Dual-Personalizing Adapter for Federated Foundation Models

Yiyuan Yang
Australian AI Institute,
Faculty of Engineering & IT
University of Technology Sydney
`Yiyuan.Yang-1@student.uts.edu.au`

Guodong Long
Australian AI Institute,
Faculty of Engineering & IT
University of Technology Sydney
`Guodong.Long@uts.edu.au`

Tao Shen
Australian AI Institute,
Faculty of Engineering & IT
University of Technology Sydney
`Tao.Sheng@uts.edu.au`

Jing Jiang
Australian AI Institute,
Faculty of Engineering & IT
University of Technology Sydney
`Jing.Jiang@uts.edu.au`

Michael Blumenstein
Australian AI Institute,
Faculty of Engineering & IT
University of Technology Sydney
`Michael.Blumenstein@uts.edu.au`

Abstract

Recently, foundation models, particularly large language models (LLMs), have demonstrated an impressive ability to adapt to various tasks by fine-tuning diverse instruction data. Notably, federated foundation models (FedFM) emerge as a privacy preservation method to fine-tune models collaboratively under federated learning (FL) settings by leveraging many distributed datasets with non-IID data. To alleviate communication and computation overhead, parameter-efficient methods are introduced for efficiency, and some research adapted personalization methods to FedFM for better user preferences alignment. However, a critical gap in existing research is the neglect of test-time distribution shifts in real-world applications, and conventional methods for test-time distribution shifts in personalized FL are less effective for FedFM due to their failure to adapt to complex distribution shift scenarios and the requirement to train all parameters. To bridge this gap, we refine the setting in FedFM, termed test-time personalization, which aims to learn personalized federated foundation models on clients while effectively handling test-time distribution shifts simultaneously. To address challenges in this setting, we explore a simple yet effective solution, a **Fed**erated **D**ual-**P**ersonalizing **A**dapter (FedDPA) architecture. By co-working with a foundation model, a global adapter and a local adapter jointly tackle the test-time distribution shifts and client-specific personalization. Additionally, we introduce an instance-wise dynamic weighting mechanism that dynamically integrates the global and local adapters for each test instance during inference, facilitating effective test-time personalization. The effectiveness of the proposed method has been evaluated on benchmark datasets across different NLP tasks with released code.

1 Introduction

Foundation models, especially the large language model (LLM) in natural language processing (NLP), have nearly exhausted public data sources for training. This necessitates alternative solutions to further improve these foundation models by leveraging private or protected data sources, such as business data in companies, smartphones, and so on. **Federated foundation models (FedFM)**[44, 41] offer a promising solution by integrating federated learning (FL) frameworks to enhance the foundation models in a decentralized manner. Built upon existing Parameter-efficient fine-tuning (PEFT) methods [33, 14, 13], FedFM is a collaboratively fine-tuning framework that leverages private datasets with privacy preservation and avoiding overfitting from client-specific fine-tuning.

Test-time distribution shift in FL [16, 29] presents a significant challenge in practical scenarios, as clients may encounter unseen learning tasks during the testing and model inference phases. For example, a client accustomed to writing emails in English may require translation assistance when working on a new project in Chinese. Therefore, it is imperative for the deployed machine learning model to be capable of tackling the test-time distribution shifts from the training data, and our paper addresses this critical issue of test-time distribution shifts within the FedFM scenario. Previous works in test-time FL[29, 16] predominantly utilize conventional deep learning models that are trained from scratch in federated settings. Recent FedFM methods mainly focus on addressing specific challenges related to data heterogeneity [2, 15] and communication overheads [35, 27]. However, none of these methods have discussed test-time distribution shifts in FedFM scenarios.

To fill this gap, we propose a novel FedFM framework that is robust to client-specific alignment and test-time distribution shifts simultaneously. With the support of a foundation model with PEFT methods, we first refine the federated setting, termed *test-time personalization*, which follows: 1) each client needs to train a personalized model using its own data from a target task, and 2) during testing, each client's personalized model needs to be robust to tackle the receiving new tasks (unseen in training) with different distributions (test-time distribution shift). Essentially, the proposed test-time personalization in FL could be simply viewed as an optimization task to seek a sweet trade-off between client-specific model personalization and model generalization to test data.

For test-time personalization in FedFM, two primary challenges—test-time distribution shifts and personalization—necessitate learning tailored to distinct objectives, and the training cost of foundation models also represents a significant concern. To address these issues, we explore a simple yet effective method, dubbed **Fed**erated **D**ual-**P**ersonalizing **A**dapter (FedDPA), where each client learns a global adapter to learn generic knowledge from the aggregation for test-time tasks and maintains a local adapter for targeted ability personalization. During the inference phase, the local and global adapters are dynamically integrated to facilitate prediction, where an instance-wise dynamic weighting mechanism is proposed to autonomously adjudicate the proportional contribution of the local and global adapters for each test instance. Experimental results demonstrate that our method achieves state-of-the-art performance on benchmarks and all data and code are released [1]. Our main contributions are summarized as follows:

- We are the first to explore the test-time distribution shifts problem in federated foundation models for practical application scenarios alignment.
- We introduce a new method, namely dual-personalizing adapter, to emphasize learning both generic and personalized knowledge in the context of FedFM with test-time personalization.
- We conduct an exhaustive analysis using heterogeneous FL benchmarks across diverse NLP tasks. The empirical outcomes reveal that our method attains state-of-the-art performance, underscoring its superior test-time personalization capabilities than existing methods.

2 Related Work

2.1 Adapter-based PEFT Methods

Given the substantial computational and storage burdens associated with directly fine-tuning foundation models, the community has shifted towards embracing parameter-efficient methods [33], with the adapter family [14] being a notable exemplar. According to different architectures, methods in

[1]https://github.com/Lydia-yang/FedDPA

the adapter family can be categorized into four types. The first one is prompt-based learning [18, 21], which is aimed at learning the continuous/soft prompt for discrete optimization. The second one is reparametrization-based methods [13, 9], achieving parameter efficiency by utilizing low-rank techniques to decompose the high-dimensional matrices. The third one is series Adapters [12], which introduce additional learnable modules in a sequential manner within specific sublayers. The last one is parallel Adapters [11], which focus on learning additional learnable modules in a parallel way with distinct sublayers. In this context, our exploration delves into the adapter-based PEFT methods of federated foundation models.

2.2 Federated Foundation Models

With the advent of foundation models, there has been a burgeoning interest [44, 41, 26, 5] in integrating these models within the FL setting. Particularly, in light of the inherent computation and communication cost, recent work [17, 43, 6] endeavors have delved deeper into integrating adapter-based parameter-efficient tuning (PEFT) methods with federated foundation models. Building upon this, a multitude of studies have emerged to navigate the challenges of incorporating federated foundation models with adapter-based PEFT methods. The paper [42] stands at the forefront, initiating the integration of instruction tuning within federated LLM frameworks. Addressing data-related issues, the paper [2] introduced a data-driven initialization approach to mitigate the primary challenges associated with LoRA in highly heterogeneous data scenarios. In addition, the research presented in [15] proposed a method to annotate unlabeled client-side data by harnessing the prowess of large models to address data scarcity concerns. To further optimize the communication and computational overheads associated with federated foundation models, the works [35, 27, 34] emphasize advancing gradient-free optimization methods suitable for devices with limited memory and computing power. For personalization, paper [38] focused on designing a specific training paradigm for LoRA to achieve more effective personalization in visual model-heterogeneous scenarios. Diverging from these approaches, our work delves into the realm of personalization with adapters in federated foundation models, extending the scope of research in this area.

