Federated Learning with Stage-wise Trajectory Matching for UAV Networks on Non-IID Data

Xiufang Shi, *Member, IEEE*, Wei Zhang, Yuheng Li, Mincheng Wu, *Member, IEEE*, Rushi Li, Shibo He, *Senior Member, IEEE*

***Abstract*—Federated learning enables multiple unmanned aerial vehicles (UAVs) to collaboratively train a global model while preserving data privacy. However, data heterogeneity across UAV swarms hinders model convergence and generalization. While existing trajectory matching methods address this problem by synthesizing distilled data from early-stage training trajectories, these trajectories are non-convergent and fail to capture the complete global distribution. To address this issue, we propose a novel federated learning framework based on stage-wise trajectory matching (FedSTM). This framework performs trajectory matching during both the early and late stages of training to synthesize distilled data that comprehensively represent the global distribution, and subsequently utilizes these distilled data to fine-tune the global model, effectively enhancing its generalization capability and stability. Additionally, we introduce a decorrelation regularization term during local training to mitigate dimensional collapse in the representation space, thereby improving the quality of the global model's trajectory. Extensive experimental results on multiple datasets demonstrate that the proposed method surpasses baseline approaches in model performance and convergence.**

***Index Terms*—Federated learning, unmanned aerial vehicles, dataset distillation, data heterogeneity**

I. Introduction

In recent years, with the rapid development of aviation technology and information technology, unmanned aerial vehicles (UAVs) have gradually become key drivers of emerging applications [1, 2], playing a critical role in fields such as environmental monitoring, disaster response, military security, and intelligent transportation. Using their flexibility, high computational capabilities, and line-of-sight (LOS) communication links, UAVs can efficiently collect diverse data across geographically dispersed areas and transmit them in real time to the cloud via powerful communication capabilities, supporting subsequent data processing and model training to better meet the needs of various application scenarios. However, the traditional cloud-centric centralized machine learning paradigm [3] requires the UAVs to directly upload the collected raw data to a central server, which may lead to significant network communication overhead and low energy efficiency for the UAV devices. Furthermore, directly transmitting raw data poses a high risk of data privacy leakage.

To address the challenges faced by cloud computing, edge computing provides a crucial solution to enhance UAV efficiency and security. Federated learning (FL), a distributed machine learning paradigm that preserves privacy, opens promising prospects for collaborative training of global models in UAVs[4]. Under the FL framework, UAVs can process data and perform model training locally or on nearby edge nodes and only need to upload their model parameters rather than raw data to the central server. Applying FL to UAV systems enables a combined advantage of privacy preservation, communication efficiency, and collaborative intelligence, making it one of the key technological pathways toward large-scale aerial computing.

However, in practical application scenarios, due to variations in deployment regions and target recognition requirements, the data collected by UAVs typically exhibit non-independent and identically distributed (non-IID) characteristics. For example, UAVs operating in urban environments may record data with dense high-rise buildings and complex traffic patterns, while those in rural areas may collect data with few obstacles in open spaces, leading to data heterogeneity between clients. In the FL paradigm, significant differences in the data distribution between clients [5] often make it difficult for local models to maintain consistent learning directions, easily causing drift of the local model. If the server only aggregates local model parameters using a weighted average, it may lead the global model to deviate from the optimal solution, not only degrading model performance but also significantly slowing convergence speed or even preventing effective convergence of the global model.

To mitigate the impact of non-IID data on FL model training, researchers have proposed various methods, mainly categorized into model-centric and data-centric approaches. Currently, most FL algorithms are model-centric [6–8], focusing on adjusting local model updates and aggregation strategies to indirectly address inconsistencies in model parameter updates, thus optimizing performance. However, these methods offer limited improvements in performance and communication costs, as the root cause of model differences lies in data heterogeneity. Therefore, addressing distribution differences at the data level is a more fundamental approach to tackling the non-IID problem. To this end, researchers have switched to data-centric solutions, including mainly data enhancement [9, 10], data selection [11], and data sharing [12].

Recently, the introduction of dataset distillation has provided a new solution for data-centric federated learning. Most

Xiufang Shi, Wei Zhang, Yuheng Li, and Mincheng Wu are with the College of Information Engineering, Zhejiang University of Technology, and also with the Zhejiang Key Laboratory of Intelligent Perception and Control for Complex Systems (e-mail: xiufangshi@zjut.edu.cn; 221122030290@zjut.edu.cn; 221124030326@zjut.edu.cn; minchengwu@zjut.edu.cn). Rushi Li is with the School of Design and Architecture, Zhejiang University of Technology, Hangzhou, China (e-mail: lirushi@zjut.edu.cn). Shibo He is with the College of Control Science and Engineering, Zhejiang University, and also with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, China (e-mail: s18he@zju.edu.cn).

Code is available upon request and will be open-sourced upon acceptance.

existing dataset distillation-based federated learning methods generally assume that distilled data possess "visual privacy" characteristics and therefore do not violate privacy-preserving principles. In these strategies, the distilled data generated on clients are uploaded to the server either once or multiple times. However, distilled data are essentially derived from local datasets and thus may still pose privacy leakage risks to some extent. In contrast, trajectory matching-based federated learning completely avoids the transmission of distilled data, and it relies solely on the trajectories of the global model to synthesize distilled data that approximate the global distribution on the server side, thereby achieving a level of privacy protection consistent with traditional FL paradigms. A representative work, called DynaFed, synthesizes the distilled data by matching the early-stage training trajectories for subsequent model fine-tuning [13]. However, since the early-stage trajectory has not yet converged and cannot represent complete global distribution knowledge, applying this type of distilled data to fine-tune the global model could induce considerable volatility as the global model approaches stability.

To address the aforementioned issues, we propose a distilled data-assisted FL method based on stage-wise trajectory matching (FedSTM). FedSTM performs trajectory matching during both the early and late stages of training to synthesize corresponding distilled data that represent the global distribution, thereby utilizing distilled data to fine-tune the global model and effectively enhancing its generalization capability and stability. Additionally, We introduce a regularization term in local training to mitigate the dimensional collapse issue in the representation space of local models under non-IID scenarios, thereby improving the quality of the global model obtained with simple aggregation of local models and obtaining more stable and high-quality global model trajectory information. We conducted extensive comparative experiments with the main baseline methods in various datasets. The experimental results demonstrate that the proposed method outperforms baselines, especially when the label distribution among clients is highly unbalanced. Furthermore, ablation studies further confirm the effectiveness of each component within the overall framework.

The remainder of this paper is organized as follows. Section II introduces related work. Section III provides the preliminaries and problem formulation. Section IV introduces the proposed framework FedSTM. Section V conducts the theoretical performance analysis. Section VI provides an experimental evaluation, and Section VII presents the conclusion.

II. Related Work

A. Federated Learning-Empowered UAV Networks

The FL framework enables multiple UAVs to collaboratively train a global model without sharing local data, providing strong privacy protection. However, due to the dynamic topology and interference-prone links in UAV networks, traditional FL architectures struggle to maintain stable model synchronization and aggregation. To address this, various improved methods have been proposed. Hierarchical et al. proposed a hierarchical FL framework that introduces an edge node layer as an intermediate aggregator between the UAV ends and the central server Tursunboev et al. [14]. This framework optimizes the global model by performing hierarchical aggregation across different network layers. Zeng et al. proposed an FL architecture entirely composed of UAVs Zeng et al. [15]. This framework establishes an FL structure with a leader UAV as the server and Follower UAVs as clients, effectively avoiding the instability issues inherent in communication links. To further enhance security and privacy protection capabilities, Wang et al. developed a blockchain-based UAV network architecture to replace traditional central servers, incorporating local differential privacy mechanisms to improve privacy safeguards Wang et al. [16]. Furthermore, Hou et al. developed an FL framework that supports covert communication, where UAVs also act as friendly jammers that emit artificial noise to conceal model uploads Hou et al. [17]. Although these studies have proposed effective improvements in architecture optimization, communication efficiency and security, they largely overlook data heterogeneity, which critically affects model generalization and convergence, posing a key challenge in current FL-enabled UAV networks.

