This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Tensor-aggregated LoRA in Federated Fine-tuning

Zhixuan Li[1*], Binqian Xu[1*], Xiangbo Shu[1†], Jiachao Zhang[2], Yazhou Yao[1], Guo-Sen Xie[1], Jinhui Tang[3]
[1]Nanjing University of Science and Technology
[2]Nanjing Institute of Technology
[3]Nanjing Forestry University
lzxlizhixuan@njust.edu.cn, xubinq11@gmail.com, shuxb@njust.edu.cn, zhangjc07@foxmail.com,
yazhou.yao@njust.edu.cn, gsxiehm@gmail.com, tangjh@njfu.edu.cn

Abstract

The combination of Large Language Models (LLMs) and Federated Learning (FL) to leverage privacy-preserving data has emerged as a promising approach to further enhance the Parameter-Efficient Fine-Tuning (PEFT) capabilities of LLMs. In real-world FL settings with resource heterogeneity, the training process of Low-Rank Adaptation (LoRA), the representative PEFT method, still faces two major challenges: aggregation noise and aggregation misalignment. In this paper, we propose a novel Tensor-aggregated LoRA (Te-LoRA) in Federated Fine-tuning based on an alternating-freeze training strategy to avoid aggregating noise without additional server-side computational costs, while mitigating aggregation suboptimality caused by parameter misalignment between heterogeneous LoRAs. Especially in addressing the aggregation suboptimality issue, we design the Pre-Aggregation Alignment strategy (PAA-strategy) and Tensor-to-Matrix strategy (T2M-strategy) for aligning heterogeneous LoRAs and aggregating them into an united tensor, which is then decomposed into matrices adapted for client download. Extensive experiments demonstrate the effectiveness and robustness of Te-LoRA in both homogeneous and heterogeneous settings.

1. Introduction

Large Language Models (LLMs) have demonstrated exceptional performance across various task scenarios, such as search engines [20], chatbots [4], healthcare [37], and more. When using pre-trained LLMs to adapt to downstream tasks, significant computational resources are required to fine-tune the model parameters. Therefore, to improve the applicability of LLMs, it is necessary to fine-tune specific datasets [9, 17, 25]. However, collecting such datasets comes with associated costs and privacy concerns [5, 18, 38]. Consequently, researchers have turned to Federated Learning (FL) as a technical approach to fine-tune LLMs using distributed client data without compromising data privacy [2, 30, 33]. In this context, researchers have explored the application of Parameter-Efficient Fine-Tuning (PEFT) [11, 19, 23] methods for federated fine-tuning of LLMs, saving substantial computational resources. Among these, Low-Rank Adaptation (LoRA) [14] has gained attention due to its significant reduction in the number of communication parameters.

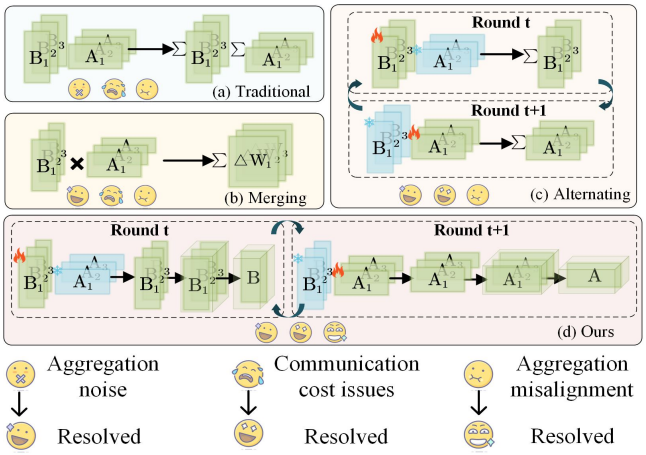


Figure 1. Comparison of different federated fine-tuning LoRAs. Compared to methods (a), (b), and (c), our Te-LoRA performs better in addressing aggregation noise, communication cost, and aggregation misalignment issues.

However, simply applying LoRA in real FL environment presents several challenges, one of the key issues being *aggregation noise*. In detail, traditional federated fine-tuning of LLMs [36, 46] averages modules A and B of local LoRAs separately, leading to discrepancies between the aggregated and ideal global LoRAs, thereby introducing noise into the

*Equal contributions
†Corresponding author

update process. To address this issue, as shown in Fig. 1, the two main solutions are as follows: 1) Merging modules A and B [3, 43], where A and B are first merged by matrix multiplication and then aggregated. 2) Alternating the training of modules A and B [22, 34], where either A or B is kept frozen for consistency at each round, with only the learning B or A being individually aggregated. Compared to the "Merging" strategy, the "Alternating" strategy reduces communication costs by nearly half, as it transmits only A or B per round.

The existing "Alternating" method [22] further reduces communication costs by selecting weights from a global LoRA with a unified initial rank. However, this homogenized assumption contradicts the core feature of resource heterogeneity in FL [16, 35, 40], where real-world scenarios involve clients with local LoRAs of different initial ranks. When LoRA modules A or B with heterogeneous ranks are uploaded and aggregated, an explicit rank dimension misalignment arises. Even when adopting the "merging" strategy for weight aggregation through matrix multiplication, semantically driven feature alignment issues may still persist. This multi-level parameter/feature mismatch caused by rank heterogeneity and data heterogeneity can be referred to as the *aggregation misalignment* problem.

To simultaneously mitigate the issues of aggregation noise and aggregation misalignment in heterogeneous settings, as shown in Fig. 1, we propose a novel Tensor-aggregated LoRA called Te-LoRA. Te-LoRA consists of two main strategies: First, due to the impact of heterogeneity, the ranks of the matrices uploaded by each client to the server are often inconsistent. We present a Pre-Aggregation Alignment strategy (PAA-strategy) to align rank while minimizing information loss or conflict. Second, to preserve the potential relationships and dependencies between different clients, we design the key Tensor-to-Matrix strategy (T2M-strategy) for decomposing the tensor stacked from aligned matrices. T2M-strategy can adapt to the aggregation of heterogeneous LoRAs while effectively preserving the key components from each client. The main contributions of this work are summarized as follows:

- **Alternating-freeze training for Tensor version of LoRA.** We propose a novel Tensor-aggregated LoRA (Te-LoRA) in Federated Fine-tuning to efficiently aggregate the main components of each client based on an alternating-freeze training approach, avoiding aggregation noise while alleviating aggregation suboptimality among heterogeneous LoRAs.
- **Alignment of heterogeneous LoRAs for server aggregation.** We design a Pre-Aggregation Alignment strategy (PAA-strategy) to align heterogeneous LoRAs, while preserving the unique features of each client and minimizing general information loss.
- **Decomposition of aggregated LoRAs for client download.** We devise a Tensor-to-Matrix strategy (T2M-strategy) that decomposes the tensor into matrices adapted for client download, after aggregating LoRAs into an united tensor covering relationships and dependencies among LoRAs.