2.3 Personalized Federated Learning

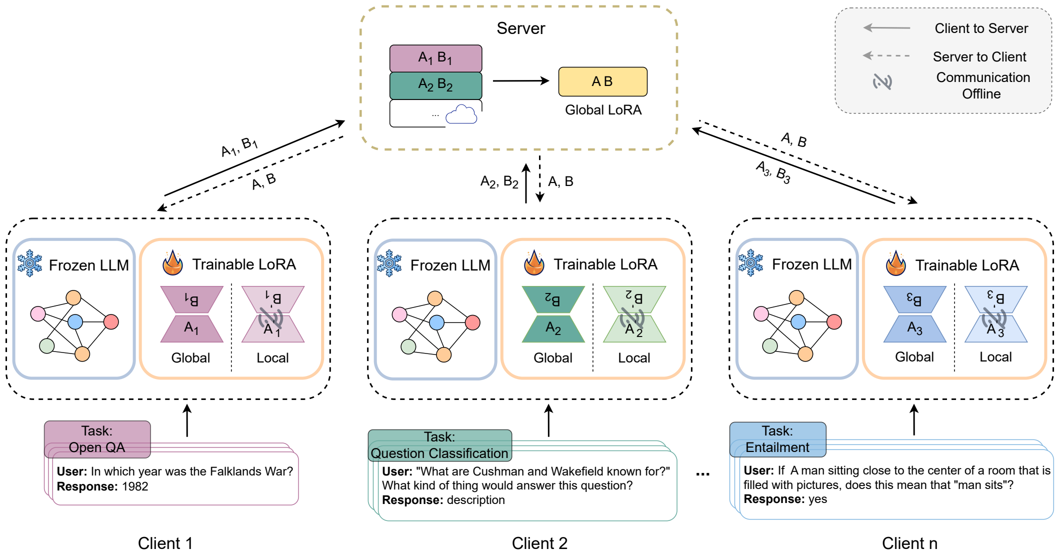
To address the necessity of personalization for individual clients, personalized Federated Learning (PFL) [28], which aims at training to cater to individual client preferences and needs, is proposed. Broadly, existing PFL methods can be categorized into two primary types: fine-tuning the global model for personalization or learning additional personalized models. Research works [10, 7] in the first category fine-tuned the whole or part of the global model with each client's local dataset for personalization. While research works [19, 22] in the second category is to learn the additional personalized layers or model through local aggregation. Nonetheless, a prevalent limitation among these PFL approaches is their concentrated focus on a specifically targeted task, often at the expense of performance when encountering test-time distribution shifts.

To fill this gap, recent research has shifted focus towards exploring different test-time distribution shifts in PFL. In contrast to studies [39, 8] in federated continual/incremental learning possessing ample annotated data from different distributions for training to address shifts, test-time FL focuses on handling distribution shifts during testing without the availability of annotated data for further training. One strand of research [3, 32] concentrates on addressing test-time distribution shifts that occur when new clients are introduced during the testing phase by module/prior adaptation. Another line of studies[29, 16] aims to tackle distribution shifts in existing clients during testing by aligning test features with existing features. Our paper falls into the second type and differs from previous work by exploring this challenge within the framework of foundation models, which are characterized by extensive parameter scales and more complex test-time distribution shifts.

3 Problem Definition

3.1 Test-time Personalization in FedFM

Considering M clients in an FL system, each client possesses its distinct local training dataset $\mathbb{D}^m_{train}$ and test dataset $\mathbb{D}^m_{test}$, where m indexes a client. One data pair in datasets is denoted as $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x}$ is the input data and $\boldsymbol{y}$ is its corresponding label.



User: In which year was the Falklands War?
Response: 1982

User: "What are Cushman and Wakefield known for?" What kind of thing would answer this question?
Response: description

User: If A man sitting close to the center of a room that is filled with pictures, does this mean that "man sits"?
Response: yes

Figure 1: The overall framework of FedDPA. Each client contains a frozen LLM, a trainable global adapter (LoRA) and a trainable local adapter (LoRA) with a specific task, where the global adapter (LoRA) is for test-time tasks and the local adapter (LoRA) is for personalization. During the training, only the parameters of the global adapter (LoRA) are transmitted to the server for aggregation.

In each client m, we will introduce the training and testing phases separately for our test-time personalization setting. In the training phase, model training utilizes solely the local dataset $\mathbb{D}^m_{train}$, which is derived from the distribution P^m_s. While in the testing phase, the test dataset $\mathbb{D}^m_{test}$ comprises two components: the test set $\mathbb{D}^m_s$ driven from the same distribution P^m_s as training data, and additional test sets $\mathbb{D}^m_t$ under data distribution shifts $P^m_s(\boldsymbol{x}, \boldsymbol{y}) \neq P^m_t(\boldsymbol{x}, \boldsymbol{y})$. Therefore, the test dataset is $\mathbb{D}^m_{test} = \mathbb{D}^m_s \cup \mathbb{D}^m_t$, and we call these datasets $\mathbb{D}^m_t$ as test-time datasets. Unlike previous works[29, 16] in test-time FL concentrating on either feature-level shifts $P^m_s(\boldsymbol{x}) \neq P^m_t(\boldsymbol{x})$ or label shifts $P^m_s(\boldsymbol{y}) \neq P^m_t(\boldsymbol{y})$, we investigate a more complex scenario where various distribution shifts, including semantic shifts, domain shifts and others, exist simultaneously. This is aligned with the practical application of foundation models, which often encounter testing data originating from diverse domains, backgrounds, or populations.

Therefore, the objective of the model in each client should not only perform well on the test set $\mathbb{D}^m_s$ (refer to personalization) but also have comparable results on the test-time dataset $\mathbb{D}^m_t$ (refer to test-time performance). This objective is consistent with the practical scenarios, since users primarily focus on the abilities they often utilize (abundant data available for training) and occasionally also introduce new tasks (limited to test data).

3.2 Challenges

The test-time personalization setting raises two pivotal considerations: *personalization* and *test-time distribution shifts*. Personalization is the primary focus, followed by optimizing test-time tasks. Our proposed method introduced in section 4 is designed to achieve personalization within FedFM while ensuring comparable results for test-time tasks, and its vital intuition is illustrated below.

Considering a foundation model, it comprises a main body $f(\boldsymbol{\theta})$, which holds most of the parameters and processes input $\boldsymbol{x}$ to produce output features $\boldsymbol{h} = f(\boldsymbol{x}; \boldsymbol{\theta})$. Additionally, there is a tail $g(\boldsymbol{\theta}_t)$ that maps these features to the output space (e.g., vocabulary), resulting in the predicted result $\hat{\boldsymbol{y}} = g(\boldsymbol{h}; \boldsymbol{\theta}_t)$. Typically, the focus in tuning and adaptations primarily lies on the main body $f(\boldsymbol{\theta})$ because the tail $g(\boldsymbol{\theta}_t)$, usually a linear function, remains unchanged (frozen) during tuning [14].

Discordance between Personalization and Test-time Tasks. The key to addressing test-time distribution shifts lies in learning generic features universally applicable across disparate distributions [1]. That is, learning a foundation model $f(\boldsymbol{\theta})$ to satisfy $P_s(f(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{y}) = P_t(f(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{y})$ although

$P_s(\boldsymbol{x}, \boldsymbol{y}) \neq P_t(\boldsymbol{x}, \boldsymbol{y})$. FL is a methodology designed to learn generic features across diverse non-IID data (different distributions) through aggregation algorithms [5, 29]. Therefore, we tailor FL training for addressing test-time tasks with the objective $\min_{\boldsymbol{\theta}} \mathcal{L}_{P_{all}}(\boldsymbol{\theta})$, where $\mathcal{L}_{P_{all}}$ represents the loss function designed for learning generic features towards all clients' distributions P_{all}. However, personalization focuses on aligning the model with the specific distribution P_s, which means learning a foundation model $f(\boldsymbol{\theta})$ with the objective $\min_{\boldsymbol{\theta}} \mathcal{L}_{P_s}(\boldsymbol{\theta})$, where $\mathcal{L}_{P_s}$ represents the loss function designed for learning personalized features towards the specific distribution. Therefore, the discordance between specific distribution alignment for personalization and generic feature learning for test-time tasks leads to inconsistent optimization objectives.

The above analysis motivates a dual model strategy—one model for test-time tasks and one model for personalization—to realize test-time personalization in FedFM. This strategy, together with our other techniques presented below for FedFM scenarios, constitutes the foundation of our method.

4 Proposed Method

To align with the application scenarios, we consider the test-time personalization setting in FedFM. Following a similar assumption from a Mixture of Experts [23], *any test-time task (distributions) to a client can be approximated as a mixture of training tasks seen by other clients in the federated learning system.* Therefore, each client primarily personalizes its model based on its local training task, while also tackling unseen test-time tasks by leveraging insights gained from other clients in the federated learning system. Discussions of other scenarios can be found in Appendix C.

