RB-LoRA: Rank-Balanced Aggregation for Low-Rank Adaptation with Federated Fine-Tuning

Sihyeon Ha, Yongjeong Oh, Yo-Seb Jeon[*]

Department of Electrical Engineering, POSTECH, South Korea
{sihyeon.ha,yongjeongoh,yoseb.jeon}@postech.ac.kr

Abstract

Federated fine-tuning of foundation models is impeded by the need to communicate billions of parameters. Low-rank adaptation (LoRA) alleviates this by updating only compact adapter matrices. However, varying client device capabilities lead to different adapter ranks, causing rank heterogeneity that undermines aggregation, and existing reconciliation methods still incur bias or inefficiency. To address this challenge, we propose *RB-LoRA*, a principled rank-balanced aggregation framework that decomposes each update into rank-wise components and aligns them using analytically derived weights. Experiments on both language and vision models demonstrate consistent improvements under one and three rounds of communication in federated learning.[1]

1 Introduction

Foundation models have achieved state-of-the-art performance across a wide spectrum of tasks (Brown et al., 2020). However, the massive scale of modern foundation models introduces substantial computational and communication overhead, thereby rendering full-parameter updates impractical in federated learning (FL) environments (Wu et al., 2025b). To mitigate this bottleneck arising from deploying foundation models in FL, parameter-efficient fine-tuning (PEFT) techniques have been studied extensively (Houlsby et al., 2019; Pfeiffer et al., 2021; Zaken et al., 2022); among these, federated Low-rank adaptation (LoRA) has gained prominence by freezing pretrained weights and updating only low-rank adapters (Hu et al., 2022; Dettmers et al., 2023; Cai et al., 2023; Cho et al., 2024).

In federated LoRA, clients retain raw data locally and transmit only the gradients of their low-rank adapters, thereby preserving privacy and substantially reducing communication costs (Wu et al., 2025a). Despite these advantages, clients in practice often adopt different adapter ranks according to their computational capacities (Cho et al., 2024). This leads to *rank heterogeneity*, where client updates reside in distinct low-dimensional subspaces. Since these subspaces differ in both dimension and orientation, optimization for aggregating heterogeneous updates remains an open problem. Existing methods attempt to mitigate rank heterogeneity through zero-padding (Cho et al., 2024), replication (Byun and Lee, 2025), and stacking (Wang et al., 2024a). However, these methods rely on heuristic designs without analytical justification. As a result, they may unintentionally prioritize high-rank clients with small local datasets or underweight low-rank clients with large datasets, thereby degrading overall performance.

To address these limitations, we present **RB-LoRA**, a principled framework that formulates rank-balanced aggregation as a weighted-alignment optimization. By decomposing adapter updates in a rank-wise manner and deriving aggregation weights based on both rank and relative dataset size, **RB-LoRA** subsumes prior heuristic approaches and balances contributions from clients with disparate ranks.

Our **contributions** can be summarized as follows:

- We propose a unified weighted alignment framework for heterogeneous rank aggregation, which subsumes prior approaches.
- We develop a factorized weighting scheme grounded in data volume and rank rarity, providing analytical justification beyond heuristics.
- We validate our approach on federated LoRA for both LLMs and vision transformers.

2 Preliminaries

LoRA. LoRA injects trainable low-rank adapters into linear layers of a pretrained model, freezing

[*] Corresponding author.

[1] Code: https://github.com/seonha01/rb-lora/

Findings of the Association for Computational Linguistics: EACL 2026, pages 1737–1746
March 24-29, 2026 ©2026 Association for Computational Linguistics