2. Related Work

2.1. Parameter-Efficient Fine-Tuning

To effectively fine-tune LLMs while reducing computational resources and storage requirements, Parameter-Efficient Fine-Tuning (PEFT) techniques have emerged. Among these, Prefix-Tuning [27] guides the pre-trained language model's generation process by optimizing a continuous sequence of task-specific vectors (called prefixes) while keeping the model parameters frozen. Soft prompts [23] enable frozen language models to perform specific tasks by propagating signals of labeled examples. Both methods are effective for task-specific adaptation. LoRA [14] reduces memory overhead by representing weight updates with low-rank matrices. AdaLoRA [47] optimizes incremental updates of pre-trained language models by adaptively allocating parameter budgets and using singular value decomposition, effectively reducing the parameter budget and avoiding intensive computation. LongLoRA [6] extends the context length of LLMs at low computational cost through sparse local attention and parameter-efficient fine-tuning. Split-LoRA [28], based on Split Federated Learning (SFL), improves training efficiency by partitioning the model and combining the parallel training of federated learning with the model splitting advantage of split learning. PEFT-based methods for LLMs, especially LoRA and its variants, have advanced fine-tuning techniques, showing excellent performance due to their simplicity and ability to reduce parameters. Thus, we fine-tune LLMs with LoRA in this paper.

2.2. Federated Fine-Tuning Homogeneous LoRA

In federated learning, several works have applied LoRA to achieve reduced communication and improved efficiency. For example, Zhang et al. [46] fine-tuned LLMs by aggregating heterogeneous instruction data from multiple clients in a federated learning framework, improving the model's generalization ability on new tasks while ensuring user privacy and data security. Wu et al. [44] introduced the FedBiOT algorithm, which compresses LLMs and splits them into two components, using a dual-layer optimization approach to achieve efficient fine-tuning and reduce computational and communication costs. FLoCoRA [12] also reduces communication overhead by freezing the originally randomized neural network parameters during training, and only training and transmitting the added LoRA adapters. However, it does not fully address the stability issues brought about by extremely low-bit quantization. Sun

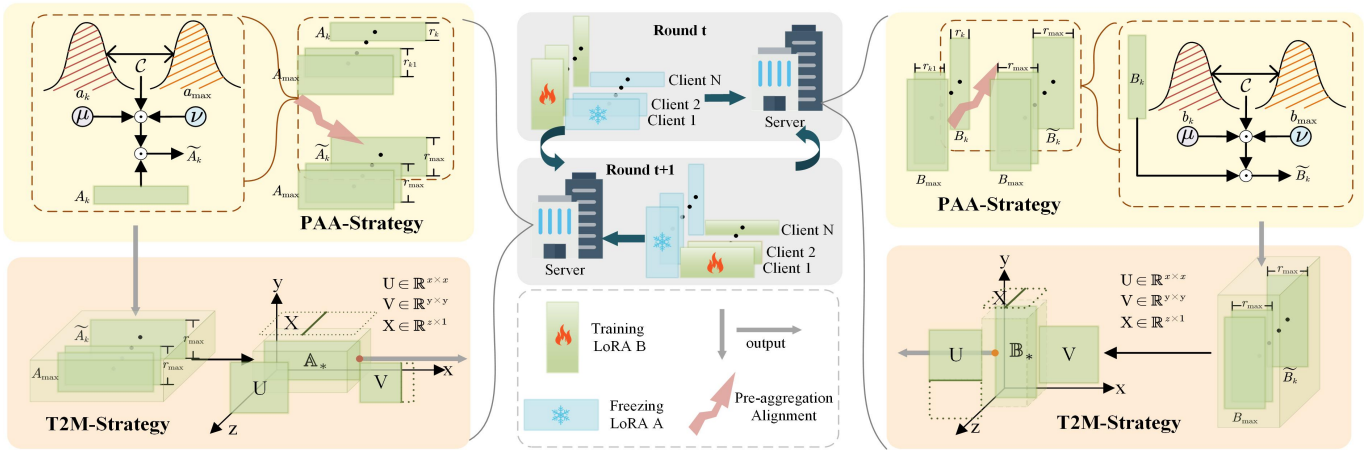


Figure 2. An overview of the proposed Te-LoRA. It is based on a framework for alternating training of the LoRA modules B and A. When the clients' LoRA have different ranks, the B or A modules are obtained after client training and subsequently uploaded to the server for aggregation. On the server, the heterogeneous B or A modules are first aligned using the PAA-strategy. In the PAA-strategy, μ and ν are all 1-column vectors of size having $r_{\max}$ and r_k, respectively. Then, the aligned B or A are aggregated using the T2M-strategy. The detailed operations will be discussed in Sec. 3.

et al. [39] pointed out that separately integrating the two LoRA matrices does not fully approximate the original, and proposed FFA-LoRA, which fixes the randomly initialized matrices while fine-tuning only the zero-initialized ones. Compared to FFA-LoRA, the proposed Te-LoRA, based on an alternating freezing framework, not only preserves the training of all LoRA modules but also effectively addresses the issue of aggregation noise.

2.3. Federated Fine-tuning Heterogeneous LoRA

Due to non-IID data heterogeneity [26] sampled from the Dirichlet distribution and heterogeneous hardware resources, clients need to adjust the LoRA rank accordingly. Existing work has proposed methods to address this issue. For instance, SLoRA [2] addresses LoRA's limitations in high data heterogeneity scenarios with data-driven initialization, achieving performance similar to full fine-tuning while reducing training time and communication costs. FLoRA [43] uses a stacked aggregation method to eliminate noise in low-rank adapter (LoRA) aggregation, ensuring noise-free global updates. It supports heterogeneous LoRA configurations across clients, allowing dynamic adjustments based on data complexity and resources. However, this approach leads to linear growth in communication complexity, higher computational and storage demands, and reduced flexibility for supporting multiple task-specific adapters. Cho et al. [8] deploy LoRA modules with different ranks across clients, using zero-padding and truncation for aggregation, combining the benefits of high- and low-rank LoRA. However, the padded zeros dilute useful information from high-quality clients, reducing optimization efficiency and potentially limiting performance. FlexLoRA [3] dynamically adjusts the LoRA rank for clients and uses Singular Value Decomposition (SVD) to reallocate global weights, enhancing resource utilization and model generalization. However, SVD truncation adds computational and storage overhead, and the resulting errors may limit performance.

