PRELORT: PREFIX-NESTED LORA FOR FEDERATED FINE-TUNING UNDER RANK HETEROGENEITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated fine-tuning of large language models using parameter-efficient methods such as LoRA enables adaptation of foundation models without directly sharing raw client data. Heterogeneous hardware resources introduce challenges, as clients with different adapter ranks cannot be directly aggregated. While existing methods enable aggregation under heterogeneous ranks, they fail to control how information is distributed across rank dimensions, leading to suboptimal use of shared low-rank representations. Instead, we propose **PreLort**: a nested low-rank formulation for federated LoRA that organizes adapter dimensions into a prefix hierarchy. Our approach encourages lower-rank dimensions to encode task-relevant information, while higher-rank dimensions capture additional capacity. Building on this, we introduce (i) a segment-wise aggregation rule that averages only over clients contributing to each rank segment, avoiding dilution from zero-padded lower-rank clients, and (ii) a prefix-nested training strategy that optimizes each adapter under multiple rank truncations, encouraging useful signal to concentrate in low-rank prefix dimensions. Together, these components encourage a consistent low-rank prefix capturing the most task-relevant information, while higher-rank dimensions learn additional capacity. This allows low-rank clients to benefit from richer information contributed by higher-rank clients, as prefix dimensions are consistently learned and aggregated. Experiments demonstrate that our method consistently outperforms prior heterogeneous federated LoRA methods in accuracy and ROUGE-L, while achieving lower or comparable perplexity across multiple base models.

1 INTRODUCTION

Federated learning (FL) enables collaborative training without sharing raw data (McMahan et al., 2017), but it does not inherently guarantee privacy, as model updates can leak sensitive information (Zhu et al., 2019; Nasr et al., 2019). Stronger privacy guarantees typically require additional techniques such as differential privacy or secure aggregation (Kairouz et al., 2021; McMahan et al., 2018; Bonawitz et al., 2017). The method in this paper addresses rank heterogeneity and is not a privacy mechanism: like standard FL, it avoids sharing raw data but does not prevent information leakage through the shared updates. Large language models (LLMs) are increasingly adapted to downstream tasks via fine-tuning, but their scale makes full-parameter training impractical in decentralized environments (Zhang et al., 2024a). To reduce computational and communication costs, parameter-efficient fine-tuning (PEFT) methods such as LoRA, prefix tuning, and prompt tuning have become the standard approach, learning low-rank updates while keeping the base model frozen (Hu et al., 2022; Houlsby et al., 2019; Pfeiffer et al., 2021; Lester et al., 2021) .

A key challenge in federated LoRA is *rank heterogeneity* (Cho et al., 2024; Wang et al., 2024). In realistic cross-device settings, clients operate under varying memory and compute constraints. As such, clients may initialize non-identical LoRA adapters, preventing direct aggregation of adapters. A naive solution is to enforce a uniform rank across clients. This is suboptimal as enforcing minimal rank across all clients will produce a model that does not have enough representational power. Recent works have provided solutions to this problem enabling aggregation across heterogeneous LoRA modules through zero-padding (Wang et al., 2024; Cho et al., 2024; Singhal et al., 2025). However, these methods focus primarily on how to combine updates algebraically, without considering how information is distributed across rank dimensions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This leads to a fundamental limitation: *heterogeneous ranks induce misaligned representations*. Even when aggregation is mathematically well-defined, combining the local updates that clients compute independently within a round may introduce inconsistencies due to misaligned representations or incompatible parameter updates, analogous to those observed when merging separately fine-tuned models (Wortsman et al., 2022; Ilharco et al., 2023; Matena & Raffel, 2022). As a result, lower-rank clients do not benefit from higher-rank updates.

In this work, we argue that addressing rank heterogeneity requires not only improved aggregation, but also *structured alignment of representations across ranks*. We introduce **PreLort**: a nested low-rank formulation that organizes LoRA adapter dimensions into a prefix structure, designed so that lower-rank dimensions capture task-relevant information and higher dimensions provide additional capacity (Kusupati et al., 2022; Cai et al., 2020; Yu et al., 2019; Yu & Huang, 2019). This structure encourages important information to concentrate hierarchically in prefix dimensions, enabling consistent aggregation across heterogeneous ranks.

Building on this formulation, we propose two key components. *First*, we design a ***nested training objective*** that explicitly trains multiple rank truncations within each client update, encouraging task-relevant signal to concentrate in shared low-rank prefixes hierarchically. In our experiments, this yields adapters whose importance concentrates in the leading dimensions and whose prefixes remain usable under truncation (Section 5.3). *Second*, we introduce a ***segment-wise aggregation rule*** that leverages this structure by aggregating adapter parameters only over the subset of clients that support each rank segment, avoiding zero-padding and preserving the contribution of higher-rank clients without introducing bias.