In test-time personalization, test-time distribution shifts and personalization are two main issues that need to be addressed, and their optimization objectives toward different distributions are inconsistent. To address these challenges and consider the efficient learning of FedFM, we propose a **Fed**erated **D**ual-**P**ersonalizing **A**dapter (FedDPA) system for each client, as shown in Fig 1. During training, a global adapter is employed to acquire generic features by FL training for test-time tasks (Sec. 4.1). Meanwhile, to address personalization, a local adapter is maintained locally to align with the client's specific distribution, and leverages generic knowledge from the global adapter for faster learning (Sec. 4.2). During the inference, the learned global and local adapters are dynamically combined using a weight generated by the instance-wise dynamic weighting mechanism for each input test instance, realizing test-time personalization (Sec. 4.3).

The Overall Objective. Considering the computation and communication cost of FedFM, we utilize the adapter-based PEFT methods, which only learn a small part of parameters $\Delta\boldsymbol{\theta}$ while keeping most of the parameters $\boldsymbol{\theta}$ frozen. Our proposed FedDPA is to learn the global adapter $\Delta\boldsymbol{\theta}_g$ and local adapters $\Delta\boldsymbol{\theta}_l^m$ *simultaneously* across M client to realize test-time personalization,

$$\begin{aligned} \min_{\Delta\boldsymbol{\theta}_g, \{\Delta\boldsymbol{\theta}_l^m\}} \quad & \sum_{m=1}^{M} [\mathcal{L}_{(\boldsymbol{x},\boldsymbol{y})\sim P_s^m}(\boldsymbol{\theta}; \Delta\boldsymbol{\theta}_g; \Delta\boldsymbol{\theta}_l^m)] \\ s.t. \quad & \Delta\boldsymbol{\theta}_g^* \in \arg\min_{\Delta\boldsymbol{\theta}_g} \mathcal{L}_{(\boldsymbol{x},\boldsymbol{y})\sim P_{all}}(\boldsymbol{\theta}; \Delta\boldsymbol{\theta}_g; \Delta\boldsymbol{\theta}_l^m) \end{aligned} \tag{1}$$

where the first part is a standard personalized FL loss to find optimal personalized models by minimizing the sum of loss on local training tasks $\mathcal{L}_{(\boldsymbol{x},\boldsymbol{y})\sim P_s^m}(.)$, and the second part is a constraint term to seek an optimal solution by minimizing the test-time loss $\mathcal{L}_{(\boldsymbol{x},\boldsymbol{y})\sim P_{all}(.)}$. Because we assume that the test-time task is unseen to a client but observed by other clients, the test-time loss can be estimated using the Empirical Risk Minimization of all client's training tasks $\min_{\Delta\boldsymbol{\theta}_g} \sum_{m=1}^{M} r_m \mathcal{L}_m(\boldsymbol{\theta}; \Delta\boldsymbol{\theta}_g)$, where P_{all} denotes all distributions of tasks in all clients, r_m denotes each client's weight for aggregation (e.g., in FedAvg, r_m is the proportion of each client's data number to all clients' data number) and $\mathcal{L}_m$ denotes the loss for each client over its local training dataset. Since the above objective cannot be solved directly, *we propose to alternatively learn the global and local adapters in a sequential manner (FedDPA-F with local adapter fine-tuning) or iterative manner (FedDPA-T with local adapter training).* Detailed algorithms of these two methods are in Appendix A.3.

Remark. To simplify the illustration, we use LLM as the backbone and adopt LoRA [13] as the adapter-based PEFT method in our framework. The overall framework is easy to adapt to other types of backbone and other adapter-based PEFT methods. LoRA decomposes the training weight into a frozen weight $\boldsymbol{\theta}$, and a trainable weight derived by the multiplication of two low-rank weights $\Delta\boldsymbol{\theta} = \Delta\boldsymbol{\theta}^b \Delta\boldsymbol{\theta}^a$. The data heterogeneity in FL with LLM primarily manifests as distribution shifts across various NLP tasks among different clients, driven by diverse backgrounds, topics, and other contextual factors, and local loss $\mathcal{L}_m$ for all NLP tasks is a standard language modeling objective [4].

4.1 Generic Learning of Global Model

Addressing test-time distribution shifts requires the acquisition of generic knowledge that is applicable across various distributions [1]. The conventional federated learning process is inherently designed to aggregate this generic knowledge among different non-IID data. Consequently, we utilize the adapter trained within the FL context as the global adapter for addressing test-time tasks. To further enhance generic learning, our model aggregation strategy is based on the client number rather than the number of data by considering the potential biases stemming from different numbers of tasks.

At each client, there consists of a frozen LLM model $f(\boldsymbol{x}; \boldsymbol{\theta})$ with a global lightweight global adapter (LoRA) $\Delta\boldsymbol{\theta}_g = \Delta\boldsymbol{\theta}_g^b \Delta\boldsymbol{\theta}_g^a$. This global adapter is used for aggregation by sending to the server. Notably, the server's role is limited to computing the aggregated adapter $\Delta\boldsymbol{\theta}_g$, thus obviating the need for maintaining a large-scale model. Similar to the standard FL process, for each client m, the adapter weight $\Delta\boldsymbol{\theta}_g^m$ is learned locally and sent to the server. Upon receipt of the adapter weights from all clients, the server employs FedAvg [24] to aggregate them and sends $\Delta\bar{\boldsymbol{\theta}}_g$ back to each client as their initialized parameter in a new round. It can be formulated as:

$$\textbf{Server: } \Delta\bar{\boldsymbol{\theta}}_g = \sum_{m=1}^{M} \frac{1}{M} \Delta\boldsymbol{\theta}_g^m, \quad \textbf{Client: } \Delta\boldsymbol{\theta}_g^m = \arg\min_{\Delta\boldsymbol{\theta}_g} \mathcal{L}_m(\boldsymbol{\theta}; \Delta\boldsymbol{\theta}_g), \text{ initialized with } \Delta\bar{\boldsymbol{\theta}}_g \quad (2)$$

Remark. Other federated algorithms like FedProx [20] can also be applied with LoRA tuning of this global model learning (in Appendix B.1). In this paper, we just take FedAvg as an example.

4.2 Personalization of Local Model

The previously developed global model, which focuses on acquiring generic features across diverse datasets, faces challenges with personalization due to inconsistent optimization objectives. To address this, we integrate a local adapter to better align with each client's specific distribution. We explore two methods as shown in Fig 2, 1) Learning sequentially: after global adapter training, the local adapter is initialized by the learned global adapter and directly fine-tuned; 2) Learning iteratively: during each communication round of global adapter training, the local adapter is re-initialized from its last state, fine-tuned alongside the frozen global adapter, and maintained locally without communication.

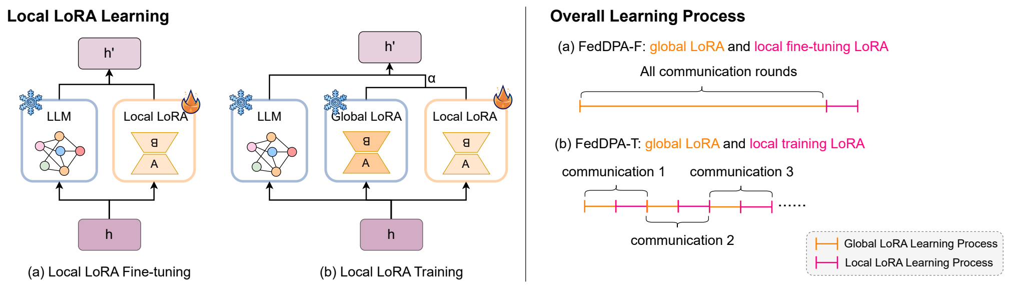


Figure 2: Frameworks of two personalized methods for local adapter (LoRA) are shown on the left, with their overall learning processes on the right.