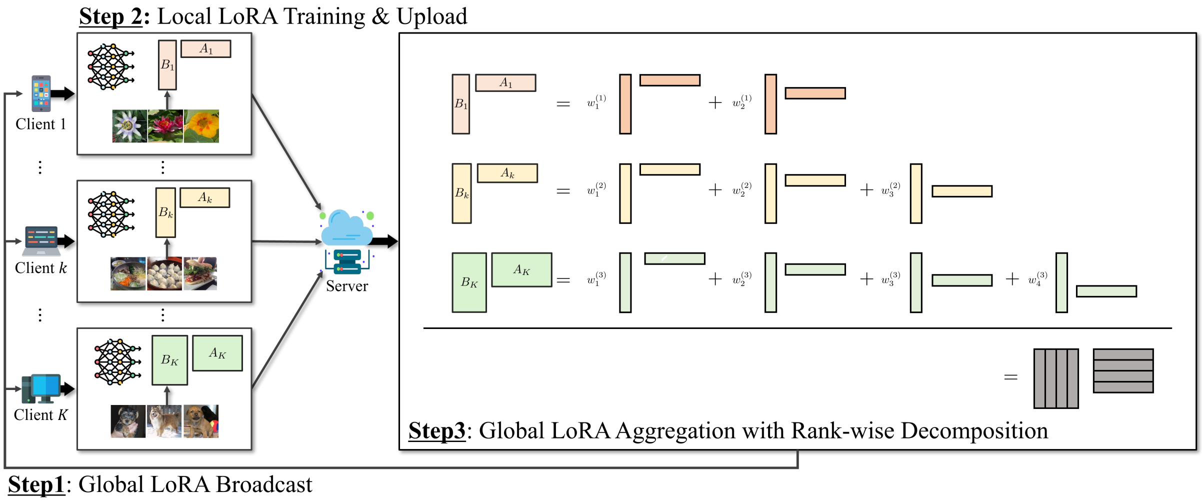

Figure 1: Overview of **RB-LoRA**

the original weights (Hu et al., 2022). Given a weight matrix $W_0 \in \mathbb{R}^{d\times d}$, LoRA represents it as

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

where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times d}$. This reduces the number of trainable parameters from d^2 to $2dr$.

Federated LoRA. In the federated setting, each of the K clients fine-tunes and transmits to the server only its adapter $(B^{(k)}, A^{(k)})$. The server aggregates the low-rank updates as

$$\Delta W_{\text{agg}} = \left(\frac{1}{K}\sum_k B^{(k)}\right)\left(\frac{1}{K}\sum_k A^{(k)}\right), \tag{2}$$

yielding the global adapter ΔW_{agg}, which serves as a unified LoRA module that reconciles information from all clients to improve generalization.

Rank Heterogeneity. In the federated LoRA setting, each client k selects its adapter rank r_k based on local capability, resulting in adapter matrices $B^{(k)} \in \mathbb{R}^{d\times r_k}$, $A^{(k)} \in \mathbb{R}^{r_k\times d}$ of various ranks. Because r_k can differ across clients, directly averaging the low-rank updates $B^{(k)}A^{(k)}$ is ill-posed. In what follows, we introduce three existing methods to reconcile these mismatched adapters.

Zero-Padding. (HETLoRA) A straightforward method is to pad smaller adapters with zeros so that all clients share the maximum rank across participants. This approach is simple and efficient but the inserted zeros dilute signals from high-rank clients—who possess richer representational capacity—biasing the global update toward lower-rank components (Cho et al., 2024).

Replication. Another method is to replicate existing adapter components until each client reaches the maximum rank. This avoids the dilution problem of zero entries and preserves all nonzero components, but it relies on a binary division of clients into high- and low-rank groups. Extending it to richer rank distributions is non-trivial, and the method often overemphasizes contributions from high-rank clients (Byun and Lee, 2025).

Stacking. (FLoRA) Stacking concatenates all client adapters along the rank dimension, preserving every client-specific direction. While this recovers the centralized update form, the resulting adapter rank grows to the sum of all local ranks, which is typically much larger than the maximum individual rank. Controlling this growth requires repeated rank reduction, adding significant overhead and undermining LoRA's parameter-efficiency benefits (Wang et al., 2024a).

Sketching. (FSLoRA) A related line of work addresses device heterogeneity through a sketching-based formulation, where each client activates only a subset of rank components using a diagonal selection matrix. Rather than explicitly reconciling adapters of different ranks, this approach controls the active subspace per client, implicitly inducing heterogeneity-aware updates. While effective for reducing computation and communication, the sketching operation performs a binary selection over rank components (Fang et al., 2025).

Discussion. Each aggregation method introduces its own bias and inefficiency, making direct comparisons challenging. Zero-padding enforces compatibility but dilutes signals from high-rank clients; replication preserves non-zero entries but assumes a binary split between high- and low-rank clients; stacking retains every client-specific direction but causes uncontrolled rank growth and costly compression. According to FedAvg (McMahan et al., 2017), client updates should be weighted by local dataset size to compensate for data heterogeneity, yet existing studies do not account for this. These limitations motivate a unified framework that treats rank and data heterogeneity consistently—an objective we pursue in Section 3.