LoRA-A2 [22] alternates freezing the A and B components of the module and dynamically selects key parameters based on the importance of each client's data, effectively reducing parameter inconsistencies during the aggregation of low-rank adapters and significantly cutting communication costs. However, the method assumes that the global LoRA has a uniform initial rank, whereas in practice, each client's LoRA begins with a different rank. Compared to the related LoRA-A2, we train the initial heterogeneous-rank LoRA directly on the clients. While LoRA-A2 focuses on adaptively selecting client ranks, we emphasize the effective alignment of heterogeneous ranks on the server side. During alignment, unlike FlexLoRA and FLoRA, which use merging strategies, we designed the PAA-strategy and T2M-strategy for adaptive alignment, preserving the effective information within clients and dependencies across clients.

3. Method

3.1. Preliminaries

Low-Rank Adaptation. The application of Low-Rank Adaptation (LoRA) [15] facilitates a reduction in the number of trainable parameters during the fine-tuning of a pre-trained model. Specifically, the weight matrix $W \in \mathbb{R}^{d\times l}$ in the fine-tuned model can be represented as the sum of the original weight matrix $W_0 \in \mathbb{R}^{d\times l}$ and an updated

weight matrix $\triangle W \in \mathbb{R}^{d\times l}$. $\triangle W$ is implemented by low-rank decomposition into smaller matrices $B \in \mathbb{R}^{d\times r}$and $A \in \mathbb{R}^{r\times l}$:

$$W = W_0 + \triangle W = W_0 + BA, \tag{1}$$

where rank r is much smaller than both dimensions d and l, thereby significantly reducing the number of trainable parameters for the weights.

Aggregation Noise. Currently, FedAvg [30], a widely used FL algorithm, updates the global model by performing a weighted average of n local clients in each communication round. When LoRA is applied to FL, training and transmitting only the updated low-rank matrices B and A significantly reduces both computational and communication costs, compared to directly updating and transmitting $\triangle W$. Combining the aforementioned FedAvg and LoRA, FedIT [46] is proposed for federated fine-tuning, where the global LoRA matrices A and B are updated by weighted averaging of the local LoRA matrices A_i and B_i from n clients, as follows:

$$\triangle W = \frac{1}{n}\sum_{i=1}^{n}\triangle W_i = \frac{1}{n}\sum_{i=1}^{n}B_i \times \frac{1}{n}\sum_{i=1}^{n}A_i. \tag{2}$$

However, when each weight is averaged and aggregated separately, there is a bias between the aggregated parameters (left) and ideal (right), as shown below:

$$\frac{1}{n}\sum_{i=1}^{n}B_i \times \frac{1}{n}\sum_{i=1}^{n}A_i \neq \frac{1}{n}\sum_{i=1}^{n}(B_iA_i). \tag{3}$$

One solution is to first merge A_i and B_i from each client by matrix multiplication and then perform average aggregation, *i.e.*, $\triangle W = \frac{1}{n}\sum_{i=1}^{n}(B_iA_i)$. Subsequently, the aggregated $\triangle W \in \mathbb{R}^{d\times l}$ must be decomposed into the new $A_i \in \mathbb{R}^{r\times l}$ and $B_i \in \mathbb{R}^{d\times r}$ for each client. This method introduces complex matrix decomposition, making post-decomposition training prone to instability. The details of another simple and effective solution are as follows.

Alternating Freezing Technique. In [22], a simple alternating freezing technique is introduced to mitigate the issue of aggregation noise when applying LoRA in FL. Specifically, in round t, the aggregated $\bar{A}^{(t)}$ remains frozen, and each client only trains $B_i^{(t)}$, as follows:

$$\frac{1}{n}\sum_{i=1}^{n}B_i^{(t)} \times \frac{1}{n}\sum_{i=1}^{n}\bar{A}^{(t)} = \frac{1}{n}\sum_{i=1}^{n}\left(B_i^{(t)}\bar{A}^{(t)}\right). \tag{4}$$

Then, in round $t+1$, the aggregated $\bar{B}^{(t+1)}$ is frozen, and only $A_i^{(t+1)}$ is trainable on each client, as shown below:

$$\frac{1}{n}\sum_{i=1}^{n}\bar{B}^{(t+1)} \times \frac{1}{n}\sum_{i=1}^{n}A_i^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n}\left(\bar{B}^{(t+1)}A_i^{(t+1)}\right). \tag{5}$$

Finally, A and B complete iterative training and aggregation through the alternating freezing technique. However, when faced with data heterogeneity and rank heterogeneity at the client, this approach does not achieve good alignment of modules A or B at the server.

3.2. Proposed Method

In real FL environments with resource heterogeneity, the training of Low-Rank Adaptation (LoRA) faces two major challenges: aggregation noise and aggregation misalignment. To address these issues, we propose a novel federated fine-tuning method with Tensor-aggregated LoRA (Te-LoRA) based on an alternating freezing training strategy, as shown in Fig. 2. Te-LoRA consists of three strategies: the alternating freezing training strategy, the Pre-Aggregation Alignment Strategy (PAA-strategy), and the Tensor Decomposition Aggregation Strategy (T2M-strategy). Based on the existing alternating freezing training strategy (Eqs. (4) and (5)), Te-LoRA first effectively avoids aggregation noise (Eq. (3)). Subsequently, the PAA-strategy aims to achieve parameter alignment between different rank modules A and B with minimal information loss. Furthermore, the T2M-strategy adaptively aligns and aggregates the main components among clients to achieve more fine-grained feature alignment. The PAA and T2M strategies complement each other, effectively mitigating the issue of aggregation misalignment.