To be more specific, a local adapter (LoRA) $\Delta\boldsymbol{\theta}_l = \Delta\boldsymbol{\theta}_l^b \Delta\boldsymbol{\theta}_l^a$ is introduced. Thus, each client contains three components: a frozen LLM $\boldsymbol{\theta}$, a global adapter (LoRA) $\Delta\boldsymbol{\theta}_g$ and a local adapter (LoRA) $\Delta\boldsymbol{\theta}_l^m$. As delineated in Fig 2 (a), for the first method, after global training, the local adapter is first initialized by the global adapter denoted as $\Delta\boldsymbol{\theta}_l^m = \Delta\boldsymbol{\theta}_g$, then fine-tuned on local data to get the final local adapter. As shown in Fig 2 (b), for the second method, during each communication round in training for each adapter layer, upon receiving an input $\boldsymbol{h}$, it simultaneously traverses the frozen LLM, the frozen global adapter and the local adapter. The process entails an initial fusion of the outputs from both the local and global adapters with a predefined weighting factor of α, followed by integration with the output of the LLM to yield the final result $\boldsymbol{h}^{'} = \boldsymbol{\theta h} + ((1-\alpha)\cdot\Delta\boldsymbol{\theta}_g\boldsymbol{h} + \alpha\cdot\Delta\boldsymbol{\theta}_l^m\boldsymbol{h})$. Therefore, the learning of the local adapter $\Delta\boldsymbol{\theta}_l^m$ for these two methods can be unified as:

$$\Delta\boldsymbol{\theta}_l^m = \arg\min_{\Delta\boldsymbol{\theta}_l^m} \mathcal{L}_m(\boldsymbol{\theta}; \Delta\boldsymbol{\theta}_g; \Delta\boldsymbol{\theta}_l^m), \quad \text{initialized with } \Delta\boldsymbol{\theta}_g \text{ or previous } \Delta\boldsymbol{\theta}_l^m \tag{3}$$

4.3 LLM-enhanced Instance-wise Dynamic Weighting Mechanism

As discussed in previous test-time FL methods [16], a dynamic combination of global components and personalized components can improve generalization while reducing the cost of hyper-parameter tuning in the deployment stage. Considering the disparate data distributions that characterize test-time tasks and local tasks and the wealth of training instances of local tasks available to each client, we propose an *instance-wise dynamic weighting mechanism* to calculate the similarity between the input instance and local instances, using this metric to determine the appropriate weight balance for the global and local adapter combination. To facilitate this, the representation of each input instance is essential. Leveraging the robust capability of pre-trained LLMs to abstract input sentences, we utilize the hidden states from the final layer of the LLM as the representation. Given that the LLM is decoder-based, with tokens attending only to preceding tokens, the embedding of the final token is considered representative of the entire input for similarity evaluation. Furthermore, to enhance the representation quality, the global adapter, which embodies generic knowledge, is incorporated into this embedding process.

More specifically, during the inference stage, for each input instance $\boldsymbol{x}$ in a client, we randomly sample S instances $\{\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_s\}$ from the local training dataset. These instances are then fed into the LLM, augmented with the global adapter, to obtain the last token's embeddings from the final layer, denoted as $\boldsymbol{w}_x$ and $\{\boldsymbol{w}_{x_0}, \boldsymbol{w}_{x_1}, ..., \boldsymbol{w}_{x_s}\}$ respectively. Subsequently, we calculate the similarity between the input representation $\boldsymbol{w}_x$ and each sampled local representation in $\{\boldsymbol{w}_{x_0}, \boldsymbol{w}_{x_1}, ..., \boldsymbol{w}_{x_s}\}$, resulting in a score range of $[0, 1]$. Finally, we average all scores to obtain the final result, represented as $\alpha_t = \lambda \cdot \sum_{i=0}^{S} \frac{1}{S}\,\mathrm{Sim}(\boldsymbol{w}_x, \boldsymbol{w}_{x_i})$, where Sim represents the function to calculate the similarity, and λ is a scale factor in $(0, 1]$ to restrict the maximum similarity score (especially for FedDPA-T).

Through this method, the balancing of weights between the global and local adapters is dynamically adjusted for each test instance, ensuring the model not only tailors to the individual client's specific needs but also benefits from the aggregated model's generic knowledge across test-time tasks.

5 Experiment

5.1 Experiment Setting

Datasets. We construct two federated datasets from Flan [31], which is a collection of various NLP tasks from over 60 datasets for instruction turning. In order to be better suitable for FL settings, we randomly select 8 NLP tasks from different datasets for each federated dataset and downsample the original datasets with more details in Appendix A.1. ROGUE-1 is taken as a metric.

Baselines and Implementation. We compare our methods with four baselines based on the same model architecture: centralized model, Local-finetuned model, FedIT [42] and FedLoRA [38]. The centralized model is trained on all data of tasks in one center. The local-finetuned model infers that only local data are used to train the model without any communication with other clients or the server.

We distribute data between clients based on the NLP task for data heterogeneity, where different NLP tasks generated from different contextual factors inherently suffer from various complex distribution shifts. Since we select 8 NLP tasks, corresponding to $M = 8$ clients.For each client, the local task serves as the primary focus for personalization, while the tasks from other clients are taken as test-time tasks. To better evaluate the effectiveness of methods, we assume that all clients are activated for every communication round and set the communication round $K = 20$. The alpaca-LoRA[2] is adapted as the base model initialized with LLaMA-7B.[3] The updating weight of local LoRA training (FedDPA-T) is $\alpha = 0.5$ ($\lambda = 0.5$) for federated dataset 1 and $\alpha = 0.3$ ($\lambda = 0.3$) for federated dataset

[2]https://github.com/tloen/alpaca-lora

[3]https://huggingface.co/huggyllama/llama-7b

2. We set $S = 5$ and choose cosine similarity for instance-wise dynamic weighting mechanism. More details are in Appendix A.2.

Table 1: Personalization and test-time personalization results of different models on federated dataset 1. FedDPA-F represents the model with the local fine-tuning adapter and FedDPA-T represents the model with the local training adapter. Linguistic represents the linguistic acceptability task, Word Dis represents the word disambiguation task, and Question CLS represents question classification task.

Methods	Federated Dataset 1								
	Para -phrase	Entail -ment	Structure to Text	Text For -matting	Linguistic Acc	Word Dis	Core -ference	Question CLS	Average
Personalization									
Centralized	77.00	82.00	72.58	96.59	70.50	63.50	77.59	89.00	78.60
FedIT	69.00	83.00	71.25	96.32	71.50	62.50	75.43	91.50	77.50
FedLoRA	77.50	84.00	71.49	96.69	73.50	**65.00**	75.27	92.00	79.43
Local-finetuned	74.50	80.00	**73.71**	**97.36**	**75.00**	54.50	68.55	89.50	76.64
FedDPA-F	79.00	**84.50**	72.06	96.90	72.00	**65.00**	73.86	92.50	79.48
FedDPA-T	**80.50**	**84.50**	72.79	96.51	73.50	62.00	**77.93**	**94.00**	**80.22**
Test-Time Personalization									
Local-finetuned	48.99	47.24	27.53	22.66	48.86	49.07	46.45	52.09	42.86
FedLoRA	75.56	76.55	75.21	74.94	76.16	74.64	74.99	76.97	75.63
FedDPA-F	**78.10**	**77.36**	**77.18**	**76.98**	**77.11**	**76.23**	**76.84**	**77.19**	**77.12**
FedDPA-T	76.20	75.51	76.19	75.63	74.86	74.60	74.77	75.96	75.47

5.2 Main Results

We compare FedDPA with other baselines on two main evaluation facets: *personalization* (scores on targeted local tasks) and *test-time personalization* (average scores on all tasks including test-time tasks). As evidenced in Table 1 and Table 2, our proposed dual-personalizing adapter methods (both fine-tuning and training) exhibit superior performance in personalization compared to other baseline models, which demonstrates the effectiveness of local adapter maintenance for enhancing performance on the targeted local task. For test-time personalization, the FedDPA-F method stands out as the most effective among all personalized models, which suggests that incorporating learning from the global adapter can be instrumental in adapting to test-time distribution shifts for a more comprehensive model achievement. Additionally, given that the global adapter aggregated on different distributions matin certain generalization capabilities, the local adapter of FedDPA-F has better generalization performance than that of FedDPA-T, which leads to better performance on most test-time tasks. More importantly, it is noteworthy that while centralized or global models may yield higher average

Table 2: Personalization and test-time personalization results of different models on federated dataset 2. FedDPA-F represents the model with the local fine-tuning adapter and FedDPA-T represents the model with the local training adapter. Reading Com represents the reading comprehension task.