3 Proposed RB-LoRA Framework

Rank-Wise Decomposition. Our **RB-LoRA** framework begins with a rank-wise decomposition of each client's LoRA update $\Delta W^{(k)}$. Specifically, $\Delta W^{(k)}$ can be expressed as a sum of r_k rank-one matrices, each formed as the outer product of two basis vectors:

$$\Delta W^{(k)} = \sum_{r=1}^{r_k} \mathbf{b}_r^{(k)} \, \mathbf{a}_r^{(k)\top}, \tag{3}$$

where $\mathbf{b}_r^{(k)}, \mathbf{a}_r^{(k)} \in \mathbb{R}^{d\times 1}$ are the rth column of $B^{(k)}$ and row of $A^{(k)}$, respectively.

Generalized Aggregation Representation. To unify heterogeneous LoRA updates, we align the two rank-r_k matrices $B^{(k)} \in \mathbb{R}^{d\times r_k}$ and $A^{(k)} \in \mathbb{R}^{r_k\times d}$ to a common rank $R = \max_k r_k$ via zero-padding:

$$P^{(k)} = \begin{bmatrix} I_{r_k} \\ \mathbf{0}_{(R-r_k)\times r_k} \end{bmatrix} \in \mathbb{R}^{R\times r_k}, \tag{4}$$

$$\tilde{B}^{(k)} = B^{(k)} P^{(k)\top} \in \mathbb{R}^{d\times R}, \tag{5}$$

$$\tilde{A}^{(k)} = P^{(k)} A^{(k)} \in \mathbb{R}^{R\times d}. \tag{6}$$

We then stack these aligned factors across clients to form $\tilde{B} = [\tilde{B}^{(1)}, \ldots, \tilde{B}^{(K)}]$ and $\tilde{A} = [\tilde{A}^{(1)}; \ldots; \tilde{A}^{(K)}]$. Accordingly, we formulate the aggregated update as

$$\Delta W_{\text{agg}} = \tilde{B}\, W\, \tilde{A},$$

where $W \in \mathbb{R}^{KR\times KR}$ encodes the weighting and alignment of client updates.

Unified Framework for the Existing Methods. Our **RB-LoRA** framework provides a unified formulation in which each prior method corresponds to a specific choice of the weighting matrix W.

- **Zero-Padding.** For client k, let $B^{(k)} \in \mathbb{R}^{d\times r_k}$ and $A^{(k)} \in \mathbb{R}^{r_k\times d}$, with local rank r_k and $R = \max_k r_k$. The original zero-padding scheme extends each adapter to R by appending zeros:

$$B_{\text{zp}}^{(k)} = \left[\, B^{(k)} \| \mathbf{0} \right], \; A_{\text{zp}}^{(k)\top} = \left[\, A^{(k)\top} \| \mathbf{0} \right]. \tag{7}$$

In our unified framework, this corresponds to

$$W_{\text{zp}} = \operatorname{diag}(\underbrace{\mathbf{1}_{R\times R}, \ldots, \mathbf{1}_{R\times R}}_{K \text{ times}}). \tag{8}$$

- **Replication.** Suppose $B^{(k)}$ has $\mathbf{b}_1^{(k)}, \cdots, \mathbf{b}_{r_k}^{(k)}$ and $A^{(k)}$ has $(\mathbf{a}_1^{(k)})^\top, \cdots, (\mathbf{a}_{r_k}^{(k)})^\top$. The replication method fills the missing dimensions by duplicating existing components:

$$B_{\text{rep}}^{(k)} = \left[B^{(k)} \| \mathbf{b}_{r_k+1}^{(\text{high})}, \cdots, \mathbf{b}_R^{(\text{high})}\right], \tag{9}$$

$$A_{\text{rep}}^{(k)\top} = \left[A^{(k)\top} \| \mathbf{a}_{r_k+1}^{(\text{high})\top} \cdots \mathbf{a}_R^{(\text{high})\top}\right]. \tag{10}$$

Within our framework, this is expressed as

$$W_{\text{rep}} = \operatorname{diag}(\mathbf{C}_{R\times R}^{(1)}, \ldots, \mathbf{C}_{R\times R}^{(K)}), \tag{11}$$

$$\mathbf{C}^{(k)} = \operatorname{diag}(\gamma_1^{(k)}, \ldots, \gamma_R^{(k)}) \cdot \mathbf{1}_{R\times R}, \tag{12}$$

or equivalently

$$\Delta W_{\text{rep}} = \sum_{k=1}^{K} \sum_{r=1}^{R} \gamma_r^{(k)} \, \mathbf{b}_r^{(k)} \mathbf{a}_r^{(k)\top}, \tag{13}$$

where $\gamma_r^{(k)} \in \{1, 2\}$ indicates whether a component is replicated.

