FedAlign-LoRA: Federated Direction Aligned LoRA Weight Aggregation

Anonymous ACL submission

Abstract

As large language models (LLMs) become increasingly prevalent, efficiently fine-tuning them across diverse, decentralized data sources poses several challenges. Various solutions have been proposed to address these problems. Among these, Federated LoRA facilitates communication-efficient fine-tuning of models across decentralized data sources. However, existing solutions depend on various approximate update formulations that introduce noise by averaging independent low-rank factors. Previous approaches have attempted to mitigate this by restricting updates to a single factor or by employing rotational transformations. Nonetheless, these methods fail to resolve a fundamental issue: structural ambiguity in the ordering and orientation of learned low-rank components across the clients. In this study, we propose a solution, FedAlign-LoRA, which identifies and formalizes the Structural Ambiguity problem in federated LoRA. This study introduces an aggregation framework that achieves invariant axis realignment using signed permutation matrices. The optimal alignment is efficiently determined by conceptualizing symmetry-aware weight matching as a bipartite assignment problem, which is solved using the Hungarian algorithm. Empirical evaluations across various datasets indicate that FedAlign-LoRA improves performance, with updates that are faster and more stable than those of federated LoRA baselines.

1 Introduction

The distributed nature of large amounts of data, coupled with privacy constraints that restrict the training of centralized models, has led to decentralized approaches such as federated learning (McMahan et al., 2016). In federated learning, data do not need to be collected at a central server, which also helps keep data private and allows models to learn in line with evolving governance policies.

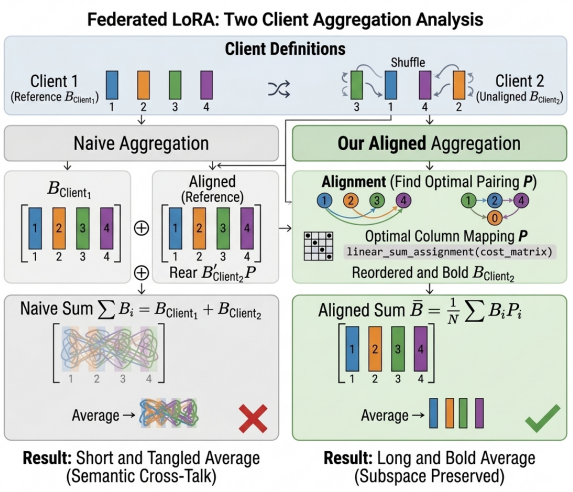

Figure 1: Comparison of Naive and direction-aligned weight aggregation, where aligned aggregation preserves the semantic information more purely.

Due to the communication cost of transferring large models, a federated learning approach using LoRA for parameter-efficient training was proposed (Hu et al., 2022). However, this poses a unique challenge: aggregation with the correct mathematical formulation. The naive aggregation of LoRA factors introduces noise. This is because factors A and B are aggregated separately, as shown in Equation 2, whereas the ideal aggregation is shown in Equation 1.

$$\Delta W^t_{\text{ideal}} = \frac{1}{N}\sum_{i=1}^{N} \Delta W^t_i = \frac{1}{N}\sum_{i=1}^{N} B^t_i A^t_i \quad (1)$$

$$\Delta W^t_{\text{naive}} = \left(\frac{1}{N}\sum_{i=1}^{N} B^t_i\right)\left(\frac{1}{N}\sum_{i=1}^{N} A^t_i\right) \quad (2)$$

This has been addressed in the literature using various approaches, such as FFA-LoRA (Sun et al., 2024) and RoLoRA (Chen et al., 2026). In FFA-Lora, a single factor B was trained with A held

constant using Gaussian initialization, whereas in RoLoRA, training was performed alternatively over the round for A or B, removing the noise term from aggregation and increasing the required training rounds owing to fewer updates per round. In this work, we approach aggregation from a different perspective by treating the LoRA factors as learned directions, factor B as the update direction vectors, and factor A as the scale of these directions. The existing literature treats these factors as sets of vectors without explicitly accounting for their directional alignment. In naive aggregation, treating each column as a direction, columns from multiple LoRA factors can cause destructive interference if they do not point in the same direction, leading to high uncertainty in the performance of the aggregated model across epochs. To mitigate this issue, we present weight aggregation with basis alignment, which can be integrated with various non-directional weight aggregation techniques to address noise terms. Viewing them from a directional perspective and aligning them in similar directions across helps mitigate destructive inference and achieve faster convergence, as demonstrated empirically in this study. Due to the noisy nature of aggregated model learning, we also propose a novel metric, FedScore, that combines multiple criteria for better comparison across methods.

Contributions:

(i) We identify and formalize the Structural Ambiguity problem in federated LoRA.

(ii) We propose FedAlign-LoRA, a novel aggregation framework that mitigates the abovementioned issue using invariant axis realignment via signed permutation matrices.

(iii) Lastly, we present a new metric 'FedScore' for better comparison of different methods, considering the non-smooth performance nature of aggregated models.

2 Related Work

Federated Learning has emerged as an effective paradigm for training models across decentralized data sources, where data sharing is restricted due to privacy, regulatory, or system constraints. In the first work on this topic (McMahan et al., 2016), the authors introduced Federated Averaging (FedAvg), in which client-side model updates are aggregated via weighted averaging based on local data sizes. This naive aggregation induces client drift, particularly with heterogeneous data, and degrades convergence. To address these challenges, several foundational extensions, such as FedProx (Mathew and Asha, 2024), SCAFFOLD (Karimireddy et al., 2020), and FedNova (Wang et al., 2021), have been proposed, mitigating these issues by introducing proximal regularization, control variates, and normalization strategies, respectively.