Methods	Federated Dataset 2								
	Para -phrase	Common -sense	Entail -ment	Text For -matting	Summari -zation	Reading Com	Senti -ment	Open QA	Average
Personalization									
Centralized	87.00	64.67	77.00	90.65	29.12	76.00	72.50	76.17	71.64
FedIT	86.00	63.13	79.00	89.80	30.36	75.50	72.00	81.06	72.07
FedLoRA	87.00	64.12	**84.50**	89.52	27.13	76.50	73.50	79.62	72.74
Local-finetuned	75.00	53.51	81.00	91.28	27.51	69.00	72.50	79.31	68.64
FedDPA-F	88.00	64.80	84.25	89.82	29.58	78.50	72.00	80.89	73.48
FedDPA-T	**90.50**	**70.54**	82.00	**91.81**	**30.75**	**81.00**	**75.00**	**91.07**	**75.33**
Test-Time Personalization									
Local-finetuned	48.21	49.07	49.75	21.86	17.35	48.57	44.04	48.19	40.88
FedLoRA	69.60	71.64	71.09	71.28	65.63	68.89	70.32	70.44	69.86
FedDPA-F	**71.64**	72.28	**72.42**	72.39	**71.12**	**70.46**	**71.00**	**71.82**	**71.64**
FedDPA-T	71.63	**72.66**	71.20	**72.58**	70.58	69.21	70.67	71.62	71.27

performances across all tasks, they fall short in excelling at specific tasks for personalization, aligning with the conclusions of the previous study [30].

6 Analysis

6.1 Convergence Analysis

We present the convergence analysis of our methods in Figure 3. Figure 3 (a) compares our methods with other baselines for personalization, with the results showcasing the average performance on target local tasks across all clients. Notably, our methods exhibit a more rapid convergence compared to FedIT and achieve notable performance enhancements after five communication rounds. Despite sharing similar trends with FedLoRA, our approaches, particularly the FedDPA-T, ultimately outperform in personalization. For a more granular insight into test-time personalization convergence, Figure 3 (b) compares average performance on all tasks, including each client's targeted local and test-time tasks. The results substantiate that our approaches demonstrate faster convergence rates, further bolstering the efficacy of our methods.

Table 3: Ablation study of instance-wise dynamic weighting mechanism (Auto). P represents personalization, and TTP represents test-time personalization.

Methods	Auto	**Fed Dataset 1**		**Fed Dataset 2**	
		P	TTP	P	TTP
FedDPA-F	✗	79.06	76.97	73.17	**71.70**
	✓	**79.48**	**77.12**	**73.48**	71.64
FedDPA-T	✗	79.57	60.06	73.75	63.57
	✓	**80.22**	**75.47**	**75.33**	**71.27**

Table 4: Ablation study of updating weight. P represents personalization, and TTP represents test-time personalization.

Methods	α	**Fed Dataset 1**		**Fed Dataset 2**	
		P	TTP	P	TTP
FedDPA-T	0.3	79.69	**75.85**	**75.33**	**71.27**
	0.5	**80.22**	75.47	74.10	70.72
	0.7	79.88	75.01	74.04	69.95

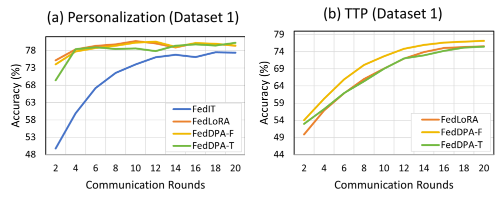


Figure 3: Average accuracy varies as communication rounds.

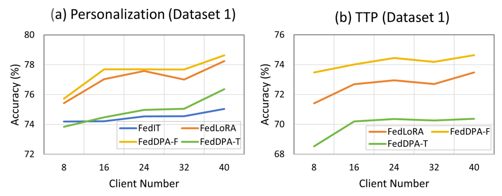


Figure 4: Average accuracy varies as different client participation numbers.

6.2 Ablation Study

Impact of Instance-Wise Dynamic Weighting Mechanism. To explore the impact of the instance-wise dynamic weighting mechanism, we implemented experiments with FedDPA methods on different datasets. As shown in Table 3, the incorporation of an instance-wise dynamic weighting mechanism contributes significantly to enhancing performance in both personalization and test-time personalization scenarios. More ablation studies are in Appendix B.2.

Impact of Updating Weight α. In this study, we investigated the influence of the updating weight α during FedDPA-T training with its value $\alpha \in \{03, 0.5, 0.7\}$. As can be seen in Table 4, for test-time personalization, increasing updating weight α will decrease the performance due to the increased proportion of the local adapter in the model, while for personalization, different updating weights α are required for different datasets to achieve their optimal results.

Impact of Client Number. To better align with the FL setting in practical application, we scaled up clients to 40 and implemented experiments with sample rate $\{0.2, 0.4, 0.6, 0.8, 1\}$. For each communication round, the server will select clients from each task based on the sample rate (more details in Appendix A.1). As shown in Figure 4, as the client participant rates increase, model accuracy also increases as more participating clients provide more data for knowledge learning. Besides, FedDPA-F outperforms all baselines, whereas FedDPA-T exhibits somewhat inferior performance, potentially due to overfitting issues when handling a small dataset.

More experiments and analyses of scalability and efficiency can be found in Appendix B.3 and B.4.

7 Conclusion

Federated Foundation Model (FedFM) is a promising direction to enhance existing Foundation Models, e.g. LLM, by leveraging private data sources. Test-time distribution shift is a critically important problem to ensure the practicability of the FedFM system. This work is the first to propose the test-time FedFM setting. To tackle this challenging scenario, we propose a novel dual-personalizing adapter for the FedFM framework. The method is evaluated on public NLP tasks that are adapted to mimic the test-time FedFM setting. This work is the first step towards this direction. We focus on defining a new learning scenario, proposing a basic learning framework, and setting up the benchmark datasets. Our future works will be in two directions: the first is to rethink this problem from a theoretical perspective, and the second is to enhance the benchmark setting with more datasets in real applications.

References

[1] Martin Arjovsky. *Out of distribution generalization in machine learning*. PhD thesis, New York University, 2020.

[2] Sara Babakniya, Ahmed Roushdy Elkordy, Yahya H Ezzeldin, Qingfeng Liu, Kee-Bong Song, Mostafa El-Khamy, and Salman Avestimehr. Slora: Federated parameter efficient fine-tuning of language models. *arXiv preprint arXiv:2308.06522*, 2023.

[3] Wenxuan Bao, Tianxin Wei, Haohan Wang, and Jingrui He. Adaptive test-time personalization for federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Zachary Charles, Nicole Mitchell, Krishna Pillutla, Michael Reneer, and Zachary Garrett. Towards federated foundation models: Scalable dataset pipelines for group-structured learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] Haokun Chen, Yao Zhang, Denis Krompass, Jindong Gu, and Volker Tresp. Feddat: An approach for foundation model finetuning in multi-modal heterogeneous federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11285–11293, 2024.

[7] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pages 2089–2099. PMLR, 2021.

[8] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10164–10173, 2022.

[9] Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*, 2022.

[10] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

[11] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[12] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[13] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[14] Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*, 2023.

[15] Jingang Jiang, Xiangyang Liu, and Chenyou Fan. Low-parameter federated learning with large language models. *arXiv preprint arXiv:2307.13896*, 2023.

[16] Liangze Jiang and Tao Lin. Test-time robust personalization for federated learning. In *ICLR*, 2023.

[17] Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. *arXiv preprint arXiv:2309.00363*, 2023.

[18] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[19] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[20] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[21] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[22] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.

[23] Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42:275–293, 2014.

[24] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[26] Chao Ren, Han Yu, Hongyi Peng, Xiaoli Tang, Anran Li, Yulan Gao, Alysa Ziying Tan, Bo Zhao, Xiaoxiao Li, Zengxiang Li, et al. Advances and open challenges in federated learning with foundation models. *arXiv preprint arXiv:2404.15381*, 2024.

[27] Jingwei Sun, Ziyue Xu, Hongxu Yin, Dong Yang, Daguang Xu, Yiran Chen, and Holger R Roth. Fedbpt: Efficient federated black-box prompt tuning for large language models. *arXiv preprint arXiv:2310.01467*, 2023.

[28] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[29] Yue Tan, Chen Chen, Weiming Zhuang, Xin Dong, Lingjuan Lyu, and Guodong Long. Is heterogeneity notorious? taming heterogeneity to handle test-time shift in federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[30] Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. How far can camels go? exploring the state of instruction tuning on open resources. *arXiv preprint arXiv:2306.04751*, 2023.

[31] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[32] Jian Xu and Shao-Lun Huang. A joint training-calibration framework for test-time personalization with label shift in federated learning. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4370–4374, 2023.

[33] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148*, 2023.

[34] M Xu, D Cai, Y Wu, X Li, and S Wang. Fwdllm: Efficient fedllm using forward gradient. 2024.