This reformulation suggests that allowing $\gamma_r^{(k)}$ to vary continuously—rather than being restricted to $\{1, 2\}$—could yield more flexible and effective aggregation.

- **Stacking.** Stacking concatenates all client adapters along the rank dimension:

$$B_{\text{stack}} = \left[\, B^{(1)} \| \cdots \| B^{(K)} \right], \tag{14}$$

$$A_{\text{stack}}{}^\top = \left[A^{(1)\top} \| \cdots \| A^{(K)\top}\right], \tag{15}$$

$$\Delta W_{\text{agg}} = B_{\text{stack}} A_{\text{stack}}. \tag{16}$$

In the unified framework, this is equivalent to simply setting $W_{\text{stack}} = I_{KR}$.

- **Sketching.** A sketching-based method activates a subset of rank components via a diagonal selection matrix. In our framework, this corresponds to a block-diagonal weighting matrix

$$W_{\text{sk}} = \operatorname{diag}\big(\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(K)}\big), \tag{17}$$

$$\mathbf{S}^{(k)} = \operatorname{diag}(s_1^{(k)}, \ldots, s_R^{(k)}). \tag{18}$$

	Uniform HETLoRA		Weighted HETLoRA		FLoRA		**RB-LoRA**	
Dataset	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**
Dolly	0.53	0.51	0.54	0.52	0.26	0.26	**0.57**	**0.57**
Alpaca	0.52	0.52	0.51	0.52	0.31	0.31	**0.54**	**0.54**

Table 1: MMLU accuracy evaluated on a 1,444-question subset under 1- and 3-shot communication settings on LLaMa3-8b.

Method	**Common Avg.**	**Advanced Avg.**
Uniform HETLoRA	0.70	0.58
Weighted HETLoRA	0.70	0.58
FLoRA	0.35	0.17
RB-LoRA	**0.71**	**0.59**

Table 2: Comparison of zero-shot accuracy on grouped benchmarks on LLaMa3-8b.

Method	**#Params/round**	**Complexity**
Uniform HETLoRA	1.00×	$O(d^2KR)$
Weighted HETLoRA	1.00×	$O(d^2KR)$
FLoRA	1.57×	$O(dK^2R^2)$
RB-LoRA	1.00×	$O(d^2KR)$

Table 3: The number of transmitted parameters and the computational complexity of each method.

Proposed Aggregation Method. According to FedAvg (McMahan et al., 2017), client contributions in federated learning should scale with local dataset size. Continuous weighting of rank components provides maximal flexibility, but directly optimizing these weights for deep networks is intractable. We therefore construct weights from two factors—client data volume and rank rarity:

$$\gamma_r^{(k)} = \alpha_k \, \beta_r, \tag{19}$$

$$\alpha_k = \frac{|D^{(k)}|}{\sum_{j=1}^{K} |D^{(j)}|}, \tag{20}$$

$$\beta_r = \frac{|D_1|}{|D_r|}, \tag{21}$$

where $|D^{(k)}|$ is the number of examples held by client k, and $|D_r|$ is the total data of all clients with adapter rank at least r. This weighting scheme is not merely heuristic but a principled formulation that unifies prior reconciliation methods under a single framework; further derivation and justification are provided in Appendix A.

Projection. In the rank-heterogeneous setting, aggregating client adapters yields a global update whose effective rank is $R = \max_k r_k$. To proceed to the next round, each client k must receive an adapter with its own rank r_k. We therefore project W_{agg} onto a rank-r_k subspace using the SVD-based procedure of FlexLoRA (Bai et al., 2024): if $W_{\text{agg}} = U\Sigma V^{\top}$, we set $W_{\text{proj}}^{(k)} = U_{1:r_k}\Sigma_{1:r_k}V_{1:r_k}^{\top}$.

4 Experimental Setup

We evaluate **RB-LoRA** on language and vision models, with detailed settings in Appendix C.

Model and Datasets. We use the LLaMA2 (Touvron et al., 2023) as our base model. All experiments are conducted on Alpaca and Dolly (Taori et al., 2023; Databricks, 2023). The dataset is distributed across 10 simulated clients with non-IID splits, ranging from 500 to 5,000 examples per client. Details of the test datasets used for zero-shot reasoning evaluation are provided in Appendix F. While some prior studies assume a larger client population, the number of participants per round is typically limited to a small number of clients through client selection. We employ full participation with 10 clients to focus exclusively on the aggregation procedure.