With the rise of large models, full-parameter training has become increasingly impractical because of communication and memory constraints. This has motivated the adoption of parameter-efficient fine-tuning (PEFT) methods in federated settings, predominantly Low-Rank Adaptation (LoRA) (Hu et al., 2022). Building on this, FedIT (Zhang et al., 2024) presented a federated learning framework with low-rank updates, demonstrating the feasibility of LoRA-based federated training. However, FLoRA (Wang et al., 2024) identified that independently averaging low-rank factors leads to an implicit noise term. Two main directions have emerged to address this issue. From an optimization perspective, frameworks such as EcoLoRA (Liu et al., 2025) and LoRA-FAIR (Bian et al., 2025) introduce communication-efficient tracking and initialization refinements to bound local subspace drift. Alternatively, structural modification approaches attempt to bypass the cross-term mismatch entirely. In FFA-LoRA, only a single factor is updated (Sun et al., 2024), whereas in RoLoRA (Chen et al., 2026), the authors alternately train factor A or B in a given round. Although this successfully eliminates cross-term noise during server-side aggregation, it inherently slows convergence and increases the required training rounds due to fewer updates per round. Other approaches, such as FedRPCA (Jhunjhunwala et al., 2025), decompose client updates into shared low-rank and sparse personalized components to better preserve client-specific information than FedAvg.

Similarly, HiLoRA (Peng et al., 2026) proposes a hierarchical low-rank adaptation that separates global and client-specific components across multiple levels, improving personalization. This perspective highlights the structured roles of low-rank components and motivates the need for consistent alignment during aggregation, which is an underexplored aspect. From a geometric perspective, LoRA constrains updates to a low-dimensional subspace in which each rank component corresponds to a directional basis. While (Hu et al., 2022) provides foundational insights into low-rank adaptation, the consistent alignment of these directional

components across clients remains largely unexplored. In concurrent work, FedRot-LoRA (Zhang et al., 2026) addressed subspace misalignment via Procrustes-based rotation, aligning client subspaces globally. However, this operates at the subspace level and does not preserve the correspondence between individual rank components, potentially combining distinct local directions into a single global axis.

In this work, we propose a framework that preserves directional consistency during LoRA aggregation. To the best of our knowledge, this is the first approach that explicitly maintains rank-wise correspondence across clients, enabling true alignment of low-rank updates while avoiding subspace ambiguity and minimizing cross-term noise.

3 Methodology

In low-rank optimization, instead of training the full model weight, low-rank factors A and B are trained and added to the model weight as ΔW as shown in Equation 3, where it can be considered as pointing the model in the direction in which it had to update the weight.

$$W_{\text{updated}} = W_0 + \Delta W \ , \tag{3}$$

where $\Delta W = BA$ and $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times d}$ with $r << d$ where d is the hidden dimension.

3.1 The Structural Ambiguity Problem in Federated LoRA

In federated learning with low-rank optimization, at round t, the client trains the B and A factors and sends their ΔW_i^t to the server, which aggregates these weights, where the ideal weight aggregation follows Equation 1. However, owing to the communication overhead of sending ΔW_i^t to the server, clients send their individual A_i^t and B_i^t, where instead of computing the product of A_i^t and B_i^t server aggregates them individually as written in Equation 2 due to space and compute overhead of calculating ΔW_i^t. Such individual aggregation results in the inclusion of a noise term, and hence $\Delta W_{Naive}^t \neq \Delta W_{ideal}^t$. To overcome this, (Sun et al., 2024; Chen et al., 2026) used a single factor per round, as discussed in Section 2. In addition, during the aggregation of weights, it is assumed that each client has the same ordered feature space, that is, B_1 is in the same direction as B_2, and the same for A_1 and A_2. This may not be true due to independent learning by individual clients, and even with the equivalent products B_1A_1 and B_2A_2, the individual B_i and A_i can differ significantly from each other. that is, a neural network can learn the exact same semantic update ΔW_i but represent it with a completely different internal ordering and sign configuration in its latent rank dimension r, as further explained in Appendix C.

3.2 Invariant Axis Realignment with Projection

To address the identified issue, we propose a projection-based Direction-Aligned Federated averaging method that aligns the learned directions across clients, thereby reducing the destructive inference caused by averaging mismatched bases. For this, we aim to find a signed permutation matrix $P \in \mathbb{R}^{r\times r}$ that reorders the rank dimensions (columns/rows) and/or flips their signs (± 1), we find the P with the optimisation problem as $\min_{P_i} \|B_iP_i - B_{\text{ref}}\|_F^2$ where B_i is i^{th} client's LoRA factor, P_i is the projection matrix for client i and B_{ref} is a reference factor. Because any signed permutation matrix merely reorders or flips the coordinate axes without changing their lengths, it is strictly orthogonal. Therefore, its transpose is equal to its inverse, $PP^\top = I$ and multiplying B with P and $P^\top$ with A does not change the product ΔW as shown below:

$$\Delta\tilde{W}_i = (B_iP_i)(P_i^\top A_i) = B_i(P_iP_i^\top)A_i = B_iIA_i = B_iA_i = \Delta W_i$$

3.3 Symmetry-Aware Weight Matching via the Hungarian Algorithm

To determine the optimal realignment matrix P_i, we consider the geometric orientations of the columns. First, we scaled them to avoid biased matching owing to different magnitude columns, thereby preserving even smaller but significant features. For this, we perform column-wise L_2 normalization on both matrices. Let $b_{\text{ref},j}$ and $b_{i,k}$ denote the j-th and k-th columns of B_{ref} and B_i respectively:

$$\hat{b}_{\text{ref},j} = \frac{b_{\text{ref},j}}{\|b_{\text{ref},j}\|_2}, \quad \hat{b}_{i,k} = \frac{b_{i,k}}{\|b_{i,k}\|_2} \tag{4}$$

The cosine similarity matrix $S \in \mathbb{R}^{r\times r}$ between all column pairs is computed as follows:

$$S_{j,k} = \hat{b}_{\text{ref},j}^\top \hat{b}_{i,k}$$

Because a sign flip in a column of B can be perfectly compensated for by flipping the correspond-

ing row of A (since $(-B)(-A) = BA$), the matching must be invariant to sign reflection. Next, we compute the cost matrix $C \in \mathbb{R}^{r \times r}$ using the absolute cosine similarity: $C_{j,k} = 1 - |S_{j,k}|$