B. Federated Learning on Non-IID Data

To alleviate the impact of non-IID data on FL model training, researchers have proposed various strategies to address these challenges. In terms of updating the local model, FedProx [6], introduces a proximal term during local training to prevent local models from deviating too far from global initialization. Karimireddy et al. proposed a model called Scaffold, which considers the non-IID issue as variance introduced among clients and introduces control variables to adjust the update direction and speed of each local model, effectively reducing the drift issue during local training and improving global model convergence [7]. In terms of improving aggregation strategies, to ensure fairness in global updates, FedNova [8] addresses the issue of inconsistent local update steps caused by differences in clients' computing capabilities or dataset sizes through normalizing and scaling each client's local updates before global model aggregation. FedAT [18] mitigates the issue caused by client lag by combining synchronous training within layers and asynchronous training between layers, bridging the gap between synchronous and asynchronous training. However, when data heterogeneity becomes much more severe, methods based on regularization and weight adjustment still struggle to achieve ideal accuracy. Moreover, addressing data heterogeneity in FL from the data level is a promising direction.

C. Federated Learning Based on Dataset Distillation

In recent years, the combination of FL and dataset distillation technology has provided new approaches to address the challenges posed by data heterogeneity. Dataset distillation technology mitigates the issue of non-IID data directly at the data level by synthesizing compact and information-rich distilled data. Due to the high quality and low overhead of local distilled data, some methods, such as DOSFL [19], FedD3 [20] and FedSynth [21], directly upload locally synthesized distilled

data to replace updates of model parameters. However, these methods inadvertently deviate from the original intention of FL and still pose potential privacy leakage risks. To further enhance privacy protection, Xiong et al. combined distribution matching with the Gaussian mechanism to synthesize distilled data, approximating the original global training loss in FL, thus improving communication efficiency and test accuracy while protecting privacy [22]. Beyond distilling local data, Jia et al. utilized a pre-trained deep generative model on the server side to distill data in the latent space, improving the quality of the distilled data and the performance of the global model [23]. Furthermore, Pi et al. proposed the DynaFed framework, which synthesizes the distilled data by matching the early-stage training trajectory of the global model and uses these data to fine-tune the subsequently aggregated global model [13]. However, early stage trajectories have not yet converged and struggle to represent the complete global distribution knowledge. This inspires our idea of matching both early-stage and late-stage training trajectories to obtain distilled data that fully represent the global distribution.

III. PRELIMINARIES AND PROBLEM FORMULATION

A. Trajectory Matching based Dataset Distillation

The core idea of dataset distillation is to transform a large-scale dataset into a smaller dataset while preserving the core features of the original dataset as much as possible [24]. Suppose that the original dataset and the distilled dataset are denoted by O and S, respectively. $|O|$ and $|S|$ represent the sizes of the original dataset and the distilled dataset, respectively. Typically, the size of the original dataset O is much larger than that of the distilled dataset S, i.e., $|S| \ll |O|$. The synthesis of distilled data can be formulated as the following optimization problem:

$$\min_{S} \mathcal{L}_{dd}\left(S, O\right), \tag{1}$$

where $\mathcal{L}_{dd}\left(S, O\right)$ represents the target function in the dataset distillation algorithm, which will be elaborated later. The optimized distilled dataset S should satisfy the requirement that the performance of the model trained on the distilled dataset S should be similar to that trained on the original dataset O.

The dataset distillation algorithm based on trajectory matching optimizes the distilled dataset by matching the training trajectories of the same model in the original dataset and the distilled dataset. Specifically, this method treats the training trajectory of the original dataset as the expert model and the training trajectory of the distilled dataset as the student model. The optimization objective is to approximate the student model with the expert model after a certain number of iterations. The optimization objective function can be expressed as

$$\begin{aligned} \mathcal{L}_{dd}(S, \Theta) &= \mathbb{E}_{\omega^{(0)} \sim \Theta}\left[\frac{\|\omega_S^{(T_s)} - \omega_O^{(T_t)}\|^2}{\|\omega_O^{(T_t)} - \omega^{(0)}\|^2}\right] \\ \text{s.t. } \omega_S^{(t)} &= \omega_S^{(t-1)} - \eta \nabla \ell(\omega_S^{(t-1)}, S)\ ; \\ \omega_O^{(t)} &= \omega_O^{(t-1)} - \eta \nabla \ell(\omega_O^{(t-1)}, O), \end{aligned} \tag{2}$$

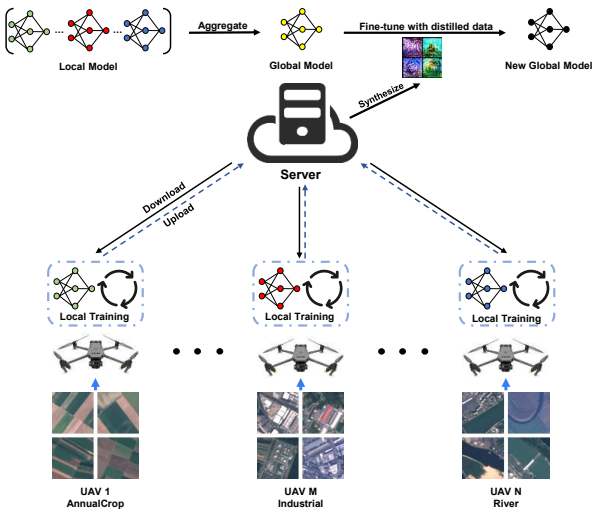


Fig. 1. UAV network architecture under FL framework

where Θ denotes the trajectory of the teacher model, $\omega^{(0)}$ is the start point of the model trajectory, $\omega_S^{(t)}$ and $\omega_O^{(t)}$ respectively denote the model parameters trained on the distilled dataset S and the original dataset O in the t-th step of model update, $\omega_S^{(T_s)}$ and $\omega_O^{(T_t)}$ respectively represent the endpoints of the trajectories formed by the training model on S and O, ℓ is the loss function of the training network, η is the learning rate.

B. Problem Formulation

We consider an FL framework, as shown in Fig. 1, which consists of a server and a number of UAVs. The UAVs are denoted by $\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$, where N represents the total number of UAVs. Each UAV u_i can be regarded as an edge client ("UAV" and "client" are used interchangeably without ambiguity hereafter) equipped with a local dataset $\mathcal{D}_i = \{(x_j, y_j)\}_{j=1}^{|\mathcal{D}_i|}$, which is collected from its target environment, x_j and y_j represent the collected data and their corresponding labels, respectively. To effectively recognize diverse targets from different geographical regions and task contexts, and meet the intelligent perception demands of cross-region and multitask scenarios, our core objective is to collaboratively train a global model with strong generalization capabilities through distributed cooperation between the server and UAVs. In this work, we assume reliable server-UAV communication and do not explicitly model error correction or retransmission mechanisms. Specifically, the training objective can be formulated as the following optimization problem:

$$\min_{\omega} F\left(\omega\right) = \min_{\omega} \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{|\mathcal{D}|} f_i\left(\omega\right) \tag{3}$$

where $F(\omega)$ is the global loss function, $f_i(\omega)$ is the local loss function of the i-th client, $|\mathcal{D}_i|$ denotes the number of local samples of the i-th client and $|\mathcal{D}| = \sum_{i=1}^{N} |\mathcal{D}_i|$ denotes the total number of samples. By optimizing this objective, we aim to obtain a robust global model capable of effectively handling the diverse data collected by different UAVs, thus supporting