LoRA Configuration. We attach LoRA adapters only to the query and key projection layers of the frozen base model. All clients share identical LoRA hyperparameters, and the adapter rank r for each client is selected from $\{4, 16, 64, 128, 256\}$ independently of dataset size. While prior work (Zhang et al., 2023b) explored adaptive rank allocation, our focus is orthogonal.

Aggregation Methods. We evaluate our **RB-LoRA** against three baseline—Uniform HETLoRA, Weighted HETLoRA, and FLoRA. FedIT framework provides a standardized FL fine-tuning setup ensuring consistent dataset partition and adapter placement (Zhang et al., 2024, 2023a). For fairness, we focused on baselines that concentrate solely on the aggregation procedure.

5 Results

We report the main evaluation results of RB-LoRA on language models. Results on vision transformers are presented in Appendix B. Rank allocation analysis is provided in Appendix E, and scaling to larger client populations is discussed in Appendix D. Additional experiments, including zero-shot reasoning as well as evaluations on other datasets and models, are reported in Appendix F.

As shown in Table 1, RB-LoRA consistently outperforms all baselines in both one-shot and three-shot settings, confirming the effectiveness of our weighting scheme in reconciling heterogeneous local updates. In contrast, FLoRA exhibits a marked drop in accuracy, as its aggregated LoRA module grows excessively in rank and the subsequent projection to each client's local rank causes notable information loss.

Table 2 reports zero-shot accuracy across grouped benchmarks. We separate results into common and advanced task groups, where RB-LoRA achieves consistently higher accuracy than other HETLoRA-based methods, demonstrating stronger generalization under heterogeneous client conditions.

Table 3 compares the number of transmitted parameters and the computational complexity of each method. FLoRA uses about 1.57× more global parameters and incurs higher aggregation complexity as the number of clients increases. Nonetheless, RB-LoRA achieves a better trade-off between accuracy and efficiency.

6 Conclusion

RB-LoRA introduces a rank-balanced aggregation framework that decomposes LoRA updates into rank-wise components and aligns with analytic weights, correcting rank heterogeneity. Experiments on language and vision models demonstrate consistent accuracy gains.

Limitations

Although RB-LoRA delivers strong empirical results, its closed-form weighting remains a heuristic rather than an optimal solution for the aggregation matrix W; end-to-end optimization or learning of these weights could further improve performance. In addition, our evaluation covers only a small set of language and vision benchmarks, and broader experimentation is required to validate the generality of our approach.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2024-00453301 and No. RS-2025-25414532)

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. 2024. Federated fine-tuning of large language models under heterogeneous tasks and client resources. *Advances in Neural Information Processing Systems*, 37:14457–14483.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Yuji Byun and Jaeho Lee. 2025. Towards federated low-rank adaptation of language models with rank heterogeneity. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 356–362.

Dongqi Cai, Yaozong Wu, Shangguang Wang, and Mengwei Xu. 2023. Fedadapter: Efficient federated learning for mobile nlp. In *Proceedings of the ACM Turing Award Celebration Conference-China 2023*, pages 27–28.

Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. 2024. Heterogeneous lora for federated fine-tuning of on-device foundation models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12903–12913.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Databricks. 2023. Databricks dolly 15k: Instruction-tuned dataset. https://github.com/databrickslabs/dolly. Accessed: 2025-07-01.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36:10088–10115.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051.

Wenzhi Fang, Dong-Jun Han, Liangqi Yuan, Seyyedali Hosseinalipour, and Christopher G Brinton. 2025. Federated sketching lora: On-device collaborative fine-tuning of large language models. *arXiv preprint arXiv:2501.19389*.

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

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. *KR*, 2012(13th):3.

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 487–503.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, and 1 others. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. 2024a. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. *Advances in Neural Information Processing Systems*, 37:22513–22533.

Ziyao Wang, Bowei Tian, Yexiao He, Zheyu Shen, Luyang Liu, and Ang Li. 2024b. One communication round is all it needs for federated fine-tuning foundation models. *arXiv preprint arXiv:2412.04650*.

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*.

Fei Wu, Jia Hu, Geyong Min, and Shiqiang Wang. 2025a. Adaptive rank allocation for federated parameter-efficient fine-tuning of language models. *arXiv preprint arXiv:2501.14406*.

Yebo Wu, Chunlin Tian, Jingguang Li, He Sun, Kahou Tam, Zhanting Zhou, Haicheng Liao, Zhijiang Guo, Li Li, and Chengzhong Xu. 2025b. A survey on federated fine-tuning of large language models. *arXiv preprint arXiv:2503.12016*.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Jianyi Zhang, Martin Kuo, Ruiyi Zhang, Guoyin Wang, Saeed Vahidian, and Yiran Chen. 2023a. Shepherd: A lightweight github platform supporting federated instruction tuning. https://github.com/JayZhang42/FederatedGPT-Shepherd.