To find the optimal permutation matrix, which is a linear sum assignment problem (bipartite matching), we find an optimal bijective permutation mapping $\pi : \{1, \ldots, r\} \to \{1, \ldots, r\}$ that minimizes the aggregate global distance:

$$\pi = \arg\min_{\sigma} \sum_{j=1}^{r} C_{j,\sigma(j)}$$

This is solved efficiently using the Hungarian algorithm. The final alignment is defined by the permuted column indices $\text{col_ind}_j = \pi(j)$ along with their corresponding orientation signs:$s_j = \text{sign}\left(S_{j,\pi(j)}\right)$ Hence, the final optimization problem becomes:

$$\min_{P_i \in \mathcal{P}_{\pm}} \|B_i P_i - B_{\text{ref}}\|_F^2 \tag{5}$$

where $\mathcal{P}_{\pm}$ denotes the set of all signed permutation matrices. For reference selection, unlike static selection, which relies on historical global states, we dynamically anchored our alignment target using a localized validation proxy. At the end of the local training round t, the central server benchmarks each client's updated parameters (B_i^t, A_i^t) on a very small, server-held validation partition $\mathcal{D}_{\text{val}}$. The factors belonging to the client that yield the highest validation accuracy are designated as the global structural reference B_{ref}, establishing an empirically optimal coordinate baseline for the remaining clients. Server-side validation is used instead of client-side validation to remove client-specific bias and prevent skewed factor importance due to data size or the adversarial nature of the client. The complete procedure is outlined in Algorithm 1 with the time complexity discussed in Appendix D.

3.4 Evaluation Metric

We evaluate our model based on the test accuracy in the final round. However, in federated learning, the model is susceptible to varying progress, and in practice, variations continue near convergence. To address this, in this paper, we propose a new metric, ***FedScore***, to evaluate the method's performance not only on the final value but also on a consolidated score that provides better information about the federated model performance, rather than just comparing noisy accuracy. These metric selections are inspired by prior work on optimization, continual learning, and federated learning (Goodfellow et al., 2016; McMahan et al., 2016; Li et al., 2020; Karimireddy et al., 2020; Kemker et al., 2018; Gama et al., 2014).

To this end, we consider a set of complementary scores that jointly capture different aspects of the method's performance.

1. LastKMean ($\hat{L}$): Computes the average accuracy over the final K rounds to estimate the stable convergence while reducing sensitivity to noisy spikes.

2. EMA Final ($\hat{M}$): Uses exponential moving averaging to smooth short-term fluctuations and estimate the underlying stable learning trajectory. This is widely used for optimization (Goodfellow et al., 2016).

3. Trend Slope ($\hat{S}$): Represents the overall improvement trend of the accuracy trajectory using. A higher value indicates stronger positive learning progress.

4. Early-Area Under the Learning Curve (Early-

Algorithm 1 FedAlign-LoRA

Require: N clients, rank r, rounds T, validation set $\mathcal{D}_{\text{val}}$.
Server Init: Global parameters $W = W_0 + \Delta W_0$, where $\Delta W_0 = B^0_{global} A^0_{global} = 0$
for round $t = 1$ **to** T **do**
  Server broadcasts $\{B^t_{\text{global}}, A^t_{\text{global}}\}$ to all clients
  for all client $i \in \{1, \ldots, N\}$ **in parallel do**
    Local Training: Optimize local B_i^t, A_i^t via SGD
  end for
  for each client $i = 1$ **to** N **do**
    $\text{Acc}_i \leftarrow \text{evaluate}(B_i^t, A_i^t; \mathcal{D}_{\text{val}})$
  end for
  $i^* \leftarrow \arg\max_i \text{Acc}_i$
  $B_{ref} = B_{i^*}$
  for all client $i \in \{1, \ldots, N\}$ **at server do**
    $\hat{B}_{\text{ref}} \leftarrow \left[\frac{b_{\text{ref},1}}{\|b_{\text{ref},1}\|_2}, \ldots, \frac{b_{\text{ref},r}}{\|b_{\text{ref},r}\|_2}\right]$
    $\hat{B}_i \leftarrow \left[\frac{b_{i,1}}{\|b_{i,1}\|_2}, \ldots, \frac{b_{i,r}}{\|b_{i,r}\|_2}\right]$
    Similarity matrix: $S \leftarrow \hat{B}_{\text{ref}}^\top \hat{B}_i$
    Cost matrix formulation: $C \leftarrow 1_{r \times r} - |S|$
    $\pi \leftarrow \text{HungarianAlgorithm}(C)$
    for each rank channel $j = 1$ **to** r **do**
      $s_j \leftarrow \text{sign}\left(S_{j,\pi(j)}\right)$
      $\tilde{b}_{i,j} \leftarrow s_j \cdot b_{i,\pi(j)}$
      $\tilde{a}_{i,j} \leftarrow s_j \cdot a_{i,\pi(j)}$ $\{a_{i,k}$ is the k-th row of $A_i\}$
    end for
    Construct: $\tilde{B}_i^t = [\tilde{b}_{i,1}, \ldots, \tilde{b}_{i,r}]$, $\tilde{A}_i^t = [\tilde{a}_{i,1}^\top, \ldots, \tilde{a}_{i,r}^\top]^\top$
  end for
  Aligned Weight Aggregation:
  $B^{t+1}_{\text{global}} = \frac{1}{N}\sum_{i=1}^N \tilde{B}_i^t, \quad A^{t+1}_{\text{global}} = \frac{1}{N}\sum_{i=1}^N \tilde{A}_i^t$
end for

AULC)($\hat{E}$): Measures the average performance during the early phase; it is generally used in continual and online learning (Kemker et al., 2018; Gama et al., 2014).

5. Total Variation Stability ($\hat{TV}$): Quantifies oscillatory or zig-zag behaviour in the accuracy trajectory using epoch-to-epoch variations. Lower values indicate smoother and more stable convergence.