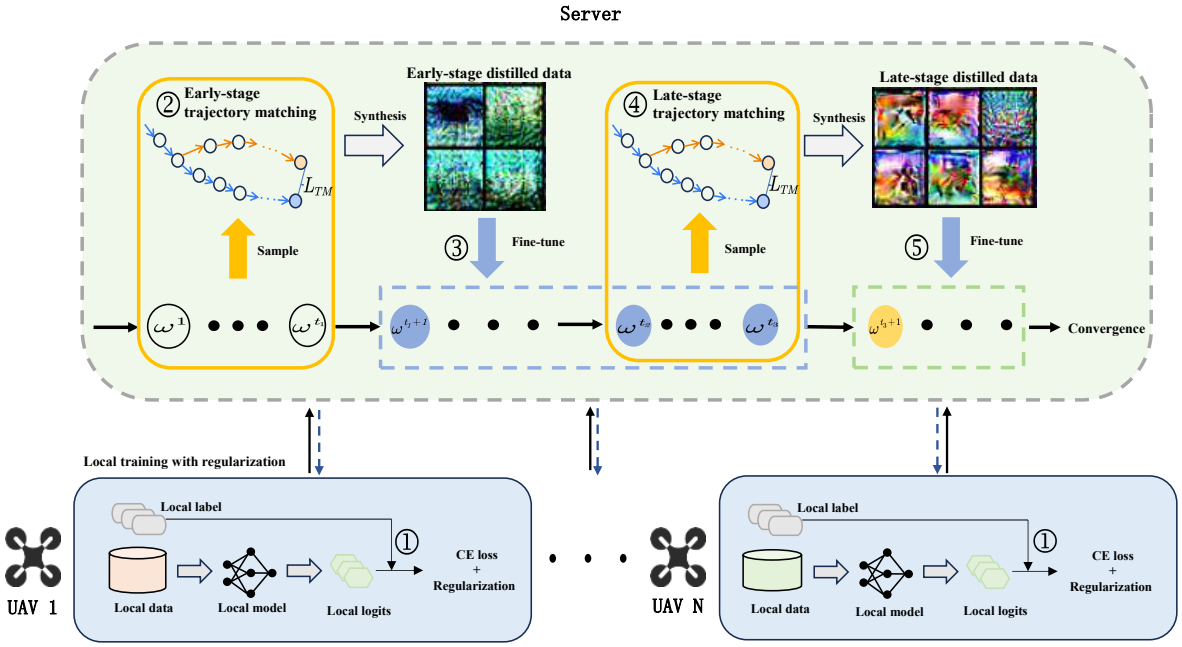


Fig. 2. The overall pipeline of FedSTM.

complex tasks such as environmental monitoring and disaster response.

As each UAV operates in distinct geographical regions and may perform different tasks, the local datasets collected by different UAVs naturally exhibit non-IID characteristics. For example, in environmental monitoring tasks, UAVs patrolling farmland regions collect images related to crops primarily, while those operating over rivers tend to gather data containing water-related features. This introduces substantial biases in the label or feature distribution. In this paper, we focus on a typical non-IID problem characterized by label imbalance [5], where the proportions of samples with each label in each client follow a Dirichlet distribution, i.e., $p_{ci} \sim Dir(\alpha)$, where p_{ci} denotes the proportion of samples with label c at client i, $Dir(\cdot)$ denotes the Dirichlet distribution, and α is a parameter characterizing the degree of imbalance in the label distribution. A lower value of α indicates a higher level of data label imbalance between clients. This Dirichlet-based modeling provides a standardized and controllable proxy for geographically induced data heterogeneity and is widely adopted in federated learning benchmarks, enabling systematic evaluation and fair comparison.

The imbalance in label distribution among UAVs significantly affects the convergence speed and generalization of the global model. To address the challenges posed by data heterogeneity, we propose an improved FL framework based on stage-wise trajectory matching (FedSTM). FedSTM introduces a stage-wise data distillation mechanism that leverages the global distribution information of UAV data to guide global model updates, thereby mitigating the adverse effects of non-IID data.

IV. The Proposed FedSTM

A. Overview

The core idea of our proposed FedSTM is to fully exploit the global data distribution information embedded in the global model trajectories. Specifically, in FedSTM, the server uses distilled data in the early training stage to capture common features that occur with high frequency in the data and are easy to learn. In addition, the server synthesizes the distilled data in the late training stage to capture fine-grained features that occur with low frequency in the data and require higher model capacity to fit, thus improving the stability and generalization of the model. Fig. 2 specifically demonstrates the five critical steps of the proposed FedSTM framework: local training with regularization, early-stage trajectory matching, early-stage model fine-tuning, late-stage trajectory matching, late-stage model fine-tuning. We will briefly introduce each step below.

1) **Local training with regularization:** Throughout the entire training process, clients perform local training and upload their local models to the server at every communication round. In particular, to mitigate dimensional collapse during local training under non-IID data, the local loss incorporates a decorrelation regularization term to optimize the local model performance, thereby improving the quality of the global model trajectory.
2) **Early-stage trajectory matching:** From the 1st to the t_1-th communication rounds, the server aggregates the received local models and forms the early stage global model trajectory $\mathcal{T}_{\text{early}} = \{\omega^1, \omega^2, ..., \omega^{t_1}\}$. Based on $\mathcal{T}_{\text{early}}$, the server performs early-stage trajectory matching to synthesize distilled data $D_{\text{syn}}^{\text{early}}$, which characterizes common features across all data of all clients, and will be utilized for subsequent global model fine-tuning.

3) **Early-stage model fine-tuning:** From the (t_1+1)-th to the t_3-th communication rounds, the server conducts early-stage global model fine-tuning. Specifically, at the t-th communication round, where $t_1+1 \leq t \leq t_3$, the server takes the aggregated global model ω^t as an initial value, and conducts model training using Stochastic Gradient Descent (SGD) method with the early-stage distilled data $D_{\text{syn}}^{\text{early}}$.
4) **Late-stage trajectory matching:** When reaching the t_2-th communication round, the accuracy of global model has reached a high level, and the growth trend is gradually stabilizing. To obtain comprehensive and sufficient trajectory information, late-stage trajectory matching begins at the t_3-th round. By matching the global trajectories from the t_2-th to t_3-th round, the server will synthesize new distilled data $D_{\text{syn}}^{\text{late}}$ to support subsequent training.
5) **Late-stage model fine-tuning:** Starting from the (t_3+1)-th round, the server will replace the early-stage distilled data $D_{\text{syn}}^{\text{early}}$ with the late-stage distilled data $D_{\text{syn}}^{\text{late}}$. The global model will be continuously fine-tuned using late-stage distilled data, which characterizes fine-grained features across all clients' data, until the FL process is fully converged.

B. Local Training with Regularization

In FL, when the model is trained on IID data, the learned feature representations are distributed across a well-dispersed high-dimensional space, where each dimension carries unique information. This allows samples from different classes to be well separated in the embedding space. However, under the non-IID setting, each client only holds a subset of classes or data with specific biases. As a result, the local model tends to focus its representational capacity on a few dimensions that are most effective for its local data, while neglecting others. This phenomenon leads to dimensional collapse, which not only degrades local model performance but also biases the global model toward dominant local features, thereby limiting its generalization ability. A decorrelation regularization term is introduced into local training to address this issue by encouraging the model to utilize the full representation space.