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.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023b. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*.

A Proposed Aggregation Method

Weighting design and motivation. Selecting (α, β) inevitably involves heuristic choices; however, our design is grounded in the theoretical formulation of RB-LoRA. It directly follows from the weighted-alignment perspective, extending standard FedAvg arguments and rank-reconciliation principles. The resulting weighting rule is

$$w_r^{(k)} = \alpha_k \beta_r.$$

Data proportionality. We define

$$\alpha_k = \frac{|D^{(k)}|}{\sum_{k'} |D^{(k')}|}$$

following the FedAvg rule, ensuring statistically efficient aggregation within each rank group.

Rank-wise fairness. In LoRA, updates are decomposed into rank components. Without correction, rank-r decompositions contribute proportionally more components than lower-rank ones, causing imbalance. Equivalently, under naïve averaging, recurrent low-rank directions are overweighted; β_r compensates for this bias. Requiring equalized rank contribution,

$$\sum_{k \in K} \alpha_k \beta_r = c \quad \forall r,$$

yields

$$\beta_r = \frac{|D_1|}{|D_r|}.$$

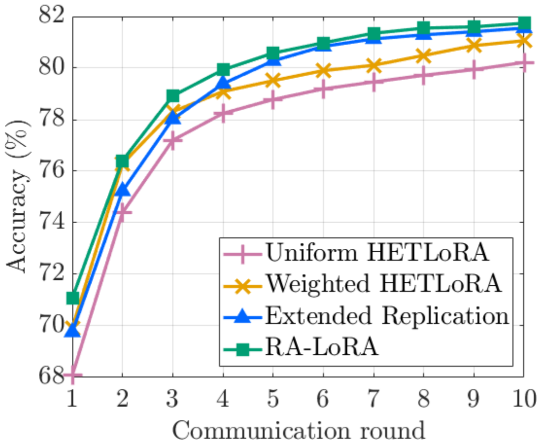

Figure 2: Top-1 accuracy over communication rounds for different aggregation methods on the Food-101 dataset using ViT backbone

Thus, $w_r^{(k)}$ ensures both dataset-proportional efficiency (via α) and fairness across rank components (via β). It reduces to FedAvg when all clients share the same rank. While not claiming formal optimality, this closed-form instantiation is theoretically motivated, yields consistent practical gains, and highlights the value of reinterpreting LoRA aggregation through a unified weighting perspective.

Additional derivation for clarity. Why α_k matches centralized training:

$$F(\mathbf{w}) = \frac{1}{\sum_{j=1}^{K} |D^{(j)}|} \sum_{k=1}^{K} |D^{(k)}| F_k(\mathbf{w}),$$

$$\nabla F(\mathbf{w}_t) = \sum_{k=1}^{K} \underbrace{\frac{|D^{(k)}|}{\sum_{j=1}^{K} |D^{(j)}|}}_{\alpha_k} \nabla F_k(\mathbf{w}_t).$$

This exactly matches the centralized gradient step (for gradients or sufficiently small local updates).

B Ablation on Vision Transformers

Model and Datasets. We extend our evaluation to visual classification by fine-tuning a vision transformer (Vaswani et al., 2017) backbone (pretrained on ImageNet) with LoRA adapters. To demonstrate modality-agnostic robustness, we conduct experiments on the Food-101 benchmark.

LoRA Configuration. For each sampled dataset, we partition the training set across four client groups, assigning adapter ranks of 2, 4, 8, and 16—together covering roughly 0.021% to 0.172% of the model's parameters. Clients perform local updates and aggregation for ten communication rounds, using the same hyperparameters (learning rate, batch size, etc.) as in our language-model experiments.

Results (Ablation Comparison). These experiments serve as a complete ablation across weighting components: *Uniform HETLoRA* (no α, β), *Weighted HETLoRA* (α-only), *Extended Replication* (β-only), and ***RB-LoRA*** (full $\alpha\beta$). As shown in Figure 2, **RB-LoRA** consistently accelerates convergence and improves final top-1 accuracy. Its linear communication and computation scaling—as opposed to the quadratic blow-up of stacking—enables efficient federated fine-tuning even on high-resolution or medical-image tasks.

C Experimental Setting

We describe additional implementation details used in the experiments reported in Section 4.