Pre-Aggregation Alignment strategy (PAA-strategy). The PAA-strategy is presented to achieve the optimal transformation from heterogeneous LoRAs to homogeneous LoRAs with minimal information loss for parameter alignment on the server side. In the implementation of the alternating freezing FL training strategy, the modules transmitted between the client and server are either B or A. Assuming that the module B is uploaded in the current round, module A remains frozen and stays on the client side. After the upload, the server receives a set of K modules B, *i.e.*, $\{B_k \in \mathbb{R}^{d\times r_k}\}_{k=1}^{K}$. The module with the highest rank is selected as the target module $B_{\max} = B_{\arg\max_{k=1,\ldots,K} r_k}$, and the remaining modules are aligned towards the target module. Specifically, the distance matrix $\mathcal{C}$ between B_k and $B_{\max}$ is calculated as follows:

$$\mathcal{C}(m,n) = \sum_{i=1}^{d}\left(b_k^{im} - b_{\max}^{in}\right), \tag{6}$$

where $\mathcal{C} \in \mathbb{R}^{r_k\times r_{\max}}$, b_k^{im} and $b_{\max}^{in}$ are the elements of the i^{th} row, m^{th} column of matrix B_k, and n^{th} column of matrix $B_{\max}$, respectively. To smooth the matching relationship between the source and target matrices, the matrix $\mathcal{C}$ is regularized: $\mathcal{K} = \exp\left(-\frac{\mathcal{C}}{\eta}\right)$, where η is the regularization parameter. Next, based on the matrix $\mathcal{K}$, we first assume that the source distribution p and the target distribution q

follow a uniform distribution, *i.e.*, $p = \left[\frac{1}{r_k}, \frac{1}{r_k}, \dots \frac{1}{r_k}\right]^\top$, $q = \left[\frac{1}{r_{\max}}, \frac{1}{r_{\max}}, \dots \frac{1}{r_{\max}}\right]^\top$. Then, using the Sinkhorn-Knopp algorithm [31], we iteratively update the adjustment factors μ and ν to make the probability matrix $\mathcal{G}$ converge to a solution that satisfies the marginal constraints of p and q, as follows:

$$\mu = \frac{p}{\mathcal{K} \cdot \nu}, \quad \nu = \frac{q}{\mathcal{K}^\top \cdot \mu}, \tag{7}$$

where the adjustment factors μ and ν are both 1-column vectors of size $r_{\max}$ and r_k, respectively. By iteratively updating until convergence, the probability matrix $\mathcal{G}$, representing the optimal alignment between the source and target features, is shown below:

$$\mathcal{G} = \mu \cdot \mathcal{K} \cdot \nu^\top. \tag{8}$$

Subsequently, the source matrix B_k is transformed using the matrix $\mathcal{G}$ to achieve alignment of different ranks with minimal information loss, as follows:

$$\widetilde{B_k} = B_k \cdot \mathcal{G}, \tag{9}$$

where $\widetilde{B_k} \in \mathbb{R}^{d \times r_{\max}}$ is the aligned matrix. The remaining module B and the module A from another round both use the same pre-aggregation process for parameter alignment.

Discussions. Although previous research in FL, such as FedOTP [24] and FedAli [10], has explored Optimal Transport (OT) for fine-grained feature alignment, FedOTP [24] uses OT to align local visual features with global and local text features, smoothing the global consensus and local personalization. To balance personalization and generalization, FedAli [10] quantifies the cost of embedding inputs into prototypes using OT, effectively aligning embeddings and reducing client inconsistencies. Compared to these studies, our PAA strategy differs in three ways: 1) The others focus on data heterogeneity in personalized FL, while our work addresses resource heterogeneity in real FL; 2) Previous work aligns personalized with global knowledge via OT, while our work aligns heterogeneous weights into homogeneous ones using OT to address rank mismatches from resource heterogeneity; 3) Previous studies focus on aligning prompts and prototypes, while our work targets alignment methods for heterogeneous LoRAs. To the best of our knowledge, this is the first research to apply OT for heterogeneous LoRA alignment in resource-heterogeneous FL.

Tensor-to-Matrix strategy (T2M-strategy). To maintain the relationships and dependencies between different clients, the Tensor-to-Matrix strategy (T2M-strategy) is designed to adaptively aggregate the LoRAs aligned by the PAA-strategy, rather than simply averaging them. Specifically, given the rank-aligned set $\{\widetilde{B_k}\}_{k=1}^K$ of modules B, $\widehat{\mathcal{B}}$ can be stacked from $\{\widetilde{B_k}\}_{k=1}^K$ to form a new tensor representing the intermediate state before aggregation, *i.e.*, $\widehat{\mathcal{B}} = \bigcup_{k=1}^K \widetilde{B_k}$, where $\widetilde{B_k} \in \mathbb{R}^{d \times r_{\max}}$ and $\widehat{\mathcal{B}} \in \mathbb{R}^{d \times r_{\max} \times K}$. Here, the aggregation process can be viewed as a tensor decomposition process. Inspired by [21], three versions of the T2M-strategy (T2M_1, T2M_2, and T2M_3) is explored to decompose $\widehat{\mathcal{B}}$ into the result $\mathbb{B}_*$. The generalized form of the decomposition is as follows:

$$\widehat{\mathcal{B}} = \mathbb{B}_* \times_1 U \times_2 V \times_3 X, \tag{10}$$

where $\mathbb{B}_*$ is a core tensor, U, V, and X serve as the factor matrices, and the symbol $\times_i$ $(i = 1, 2, 3)$ denotes mode-i tensor-matrix multiplication. T2M_1 is as follows:

$$\widehat{\mathcal{B}} = \mathbb{B}_*^{(1)} \times_1 U^{(1)} \times_2 V^{(1)} = \left[\!\left[\mathbb{B}_*^{(1)}; U^{(1)}, V^{(1)}, \mathbb{I}\right]\!\right], \tag{11}$$

where $\mathbb{B}_*^{(1)} \in \mathbb{R}^{1 \times 1 \times K}$, $U^{(1)} \in \mathbb{R}^{d \times 1}$, $V^{(1)} \in \mathbb{R}^{r_{max} \times 1}$, and $X = \mathbb{I}$ is the $K \times K$ identity matrix. $[\![\cdot]\!]$ denotes the shorthand. $\mathbb{B}_*^{(1)}$ represents the weight coefficient of different clients, reflecting the importance scores. And the aggregated module is $\bar{B}$, where $\bar{B}_{i,j} = \sum_{k=1}^K \widehat{\mathcal{B}}_{i,j,k} \cdot \mathbb{B}_{*\,1,1,k}^{(1)}$. The formulate of T2M_2 is as follows:

$$\widehat{\mathcal{B}} = \mathbb{B}_*^{(2)} \times_3 X^{(2)} = \left[\!\left[\mathbb{B}_*^{(2)}; \mathbb{I}, \mathbb{I}, X^{(2)}\right]\!\right], \tag{12}$$

Algorithm 1: Te-LoRA

Initialization: each client k initializes the LoRA $\triangle W_k = B_k A_k$ with $B_k \in \mathbb{R}^{d \times r_k}$ and $A_k \in \mathbb{R}^{r_k \times d}$.