The goal of the regularization term is to promote feature diversity by minimizing correlations between different dimensions of the feature representations. Ideally, we aim to equalize the singular values of the feature covariance matrix, which would indicate balanced use of all dimensions. However, directly optimizing the singular values is computationally expensive. Instead, by referring to [25], we adopt a tractable surrogate: minimizing the Frobenius norm of the correlation matrix of the feature representations. Let K_{ω,ξ_i} denote the correlation matrix computed from a mini-batch ξ_i, the regularization term is defined as:

$$R(\omega,\xi_i)=\frac{1}{d^2}\|K_{\omega,\xi_i}\|_F^2, \tag{4}$$

where d denotes the feature dimension. Minimizing this term encourages the correlation matrix to approximate the identity matrix, thereby reducing inter-dimensional correlations while preserving variance in each dimension. The total loss function in local training is

$$\mathcal{L}(\omega,\xi_i)=\mathcal{L}'(\omega,\xi_i)+\lambda R(\omega,\xi_i) \tag{5}$$

where $\mathcal{L}'(\omega,\xi_i)$ is a typical loss term, e.g. cross-entropy loss, in supervised learning, and λ is the hyperparameter of regularization strength. The entire local training process is defined as a function LocalTraining $(\omega^t, D_i, E_1, \lambda)$.

The quality of the global model's trajectory is highly dependent on the diversity and richness of the local representations. Without regularization, local models suffering from dimensional collapse produce biased and low-quality updates. When aggregated, these updates lead to a global model that learns a biased representation, which in turn biases the distilled data synthesized via trajectory matching. By incorporating the decorrelation term, local models can learn more balanced and diverse representations. This improves the quality of both local updates and the global trajectory, enabling the distilled data to better approximate the true global distribution.

C. Stage-wise Trajectory Matching

1) Early-stage Trajectory Matching: In the traditional FL training process, the server updates the global model parameters by averaging local model parameters uploaded by clients. When the data across clients are non-IID, the global model may stay in the lower accuracy region and oscillate around the region repeatedly, which makes it difficult to achieve stable convergence. However, the iterative update of the global model is still driven by the data of all clients, and therefore its dynamic changes implicitly contain knowledge about the global data distribution [13]. Based on this fact, the goal of early stage trajectory matching is to extract feature distribution knowledge from the early stage trajectory and transfer it to a synthetic distilled dataset, enabling further fine-tuning of the global model.

Fig. 3 shows the synthesis process of the distilled data based on the trajectory matching. Suppose that the original early stage trajectory of the global models obtained by model aggregation is $\mathcal{T}_{\text{early}}=\left\{\omega^1,\omega^2,...,\omega^{t_1}\right\}$. During each update of the distilled dataset, the server randomly selects a starting point ω^t and its corresponding endpoint ω^{t+L_o} from $\mathcal{T}_{\text{early}}$ for trajectory matching. Let $\hat{\omega}^{t+L_s}$ denote the updated model using L_s epochs of SGD in the distilled dataset D_{syn}^{early} with an initial model ω^t. As defined in Eq.(2), the distilled data is optimized by minimizing the following trajectory matching loss function:

$$\mathcal{L}_{dd}(D_{syn}^{early},\mathcal{T}_{\text{early}})=\mathbb{E}_{\omega^t\sim\mathcal{T}_{\text{early}}}\left[\frac{\|\hat{\omega}^{t+L_s}-\omega^{t+L_o}\|_2^2}{\|\omega^t-\omega^{t+L_o}\|_2^2}\right] \tag{6}$$

Intuitively, this optimization process can be viewed as aligning the quality of the distilled model through the L_s steps of SGD in D_{syn}^{early} with that of the global model through the L_o rounds of update of the aggregation. We refer to this process of distilled data synthesis via trajectory matching as DataSyn.

Note that the size of early-stage distilled data is usually much smaller than the size of datasets on the client side. To ensure reasonableness of trajectory matching, the related

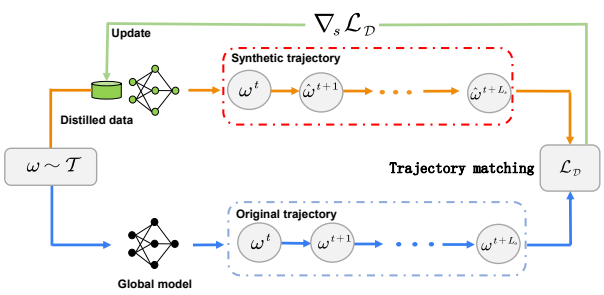


Fig. 3. Overview of trajectory matching-based dataset distillation, where $\omega \sim \mathcal{T}$ denotes ω is sampled from the global model trajectory $\mathcal{T}$.

variables must satisfy the following constraints: the round t of the starting point ω^t should satisfy $t \leq t_1 - L_o$ such that the endpoint of the trajectory matching still lies within $\mathcal{T}_{\text{early}}$; Through the above design, knowledge of the global distribution in early stage trajectories $\mathcal{T}_{\text{early}}$ can be effectively transferred to the distilled dataset D_{syn}^{early}, making it an approximate representation of the global data distribution, which provides support to fine-tuning the global model.

In the case of high data heterogeneity, when the global model is first fine-tuned using distilled data from the early stages D_{syn}^{early}, the model accuracy shows a significant improvement. However, as training progresses, although fine-tuning based on D_{syn}^{early} can gradually improve the convergence of the global model, it also tends to cause considerable fluctuations in accuracy. Such fluctuations are mainly attributed to the fact that early-stage trajectory matching relies on trajectories of the global model at a low-accuracy stage to synthesize distilled data, which is insufficient to fully cover the global data distribution, thereby introducing instability in subsequent fine-tuning. Specifically, distribution bias is closely related to the dynamical characteristics during the early stage of model training [26]. At this stage, the direction of parameter updates is primarily driven by common features that frequently appear in the data distribution. These features typically have lower complexity and clear pattern boundaries.As a result, the model tends to learn these common features in the early stages. Consequently, the distilled data $D_{\text{syn}}^{\text{early}}$ synthesized by early-stage trajectory matching primarily captures these dominant features of the global data distribution. However, its coverage of low-frequency, complex, and long-tail features remains insufficient, which limits its ability to fully characterize the global distribution and ultimately causes accuracy fluctuations during subsequent fine-tuning. Therefore, relying solely on the distilled data in the early stage D_{syn}^{early} is insufficient to meet the learning needs of the model for fine-grained features in the later stage of training.

2) Late-stage Trajectory Matching: The late stage trajectories of the global model can include both common and fine-grained features of the data. Thus, it is necessary to perform trajectory matching again to synthesize new distilled data at the late stage. Specifically, we choose a range of communication rounds, say $[t_2, t_3]$, where the global model accuracy has reached a high level and the growing trend tends to level off, indicating that the performance improvement from D_{syn}^{early} is close to a bottleneck. The late stage trajectory is defined as the sequence of global model parameters $\mathcal{T}_{late} = \{\omega^{t_2}, \omega^{t_2+1}, ..., \omega^{t_3}\}$ from the t_2-th to the t_3-th round. The distilled dataset synthesized by late-stage trajectory matching is denoted as D_{syn}^{late}. The distillation process of the trajectory-matching-based dataset is the same as that in the early stage trajectory matching. It is important to note that the size of D_{syn}^{late} should be larger than that of D_{syn}^{early}. That is because fine-grained features of the data typically involve rare or long-tailed features, which require sufficient samples to accurately capture and represent them. With a small size of samples, a distilled dataset tends to focus more on the basic features, and fine-grained features might be overlooked because of insufficient samples. By increasing the size of the distilled dataset, it can simultaneously reflect both common and fine-grained features, thereby enhancing the model's ability to learn boundary samples in the late stage and further improving overall performance.

D. Global Model Fine-tuning

The global model fine-tuning occurs in both the early stage and the late stage. In the early training stage, the model is fine-tuned in the distilled data set in the early stages D_{syn}^{early} to strengthen its learning of common features across client data. In the late training stage, model fine-tuning is performed on the distilled data set in the late stage D_{syn}^{late} to continuously improve the learning performance of the global model.