Our approach differs from prior work in that it jointly addresses representation alignment across rank dimensions, training dynamics, and aggregation under heterogeneous ranks. Averaging each parameter only over the clients that hold it is an established principle in width-heterogeneous FL (Horváth et al., 2021; Diao et al., 2021; Alam et al., 2022); our contribution is to couple it with all-prefix nested training, so that the LoRA rank coordinates being aggregated are consistently ordered across clients.

Our main contributions are summarized as follows:

- We identify representation misalignment across rank dimensions as a key limitation in heterogeneous federated LoRA.
- We propose a nested low-rank formulation that imposes a common prefix structure over adapter dimensions, and couple nested training with segment-wise aggregation in the update space so that adapters of different ranks can be aggregated without zero-padding.
- We empirically demonstrate consistent improvements in model quality and stability over prior heterogeneous federated LoRA methods on instruction-tuning benchmarks.

2 Related Work

Parameter-Efficient Fine-Tuning. PEFT methods adapt large language models by training a small set of parameters while keeping the backbone frozen, e.g., adapters (Houlsby et al., 2019; Pfeiffer et al., 2021), prompt and prefix tuning (Lester et al., 2021; Li & Liang, 2021), BitFit (Ben Zaken et al., 2022), and LoRA (Hu et al., 2022) with variants such as AdaLoRA (Zhang et al., 2023), DoRA (Liu et al., 2024), and LoFT (Tastan et al., 2026). These methods target centralized training and do not address the heterogeneous resource constraints and rank variability of federated learning.

Federated Fine-Tuning and Heterogeneous LoRA. Federated fine-tuning adapts large language models using decentralized data while avoiding direct data sharing. To reduce communication and computation costs, recent methods combine federated learning with parameter-efficient fine-tuning, especially LoRA. FedIT (Zhang et al., 2024a) applies LoRA with FedAvg for federated instruction tuning, but direct factor-wise averaging introduces aggregation noise because averaging LoRA factors does not correspond to averaging their induced updates. Recent work has also explored stronger privacy requirements in federated adaptation: BlindFed (Tastan & Nandakumar, 2025) considers a double-blind setting in which neither private client data nor the proprietary foundation model is directly exposed, using encrypted inference, low-rank parallel adapters, and secure aggregation. FLoRA (Wang et al., 2024) addresses heterogeneous LoRA aggregation with

stacking-based aggregation, which is exact in update space and naturally supports heterogeneous ranks. HetLoRA (Cho et al., 2024) further considers rank heterogeneity across clients through rank self-pruning and sparsity-weighted aggregation. FlexLoRA (Bai et al., 2024) aggregates client updates in the full weight space and uses SVD to redistribute rank-specific low-rank approximations to clients.

These methods enable heterogeneous LoRA aggregation, but they mainly focus on how to combine adapters after local training. In contrast, our work focuses on aligning the internal rank structure during training itself. By enforcing a prefix-nested organization of LoRA dimensions and aggregating rank segments only over contributing clients, PreLort encourages shared low-rank prefixes to capture task-relevant information while higher-rank dimensions provide additional capacity.

Slimmable Networks and Partial Aggregation. Slimmable networks, introduced by Yu et al. (2019), train a single model that can operate at multiple widths, enabling dynamic trade-offs between model capacity and efficiency. This idea has inspired follow-up work in federated learning, mainly for resource-aware training (Mei et al., 2022; Horváth et al., 2021), communication and computational efficiency (Wang et al., 2022), and neural architecture search (Yu & Huang, 2020). Recent work such as Aequa (Tastan et al., 2025) further leverages slimmable networks for fair federated learning by allocating model capacity (width) to participants based on their contributions. Model-heterogeneous FL methods such as FjORD (Horváth et al., 2021), HeteroFL (Diao et al., 2021), and FedRolex (Alam et al., 2022) train width-reduced sub-models and average each parameter only over the clients that hold it, and Tang et al. (2024) similarly avoid zero-padding when aggregating compressed gradients. These methods operate on identifiable width or channel coordinates, whereas LoRA factorizations are not identifiable: $BA = (BQ)(Q^{-1}A)$ for any invertible Q, so rank coordinates of separately updated factors are not automatically aligned across clients. Our contribution is therefore not support-aware averaging itself, but its coupling with all-prefix nested training in the LoRA update space. To the best of our knowledge, our work is the first to exploit this paradigm for addressing rank heterogeneity in federated LoRA.