1. for t = 1, 2, . . . , T do:
2. Select K participants for round t
3. if t%2 = 1 then:
4. for k = 1, 2, · · · , K in parallel do:
5. $B_k^{(t+1)} = \text{LocalTraining}\left(B^{(t)}, k\right)$
6. $\widetilde{B}_k^{(t+1)} \leftarrow \text{PAA}\left(B_k^{(t+1)}\right)$
7. $\mathbb{B}_*^{(t+1)} \leftarrow \text{T2M}\left(\widetilde{B}_1^{(t+1)}, \widetilde{B}_2^{(t+1)}, ..., \widetilde{B}_{\text{K}}^{(t+1)}\right)$
8. $\overline{A}^{(t+1)} = A^{(t)}$
9. end for
10. else:
11. for k = 1, 2, · · · , K in parallel do:
12. $A_k^{(t+1)} = \text{LocalTraining}\left(A^{(t)}, k\right)$
13. $\widetilde{A}_k^{(t+1)} \leftarrow \text{PAA}\left(A_k^{(t+1)}\right)$
14. $\mathbb{A}_*^{(t+1)} \leftarrow \text{T2M}\left(\widetilde{A}_1^{(t+1)}, \widetilde{A}_2^{(t+1)}, ..., \widetilde{A}_{\text{K}}^{(t+1)}\right)$
15. $\overline{B}^{(t+1)} = B^{(t)}$
16. end for
17. end if
18. end for

where $\mathbb{B}_*^{(2)} \in \mathbb{R}^{d \times r_{max} \times 1}$, $X^{(2)} \in \mathbb{R}^{K \times 1}$, $U = \mathbb{I}$ and $V = \mathbb{I}$ are the $d \times d$ and $r_{max} \times r_{max}$ identity matrices. $\mathbb{B}_*^{(2)}$ is the aggregation result capturing the common and representative information, *i.e.*, $\bar{B} = \mathbb{B}_*^{(2)}$. T2M_3 is a combined version of T2M_1 and T2M_2, as shown below:

$$\mathbb{B}_*^{(3)} = \text{T2M}_2(\widehat{\mathcal{B}} \odot \text{T2M}_1(\widehat{\mathcal{B}})), \tag{13}$$

where $\mathbb{B}_*^{(1)} = \text{T2M}_1(\widehat{\mathcal{B}})$, $\odot$ denotes hadamard product used to obtain the new weighted tensor $\widehat{\mathcal{B}}^{(1)} \in \mathbb{R}^{d \times r_{\max} \times K}$, *i.e.*, $\widehat{\mathcal{B}}_{i,j,k}^{(1)} = \widehat{\mathcal{B}}_{i,j,k} \cdot \mathbb{B}_{*1,1,k}^{(1)}$. The final outcome, $\mathbb{B}_*^{(3)} \in \mathbb{R}^{d \times r_{\max} \times 1}$, is the aggregated module, *i.e.*, $\bar{B} = \mathbb{B}_*^{(3)}$. After completing aggregation on the server side, the module $\bar{B}$ is distributed to each client for the next round of training. The T2M-strategy is also applicable to the decomposition of module A when module B is frozen for aggregation. Algorithm 1 provides the pseudo-code for Te-LoRA.

4. Experiments

4.1. Experiment Setup

Comparison methods. We compare the proposed Te-LoRA with three state-of-the-art approaches that support federated fine-tuning of heterogeneous LoRA. "Zero-padding" [8] scales the lower ranks of heterogeneous LoRAs from clients by padding them with zeros up to the maximum rank on the server side. "FLoRA" [43] stacks all heterogeneous LoRAs together and then performs matrix multiplication for aggregation on the server side. "FlexLoRA" [3] aggregates heterogeneous LoRAs after matrix multiplication, then decomposes them into modules A and B via SVD. "Local" refers to the average test results from all clients independently training on their local data, serving as an important baseline to validate the effectiveness of FL. To ensure a fair comparison, we reproduced the comparison methods based on the existing codebase [45], with the same LLM, data splits, and training configurations, and conducted all the experiments.

Datasets and evaluation. We used three datasets to train Te-LoRA: Alpaca-GPT4 [32], an English instruction-following dataset consisting of 52k samples generated by GPT-4 [1] based on Alpaca's self-learning [42]; Databricks-dolly-15k (Dolly) [46], an open-source dataset created by Databricks' staff, which includes tasks such as brainstorming, classification, closed quizzes, open quizzes, and summarization; and Wizard dataset [29] is a quiz dataset including 70k instruction-output pairs. For federated training, datasets were partitioned using a Dirichlet distribution ($\alpha = 0.5$). Furthermore, we separately consider closed [7] and open evaluation benchmarks, using MMLU [13] for closed evaluation and MT-Bench [48] for open evaluation. MMLU is a widely used Q&A dataset for LLM fine-tuning, including 14,024 questions across 57 different topics to assess the logical reasoning ability of LLMs. MT-Bench is a set of challenging multi-round open-ended questions for assessing chat assistants [48].

Method	MMLU			MT-Bench
Homo	Wizard	Dolly	Alpaca-GPT4	Wizard
Local	20.87	25.13	38.66	2.92
Zero-padding [8]	22.03	26.59	44.88	3.17
FLoRA [43]	22.91	27.73	45.15	3.22
FlexLoRA [3]	22.97	28.21	45.34	3.26
Te-LoRA (Ours)	**23.35**	**28.44**	**45.86**	**3.31**

Table 1. Comparison of Te-LoRA with other methods on MMLU and MT-bench. 'Homo' represents settings with homogeneous LoRA ranks.

Training details. For the model configurations, we use the pre-trained Llama2-7B [41] as the base model and conduct 200 rounds of FL communication. The initial learning rate is set to 5e-5 and decays to 1e-6 after 200 rounds using a cosine scheduler. In each round, two clients are randomly selected, each training for 10 epochs with a batch size of 16. Referring to FLoRA's setup [43] in Homogeneous LoRA, the rank of LoRA is 16. Referring to FLoRA's simulation of a real-world scenario where clients have heterogeneous computational resources, in Heterogeneous LoRA, we assign [64, 32, 16, 16, 8, 8, 4, 4, 4, 4] as different local LoRA ranks applied to 10 clients. All experiments are conducted on a single NVIDIA 3090 GPU.