Specifically, after the t-th round of global aggregation, the server obtains the current global model parameters ω^t. If $t > t_1$, the global model is fine-tuned in the distilled dataset for E_2 epochs using SGD, with the following update rule:

$$\omega^{t,k+1} \leftarrow \omega^{t,k} - \eta_{t,k} \nabla \left[\mathcal{L}\left(\omega, \xi^k\right)\right]_{\omega=\omega^{t,k}}, \tag{7}$$

where $\eta_{t,k}$ is the learning rate, $\xi^k \in D_{syn}^t$ denotes the k-th mini-batch in the distilled dataset D_{syn}^t, which is defined by

$$D_{syn}^t = \begin{cases} D_{syn}^{early}, & t_1 < t \leq t_3 \\ D_{syn}^{late}, & t > t_3 \end{cases} \tag{8}$$

and $\mathcal{L}\left(\omega, \xi^k\right)$ represents the loss function based on the current mini-batch. For simplicity, the entire fine-tuning process is defined as a function FineTune $\left(\omega^t, D_{syn}^t, E_2\right)$.

The overall implementation procedure of the proposed FedSTM is shown in Algorithm 1. In FedSTM, each client performs local training with regularization in each local round to mitigate the dimensional collapse of model representations, while the server fine-tunes the global model using stage-wise distilled datasets. The synergy between these two components creates a positive cycle between high-quality trajectories and high-quality distilled data, continuously enhancing the overall performance of the model.

V. Performance Analysis

A. Convergence analysis

To facilitate the subsequent theoretical analysis, we first introduce several standard assumptions used in the analysis of federated optimization algorithms.

Algorithm 1 FedSTM

Input: D_i, T, E_1, E_2, t_1, t_2, t_3, L_s, L_o, λ, η, I
Output: Global model ω^t

Randomly initialize global model parameters ω^0; set trajectory sequence $\mathcal{T} \leftarrow [\,]$
for $t = 0, 1, 2, \cdots, T$ **do**
  Server has the current global model ω^t
  if $t < t_1$ **then**
    $\tilde{\omega}^t = \omega^t$
  else if $t = t_1$ **then**
    $D_{\text{syn}}^{\text{early}} \leftarrow \text{DataSyn}(\mathcal{T}_{\text{early}} = \{\omega^t\}_{t=1}^{t_1}, L_o, L_s, \eta, I)$
    $\tilde{\omega}^t = \omega^t$
  else if $t_1 < t < t_3$ **then**
    $\tilde{\omega}^t = \omega^{t,E_2} \leftarrow \text{Finetune}(D_{\text{syn}}^{\text{early}}, \omega^t, E_2)$
  else if $t = t_3$ **then**
    $D_{\text{syn}}^{\text{late}} \leftarrow DataSyn(\mathcal{T}_{\text{late}} = \{\omega^t\}_{t=t_2}^{t_3},\ L_o, L_s, \eta, I)$
    $\tilde{\omega}^t = \omega^{t,E_2} \leftarrow \text{Finetune}(D_{\text{syn}}^{\text{early}}, \omega^t, E_2)$
  else if $t > t_3$ **then**
    $\tilde{\omega}^t = \omega^{t,E_2} \leftarrow \text{Finetune}(D_{\text{syn}}^{\text{late}}, \omega^t, E_2)$
  end if
  Server randomly selects a subset of clients $\mathcal{K}^t$, sends the global model $\tilde{\omega}^t$ to them
  for $i \in \mathcal{K}^t$ **do**
    $\omega_i^{t+1} \leftarrow \text{LocalTraining}(\tilde{\omega}^t, D_i, E_1, \lambda)$
    Upload the updated model ω_i^{t+1} to the server.
  end for
  Server aggregate local updates as $\omega^{t+1} \leftarrow \sum_{i \in \mathcal{K}^t} \frac{|\mathcal{D}_i|}{|\mathcal{D}|}\, \omega_i^{t+1}$ and append it to $\mathcal{T}$.
end for
return ω^T

Assumption 1 ($\tilde{L}$-smoothness): The loss function $\mathcal{L}(\cdot, \mathcal{D})$ is $\tilde{L}$-smooth, i.e.,

$$||\nabla\mathcal{L}(\omega_1) - \nabla\mathcal{L}(\omega_2)|| \leq \tilde{L}||\omega_1 - \omega_2||, \forall\omega_1, \omega_2. \tag{9}$$

Assumption 2 (μ-strongly convex): The loss function $\mathcal{L}(\cdot, \mathcal{D})$ is μ-strongly convex, i.e.,

$$\mathcal{L}(\omega_1) \geq \mathcal{L}(\omega_2) + \langle\nabla\mathcal{L}(\omega_2),\, \omega_1 - \omega_2\rangle + \frac{\mu}{2}||\omega_1 - w_2||^2, \forall\, \omega_1,\ \omega_2. \tag{10}$$

Assumption 3 (Bounded gradient variance). Let $\xi_i \sim \mathcal{D}_i$ be a mini-batch sample drawn from the local dataset of client i. The stochastic gradient satisfies

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i}\left[\, ||\nabla\mathcal{L}(\omega, \xi_i) - \nabla\mathcal{L}(\omega, _i)||^2 \,\right] \leq \sigma_i^2. \tag{11}$$

Assumption 4 (Bounded stochastic gradients). The expected squared norm of stochastic gradients is bounded. For all clients $i = 1, 2, \ldots, N$,

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i}\left[\, ||\nabla\mathcal{L}_i(\omega, \xi_i)||^2 \,\right] \leq G_1^2. \tag{12}$$

Furthermore, there exists $G_2 > 0$ such that for both early and late-stage distilled data,

$$\mathbb{E}_{\xi \sim D_{\text{syn}}}\left[||\nabla\mathcal{L}_i(\omega, \xi)||^2\right] \leq G_2^2, \tag{13}$$

where D_{syn} is the distilled dataset, in the early-stage $D_{\text{syn}} = D_{\text{syn}}^{\text{late}}$, and in the late-stage, $D_{\text{syn}} = D_{\text{syn}}^{\text{late}}$.

Similar to [13], we additionally adopt the following assumption about synthetic gradient error.

Assumption 5 (Bounded synthetic gradient error). For the distilled dataset $\mathcal{D}_{\text{syn}}$,

$$||\nabla\mathcal{L}(\omega^t, D_{\text{syn}}) - \nabla\mathcal{L}(\omega^t, D)|| \leq \delta||\nabla\mathcal{L}(\omega^t, \mathcal{D})|| + \varepsilon, \tag{14}$$

where $\delta \geq 0$ and $\varepsilon \geq 0$ are two small constants.

The core of our theoretical argument lies in the quality difference between the distilled data synthesized at different stages. Since the late-stage trajectory $\mathcal{T}_{\text{late}}$ is obtained from a more converged and stable global model, the corresponding distilled data $D_{\text{syn}}^{\text{late}}$ captures the global distribution more comprehensively, including both common and fine-grained features. In contrast, the early-stage distilled data $D_{\text{syn}}^{\text{early}}$, synthesized from a non-convergent trajectory, primarily captures high-frequency, common features. Therefore, we reasonably assume that the late-stage distilled data has a strictly smaller synthetic gradient error, i.e,

$$\delta_{\text{late}} \leq \delta_{\text{early}}, \qquad \varepsilon_{\text{late}} \leq \varepsilon_{\text{early}} \tag{15}$$

with at least one inequality being strict.

In the proposed FedSTM, when $t_1 < t \leq t_3$, the global model is fine-tuned using $D_{\text{syn}}^{\text{early}}$, while when $t > t_3$, the global model is fine-tuned using $D_{\text{syn}}^{\text{late}}$. In contrast, the existing single-stage baseline, e.g., DynaFed, keeps to fine-tune the global model using only the early-stage distilled data $D_{\text{syn}}^{\text{early}}$ even when $t > t_3$.