Training Setup. Each client trains with a local batch size of 32 and a micro batch size of 16. We use stochastic gradient descent (SGD) as the optimizer with a local learning rate of 0.0003 and apply linear learning rate decay. Training is performed for one local epoch per communication round. LoRA adapters are inserted into the query and value projection layers, with LoRA alpha set to 16 and dropout rate set to 0.05. Clients train on inputs only, without label supervision, and sequence length grouping is disabled.

Communication Setup. We conduct up to three communication rounds for each aggregation method. After each round, clients transmit their adapted parameters to the server for aggregation. While recent work suggests that a single communication round can suffice for federated fine-tuning of foundation models (Wang et al., 2024b), we evaluate up to three rounds to investigate how performance evolves with additional communication.

Hardware. All experiments are run on a machine equipped with four NVIDIA RTX 6000 Ada Generation GPUs, each with 48 GB of memory.

Evaluation Metrics. For evaluation, we use the MMLU benchmark (Hendrycks et al., 2020). We also measure perplexity (PPL) on WikiText-2 (Merity et al., 2016) and PTB (Marcus et al., 1993).

D Scaling to Larger Client Populations

To investigate scalability, we further increased the number of clients to 50. For this setting, we adopted the UltraChat dataset (Ding et al., 2023), which provides sufficient scale to support a larger federated configuration. Table 5 presents zero-shot reasoning accuracy and perplexity results under this large-client setup. *RB-LoRA* remains stable in this more challenging environment, highlighting its ability to handle both rank heterogeneity and larger client populations effectively.

E Rank Allocation Analysis

To examine whether the proposed aggregation mechanism of **RB-LoRA** operates orthogonally to rank allocation strategies, we conduct an additional study comparing two rank assignment schemes. In the first setting, the local rank r_k is allocated in proportion to the dataset size of each client—clients with more local data are assigned larger ranks, reflecting their greater contribution potential. In the second setting, ranks are randomly assigned regardless of dataset size. As shown in Table 4, RB-LoRA achieves comparable performance under both schemes, indicating that its aggregation principle remains robust and orthogonal to specific rank allocation policies.

F Additional Experimental Results

In this section, we provide supplementary experiments to further validate the effectiveness of RB-LoRA under diverse evaluation settings. We extend our analysis along two directions: (i) additional datasets for zero-shot reasoning and perplexity evaluation, and (ii) rank allocation analysis to test orthogonality.

F.1 Zero-Shot Reasoning and Perplexity

We evaluate zero-shot reasoning and language modeling performance to assess the generalization ability of RB-LoRA. For zero-shot reasoning evaluation, we employ a diverse set of benchmarks consisting of *common tasks* (ARC-E, BoolQ, HellaSwag, OpenBookQA, PIQA, Winograd) and *advanced tasks* (ARC-C, BBH, SciQ, MathQA) (Clark et al., 2018, 2019; Zellers et al., 2019; Mihaylov et al., 2018; Bisk et al., 2020; Levesque et al., 2012; Suzgun et al., 2022; Welbl et al., 2017; Amini et al., 2019). We additionally measure perplexity (PPL) on WikiText-2 (Merity

	Uniform HETLoRA	**Weighted HETLoRA**	**FLoRA**	***RB-LoRA***
MMLU (prop.)	30.37	33.78	25.34	35.52
MMLU (Rand.)	30.32	34.72	27.40	38.82
Common Avg. (prop.)	0.65	0.66	0.33	0.66
Common Avg. (Rand.)	0.66	0.66	0.39	0.67
Advanced Avg. (prop.)	0.50	0.51	0.17	0.52
Advanced Avg. (Rand.)	0.50	0.51	0.16	0.52

Table 4: Comparison of rank allocation strategies under a 10-client federated setup on LLaMa2-7b.

	Uniform HETLoRA	**Weighted HETLoRA**	**FLoRA**	***RB-LoRA***
PPL(Wikitext-2)	6.23	6.23	9111224306.69	6.32
PPL(PTB)	22.12	22.05	2817261908.48	21.74
ARC-E	0.71	0.71	0.25	0.69
BoolQ	0.76	0.76	0.30	0.73
HellaSwag	0.70	0.70	0.31	0.70
OpenBookQA	0.28	0.28	0.14	0.33
PIQA	0.79	0.80	0.46	0.78
Winogrande	0.69	0.69	0.44	0.72
Common Avg.	0.66	0.66	0.32	**0.66**
ARC-C	0.42	0.43	0.26	0.44
BBH	0.39	0.39	0.00	0.39
SciQ	0.93	0.93	0.18	0.95
MathQA	0.25	0.26	0.18	0.27
Advanced Avg.	0.50	0.51	0.16	**0.52**

Table 5: Perplexity (PPL) and zero-shot accuracy for various LoRA aggregation methods, fine-tuned on the **UltraChat** dataset with a 50-client federated setup on LLaMa2-7b.