4.2. Comparison Experiment

Homogeneous LoRA. In the homogeneous LoRA setup, we configure all clients to share the same LoRA rank of 16. As shown in Tab. 1, we draw the following conclusions: 1) All four federated fine-tuning LoRA methods outperform "Local" with Te-LoRA achieving a 7.2 improvement in the MMLU benchmark compared to "Local" under Alpaca-GPT4 training. 2) Among the comparison methods, Zero-padding, which is affected by aggregation noise, performs the worst. Notably, in homogeneous LoRA, Zero-padding degrades to a traditional approach where modules B and A are aggregated separately on the server. 3) FlexLoRA outperforms FLoRA. Both aggregate the matrix-multiplied weights from modules B and A, but FlexLoRA uses SVD to decompose the aggregated weights and sends new modules to clients, while FLoRA merges the aggregated weights with the client LLM and reinitialized LoRA training. The reinitialized LoRA may cause instability in client training. 4) Te-LoRA outperforms all methods across tasks by effectively resolving aggregation noise and misalignment.

Heterogeneous LoRA. As described in Sec. 4.1, to simulate the scenario of resource heterogeneity among clients in real-world settings, we apply different local LoRA ranks of [64, 32, 16, 16, 8, 8, 4, 4, 4, 4] to 10 clients. By comparing the results of Heterogeneous LoRA (Heter) in Tab. 2

Method		MMLU		MT-Bench
Heter	Wizard	Dolly	Alpaca-GPT4	Wizard
Local	20.53	25.10	38.54	2.89
Zero-padding [8]	21.86	26.48	44.47	3.15
FLoRA [43]	22.79	27.60	44.94	3.20
FlexLoRA [3]	23.49	28.18	45.48	3.28
Te-LoRA (Ours)	**23.71**	**28.37**	**46.16**	**3.33**

Table 2. Comparison of Te-LoRA with other methods on MMLU and MT-bench. 'Heter' represents settings with heterogeneous LoRA ranks.

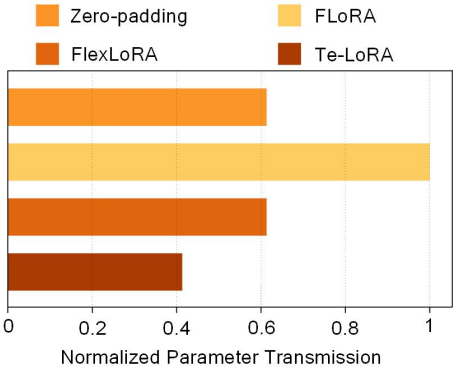


Figure 3. Comparison of the communication cost. Te-LoRA achieves lower overhead compared to the other methods.

with Homogeneous LoRA (Homo) in Tab. 1, we make several interesting observations. First, both Zero-padding and FLoRA show a performance drop in the Heter configuration compared to Homo, with Zero-padding decreasing by up to 0.41 and FLoRA by up to 0.21. Since FLoRA mitigates aggregation noise, it still outperforms Zero-padding. Second, Te-LoRA and FlexLoRA under Heter outperform Homo in most evaluation tasks, demonstrating their ability to effectively handle resource heterogeneity and optimize resource utilization. Because Te-LoRA more effectively addresses the aggregation misalignment issue, it still maintains the best performance under Heter. Finally, Te-LoRA and FlexLoRA show slight performance drops when transitioning from Homo to Heter on the Dolly dataset. This may be due to Dolly's smaller dataset (15k) compared to Alpaca-GPT4 (52k) and Wizard (70k). In the Heter configuration, the 64 and 32 rank LoRA models may overfit on clients with less data, leading to poorer aggregation results.

Communication overhead. Communication cost plays a crucial role in real-world FL. Therefore, we compared the total upload and download communication costs in a single round across different methods. To ensure a fair comparison, we normalized the transmission parameters. As shown in Fig. 3, the proposed Te-LoRA has the lowest communication cost, as it uploads only module A or B per round. In contrast, FlexLoRA and Zero-padding require both modules to be uploaded and downloaded each round, while

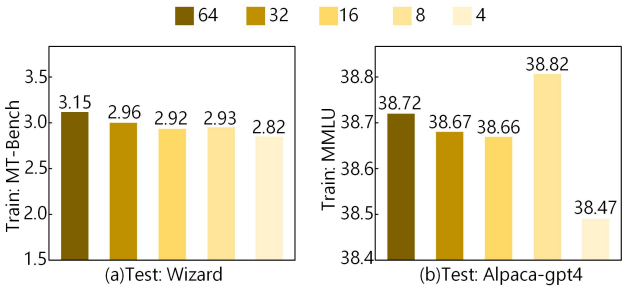


Figure 4. Impact of different ranks (64, 32, 16, 8, and 4).

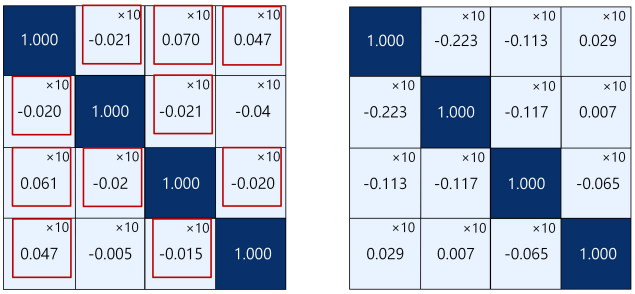
FLoRA incurs the highest cost due to transmitting the result of multiplying matrices B and A during download. The quantitative comparison highlights the advantage of the alternating freezing technique.

Impact of rank. To further investigate the impact of different rank configurations on homogeneous-rank clients, we conducted experiments on the "Local" baseline with five rank settings (64, 32, 16, 8, and 4). Training was performed on the Wizard and Alpaca-GPT4 datasets, and evaluation was conducted using MT-Bench and MMLU, respectively. As illustrated in Fig. 4, setting the rank to 4 for all clients, which aligns with the most resource-constrained ones, leads to a significant performance decline. While higher ranks can improve performance, they are not feasible for resource-limited clients. Therefore, adopting a heterogeneous rank configuration enables better resource utilization and further enhances performance.