6. Post-Peak Stability ($\hat{P}$): Computes the standard deviation of accuracies after the peak round to evaluate the consistency of performance after convergence.

FedScore combines the performance, convergence, and stability metrics into a single score for a comprehensive comparison across methods. The values mentioned in Table 1 are normalized across the experiments. The weights used for these metrics were selected based on their relative importance and to balance the different aspects, as determined by the empirical results. The complete formulation is provided in Appendix E.

$$\textbf{FedScore} = 0.30 \cdot \hat{L} + 0.20 \cdot \hat{M} + 0.20 \cdot \hat{S} + 0.20 \cdot \hat{E} - 0.10 \cdot \hat{TV} - 0.10 \cdot \hat{P} \quad (6)$$

4 Experiments

FedAlign-LoRA was evaluated using standard and recent federated baselines for natural language understanding tasks. The comparison is performed based on different numbers of clients, ranks, data heterogeneity, and LoRA α. We will open-source the code upon its acceptance.

4.1 Datasets and Model

For natural language understanding tasks, we evaluate all methods on three datasets from GLUE (Wang et al., 2018): SST2, CoLA, and RTE, and the Cornell University rotten_tomatoes movie reviews(CMRT) (Pang and Lee, 2005). We use the pretrained RoBERTa-Large (Liu et al., 2020) model from the HuggingFace Transformers library (Wolf et al., 2020). Detailed data description is provided in Appendix B.

4.2 Baselines

We compared FedAlign-LoRA with three federated LoRA baselines. FedIT (Zhang et al., 2024) trains both LoRA factors A and B and aggregates them via naive averaging ($\bar{B}\bar{A}$), preserving full expressivity but being susceptible to aggregation errors due to subspace misalignment. RoLoRA (Chen et al., 2026) alternates between updating B (with A frozen) and updating A (with B frozen) across communication rounds. RoLoRA avoids bilinear factor aggregation but achieves it by restricting the effective parameter space, which can reduce expressivity and result in slower convergence. FedRot-LoRA (Zhang et al., 2026) aligns local LoRA updates via client-side rotations prior to aggregation via Procrustes rotation, which considers the ordering of factor direction to be the same across clients. In our implementation, we used $\lambda = 0.5$ as the soft rotation parameter, since no specific value was provided for all results, and the suggested range was 0 to 1.

4.3 Implementation details

All experiments were conducted with the data split into homogeneous groups, unless otherwise stated, on a set of NVIDIA A6000, A100, and H200 GPUs. The total time for the final experiment was 2436 GPU hours, with FedAlign-Lora and baselines for 573 and 1863 GPU hours, respectively. The results presented are with a random seed of 1 and for single runs, unless otherwise mentioned. We trained the model with the selected learning rate (η) of $1e-5$ based on initial testing over a few rounds on the GLUE-RTE dataset, using the set $\eta = 1e-4, 1e-5, 1e-6$. We set the number of communication rounds to 100 and 200, respectively, for homogenous and heterogenous data split with local training epoch set to one. The batch size was set to 4, 16, 32, and 32 for the SST2, cola, RTE, and CMRT datasets, respectively, for the homogeneous case; for the heterogeneous case, it was set to 4 RTE and 16 others. The maximum sequence length is 128, except for CMRT, which is 256. Following the common practice in fine-tuning, we used AdamW as the optimizer and set the scaling factor to 2× the LoRA rank for all algorithms, unless stated otherwise.

5 Results

Table 1 reports the performance of all methods across four benchmarks(RTE, CoLA, CMRT, and SST2) under varying numbers of clients. We evaluated both accuracy and FedScore, which reflect robustness and generalization in federated settings. Across most settings, the proposed ap-

Dataset	Clients	FedIT		RoLoRA		FedRot-LoRA		FedAlign-LoRA(ours)	
		Acc	FedScore	Acc	FedScore	Acc	FedScore	Acc	FedScore
RTE	3	0.756	0.3313	0.716	0.2597	0.76	0.3205	**0.764**	***0.3813***
	10	0.664	0.2364	0.496	-0.1222	0.496	-0.1171	**0.68**	***0.2452***
	20	0.524	***0.0325***	0.448	-0.1728	0.46	-0.1273	**0.528**	-0.0206
	30	0.492	-0.0147	0.456	-0.0377	0.464	-0.0358	**0.504**	***-0.0109***
CoLA	3	0.8456	0.4288	0.8509	0.4735	0.8498	0.4598	**0.8584**	***0.4802***
	10	**0.8477**	***0.4853***	0.8072	0.3956	0.8317	0.4222	0.8392	0.4774
	20	0.8168	0.4601	0.7966	0.3988	0.8136	0.5679	**0.8275**	***0.4711***
	30	0.8115	0.4603	0.7593	0.3527	0.8019	0.4326	**0.82**	***0.4658***
CMRT	3	0.9193	0.533	0.9146	0.5278	0.9184	0.5423	**0.9203**	***0.5453***
	10	**0.9146**	0.5438	0.9081	0.5181	0.909	0.5241	**0.9146**	***0.5447***
	20	**0.9146**	0.5535	0.8959	0.5446	0.9062	***0.5679***	0.909	0.547
	30	**0.9043**	0.5542	0.894	***0.6005***	0.8959	0.5415	**0.9043**	0.5502
SST2	3	0.9567	0.594	0.9567	0.5799	**0.9592**	0.5851	0.9554	***0.5968***
	10	0.958	***0.6012***	0.9529	0.5751	0.9567	0.5824	**0.958**	0.6007
	20	0.9529	***0.5984***	0.9529	0.579	0.9541	0.5972	**0.9554**	0.5957
	30	0.9541	0.5955	0.9465	0.5716	0.9465	0.5885	**0.9567**	***0.6041***
Sum		13.2321	6.994	12.7516	6.0441	12.923	6.4515	***13.2948***	***7.074***
Average		0.827	0.4371	0.797	0.3778	0.8077	0.4032	***0.8309***	***0.4421***