	Uniform HETLoRA	**Weighted HETLoRA**	**FLoRA**	***RB-LoRA***
PPL(Wikitext-2)	7.29	7.29	3204986725.18	7.46
PPL(PTB)	11.74	11.72	9612003644.05	11.89
ARC-E	0.82	0.82	0.25	0.81
BoolQ	0.79	0.80	0.37	0.79
HellaSwag	0.68	0.68	0.34	0.69
OpenBookQA	0.31	0.30	0.13	0.38
PIQA	0.82	0.82	0.45	0.81
Winogrande	0.76	0.75	0.53	0.75
Common Avg.	0.70	0.70	0.35	**0.71**
ARC-C	0.43	0.43	0.25	0.50
BBH	0.62	0.62	0.00	0.62
SciQ	0.96	0.96	0.22	0.96
MathQA	0.31	0.32	0.20	0.29
Advanced Avg.	0.58	0.58	0.17	**0.59**

Table 6: Perplexity (PPL) and zero-shot accuracy for various LoRA aggregation methods, fine-tuned on the **Dolly** dataset with a 10-client federated setup on LLaMa3-8b.

et al., 2016) and PTB (Marcus et al., 1993). Tables 6, 7, 8 and 9 summarize the results. Overall, RB-LoRA achieves comparable perplexity to existing aggregation schemes while yielding consistent improvements in zero-shot accuracy across both common and advanced benchmarks, confirming its stronger generalization under heterogeneous client conditions.

	Uniform HETLoRA		Weighted HETLoRA		FLoRA		**RB-LoRA**	
Dataset	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**	**1-Shot**	**3-Shot**
Alpaca	0.31	0.30	0.34	0.35	0.27	0.27	**0.39**	**0.39**
Dolly	0.30	0.30	0.30	0.30	0.25	0.25	**0.35**	**0.35**

Table 7: MMLU accuracy evaluated on a 1,444-question subset under 1- and 3-shot communication settings.

	Uniform HETLoRA	**Weighted HETLoRA**	**FLoRA**	***RB-LoRA***
PPL(Wikitext-2)	6.24	6.29	58764.19	6.53
PPL(PTB)	22.11	21.95	63830.71	22.19
ARC-E	0.71	0.70	0.21	0.70
BoolQ	0.76	0.73	0.67	0.75
HellaSwag	0.70	0.71	0.30	0.70
OpenBookQA	0.28	0.33	0.14	0.36
PIQA	0.80	0.80	0.49	0.80
Winogrande	0.70	0.69	0.50	0.71
Common Avg.	0.66	0.66	0.39	**0.67**
ARC-C	0.43	0.43	0.24	0.44
BBH	0.39	0.39	0.00	0.39
SciQ	0.93	0.94	0.21	0.94
MathQA	0.26	0.27	0.19	0.30
Advanced Avg.	0.50	0.51	0.16	**0.52**

Table 8: Perplexity (PPL) and zero-shot accuracy for various LoRA aggregation methods, fine-tuned on the **Alpaca** dataset with a 10-client federated setup on LLaMa2-7b.

	Uniform HETLoRA	**Weighted HETLoRA**	**FLoRA**	***RB-LoRA***
PPL(Wikitext-2)	6.23	6.23	9111224306.69	6.32
PPL(PTB)	22.12	22.05	2817261908.48	21.74
ARC-E	0.71	0.72	0.20	0.74
BoolQ	0.76	0.76	0.30	0.78
HellaSwag	0.70	0.69	0.30	0.70
OpenBookQA	0.28	0.29	0.12	0.32
PIQA	0.78	0.79	0.48	0.79
Winogrande	0.70	0.69	0.47	0.73
Common Avg.	0.66	0.66	0.31	**0.68**
ARC-C	0.42	0.43	0.25	0.44
BBH	0.40	0.40	0.00	0.40
SciQ	0.93	0.93	0.19	0.93
MathQA	0.26	0.26	0.25	0.27
Advanced Avg.	0.50	0.50	0.17	**0.51**

Table 9: Perplexity (PPL) and zero-shot accuracy for various LoRA aggregation methods, fine-tuned on the **Dolly** dataset with a 10-client federated setup on LLaMa2-7b.