[35] Mengwei Xu, Yaozong Wu, Dongqi Cai, Xiang Li, and Shangguang Wang. Federated fine-tuning of billion-sized language models across mobile devices. *arXiv preprint arXiv:2308.13894*, 2023.

[36] Xiangpeng Yang, Linchao Zhu, Hehe Fan, and Yi Yang. Eva: Zero-shot accurate attributes and multi-object video editing. *arXiv preprint arXiv:2403.16111*, 2024.

[37] Xiangpeng Yang, Linchao Zhu, Xiaohan Wang, and Yi Yang. Dgl: Dynamic global-local prompt tuning for text-video retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6540–6548, 2024.

[38] Liping Yi, Han Yu, Gang Wang, and Xiaoguang Liu. Fedlora: Model-heterogeneous personalized federated learning with lora tuning. *arXiv preprint arXiv:2310.13283*, 2023.

[39] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pages 12073–12086. PMLR, 2021.

[40] Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, et al. Unleashing the power of multi-task learning: A comprehensive survey spanning traditional, deep, and pretrained foundation model eras. *arXiv preprint arXiv:2404.18961*, 2024.

[41] Sixing Yu, J Pablo Muñoz, and Ali Jannesari. Federated foundation models: Privacy-preserving and collaborative learning for large models. *arXiv preprint arXiv:2305.11414*, 2023.

[42] Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Guoyin Wang, and Yiran Chen. Towards building the federated gpt: Federated instruction tuning. *arXiv preprint arXiv:2305.05644*, 2023.

[43] Zhuo Zhang, Yuanhang Yang, Yong Dai, Qifan Wang, Yue Yu, Lizhen Qu, and Zenglin Xu. Fedpetuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models. In *Annual Meeting of the Association of Computational Linguistics 2023*, pages 9963–9977. Association for Computational Linguistics (ACL), 2023.

[44] Weiming Zhuang, Chen Chen, and Lingjuan Lyu. When foundation model meets federated learning: Motivations, challenges, and future directions. *arXiv preprint arXiv:2306.15546*, 2023.

Appendix

A Implementation Details

A.1 Datasets

In this paper, we have developed two federated datasets derived from the Flan [31], and details to construct these datasets are elucidated in this section. Flan encompasses a diverse array of NLP tasks, each comprising multiple datasets. These tasks, generated from different contextual factors, inherently experience various complex distribution shifts. To align with FL settings, we employed a stratified selection process, randomly choosing one dataset from each of the eight distinct tasks from Flan to form each federated dataset. In addition, to simulate client local data scarcity [24], we implemented a downsampling strategy, reducing the size of each selected local dataset to 300 training instances and 200 testing instances. Consequently, each constructed federated dataset encompasses eight distinct NLP tasks, culminating in a whole dataset comprising 2400 training examples and 1600 testing examples across all tasks. The specific tasks and datasets included in each federated dataset are cataloged in Table 5.

The NLP tasks within these datasets can be broadly divided into two types: generation tasks and classification tasks. To facilitate uniform processing by LLM, all tasks are converted into a generative format, employing distinct instructions for each dataset. Illustrative examples of these data for both classification and generation tasks are provided in Table 6. For the input of the LLM, we adopted a simple template, the details of which are delineated in Table 7.

Dataset Partitioning for Ablation Study. In our ablation study in section 6.2 examining the client number to align with FL settings, we divided each task in our constructed federated datasets into five subsets, each comprising an equal number of training data. Based on our assumption that each client is associated with a single task, this division results in a total of 40 clients, with each client possessing a local dataset of 60 training examples. To mimic real-world FL communication dynamics, we employed a randomized selection process for clients (subsets) within each task according to specified sample rates. Accordingly, for sample rates specified as $\{0.2, 0.4, 0.6, 0.8, 1\}$, we selected 1,2,3,4, and 5 clients (subsets) per task, leading to 8, 16, 24, 32, and 40 clients participating in federated communications, respectively. The evaluation phase involves computing the average results across these selected clients for each specified sample rate, which provides a comprehensive analysis of how client numbers influence the performance of our method.

Different with Multi-Task Learning. Although both multi-task learning and our test-time personalization setting involve multiple tasks during training and testing, multi-task learning operates in a centralized setting, whereas our setting is based on federated learning, a distributed setting. More importantly, our setting accounts for test-time distribution shifts, a challenge that is typically overlooked in conventional multi-task learning. Additionally, fine-tuning on the combination of multi-task data from a centralized foundation model serves as a strong baseline for multi-task learning. In foundation models, all tasks are standardized into a uniform format, and the model benefits from task-agnostic token embeddings learned through extensive pre-training on diverse data. Thus, directly fine-tuning on this multi-task data represents the implementation of multi-task learning using foundation models [40]. We have included this baseline, referred to as "Centralized," in our experimental comparisons.

A.2 Baselines and Implementation

In this section, detailed descriptions of the implementation of FedDPA and each baseline compared in this study will be provided:

- **Centralized model:** This model is formulated by aggregating all available data from various tasks at a single centralized center for training purposes, with 50 epochs to optimize.
- **Local-finetuned model:** This model trains independently without any external communication from other clients or a central server. It is specifically trained on data pertaining to a single task, dedicating 50 epochs to optimize for task-specific performance without the influence of external data.

Table 5: Tasks and datasets of constructed federated dataset 1 and federated dataset 2.

Federated Dataset 1		**Federated Dataset 2**	
Task	Dataset	Task	Dataset
Paraphrase	glue_qqp	Paraphrase	paws_wiki
Entailment	snli	Commonsense	hellaswag
Structure to text	web_nlg_en	Entailment	qnli
Text formatting	fix_punct	Text formatting	word_segment
Linguistic acceptability	cola	Summarization	gigaword
Word disambiguation	wic	Reading comprehension	bool_q
Coreference	definite_pronoun_resolution	Sentiment	sentiment140
Question classification	trec	Open-domain QA	acr_easy

Table 6: Examples of data in our constructed federated datasets.

Data Examples	
Input:	The father convinced his son that it is possible for him to one day become a knight, but he may never achieve such status coming from a peasant family. Who is "he"? OPTIONS: - The father - his son
Output:	His son
Input:	Police are seeking a former village chief in north china for allegedly killing his political rivals in an attack apparently motivated by local power plays, state press reported monday . Can you generate a short summary of the above paragraph?
Output:	Former chinese village head wanted for political murders

- **FedIT model [42]:** The FedIT model is the final aggregated global model derived from diverse local client datasets after training. It embodies the essence of collaborative learning inherent to federated learning, assimilating knowledge from a multitude of client-specific data sources.
- **FedLoRA model [38]:** Here, we adapt the training paradigm in paper [38] to NLP tasks by focusing on training the lightweight LoRA for aggregation while keeping the majority of the LLM parameters frozen. Subsequently, a personalized adaptation process is employed, where the globally aggregated LoRA undergoes further local training on each local client's dataset to tailor the learning outcomes to individual client needs.
- **FedDPA-F:** FedDPA-F is the combination of the global adapter and the local fine-tuning adapter. During the inference, the scale factor is set to $\lambda = 1$ in the instance-wise dynamic weighting mechanism.
- **FedDPA-T:** FedDPA-T is the combination of the global adapter and the local training adapter. Since the global adapter contributes to the training of the local adapter for personalization, it is essential to restrict the similarity score during the inference. This adjustment is necessary to ensure that the integration of the global and local adapters achieves optimal personalization outcomes. Thus, the scale factor λ is set equal to the updating weight α used in the local adapter training.

All models are implemented using LoRA to enhance learning efficiency, with the rank of LoRA set as $r = 8$ and only applied to $\boldsymbol{W}_q$ and $\boldsymbol{W}_v$. For FL methods, each client conducts 10 local epochs

Table 7: Prompt Template.

	Template
Prompt Input	**Instruction**: {instruction} **Response**:

with a batch size of 32. We implement all the methods using PyTorch and conduct all experiments on NVIDIA Quadro RTX 6000 GPU.