Table 1: Accuracy and FedScore comparison under a homogenous data setting with rank 4 across multiple datasets for different numbers of clients. The table reports the accuracy(Acc) achieved by each model-dataset pair under various model aggregation methods. The bold values indicate the highest accuracy, and the bold and italic values indicate the highest FedScore achieved. This comparison highlights the effectiveness of the proposed method.

proach FedAlign-Lora achieves the best or competitive performance in both metrics, demonstrating an improvement over FedIT, RoLoRA, and FedRot-LoRA. On RTE, FedAlign-LoRA consistently achieves the highest accuracy across all client settings, followed by FedIT. At 10 clients, it improves the accuracy to 0.68, compared to 0.664 (FedIT), and significantly outperforms RoLoRA and FedRot-LoRA (both 0.469). At 30 clients, it maintains the highest accuracy of 0.504 while also achieving the best FedScore of -0.0109. Although FedIT slightly outperforms at 20 clients with a FedScore of 0.0326, our method maintains a higher accuracy. On CoLA, FedAlign-LoRa performs the best in most cases, particularly in terms of FedScore. With 3 clients, it achieves the best accuracy of 0.8584 and a FedScore of 0.481. FedIT shows a slight advantage at 10 clients, but our method remains competitive and demonstrates stability across varying client counts. On CMRT, all methods perform strongly, but FedAlign-LoRA still provides consistent gains. It achieves the best accuracy and FedScore at 3 clients $(0.9203/0.5462)$. At 10 clients, it matches FedIT in accuracy(0.9146) with an improved FedScore. At 20 clients, FedRot-LoRA attains the highest FedScore(0.5679), whereas FedIT achieves the highest accuracy (0.9146). However, FedAlign-LoRA maintains a balanced performance across both metrics. However, at client 30, RoLoRA achieves maximum accuracy with a second-best FedScore of 0.8959. Overall, our method demonstrates robust and stable performance without sacrificing accuracy for robustness On SST2, all methods perform similarly in terms of accuracy, with FedAlign-LoRA performing the best in all cases except for client 3, where FedRot-LoRA achieved the best accuracy of 0.9592. However, differences emerged in FedScore, with FedIT and FedAlign-LoRA performing equally well. Overall, FedAlign-LoRA performs the best, with an average accuracy of 0.8309 and a FedScore of 7.074. These results suggest that while accuracy saturates on SST2, FedAlign-LoRA still improves federated stability and consistency.

5.1 Ablation Studies

5.1.1 Evaluating Performance Beyond Final-Round Volatility

Table 2 presents a detailed evaluation against competitive benchmarks using our comprehensive FedScore framework, which aggregates multiple distinct dimensions of the model learning trajectory, as discussed in Section 3.4. Across all evaluated datasets, FedAlign-LoRA consistently outperforms the alternative approaches, securing the highest

Dataset	Method	FedScore ↑	Last10 Mean ↑	Early-AULC ↑	Total Variation Stability ↑	EMA Final ↑	Trend Slope ↓	Post-Peak Stability ↓
RTE	FedAlign-LoRA	**0.3813**	**0.7636**	**0.5034**	0.0104	**0.7635**	0.0030	**0.0000**
	FedIT	0.3313	0.7528	0.4988	0.0103	0.7531	**0.0029**	0.0075
	FedRot-LoRA	0.3205	0.7532	0.4784	0.0129	0.7512	0.0035	0.0099
	RoLoRA	0.2597	0.7144	0.4774	**0.0149**	0.7125	0.0032	0.0103
CMRT	FedAlign-LoRA	**0.5453**	**0.9222**	**0.8801**	0.0041	**0.9214**	**0.0005**	0.0025
	RoLoRA	0.5423	0.9170	0.8375	0.0060	0.9174	0.0009	0.0015
	FedIT	0.5333	0.9196	0.8711	0.0058	0.9190	0.0006	0.0031
	FedRot-LoRA	0.5278	0.9102	0.8071	**0.0095**	0.9105	0.0012	**0.0000**
CoLA	FedAlign-LoRA	**0.4802**	**0.8515**	**0.7659**	0.0050	**0.8530**	**0.0010**	**0.0000**
	FedRot-LoRA	0.4735	0.8435	0.7305	0.0052	0.8446	0.0013	**0.0000**
	RoLoRA	0.4598	0.8495	0.7545	0.0051	0.8497	0.0011	0.0036
	FedIT	0.4288	0.8511	0.7609	**0.0065**	0.8508	**0.0010**	0.0085
SST2	FedAlign-LoRA	**0.5968**	0.9566	0.9459	0.0021	0.9566	**0.0001**	**0.0014**
	FedIT	0.5940	0.9561	**0.9475**	0.0018	0.9560	**0.0001**	0.0020
	FedRot-LoRA	0.5851	**0.9581**	0.9358	0.0046	**0.9582**	0.0003	0.0019
	RoLoRA	0.5799	0.9566	0.9159	**0.0060**	0.9565	0.0005	0.0017

Table 2: A detailed comparison of FedScore and individual metrics for 3 clients and rank 4. This highlights the sensitivity of the method behavior toward performance and learning robustness. This highlights the robustness of the proposed method. Bolded values indicate the highest accuracy, and ↑ and ↓ denote the preference of value with methods arranged in the ranked order of FedScore.

