled Federated Learning with Limited and Unreliable Communications," *IEEE Trans. Mob. Comput.*, vol. 23, no. 5, pp. 4819–4831, 2024.

[26] S. Seo, J. Lee, H. Ko, J. Park, J. Kim, S. Yoon, and J. Kim, "Situation-Aware Cluster and Quantization Level Selection Algorithm for Fast Federated Learning," *IEEE Internet Things J.*, vol. 10, no. 15, pp. 13 292–13 302, 2023.

[27] Y. Kang and B. Li, "POLARIS: Accelerating Asynchronous Federated Learning with Client Selection," *IEEE Trans. Cloud Comput.*, vol. 12, no. 2, pp. 446–458, 2024.

[28] E. Malan, V. Peluso, A. Calimera, and E. Macii, "Automatic Layer Freezing for Communication Efficiency in Cross-Device Federated Learning," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6072–6083, 2023.

[29] Z. Huang, L. Shen, J. Yu, B. Han, and T. Liu, "FlatMatch: Bridging Labeled Data and Unlabeled Data with Cross-Sharpness for Semi-Supervised Learning," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, Vancouver, Canada, 2024.

[30] W. P. Amorim, E. C. Tetila, H. Pistori, and J. P. Papa, "Semi-Supervised Learning with Convolutional Neural Networks for UAV Images Automatic Recognition," *Comput. Electron. Agric.*, vol. 164, p. 104932, 2019.

[31] Z. Wei, M. Zhu, N. Zhang, L. Wang, Y. Zou, Z. Meng, H. Wu, and Z. Feng, "UAV-assisted Data Collection for Internet of Things: A Survey," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15 460–15 483, 2022.

[32] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying Semi-Supervised Learning with Consistency and Confidence," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, New Orleans, USA, 2020, pp. 596–608.

[33] J. Ouyang, Y. Liu, and H. Liu, "Two-Timescale Energy Optimization for Wireless Federated Learning," in *IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2024, pp. 1–6.

[34] D. Stripelis, P. M. Thompson, and J. L. Ambite, "Semi-Synchronous Federated Learning for Energy-Efficient Training and Accelerated Convergence in Cross-Silo Settings," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 5, pp. 1–29, 2022.

[35] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-Aware Minimization for Efficiently Improving Generalization," in *Int. Conf. Learn. Represent. (ICLR)*, Vienna, Austria, 2021, pp. 1–19.

[36] S. Basu, S. Ganguly, S. Mukhopadhyay, R. DiBiano, M. Karki, and R. Nemani, "Deepsat: A Learning Framework for Satellite Imagery," in *Proc. Int. Conf. Adv. Geogr. Inf. Syst. (SIGSPATIAL)*, 2015, pp. 1–10.

[37] A. Shamsoshoara, F. Afghah, A. Razi, L. Zheng, P. Z. Fulé, and E. Blasch, "Aerial Imagery Pile Burn Detection Using Deep Learning: The FLAME Dataset," *Comput. Netw.*, vol. 193, p. 108001, 2021.

[38] S. Althunibat, O. S. Badarneh, and R. Mesleh, "Random waypoint mobility model in space modulation systems," *IEEE Commun. Lett.*, vol. 23, no. 5, pp. 884–887, 2019.

Mingxiong Zhao (S'11-M'16)received the B.S. degree in Electrical Engineering and the Ph.D. degree in Information and Communication Engineering from South China University of Technology (SCUT), Guangzhou, China, in 2011 and 2016, respectively. He was a visiting Ph.D. student at University of Minnesota (UMN), Twin Cities, MN, USA, from 2012 to 2013 and Singapore University of Technology and Design (SUTD), Singapore, from 2015 to 2016, respectively. Since 2016, he has been with the School of Software, Yunnan University, Kunming, China, where he is currently an Associate Professor. His research interests include network security, mobile edge computing, and edge AI techniques.