A.3 Algorithm

Algorithm 1: FedDPA-F

```
Require: Number of clients $M$; number of communication rounds $K$; local step size $\eta$; freeze
         foundation model $\boldsymbol{\theta}$; initial global adapter $\Delta\boldsymbol{\theta}_g$ and local adapters $\Delta\boldsymbol{\theta}_l^1, \cdots, \Delta\boldsymbol{\theta}_l^M$;
         local training datasets $\boldsymbol{D}_{train}^1, \cdots, \boldsymbol{D}_{train}^M$.
// Learn Global Adapter
for $k \leftarrow 1$ to $K$ do
    Sample clients $\mathcal{S} \subseteq \{1, \cdots, M\}$;
    Communicate $\Delta\boldsymbol{\theta}_g$ to all clients $m \in \mathcal{S}$;
    for each client $m \in \mathcal{S}$ in parallel do
        Initialize $\Delta\boldsymbol{\theta}_g^m \leftarrow \Delta\boldsymbol{\theta}_g$;
        $\Delta\boldsymbol{\theta}_g^m \leftarrow$ Client local training$(\{\boldsymbol{\theta}, \Delta\boldsymbol{\theta}_g\}, \boldsymbol{D}_{train}^m, \eta)$;        [Equation 2]
        Communicate $\Delta\boldsymbol{\theta}_g^m$ to the server;
    Construct $\Delta\boldsymbol{\theta}_g = \sum_{m \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \Delta\boldsymbol{\theta}_g^m$;        [Equation 2]
// Learn Local Adapters
for each client $m \in \{1, \cdots, M\}$ in parallel do
    Initialize $\Delta\boldsymbol{\theta}_l^m \leftarrow \Delta\boldsymbol{\theta}_g$;
    $\Delta\boldsymbol{\theta}_l^m \leftarrow$ Client local training$(\{\boldsymbol{\theta}, \Delta\boldsymbol{\theta}_l^m\}, \boldsymbol{D}_{train}^m, \eta)$;        [Equation 3]
return $\Delta\boldsymbol{\theta}_g, \{\Delta\boldsymbol{\theta}_l^1, \cdots, \Delta\boldsymbol{\theta}_l^M\}$.
```

Algorithm 2: FedDPA-T

```
Require: Number of clients $M$; number of communication rounds $K$; local step size $\eta$; freeze
         foundation model $\boldsymbol{\theta}$; initial global adapter $\Delta\boldsymbol{\theta}_g$ and local adapters $\Delta\boldsymbol{\theta}_l^1, \cdots, \Delta\boldsymbol{\theta}_l^M$;
         local training datasets $\boldsymbol{D}_{train}^1, \cdots, \boldsymbol{D}_{train}^M$.
for $k \leftarrow 1$ to $K$ do
    Sample clients $\mathcal{S} \subseteq \{1, \cdots, M\}$;
    Communicate $\Delta\boldsymbol{\theta}_g$ to all clients $m \in \mathcal{S}$;
    for each client $m \in \mathcal{S}$ in parallel do
        Initialize $\Delta\boldsymbol{\theta}_g^m \leftarrow \Delta\boldsymbol{\theta}_g$ and $\Delta\boldsymbol{\theta}_l^m \leftarrow \Delta\boldsymbol{\theta}_l^{m,k-1}$;
        // Learn Global Adapter
        $\Delta\boldsymbol{\theta}_g^m \leftarrow$ Client local training$(\{\boldsymbol{\theta}, \Delta\boldsymbol{\theta}_g\}, \boldsymbol{D}_{train}^m, \eta)$;        [Equation 2]
        // Learn Local Adapter
        $\Delta\boldsymbol{\theta}_l^{m,k} \leftarrow$ Client local training$(\{\boldsymbol{\theta}, \Delta\boldsymbol{\theta}_g, \Delta\boldsymbol{\theta}_l^m\}, \boldsymbol{D}_{train}^m, \eta)$;        [Equation 3]
        Communicate $\Delta\boldsymbol{\theta}_g^m$ to the server;
        Maintain $\Delta\boldsymbol{\theta}_l^{m,k}$ locally;
    Construct $\Delta\boldsymbol{\theta}_g = \sum_{m \in \mathcal{S}} \frac{1}{|\mathcal{S}|} \Delta\boldsymbol{\theta}_g^m$;        [Equation 2]
return $\Delta\boldsymbol{\theta}_g, \{\Delta\boldsymbol{\theta}_l^1, \cdots, \Delta\boldsymbol{\theta}_l^M\}$.
```

B Additional Experiments

B.1 Adaptability Analysis

To enhance applicability across diverse non-IID environments, our method is meticulously designed with a high degree of flexibility for its adaption across various global learning frameworks, backbones and PEFT methods for different scenarios. This adaptability is simply achieved through the straightforward substitution of the FedAvg, LLM and LoRA with alternative aggregation methods, transformer-based foundation models and adapter-based PEFT methods during the training. In our experiment, we employ FedAvg, LLM and LoRA as representative examples, demonstrating our methods' superior performance compared to other baselines as indicated in Table 1 and Table 2. To further validate the effectiveness and versatility of our approach within different federated learning contexts, we adapt our methods to include the FedProx[20] framework and also implement other baselines (FedIT and FedLoRA) within FedProx to maintain a fair comparison.

Results presented in Table 8 indicate that our methods, both training and fine-tuning methods, outperform competing approaches. Specifically, FedDPA-T excels in personalization, while FedDPA-F leads in test-time personalization, maintaining consistent performance with FedAvg. These findings underscore the robustness and consistent efficacy of our methods across various global learning paradigms for different non-IID scenarios.

Table 8: Personalization and test-time personalization results of different models with FedProx framework on federated dataset 1. FedDPA-F represents the model with the local fine-tuning adapter and FedDPA-T represents the model with the local training adapter. Linguistic represents the linguistic acceptability task, Word Dis represents the word disambiguation task, and Question CLS represents question classification task.

Methods	**Federated Dataset 1**								
	Para -phrase	Entail -ment	Structure to Text	Text For -matting	Linguistic Acc	Word Dis	Core -ference	Question CLS	Average
Personalization									
FedIT	71.00	83.00	70.38	96.32	69.50	63.50	73.55	91.50	77.34
FedLoRA	79.00	85.00	70.63	96.60	69.00	63.00	**78.85**	88.50	78.82
FedDPA-F	77.50	85.00	**71.92**	96.78	73.00	**63.50**	77.87	89.50	79.38
FedDPA-T	**79.50**	**85.50**	71.62	**96.89**	**76.00**	60.00	77.28	**93.50**	**80.04**
Test-Time Personalization									
FedLoRA	76.76	75.82	74.11	74.36	75.01	72.27	77.16	74.87	75.05
FedDPA-F	**77.25**	**76.01**	**76.72**	**76.95**	**77.21**	**75.48**	**77.31**	**76.14**	**76.63**
FedDPA-T	75.64	74.62	75.58	75.01	74.90	74.21	74.80	75.35	75.01

B.2 Instance-Wise Dynamic Weighting Mechanism Analysis

In this section, we further examine the impact of the instance-wise dynamic weighting mechanism, including the similarity metric, the selected local instance number and the type of instance representation.

Impact of Similarity Metric. In Section 4.3, we employ the similarity metric to calculate the average similarity scores of each input instance, which serves as the weight α_t to dynamically balance the global and local adapters. For this purpose, cosine similarity is selected in our experiment due to its better robustness and normalization with high-dimensional vectors than other metrics, and its superiority has been demonstrated in many NLP/CV works[25, 37, 36]. Additionally, we conducted an ablation study comparing other metrics like L2-norm and Pearson correlation, and the results in Table 9 demonstrate that cosine similarity outperforms other similarity metrics.

Impact of Instance Number S. In Section 4.3, the selection of S, representing the number of local instances for similarity calculation, is pivotal. To comprehensively evaluate the effect of varying the number of these instances, we conduct a series of experiments employing distinct local instance numbers, specifically $S \in \{1, 3, 5, 7, 9\}$. The accuracy results, as depicted in Figure 5, illustrate the dependency of model performance on different instance numbers S. As demonstrated in Figure 5 (a), in the context of personalization, it is observed that our models attain a plateau in accuracy when the instance number exceeds 5. This indicates a stabilization in model performance beyond this threshold of local instances. Furthermore, Figure 5 (b) delves into the realm of test-time personalization. The findings here reveal similar results, indicating that variations in the instance number do not markedly impact the model's performance in test-time personalization.