overall FedScore. FedAlign-LoRA demonstrates superior final performance while maintaining exceptional post-peak consistency, achieving a Post-Peak Stability Std of exactly 0.0 in three out of four datasets. This highlights its capability to strike an optimal balance between aggressive optimization and late-stage convergence stability. A primary motivation for introducing FedScore is to expose baseline models that exhibit high but deeply unstable final accuracies. This phenomenon is clearly demonstrated in the CoLA dataset, where relying purely on the traditional Last10Mean metric would falsely position the volatile FedIT baseline (0.8511) as being superior to RoLoRA (0.8435). However, FedScore correctly penalizes FedIT for its severe late-stage volatility (a Post-Peak Stability Std of 0.0085 and a high smoothness penalty of 0.0065), ultimately ranking it last, while rewarding RoLoRA for its stable, zero-variance post-peak behavior. For the other datasets, RoLoRA also shows the best Total Variational Stability by achieving the highest value in each case, which can be attributed to its learning of only one LoRA factor per round. For the CMRT dataset, FedRot-LoRA showed the best trend stability, achieving the highest value of 0.0095, along with the best post-peak stability of 0.0. Similarly, in the SST2 dataset, although certain baselines closely match the final accuracy of our method, they fail to maintain stability after reaching their peak performance. By capturing these subtle oscillatory degradation patterns alongside the overall learning trend slope, FedScore provides a highly dependable and holistic evaluation that reflects the true deployment readiness in highly variable federated settings.

5.1.2 Rank Sensitivity Analysis

To further assess ranking sensitivity and consistency across methods, Table 3 presents results for the varying rank settings of $4, 8$, and 16on the RTE and CMRT datasets. Across the different rank configurations, FedAlign-LoRA consistently achieves the best or near-best performance, demonstrating its robustness to changes in adaptation capacity. Unlike baselines, whose performance fluctuates more significantly with rank, our method maintains stable, strong accuracy across both datasets. On the RTE dataset, the overall best accuracy of 0.8 is achieved by FedAlign-LoRA at rank 8, followed by FedAlign-LoRA and FedIT at rank 16, with accuracies of 0.796 and 0.792, respectively, demonstrating the method's ability to achieve higher accuracy at lower ranks in some cases. Similarly, on the CMRT dataset, FedAlign-LoRA achieves an accuracy of 0.9203 at rank 4; however, the best accuracy of 0.9231 was achieved by both FedRot-LoRA at rank 8 and FedAlign-LoRA at rank 16, demonstrating the model's ability to perform well at lower-rank levels.

Rank	Method	RTE	CMRT
4	FedIT	0.7560 0.3313	0.9193 0.5332
	RoLoRA	0.7160 0.2597	0.9146 0.5278
	FedRot-LoRA	0.7600 0.3205	0.9184 0.5423
	FedAlign-LoRA	**0.7640** **0.3813**	**0.9203** **0.5453**
8	FedIT	0.7880 0.3095	0.9203 0.5350
	RoLoRA	0.7600 0.3501	0.9156 0.5368
	FedRot-LoRA	0.7720 0.3281	**0.9231** **0.5438**
	FedAlign-LoRA	**0.8000** **0.3615**	0.9165 0.5284
16	FedIT	0.7920 0.3589	0.9221 0.5337
	RoLoRA	0.7600 0.3367	0.9165 **0.5349**
	FedRot-LoRA	0.7720 0.2849	0.9184 0.5295
	FedAlign-LoRA	**0.7960** **0.3790**	**0.9231** 0.5229

Table 3: Comparison of LoRA rank sensitivity across different methods on the RTE and CMRT datasets (3 clients). The second row for each method indicates the FedScore.

5.1.3 Evaluation Under Non-IID Settings

Table 4 details performance under Dirichlet-based heterogeneity, where lower α dictates extreme non-IID conditions. While all methods degrade as α drops from 1.0 to 0.1 across both client configurations, FedAlign-LoRA consistently mitigates this impact. It outperforms all baselines in nearly every setup, peaking at 0.800 (3 clients) and 0.668 (10 clients) at $\alpha = 1.0$. Detailed FedScore results are provided in Appendix F.

N	Method	α = 0.1	α = 0.5	α = 1.0
3	FedAlign-LoRA	0.520	0.684	0.800
	FedIT	0.532	0.636	0.792
	RoLoRA	0.512	0.512	0.772
	FedRot-LoRA	0.488	0.512	0.784
10	FedAlign-LoRA	0.516	0.616	0.668
	FedIT	0.548	0.524	0.608
	RoLoRA	0.512	0.512	0.512
	FedRot-LoRA	0.488	0.512	0.504

Table 4: Performance comparison with 3 and 10 Clients denoted by N under varying Dirichlet α on RTE dataset with methods arranged based on accuracy.

Method	2	4	8
FedAlign-LoRA	0.720	0.720	**0.764**
FedRot-LoRA	0.700	0.740	**0.760**
FedIT	0.720	0.720	**0.756**
RoLoRA	0.616	0.676	**0.716**

Table 5: Comparison over LoRA α for rank 4 on RTE dataset with methods arranged based on accuracy.

5.1.4 Impact of LoRA Alpha (α)

Table 5 demonstrates the accuracy variation across different α values for rank 4 in 3 client setting. All methods show superior performance at $\alpha = 8$ with our proposed method, FedAlign-LoRA, matching or outperforming the baselines across all tiers. Notably, it achieves the highest overall performance score in the matrix (0.764), marginally edging out FedRot-LoRA.

5.1.5 Rotation Effect on Aligned Factors

Table 6 compares FedAlign-LoRA with applying Procrustes Rotation on Factor B on the aligned factor for 3 clients with rank 4. FedAlign-LoRA performed better without the subspace rotation.

Dataset	F1	F1 with Rotaion(F2)	Δ (F1 - F2)
RTE	0.764	0.752	0.012
CoLA	0.8584	0.8466	0.0118
CMRT	0.9203	0.9193	0.001

Table 6: Performance difference between FedAlign-LoRA(F1) and FedAlign-LoRA with Procrustes Rotation on Factor B(F2)

6 Conclusion

In this paper, we identify an issue with the alignment of LoRA factors in federated weight averaging from the perspective of B's column as learned directions. With this, we propose **FedAlign-LoRA**, an Invariant Axis Realignment with the Projection matrix, which aims to find the best projection matrix with respect to the reference and align client weights before aggregation. To demonstrate the effectiveness of the proposed approach, we conduct extensive experiments across clients, ranks, datasets and data heterogeneity. Lastly, to navigate the inherent volatility of federated learning trajectories, we introduce **FedScore**, a novel metric that consolidates performance, convergence, and stability into a single holistic score, capturing nuanced learning dynamics that traditional final-round metrics overlook.