4.3. Ablation Experiment

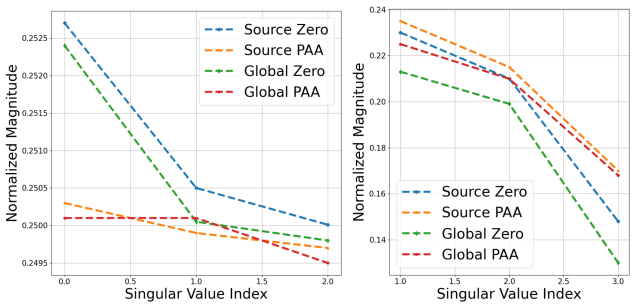
To demonstrate the effectiveness of the PAA-strategy and T2M-strategy in Te-LoRA, we conducted several ablation experiments. Specifically, "w/o PPA" refers to replacing the PAA-strategy with zero-padding in the heterogeneous experiment, which combines the alternating freezing method with zero-padding. "w/o T2M" indicates replacing the T2M-strategy with the weighted average aggregation of FedAvg in the heterogeneous experiments.

The effects of PAA-strategy. To compare with the alignment effect of zero-padding and PAA-strategy, we visualized the cosine similarity between clients after alignment using zero-padding and PAA-strategy, as well as the change curves of the singular value distribution before and after alignment. As shown in Fig. 5, we visualized the cosine similarity between four clients, where most clients exhibit higher cosine similarity after alignment with the PAA strategy compared to zero-padding. This indicates that the PAA-strategy not only aligns the parameters between heterogeneous LoRAs on the server side but also reduces the information loss caused by the alignment. To better illustrate the differences before and after alignment, we compared the normalized singular value distribution curves, as shown in Fig. 6. It can be observed that the distribution of singular



(a) On PAA-strategy. (b) On Zero-padding.

Figure 5. Cosine similarity visualization of the local model processed by PAA-strategy or Zero-padding with the global model.



(a) In round four. (b) In round seven.

Figure 6. Singular value distribution curves. The source LoRA and those transformed by zero-padding and the PAA strategy undergo SVD to plot the singular value distribution curves.

values after applying the PAA strategy is closer to that of the original LoRA, while the distribution after zero-padding is relatively farther away. This indicates that the PAA strategy retains more of the major components to a certain extent and is closer to the original LoRA compared to zero-padding.

To further validate the effect of PPA quantitatively, we conducted an ablation study, as shown in Tab. 3. In all scenarios, "w/o PAA" performs worse than T2M_2, with a performance drop of at least 1.3 points in each case. This validates that the PAA-strategy, compared to Zero-padding, can more effectively reduce information loss when aligning parameters between different levels of the A and B modules. Meanwhile, compared to the FlexLoRA in Tab. 2, "w/o T2M" performs better in both the Dolly and Alpaca-GPT4 scenarios. This suggests that combining the PAA-strategy and alternating freezing strategy avoids aggregation noise while achieving parameter alignment between different levels of modules A and B, reducing information loss.

The effects of T2M-strategy. To evaluate the impact of this strategy, as shown in Tab. 3, we performed an ablation experiment with "w/o T2M". First, "w/o T2M" performs worse in all three scenarios compared to the T2M_2 method, especially in the Alpaca-GPT4 dataset, where w/o T2M is about 0.63 points lower than T2M_2. This validates that the T2M-strategy, compared to the weighted average

Method		MMLU	
Heter	Wizard	Dolly	Alpaca-GPT4
w/o PAA	22.40	27.10	44.77
w/o T2M	23.38	28.21	45.53
T2M_1	22.91	28.14	45.30
T2M_2	**23.71**	**28.37**	**46.16**
T2M_3	23.67	28.30	45.99

Table 3. Validation of Te-LoRA components. 'Heter' represents settings with heterogeneous LoRA ranks.

aggregation operation of FedAvg, better preserves the key information from each client, enabling finer feature alignment. In comparison to Zero-padding in Tab. 2, "w/o PAA" achieves higher MMLU scores for all three datasets: 0.54, 0.38, and 0.3 points higher on Wizard, Dolly, and Alpaca-GPT4, respectively. This indicates that the T2M-strategy combined with alternating freezing more effectively aligns features while avoiding aggregation noise.

The effects of T2M_1, T2M_2, and T2M_3. As observed from Tab. 3, T2M_1 performs slightly worse than T2M_2. In the Wizard, Dolly, and Alpaca-GPT4 datasets, T2M_2 scores 0.2, 0.23, and 0.86 points higher than T2M_1, respectively. While T2M_3 outperforms T2M_1 on all three datasets, it still trails T2M_2, though the results are nearly identical. For example, in the Wizard, Dolly, and Alpaca-GPT4 datasets, T2M_2 scores 0.004, 0.007, and 0.005 points higher than T2M_3. This suggests that although the tensor $\mathbb{B}_*^1 \in \mathbb{R}^{1\times 1\times k}$ obtained from the compression of $\widehat{\mathcal{B}} \in \mathbb{R}^{d\times r_{\max}\times k}$ can preserve the proportion of each client, it is slightly less effective in retaining client information. On the other hand, the tensor $\mathbb{B}_* \in \mathbb{R}^{d\times r_{\max}\times 1}$ obtained from the compression of $\widehat{\mathcal{B}}$ better preserves the key information from each client and does not introduce redundancy in information like $\mathbb{B}_*^3 \in \mathbb{R}^{d\times r_{\max}\times 1}$ does.

5. Conclusion

In this paper, we introduce a novel Tensor-aggregation LoRA method (Te-LoRA) to address the noise and misalignment issues in the aggregation of low-rank adaptation within federated learning under resource heterogeneity. Building upon an alternating freezing strategy, Te-LoRA incorporates Pre-Aggregation Alignment (PAA) and Tensor Decomposition Aggregation (T2M) strategies. The PAA strategy utilises OT technology to align the weights of heterogeneous LoRA modules A and B from different clients into homogeneous weights, while the T2M-strategy adaptively aggregates key components to preserve interdependencies. Experimental results demonstrate that Te-LoRA effectively reduces communication costs and significantly improves fine-tuning performance, offering a viable solution for efficient federated fine-tuning in resource-constrained environments.

Acknowledgements. The work is supported by the National Natural Science Foundation of China (Grant No. 62222207, 62427808, 62472208, 62276134).