To analyze the benefit brought by stage-wise trajectory matching, except for the difference on fine-tuning, FedSTM and the single-stage method adopt the same setting. Let $\Delta_t = \mathbb{E}||\omega^t - \omega^*||^2$ denote the expected error to the optimal model ω^* at communication round t, $\mathcal{V}_T^{(\text{single})}$ and $\mathcal{V}_T^{(\text{stage})}$ denote upper bounds of the expected error after T rounds respectively for the single-stage approach and FedSTM. Through recursive convergence analysis under the stated assumptions, we can obtain the following results:

Theorem 1: Under Assumptions 1-5, and given that the late-stage distilled data exhibits a strictly smaller synthetic gradient error, i.e., $\delta_{\text{late}} \leq \delta_{\text{early}}$ and $\varepsilon_{\text{late}} \leq \varepsilon_{\text{early}}$ with at least one inequality being strict, the proposed FedSTM framework with stage-wise fine-tuning achieves a tighter upper bound on the expected optimization error compared to the single-stage approach after T communication rounds, namely,

$$\mathcal{V}_T^{(\text{stage})} < \mathcal{V}_T^{(\text{single})}. \tag{16}$$

The detailed step-by-step derivation about this theorem is provided in the supplymentary material. This theorem provides a theoretical guarantee that our stage-wise design effectively leverages the improving quality of the global model's trajectory to synthesize better distilled data, thereby leading to faster convergence and superior final performance.

TABLE I
PARAMETER SETTINGS OF FEDSTM

Parameter Type	Parameter	Value	Function Description
Server	N	80	Total numbers of clients
	T	400	Total communication rounds between clients and server
	C	40%	Client participation ratio per round
Client	E_1	1	Local epochs
	η_1	0.001	Learning rate of local update
	B_1	32	Batch size of local training
	λ	0.05	The regularization strength hyperparameter
Data Distillation	$\lvert D_{\text{syn}}^{\text{early}}\rvert$	150(CIFAR10, CINIC10) 450(NWPU-RESISC45)	Size of early-stage distilled data (IPC × number of classes)
	$\lvert D_{\text{syn}}^{\text{late}}\rvert$	500(CIFAR10, CINIC10) 675(NWPU-RESISC45)	Size of late-stage distilled data (IPC × number of classes)
	I	3000	Optimization rounds for distilled data
	η_2	0.05	Learning rate of distilled data optimization
	B_2	32	Batch size of distilled data
	L_o	20	Step length of original trajectories
	L_s	3	Step length of synthetic trajectories
	t_1	25	Ending round for early-stage trajectory
	t_2	100	Starting round for late-stage trajectory
	t_3	150	Ending round for late-stage trajectory
Model fine-tuning	E_2	20	Rounds of model fine-tuning
	B_3	256	Batch size of model fine-tuning
	η_3	0.01	Learning rate of model fine-tuning on distilled data

B. Computational Complexity

Compared with the classical FL framework, the proposed FedSTM introduces additional computational complexity on the server side, due to the incorporation of stage-wise trajectory matching and global model fine-tuning. In the process of stage-wise trajectory matching, the server performs I iterations to synthesize the distilled data. In each iteration, the server first randomly samples a starting point from the global model trajectory, and conducts L_s steps of SGD on the current distilled data to train the student model, with a computational complexity of $\mathcal{O}(L_s \cdot |D_{\text{syn}}| \cdot |w|)$, where $|D_{\text{syn}}|$ denotes the size of the distilled dataset, and $|w|$ is the size of model parameters. Subsequently, the server computes the distance loss between the student model and the teacher model. Based on this loss, the distilled dataset is then optimized via backpropagation, with a computational complexity of $\mathcal{O}(|D_{\text{syn}}| \cdot |w|)$. Therefore, the overall computational complexity of trajectory matching is $\mathcal{O}(I \cdot L_s \cdot |D_{\text{syn}}| \cdot |w|)$. The corresponding computational complexities for generating the dataset at the early and late stages are respectively $\mathcal{O}(I \cdot L_s \cdot |D_{\text{syn}}^{early}| \cdot |w|)$ and $\mathcal{O}(I \cdot L_s \cdot |D_{\text{syn}}^{late}| \cdot |w|)$. For the process of fine-tuning, when $t > t_1$, the server conducts E_2 steps of SGD to update the global model based on the distilled dataset. The overall computational complexity for fine-tuning is $\mathcal{O}((T - t_1) \cdot E_2 \cdot B_3 \cdot |w|)$, where T is the total number of communication rounds and B_3 is the batch size used for fine-tuning.

VI. EXPERIMENTAL EVALUATION

In this section, we first introduce the experimental setup. Subsequently, we compare the performance of the FedSTM algorithm with the baseline methods under varying data heterogeneity and different datasets. Finally, through ablation studies, we validate the effectiveness of each component in the algorithm.

A. Experimental Setup

1) Dataset and Model: This experiment selects three classic image classification datasets: CIFAR10 [27], CINIC10 [28] and NWPU-RESISC45[29] to evaluate the performance of the proposed FedSTM. Specifically, the CIFAR10 dataset consists of 10 categories and in total 60,000 color images with 32×32 pixels. Compared to CIFAR10, CINIC10 incorporates 10 categories of high-resolution images from ImageNet, which can offer more diverse training and testing samples. The NWPU-RESISC45 dataset contains 31,500 high-resolution remote sensing images covering 45 scene categories. This dataset captures remote sensing images from various aerial perspectives and provides diverse spatial patterns, textures, and observation angles, making it well-suited for evaluating UAV-related classification tasks.

To simulate data heterogeneity across clients under FL framework, we use the Dirichlet distribution to partition the dataset for each UAV. For each benchmark dataset, we consider four levels of data heterogeneity setting α = 0.01, 0.02, 0.04 and 0.1. It is important to note that the hyperparameter α controls the degree of heterogeneity: the smaller the α, the more severe the label-skewed non-IID distribution.

To ensure fair comparison across experiments, we use the same ConvNet architecture in our experiments, which consists of three repeated convolutional blocks, each comprising the following layers: a convolutional layer with 128 filters, an instance normalization layer, a ReLU activation function layer, and an average pooling layer.

2) Baselines: We select the following typical algorithms as comparative baselines, including classic algorithms and state-of-the-art methods combining trajectory matching with FL:

- FedAvg [30], a classic FL algorithm, which updates the global model by directly weighting averaging the parameters of local models.