Limitations

FedAlign-LoRA considers the availability of a very small validation sample. Although such a small sample can be taken from the open source of the same distribution, it may not be available in all cases. In such a case, different techniques can be applied to find the reference factor, and a base approach is to use the previous global round; however, we have not benchmarked these approaches in the current work and leave them for further exploration by interested researchers. In addition, the current work is limited to a smaller model, and further exploration is needed to determine the effectiveness of FedAlign-LoRA for larger models and different take settings, like text generation.

References

Jieming Bian, Lei Wang, Letian Zhang, and Jie Xu. 2025. Lora-fair: Federated lora fine-tuning with aggregation and initialization refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3737–3746.

Shuangyi Chen, Yuanxin Guo, Yue Ju, Hardik Dalal, Zhongwen Zhu, and Ashish J Khisti. 2026. Robust federated finetuning of LLMs via alternating optimization of loRA. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3.

Divyansh Jhunjhunwala, Arian Raje, Madan Ravi Ganesh, Chaithanya Kumar Mummadi, Chaoqun Dong, Jiawei Zhou, Wan-Yi Lin, Gauri Joshi, and Zhenzhen Li. 2025. Fedrpca: Enhancing federated lora aggregation using robust pca. *arXiv preprint arXiv:2506.01194*.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pages 5132–5143. PMLR.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. 2018. Measuring catastrophic forgetting in neural networks. In *AAAI*.

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

Han Liu, Ruoyao Wen, Srijith Nair, Jia Liu, Wenjing Lou, Chongjie Zhang, William Yeoh, Yevgeniy Vorobeychik, and Ning Zhang. 2025. Ecollora: Communication-efficient federated fine-tuning of large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20743–20757.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Cina Mathew and P Asha. 2024. Fedprox: Fedsplit algorithm based federated learning for statistical and system heterogeneity in medical data communication. *J. Internet Serv. Inf. Secur.*, 14(3):353–370.

H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. 2016. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

Zihao Peng, Nan Zou, Jiandian Zeng, Guo Li, Ke Chen, Boyuan Li, and Tian Wang. 2026. Hilora: Hierarchical low-rank adaptation for personalized federated learning. *CVPR*.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. 2024. Improving lora in privacy-preserving federated learning. *arXiv preprint arXiv:2403.12313*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium. Association for Computational Linguistics.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. 2021. A novel framework for the analysis and design of heterogeneous federated learning. *IEEE Transactions on Signal Processing*, 69:5234–5249.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. 2024.

FLoRA: Federated fine-tuning large language models with heterogeneous low-rank adaptations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online. Association for Computational Linguistics.

Haoran Zhang, Dongjun Kim, Seohyeon Cha, and Haris Vikalo. 2026. Fedrot-lora: Mitigating rotational misalignment in federated lora.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. 2024. 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.

A LLM Disclosure

In accordance with the ACL Policy on LLMs, I certify that a Large Language Model was utilised exclusively for grammatical refinement and stylistic polishing. The core research conceptualisation, methodology, and experimental execution remain entirely our own independent intellectual work.

B Datasets

The experiments are conducted on four benchmark text classification datasets: GLUE RTE, CoLA, Stanford NLP Group SST2, and Cornell University Rotten Tomatoes. RTE (Recognizing Textual Entailment) is a binary natural language inference task which the model determines whether a hypothesis sentence is entailed by a premise sentence. CoLA (Corpus of Linguistic Acceptability) is a binary classification task that identifies whether a sentence is grammatically correct. SST -2 (Stanford Sentiment Treebank-2) and Rotten Tomatoes are binary sentiment classification datasets that aim to classify movie reviews as positive or negative. For datasets in which official test labels are unavailable (RTE, CoLA, and SST2), the original validation split is further divided into 10 % validation and 90% test subsets for selecting the reference LoRA factor and final evaluation, respectively. The details of the data split are presented in Table 7.

C Subspace Alignment Example

In the case of considering B and A as sets of directions for the updates and their magnitudes, respectively. The naive aggregation can introduce noise if the columns do not point in similar directions, as we directly aggregated the ith column of $client_1$ with the ith column of $client_2$. For example, if the 1st column of $client_1$ points toward the "heights", while the 1st column of $client_2$ points toward the " color " directions, then a simple averaging of these directions can introduce noise. Hence, this naive aggregation mixes mismatched features, leading to destructive interference and severe degradation in performance.

D Complexity Analysis

FedAlign-LoRA is designed to be computationally efficient, adding only a small overhead per round, which is governed by the LoRA rank and number of clients. The aggregation is categorized into two main parts: 1) evaluating the client on a very small validation set, which incurs a cost of $\mathcal{O}(N \cdot |\mathcal{D}_{\text{val}}| \cdot C_{\text{fwd}} + N)$ and 2) Aligning the LoRA factors, which is done in five steps. 1. Column normalization with a cost of $\mathcal{O}(r \cdot d)$ 2. The similarity matrix costs $\mathcal{O}(r^2 \cdot d)$ due to the multiplication of size $(r \times d)$ and $(d \times r)$ 3. Cost matrix computed in $\mathcal{O}(r^2)$ 4. For the Hungarian matrix, $\mathcal{O}(r^3)$ is required lastly 5. Permutation and sign correction are performed in $\mathcal{O}(r \cdot (d + k))$, where k is the output dimension, resulting in a total time complexity of $\mathcal{O}(r^2 d + r^3 + rk)$ for a client. Because $(r \ll d)$, the overhead of the proposed algorithm is minimal to the local training cost without introducing a communication overhead.