Shihao Zhao received his B.E. degree in Computer Science and Technology from Kunming University of Science and Technology (KUST), Kunming, China, in 2022. Since 2022, he has been pursuing his master's degree in Software Engineering at the School of Software, Yunnan University, Kunming, China. His research interests include network security, mobile edge computing, and edge AI techniques.

Chenyuan Feng (S'16-M'21) received the B.E. degree in electrical and electronics engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016, and the Ph.D. degree in information system technology and design from Singapore University of Technology and Design (SUTD), Singapore, in 2021, respectively. Currently she is a research fellow at Eurecom, France. Her research interests include edge intelligence, multimedia intelligence, as well as AI for network and communication.

Dr. Feng is a receipt of the 2021 IEEE ComComAp Best Paper Award. Dr. Feng was invited to deliver several tutorials and invited talk at International conferences in the area of machine learning for communication, such as IEEE PIMRC'2024, IEEE VCC'2024, IEEE ICCT'2022 and IEEE ICCT'2024. Dr. Feng serves as an Associate Editor for the IEEE INTERNET OF THINGS JOURNAL and the IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY. Dr. Feng is a Marie Skłodowska-Curie Scholar.

Howard H. Yang (S'13–M'17) received the B.E. degree in Communication Engineering from Harbin Institute of Technology (HIT), China, in 2012, and the M.Sc. degree in Electronic Engineering from Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2013. He earned the Ph.D. degree in Electrical Engineering from Singapore University of Technology and Design (SUTD), Singapore, in 2017. He was a Postdoctoral Research Fellow at SUTD from 2017 to 2020, a Visiting Postdoc Researcher at Princeton University from 2018 to 2019, and a Visiting Student at the University of Texas at Austin from 2015 to 2016. Currently, he is an assistant professor at the Zhejiang University/University of Illinois at Urbana-Champaign Institute (ZJU-UIUC Institute), Zhejiang University, Haining, China. He is also an adjunct assistant professor with the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, IL, USA.

Dr. Yang's research interests include wireless communications, networking, signal processing, the modeling of modern wireless networks, high dimensional statistics, graph signal processing, and machine learning. He serves as an editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He received the IEEE ComSoc Asia Pacific Oustanding Young Researcher Award in 2023, the IEEE Signal Processing Society Best Paper Award in 2022, the IEEE WCSP 10-Year Anniversary Excellent Paper Award in 2019, and the IEEE WCSP Best Paper Award in 2014.

Dusit Niyato (S'99-M'08-F'21) received BEng degree from King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand, in 1999 and the Ph.D. degree in electrical and computer engineering from the University of Manitoba, Canada, in 2008. He is currently a professor in the School of Computer Science and Engineering, with Nanyang Technological University, Singapore. His research interests include the Internet of Things, machine learning, and incentive mechanism design.

Dr. Niyato serves as a senior editor of the IEEE WIRELESS COMMUNICATIONS LETTER, an area editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, an editor of IEEE TRANSACTIONS ON COMMUNICATIONS, an associate editor of the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and the IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING. He is a Fellow of IEEE.

Tony Q. S. Quek (S'98-M'08-SM'12-F'18) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 2008. Currently, he is the Cheng Tsang Man Chair Professor with Singapore University of Technology and Design (SUTD) and ST Engineering Distinguished Professor. He also serves as the Director of the Future Communications R & D Programme, the Head of ISTD Pillar, and the Deputy Director of the SUTD-ZJU IDEA. His current research topics include wireless communications and networking, network intelligence, non-terrestrial networks, open radio access network, and 6G.

Dr. Quek has been actively involved in organizing and chairing sessions, and has served as a member of the Technical Program Committee as well as symposium chairs in a number of international conferences. He is currently serving as an Area Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.

Dr. Quek was honored with the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards – Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, the 2020 IEEE Communications Society Young Author Best Paper Award, the 2020 IEEE Stephen O. Rice Prize, the 2020 Nokia Visiting Professor, and the 2022 IEEE Signal Processing Society Best Paper Award. He is the AI on RAN Working Group Chair in AI-RAN Alliance. He is a Fellow of IEEE, a Fellow of WWRF, a Fellow of the Academy of Engineering Singapore.