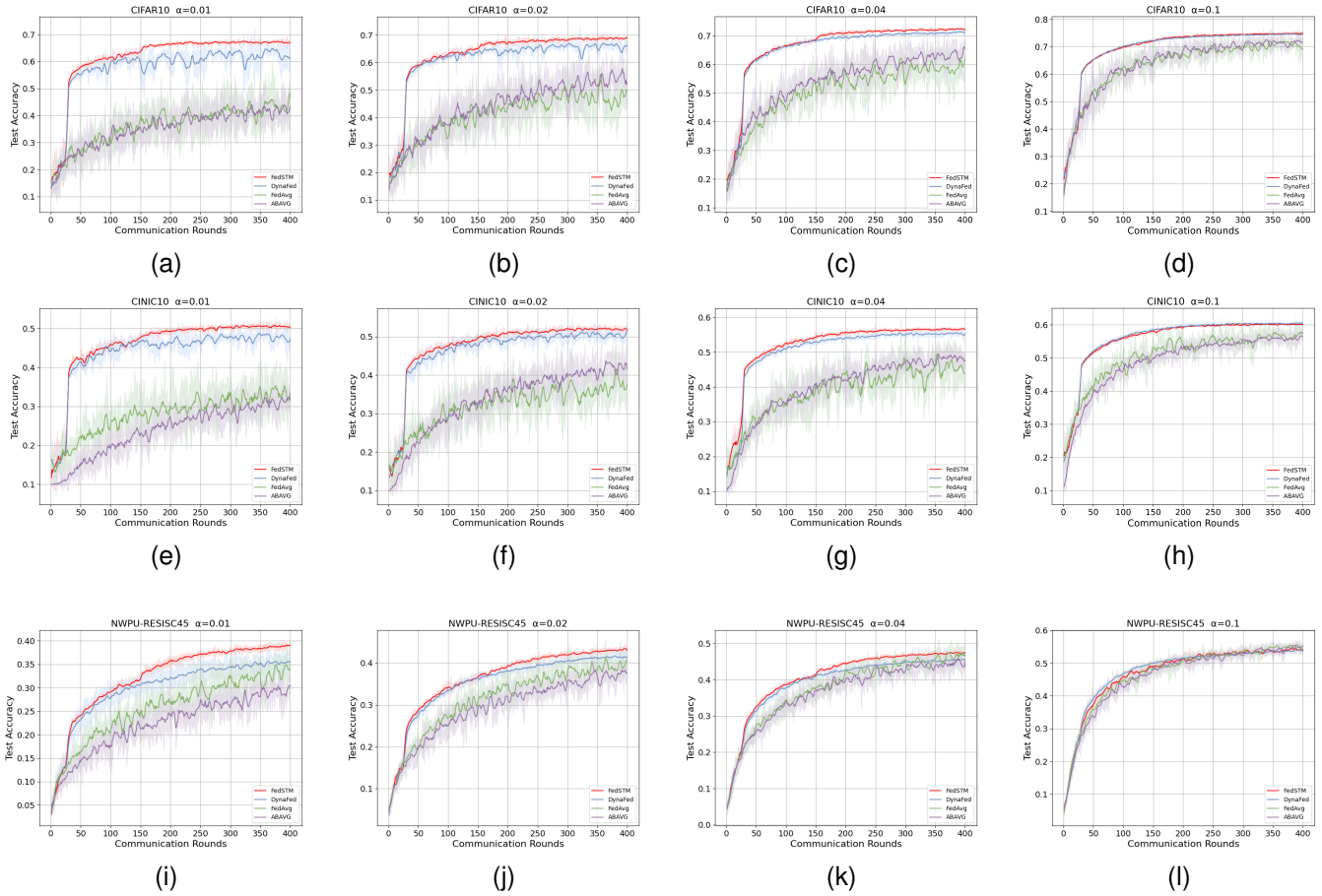


Fig. 4. Global model's test accuracies of different methods throughout the communication rounds on three datasets.

- ABAVG [31], an FL algorithm with an improved aggregation mechanism, which adopts a weight allocation mechanism based on the validation accuracy of local models.
- DynaFed [13], a state-of-the-art FL algorithm based on trajectory matching, which uses the early-stage trajectories of the global model on the server to synthesize the distilled data for subsequent fine-tuning.

These three baseline methods provide comparative references from different dimensions for FL research, facilitating a comprehensive evaluation of FedSTM's advantages in complex scenarios without IID data.

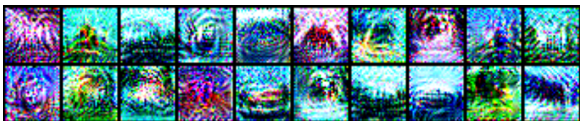
3) Implementation details: Our experiments on the CIFAR10 and CINIC10 datasets are conducted on an NVIDIA RTX 4090 GPU, while the experiments on the NWPU-RESISC45 dataset are conducted on an NVIDIA A800 GPU. All experiments are implemented using PyTorch. In the implementation of FedSTM, detailed parameter settings are listed in Table I. In the setting of baseline methods, to ensure fairness, DynaFed, except for not requiring late-stage trajectory matching and local training regularization, adopts identical parameter settings to FedSTM. FedAvg and ABAVG use the same number of clients, global communication rounds, proportion of participating clients per round, optimizer, and learning rate as FedSTM.

B. Comparative experiment

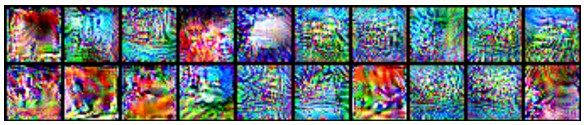
This subsection validates the performance of the proposed FedSTM across different datasets and varying levels of data heterogeneity through comparative experiments. We use a Dirichlet distribution to partition datasets. For each dataset, we consider four levels of data heterogeneity : α = 0.01, 0.02, 0.04, and 0.1. For each case in the experiment, we repeat the procedure three times and record the mean and standard deviation of the results. Fig. 4 shows the learning curves of different methods in three datasets with varying heterogeneity over 400 communication rounds.

From Fig. 4, we can see that when α = 0.01, 0.02, 0.04, and 0.1, the proposed FedSTM consistently outperforms both FedAvg and ABAVG in all three datasets in terms of convergence speed, final test accuracy, and stability. Specifically, FedSTM achieves higher accuracy in fewer communication rounds and exhibits lower variance across training. When α = 0.01, the label distribution is highly imbalanced, as shown in Fig. 4a, Fig. 4e and Fig. 4i, the performance gap between FedSTM and FedAvg or ABAVG is very large. As α increases, the level of heterogeneity of the data decreases, and the performance gap between FedSTM and FedAvg or ABAVG gradually narrows.

The performance of FedSTM is comparable to that of DynaFed, and it exhibits superior results under high data heterogeneity. Specifically, when α = 0.01, the FedSTM test



(a) Early-stage distilled data



(b) Late-stage distilled data

Fig. 5. Visualization of distilled data at different stages on CIFAR10.

precision is clearly higher than that of DynaFed on the CIFAR10 and CINIC10 datasets. Additionally, this precision advantage becomes even more pronounced on the NWPU-RESISC45 dataset. As shown in Fig. 4a, Fig. 4e and Fig. 4i, between rounds 26 and 150, when the server fine-tunes the global model using only distilled data from the early stages, the test accuracy of FedSTM is clearly higher than that of DynaFed. This advantage primarily stems from the introduction of the decorrelation regularization term in local model updates, which effectively mitigates dimensional collapse during local training, providing a low-bias prior distribution for synthesizing high-quality early-stage distilled data. A visualization of distilled data at different stages is provided in Fig. 5. Upon reaching the 150th round, in FedSTM, the server switches to late-stage distilled data for fine-tuning, resulting in another noticeable accuracy improvement across all four datasets.

With the increase of α, the performance of FedSTM gradually approaches that of DynaFed. This is because, as data heterogeneity decreases, the local clients' training data progressively resembles a uniform sampling of the global data distribution. Consequently, the early-stage distilled data becomes more representative and general, rather than being heavily biased toward specific patterns or categories. Therefore, during the subsequent fine-tuning process, the global model continuously encounters this nearly unbiased and broadly representative data distribution, allowing it to naturally and smoothly transit from learning simple patterns to complex patterns without relying on additional late-stage distilled data. Furthermore, reducing data heterogeneity mitigates the issue of dimensional collapse in model representation, thereby reducing the marginal benefit of the synergy between trajectory quality and distilled data in the optimization space.

Overall, these results show that FedSTM significantly outperforms baseline methods in highly non-IID scenarios, and its advantage becomes more pronounced as data heterogeneity increases.

C. Ablation Experiment

To validate the effectiveness of local training decorrelation regularization and late stage trajectory matching in FedSTM, we perform ablation experiments on the CIFAR10 dataset with $\alpha = 0.01$. In experiments, the symbol "w/o PTM" denotes the FedSTM without late-stage trajectory matching, and the symbol "w/o REG" denotes the FedSTM without decorrelation regularization in local training. In particular, DynaFed is used as the comparison baseline, as it represents FedSTM without local training regularization and late-stage trajectory matching.