E Evaluation Metrics

In this section, we present the complete mathematical formulation used for different metrics, where a_t is the accuracy at round t.

$$\text{LastKMean}(\hat{L}) = \frac{1}{K} \sum_{t=T-K+1}^{T} a_t \qquad (7)$$

K is the number of last rounds to consider, with K equal to 10 used in this paper.

$$\text{EMAFinal}(\hat{M}) = \text{EMA}_T \qquad (8)$$

where $\text{EMA}_t = \alpha a_t + (1 - \alpha)\text{EMA}_{t-1}$ In the paper, we considered α as 0.2. and $\text{EMA}_1 = a_1$.

Dataset	Train split	Validation split	Test split
GLUE-RTE(RTE)	2,490	28	249
GLUE-CoLA (CoLA)	8,551	104	939
Stanford NLP Group-SST2 (SST2)	67,349	87	785
Rotten Tomatoes Movie Review(RTMR)	8,530	1,066	1,066

Table 7: Data split used for experiment

The trend slope shows the overall learning tendency using linear regression over the accuracy trajectory.

$$\mathrm{TrendSlope}(\hat{S}) = \frac{T\sum_{t=1}^{T} t a_t - \left(\sum_{t=1}^{T} t\right)\left(\sum_{t=1}^{T} a_t\right)}{T\sum_{t=1}^{T} t^2 - \left(\sum_{t=1}^{T} t\right)^2} \tag{9}$$

$$\mathrm{EarlyAULC}(\hat{E}) = \frac{1}{K}\sum_{t=1}^{K} a_t \tag{10}$$

K is the number of first rounds to consider, with K equal to 20 used in this paper.

$$\mathrm{TotalVariationStability}(\hat{TV}) = \frac{1}{T-1}\sum_{t=2}^{T} |a_t - a_{t-1}| \tag{11}$$

$$\mathrm{PostPeakStability}(\hat{P}) = \sqrt{\frac{1}{T-p+1}\sum_{t=p}^{T} (a_t - \bar{a}_{\mathrm{post}})^2} \tag{12}$$

where p is the index for the peak accuracy and $\bar{a}_{\mathrm{post}}$ is the mean value of the accuracies from the peak index p.

F Experiments

Dataset	Clients	Alpha	Method	Score	Last10Mean	Early AULC	Trend Slope	EMA Final	T. V. Stability
RTE	3	0.1	RoLoRA	0.0425	0.5120	0.5118	0.0007	0.5120	0.0034
			FedAlign-LoRA	0.0315	0.5168	0.4790	0.0082	0.5156	0.0103
			FedIT	0.0253	0.5284	0.4992	0.0107	0.5296	0.0109
			FedRot-LoRA	-0.0880	0.5020	0.5120	0.0075	0.4950	0.0251
		0.5	FedIT	0.2690	0.6316	0.5120	0.0073	0.6348	0.0120
			FedAlign-LoRA	0.2148	0.6676	0.4954	0.0195	0.6688	0.0249
			RoLoRA	0.0568	0.5120	0.5122	0.0000	0.5120	0.0003
			FedRot-LoRA	0.0389	0.5120	0.5120	0.0024	0.5119	0.0017
		1.0	FedAlign-LoRA	0.5801	0.7988	0.5120	0.0074	0.7997	0.0054
			RoLoRA	0.5383	0.7704	0.5068	0.0095	0.7699	0.0069
			FedRot-LoRA	0.5280	0.7840	0.5182	0.0078	0.7838	0.0074
			FedIT	0.5089	0.7828	0.5104	0.0105	0.7848	0.0081
	10	0.1	FedIT	0.1103	0.5508	0.4720	0.0089	0.5507	0.0106
			FedAlign-LoRA	0.0742	0.5428	0.5120	0.0045	0.5409	0.0114
			RoLoRA	0.0380	0.5124	0.5120	0.0016	0.5126	0.0041
			FedRot-LoRA	-0.0817	0.4876	0.5176	0.0078	0.4887	0.0178
		0.5	FedAlign-LoRA	0.1764	0.6052	0.5120	0.0073	0.6070	0.0149
			FedIT	0.0556	0.5240	0.5120	0.0019	0.5232	0.0040
			RoLoRA	0.0543	0.5120	0.5120	0.0001	0.5120	0.0009
			FedRot-LoRA	0.0041	0.5120	0.5224	0.0032	0.5118	0.0098
		1.0	FedAlign-LoRA	0.3634	0.6692	0.4900	0.0078	0.6688	0.0076
			FedIT	0.2334	0.6136	0.5082	0.0029	0.6134	0.0086
			RoLoRA	0.0458	0.5120	0.5120	0.0005	0.5120	0.0026
			FedRot-LoRA	-0.0279	0.5032	0.5120	0.0047	0.5068	0.0162
CoLA	3	0.1	FedAlign-LoRA	0.5163	0.7866	0.6944	0.0067	0.7868	0.0087
			FedRot-LoRA	0.4882	0.7765	0.6983	0.0065	0.7766	0.0062
			FedIT	0.3978	0.7334	0.6944	0.0054	0.7331	0.0103
			RoLoRA	0.3018	0.7475	0.6965	0.0297	0.7482	0.0200
		0.5	FedRot-LoRA	0.6417	0.8535	0.7280	0.0046	0.8536	0.0021
			FedIT	0.6257	0.8534	0.7669	0.0046	0.8530	0.0037
			FedAlign-LoRA	0.6155	0.8463	0.7523	0.0041	0.8459	0.0038
			RoLoRA	0.6029	0.8370	0.7007	0.0057	0.8356	0.0050
		1.0	RoLoRA	0.6413	0.8493	0.7081	0.0028	0.8496	0.0021
			FedRot-LoRA	0.6169	0.8491	0.7369	0.0036	0.8501	0.0040
			FedIT	0.6135	0.8414	0.7409	0.0038	0.8414	0.0029
			FedAlign-LoRA	0.6118	0.8436	0.7401	0.0039	0.8430	0.0037

Table 8: Experimental Results across Different Datasets, Client Sizes, and Data Heterogeneity