Impact of Instance Representation. In Section 4.3, our method entails utilizing the embedding of the final token from the last hidden layer of the LLM, denoted as 'LAST', as the input instance representation for the purpose of similarity calculation. In this exploration, we delve into another instance representation strategy, which involves employing the average embedding of all tokens from the final hidden layer of the LLM, herein referred to as 'AVG'. The comparative analysis, as presented

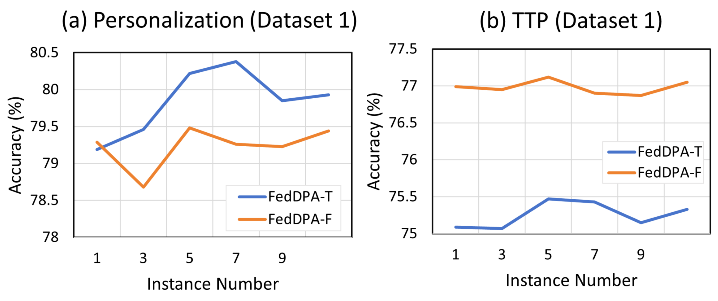


Figure 5: Average accuracy varies as different instance numbers. TTP represents test-time personalization.

in Table 10, demonstrates that employing the embedding of the last token yields superior performance relative to the strategy of averaging the embeddings of all tokens. This observed difference in performance can be attributed to the decoder structure inherent to LLMs, wherein the final token is capable of attending to all preceding tokens, thereby encapsulating comprehensive sentence-level information.

Table 9: Ablation study of similarity metric (Sim). P represents personalization, and TTP represents test-time personalization. -L2 represents using the L2-Norm as metric, Pearson represents using the Pearson correlation as metric, and Cosine represents using the cosine similarity as the metric.

Methods	Sim	**Fed Dataset 1**	
		P	TTP
FedDPA-F	-L2	77.69	77.10
	Pearson	79.09	76.78
	Cosine	**79.48**	**77.12**
FedDPA-T	-L2	77.58	**77.30**
	Pearson	79.73	75.27
	Cosine	**80.22**	75.47

Table 10: Ablation study of instance representations (Emb). P represents personalization, and TTP represents test-time personalization. LAST represents using the embedding of the final token from the final hidden layer of LLM as instance representation, and AVG represents using the average embedding of all tokens from the final hidden layer of LLM as instance representation.

Methods	Emb	**Fed Dataset 1**	
		P	TTP
FedDPA-F	AVG	79.30	76.77
	LAST	**79.48**	**77.12**
FedDPA-T	AVG	79.65	73.36
	LAST	**80.22**	**75.47**

B.3 Model Scalability Analysis

In order to examine the effectiveness of model scalability, we conduct experiments based on a larger model, LLaMA-13B. The outcomes, as presented in Table 11, elucidate that larger models exhibit superior performance over their smaller counterparts across all personalization methods evaluated. Furthermore, it is noteworthy that FedDPA-T surpasses FedDPA-F in terms of personalization and achieves comparable results in test-time personalization. This analysis underscores the inherent advantages of larger models in enhancing model performance, alongside the advance of the FedDPA-T approach in the context of personalization and adaptability to test-time conditions.

Table 11: Ablation study of model size. P represents personalization, and TTP represents test-time personalization.

Methods	Size	**Fed Dataset 1**	
		P	TTP
FedDPA-F	7B	79.48	77.12
	13B	**81.52**	**80.55**
FedDPA-T	7B	80.22	75.47
	13B	**82.76**	**80.47**

B.4 Communication and Computation Analysis

In this section, we undertake a detailed examination of both the communication and computation overhead associated with our proposed model in comparison to other baseline models. The results, as detailed in Table 12, delineate the communication and computation burdens imposed by various models. Given that these models are all based on the LoRA framework and exclusively transmit LoRA weights for aggregation (with our methods specifically transmitting only the global LoRA weights), they inherently sustain a minimal communication overhead. Regarding the computation overhead, the LoRA architecture permits the training of both local and global LoRAs in parallel, resulting in a marginal increase in computational demands for FedDPA-T. Conversely, FedDPA-F learns the local LoRA through an additional fine-tuning phase, thereby not imposing any additional computational overhead during the training phase.

Additionally, we have conducted an analysis of the inference time associated with our models. This examination involved calculating the average inference time per instance for FedLoRA, FedDPA without the instance-wise dynamic weighting mechanism, and FedDPA. As illustrated in Table 13, it is observed that our methods incur slightly higher inference time compared to FedLoRA. This marginal increase in inference time underscores the efficiency of our proposed methods, demonstrating that the enhanced performance and capabilities are achieved with a minimal impact on computational efficiency during inference.

Table 12: The communication and computation overhead of FedDPA and other baselines on Federated Dataset 1.

Methods	Comm.Overhead	Comp.Overhead
FedIT	4.2M(0.06%)	0.277 TFLOPS
FedLoRA	4.2M(0.06%)	0.277 TFLOPS
FedDPA-F	4.2M(0.06%)	0.277 TFLOPS
FedDPA-T	4.2M(0.06%)	0.281 TFLOPS

Table 13: Average inference time per instance. Auto represents the instance-wise dynamic weighting mechanism.

Methods	Time
FedLoRA	3.84s
FedDPA (w/o auto)	3.91s
FedDPA	4.13s

C Discussions

C.1 Extend to Other Scenarios of Test-Time FL

In this paper, we primarily consider an ideal scenario for our proposed test-time personalization setting, where all tasks are included for all clients. In this section, we will discuss our methods under alternative scenarios.

In practical applications, it is possible that some tasks remain unseen by all clients during training and may only appear during the testing phase for certain clients. In this scenario, test-time distribution shifts arise from these unseen tasks. According to previous works [5, 29], generic features learned through FL are robust to distribution shifts, even those originating from unseen test-time tasks. Consequently, our method can be directly adapted to this scenario. We conducted experiments to evaluate our methods against other baselines on three unseen test-time tasks. All methods and baselines are trained on our constructed Federated Dataset 1 and tested on three tasks not included in Federated Dataset 1. For personalized methods, we report the average score across all clients and the maximum score among all clients to provide a comprehensive comparison.

As shown in Table 14, FedDPA-T outperforms all other models, indicating the effectiveness of our method on unseen test-time tasks. Additionally, FedDPA-F surpasses FedLoRA, suggesting that the generic features learned through FL across diverse data distributions are robust to various distribution shifts, consistent with the findings in [29]. Despite this, other techniques targeting these unseen test-time tasks could further enhance our proposed methods, which we leave for future work to explore.

Our methods are also applicable to scenarios involving the introduction of new clients. As previously analyzed, our methods are robust to different distribution shifts. By computing the similarity between instances from new clients and existing clients through our instance-wise dynamic weighting mechanism, we can identify the most similar existing client. The model of this identified client can

Table 14: Test-time performance on unseen tasks. All models are trained on Federated Dataset 1, and these unseen test-time tasks are not included in Federated Dataset 1. AVG represents the average score across all clients for this task, while MAX represents the highest score among these clients for this task. The best performance for AVG is bolded, and the best performance for MAX is underlined.

Test-Time Task	FedIT	FedLoRA		FedDPA-F		FedDPA-T	
		AVG	MAX	AVG	MAX	AVG	MAX
Summarization	22.40	22.25	22.42	22.37	22.52	**22.46**	<u>22.65</u>
Reading Comprehension	69.50	68.88	73.00	69.44	72.00	**71.88**	<u>76.00</u>
Open Domain QA	78.32	76.46	78.96	78.01	79.61	**78.76**	<u>79.92</u>

then be used as the initial model for the new clients, providing a more effective starting point for further training.

C.2 Limitations

The proposed dual-personalizing adapter architecture is limited by 1) model scales: the proposed methods rely on the foundation model, presupposing that each client possesses sufficient memory capacity and computational resources to store and train the foundation model with PEFT methods. 2) secure issues: the framework operates under the assumption that all clients are trusted and legally entitled to access and utilize data stored on them, and the whole process does not suffer from any attacks.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims have been made clearly in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Appendix C.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to the experiment section 5 and Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Please refer to the supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to Appendix A.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: All experiments are conducted in the same setting with stable results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to Appendix A.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Although Federated Learning technology has positive societal impacts on privacy preservation, our paper aims to enhance federated learning in the application scenario with foundation models. We only use publicly available NLP datasets to evaluate the effectiveness of the proposed method. Therefore, we would like to rate our work's societal impacts as N/A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Since our paper proposes a new framework for federated learning and the models/datasets we use are publicly available, there are no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Please refer to the section 5.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Please refer to the supplementary material.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.