As shown in Fig. 6, the method "w/o PTM", which benefits from the incorporation of local training regularization,

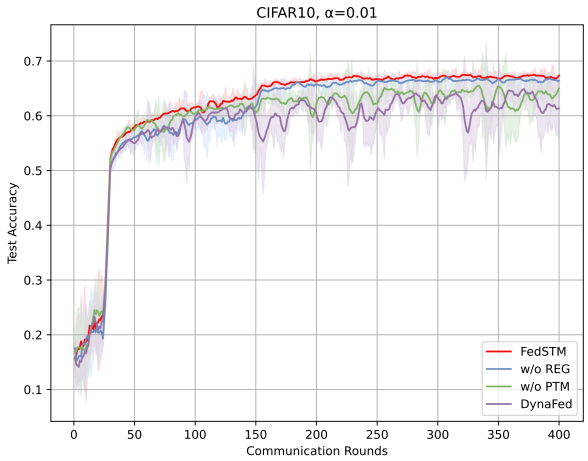


Fig. 6. Ablation experiment with regards to local training decorrelation regularization and late-stage trajectory matching.

achieves higher final test accuracy than DynaFed. The method "w/o REG", which utilizes both early stage and late stage trajectory matching, maintains an accuracy similar to that of DynaFed from rounds 25 to 150 but demonstrates a notable improvement in the final stages. The complete FedSTM, which combines local training regularization and late-stage trajectory matching, achieves the highest test accuracy. The above results clearly demonstrate that both local training regularization and late-stage trajectory matching positively contribute to model performance in FedSTM.

VII. Conclusion

In this article, we propose a novel FL framework, FedSTM, based on stage-wise trajectory matching to effectively deal with the performance degradation problem due to the imbalance of the data distribution (data heterogeneity) between UAVs in UAV networks. Specifically, through a stage-wise trajectory matching mechanism and local training regularization strategy, we effectively address the issue of insufficient access to global data distribution information under non-IID data settings. The experimental results show that FedSTM significantly outperforms existing methods in various heterogeneity scenarios. Ablation studies validate the effectiveness of each component.

While the proposed FedSTM framework demonstrates significant advantages in handling non-IID data, it is subject to certain limitations: 1) Computation overhead. The stage-wise trajectory matching process requires additional computation on

the server side to synthesize distilled data. Although this improves model performance, it may increase the overall resource consumption in resource-constrained UAV environments. 2) Data privacy. Although distilled data are synthetic, they still capture global distribution information. In highly sensitive applications, further analysis is needed to ensure that these data do not inadvertently reveal private information from local datasets. 3) Data modality. Our experiments are conducted primarily on image classification tasks. The effectiveness of FedSTM for other types of data, e.g., sequential data, text, or sensor signals, remains to be validated. 4) Model architecture. FedSTM requires all clients to use the same model architecture, which may not hold in practical UAV networks with heterogeneous hardware. 5) Communication reliability. UAV communication is often unreliable in practice. How such unreliability affects server-side trajectory synthesis and the performance of FedSTM has not been explicitly considered in this work.

We plan to address these limitations in future work by developing lightweight trajectory matching mechanisms to reduce overhead, investigating privacy-enhancing techniques for distilled data synthesis, extending FedSTM to multimodal data and heterogeneous model architectures, and exploring robust strategies to handle unreliable communication.

REFERENCES

[1] B. Fan, Y. Li, R. Zhang, and Q. Fu, "Review on the technological development and application of uav systems," *Chinese Journal of Electronics*, vol. 29, no. 2, pp. 199–207, 2020.

[2] Y. Wang, G. Sun, Z. Sun, J. Wang, J. Li, C. Zhao, J. Wu, S. Liang, M. Yin, P. Wang *et al.*, "Toward realization of low-altitude economy networks: Core architecture, integrated technologies, and future directions," *arXiv preprint arXiv:2504.21583*, 2025.

[3] A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," *Big Data Mining and Analytics*, vol. 5, no. 1, pp. 32–40, 2021.

[4] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[5] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," in *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE, 2022, pp. 965–978.

[6] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[7] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*. PMLR, 2020, pp. 5132–5143.

[8] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[9] M. Shin, C. Hwang, J. Kim, J. Park, M. Bennis, and S.-L. Kim, "Xor mixup: Privacy-preserving data augmentation for one-shot federated learning," *arXiv preprint arXiv:2006.05148*, 2020.

[10] T. Yoon, S. Shin, S. J. Hwang, and E. Yang, "Fedmix: Approximation of mixup under mean augmented federated learning," *arXiv preprint arXiv:2107.00233*, 2021.

[11] Y.-T. Cao, Y. Shi, B. Yu, J. Wang, and D. Tao, "Knowledge-aware federated active learning with non-iid data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 22 279–22 289.

[12] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[13] R. Pi, W. Zhang, Y. Xie, J. Gao, X. Wang, S. Kim, and Q. Chen, "Dynafed: Tackling client data heterogeneity with global dynamics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 177–12 186.

[14] J. Tursunboev, Y.-S. Kang, S.-B. Huh, D.-W. Lim, J.-M. Kang, and H. Jung, "Hierarchical federated learning for edge-aided unmanned aerial vehicle networks," *Applied Sciences*, vol. 12, no. 2, p. 670, 2022.

[15] T. Zeng, O. Semiari, M. Mozaffari, M. Chen, W. Saad, and M. Bennis, "Federated learning in the sky: Joint power allocation and scheduling with uav swarms," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[16] Y. Wang, Z. Su, N. Zhang, and A. Benslimane, "Learning in the air: Secure federated learning for uav-assisted crowdsensing," *IEEE Transactions on network science and engineering*, vol. 8, no. 2, pp. 1055–1069, 2020.

[17] X. Hou, J. Wang, C. Jiang, X. Zhang, Y. Ren, and M. Debbah, "Uav-enabled covert federated learning," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6793–6809, 2023.

[18] Z. Chai, Y. Chen, L. Zhao, Y. Cheng, and H. Rangwala, "Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data," *ArXivorg*, 2020.

[19] Y. Zhou, G. Pu, X. Ma, X. Li, and D. Wu, "Distilled one-shot federated learning," *arXiv preprint arXiv:2009.07999*, 2020.

[20] R. Song, D. Liu, D. Z. Chen, A. Festag, C. Trinitis, M. Schulz, and A. Knoll, "Federated learning via decentralized dataset distillation in resource-constrained edge environments," in *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2023, pp. 1–10.

[21] S. Hu, J. Goetz, K. Malik, H. Zhan, Z. Liu, and Y. Liu, "Fedsynth: Gradient compression via synthetic data in federated learning," *arXiv preprint arXiv:2204.01273*, 2022.

[22] Y. Xiong, R. Wang, M. Cheng, F. Yu, and C.-J. Hsieh, "Feddm: Iterative distribution matching for communication-efficient federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 323–16 332.

[23] Y. Jia, S. Vahidian, J. Sun, J. Zhang, V. Kungurtsev, N. Z. Gong, and Y. Chen, "Unlocking the potential of federated learning: The symphony of dataset distillation via deep generative latents," in *European Conference on Computer Vision*. Springer, 2024, pp. 18–33.

[24] R. Yu, S. Liu, and X. Wang, "Dataset distillation: A comprehensive review," *IEEE transactions on pattern analysis and machine intelligence*, vol. 46, no. 1, pp. 150–170, 2023.

[25] Y. Shi, J. Liang, W. Zhang, V. Y. Tan, and S. Bai, "Towards understanding and mitigating dimensional collapse in heterogeneous federated learning," *arXiv preprint arXiv:2210.00226*, 2022.

[26] Z. Guo, K. Wang, G. Cazenavette, H. Li, K. Zhang, and Y. You, "Towards lossless dataset distillation via difficulty-aligned trajectory matching," *arXiv preprint arXiv:2310.05773*, 2023.

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

[28] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey, "Cinic-10 is not imagenet or cifar-10," *arXiv preprint arXiv:1810.03505*, 2018.

[29] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, 2017.

[30] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A.

y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[31] J. Xiao, C. Du, Z. Duan, and W. Guo, "A novel server-side aggregation strategy for federated learning in non-iid situations," in *2021 20th international symposium on parallel and distributed computing (ISPDC)*. IEEE, 2021, pp. 17–24.