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 6

[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. 1, 3

[3] Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. Federated fine-tuning of large language models under heterogeneous language tasks and client resources. *arXiv e-prints*, pages arXiv–2402, 2024. 2, 3, 6, 7

[4] Desirée Bill and Theodor Eriksson. Fine-tuning a llm using reinforcement learning from human feedback for a therapy chatbot application, 2023. 1

[5] Daoyuan Chen, Yilun Huang, Zhijian Ma, Hesen Chen, Xuchen Pan, Ce Ge, Dawei Gao, Yuexiang Xie, Zhaoyang Liu, Jinyang Gao, et al. Data-juicer: A one-stop data processing system for large language models. In *Companion of the 2024 International Conference on Management of Data*, pages 120–134, 2024. 1

[6] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023. 2

[7] Yew Ken Chia, Pengfei Hong, Lidong Bing, and Soujanya Poria. Instructeval: Towards holistic evaluation of instruction-tuned large language models. *arXiv preprint arXiv:2306.04757*, 2023. 6

[8] Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. Heterogeneous lora for federated fine-tuning of on-device foundation models. *arXiv preprint arXiv:2401.06432*, 2024. 3, 6, 7

[9] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023. 1

[10] Sannara Ek, Kaile Wang, François Portet, Philippe Lalanda, and Jiannong Cao. Fedali: Personalized federated learning with aligned prototypes through optimal transport. *arXiv preprint arXiv:2411.10595*, 2024. 5

[11] Junyao Gao, Yanan Sun, Fei Shen, Xin Jiang, Zhening Xing, Kai Chen, and Cairong Zhao. Faceshot: Bring any character into life. *arXiv preprint arXiv:2503.00740*, 2025. 1

[12] Lucas Grativol, Mathieu Leonardon, Guillaume Muller, Virginie Fresse, and Matthieu Arzel. Flocora: Federated learning compression with low-rank adaptation. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pages 1786–1790. IEEE, 2024. 2

[13] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020. 6

[14] 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. 1, 2

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 3

[16] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2021. 2

[17] Xin Jiang, Hao Tang, Junyao Gao, Xiaoyu Du, Shengfeng He, and Zechao Li. Delving into multimodal prompting for fine-grained visual classification. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2570–2578, 2024. 1

[18] Xin Jiang, Hao Tang, and Zechao Li. Global meets local: Dual activation hashing network for large-scale fine-grained image retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 2024. 1

[19] Xin Jiang, Hao Tang, Rui Yan, Jinhui Tang, and Zechao Li. Dvf: Advancing robust and accurate fine-grained image retrieval with retrieval guidelines. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 2379–2388, 2024. 1

[20] Dominique Kelly, Yimin Chen, Sarah E Cornwell, Nicole S Delellis, Alex Mayhew, Sodiq Onaolapo, and Victoria L Rubin. Bing chat: The future of search engines? *Proceedings of the Association for Information Science and Technology*, 60(1):1007–1009, 2023. 1

[21] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 5

[22] Jabin Koo, Minwoo Jang, and Jungseul Ok. Towards robust and efficient federated low-rank adaptation with heterogeneous clients. *arXiv preprint arXiv:2410.22815*, 2024. 2, 3, 4

[23] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 1, 2

[24] Hongxia Li, Wei Huang, Jingya Wang, and Ye Shi. Global and local prompts cooperation via optimal transport for federated learning. In *CVPR*, pages 12151–12161, 2024. 5

[25] Pengpeng Li, Xiangbo Shu, Chun-Mei Feng, Yifei Feng, Wangmeng Zuo, and Jinhui Tang. Surgical video workflow analysis via visual-language learning. *npj Health Systems*, 2 (1):5, 2025. 1

[26] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019. 3

[27] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 2

[28] Zheng Lin, Xuanjie Hu, Yuxin Zhang, Zhe Chen, Zihan Fang, Xianhao Chen, Ang Li, Praneeth Vepakomma, and Yue Gao. Splitlora: A split parameter-efficient fine-tuning framework for large language models. *arXiv preprint arXiv:2407.00952*, 2024. 2

[29] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023. 6

[30] 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. 1, 4

[31] Edouard Pauwels and Samuel Vaiter. The derivatives of sinkhorn–knopp converge. *SIAM Journal on Optimization*, 33(3):1494–1517, 2023. 5

[32] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023. 6

[33] Zhen Qin, Daoyuan Chen, Bingchen Qian, Bolin Ding, Yaliang Li, and Shuiguang Deng. Federated full-parameter tuning of billion-sized language models with communication cost under 18 kilobytes. *arXiv preprint arXiv:2312.06353*, 2023. 1

[34] Riccardo Salami, Pietro Buzzega, Matteo Mosconi, Jacopo Bonato, Luigi Sabetta, and Simone Calderara. Closed-form merging of parameter-efficient modules for federated continual learning. *arXiv preprint arXiv:2410.17961*, 2024. 2

[35] Xiangbo Shu, Jinhui Tang, Guo-Jun Qi, Wei Liu, and Jian Yang. Hierarchical long short-term concurrent memory for human interaction recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(3):1110–1118, 2019. 2

[36] Xiangbo Shu, Liyan Zhang, Guo-Jun Qi, Wei Liu, and Jinhui Tang. Spatiotemporal co-attention recurrent neural networks for human-skeleton motion prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):3300–3315, 2021. 1

[37] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620 (7972):172–180, 2023. 1

[38] Tianyu Song, Shumin Fan, Pengpeng Li, Jiyu Jin, Guiyue Jin, and Lei Fan. Learning an effective transformer for remote sensing satellite image dehazing. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023. 1

[39] Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. *CoRR*, 2024. 3

[40] Jinhui Tang, Xiangbo Shu, Rui Yan, and Liyan Zhang. Coherence constrained graph lstm for group activity recognition. *IEEE transactions on pattern analysis and machine intelligence*, 44(2):636–647, 2019. 2

[41] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 6

[42] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*, 2022. 6

[43] Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. *arXiv preprint arXiv:2409.05976*, 2024. 2, 3, 6, 7

[44] Feijie Wu, Zitao Li, Yaliang Li, Bolin Ding, and Jing Gao. Fedbiot: Llm local fine-tuning in federated learning without full model. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3345–3355, 2024. 2

[45] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6137–6147, 2024. 6

[46] Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the federatedgpt: Federated instruction tuning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6915–6919. IEEE, 2024. 1, 2, 4, 6

[47] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023. 2

[48] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023. 6