Model Merging. Merging methods such as Task Arithmetic (Ilharco et al., 2023), TIES-Merging (Yadav et al., 2023), DARE (Yu et al., 2024), LoraHub (Huang et al., 2023), and LoRA-LEGO (Zhao et al., 2025) combine separately fine-tuned models or adapters. Appendix E.2 explains why we do not use them as direct baselines.

3 PRELIMINARIES

We consider K clients, where client k holds a private dataset $\mathcal{D}_k$ of size n_k and $N = \sum_k n_k$; the goal is to minimize $F(W) = \sum_k \frac{n_k}{N} F_k(W)$, where F_k is the local empirical loss. In each round, a subset of clients trains locally and the server aggregates their updates. LoRA (Hu et al., 2022) freezes a pre-trained weight $W_0 \in \mathbb{R}^{m\times n}$ and learns $\Delta W = BA$ with $B \in \mathbb{R}^{m\times r}$, $A \in \mathbb{R}^{r\times n}$, and $r \ll \min(m, n)$; the rank r trades off capacity and resource cost.

Aggregating LoRA factors. Each client trains (A_k, B_k) starting from the current global adapter and sends pseudo-gradients $\nabla A_k, \nabla B_k$, the change of its factors during local training. Averaging factors does not average the induced updates, since $\big(\sum_k \frac{n_k}{N} B_k\big)\big(\sum_k \frac{n_k}{N} A_k\big) \neq \sum_k \frac{n_k}{N} B_k A_k$ introduces cross-client terms $B_i A_j$ (Wang et al., 2024; Singhal et al., 2025). Exact aggregation is possible by stacking factors, at the cost of a rank that grows with the number of clients (Wang et al., 2024), or by aggregating in the full weight space (Bai et al., 2024). Rank heterogeneity adds a second problem: factors of different shapes must be aligned, typically by zero-padding low-rank clients (Cho et al., 2024), which dilutes the higher-rank dimensions. PreLort keeps a fixed-rank persistent adapter and targets this second problem, together with the misalignment of rank coordinates across clients (Section 4.3); like other factor-averaging methods, it does not remove cross-client terms.

4 METHOD

We address rank heterogeneity in federated LoRA by jointly designing a nested training strategy and a segment-wise aggregation rule. Let $r_1 < r_2 < \cdots < r_L$ be the distinct ranks used by clients

and $r_0 = 0$; we call the index range $[r_{l-1}, r_l)$ the l-th *rank segment*. The key idea is to encourage lower-rank dimensions to encode meaningful and shared representations across clients, while higher-rank dimensions capture additional capacity. To achieve this, we train each client to make its adapter useful under multiple truncations, encouraging task-relevant signal to concentrate in low-rank prefix dimensions. We then aggregate adapters in a segment-wise manner, combining only the clients that contribute to each rank segment. This coupling between training and aggregation enables consistent knowledge sharing across heterogeneous ranks.

4.1 Nested Low-Rank Structure

We impose a prefix-structured organization over adapter dimensions. For a client with rank r_k, only the first r_k dimensions are active. During local training, however, the adapter is optimized not only at full rank but also under multiple truncated configurations, where only prefix subsets of dimensions are used.

As a result, every prefix of a high-rank adapter is trained to be usable on its own. This does not partition information by rank in a semantic sense; rather, the i-th rank component participates in every prefix objective of width $r \geq i$, so lower-index components receive supervision from more prefix configurations. Empirically, this concentrates importance in the leading dimensions (Section 5.3.1) and keeps prefixes usable under truncation (Section 5.3.2). As illustrated in Figure 1, a higher-rank adapter naturally contains multiple lower-rank sub-adapters within its leading dimensions.

4.2 Local Training Strategy

To encourage prefix dimensions to encode meaningful representations, we train each client to perform well not only at its full rank but also at the lower rank levels that appear in the federation.

Let $\mathcal{R} = \{r_1 < r_2 < \cdots < r_L\}$ denote the set of distinct rank levels in the system. For a client with rank r_k, we define its active training ranks as

$$\mathcal{R}_k = \{r \in \mathcal{R} \mid r \leq r_k\}. \tag{1}$$

Thus, a client is trained at its full rank as well as all lower rank levels that may participate in aggregation with it.

For each $r \in \mathcal{R}_k$, we construct a truncated adapter by activating only the first r dimensions and masking the remaining dimensions. Denoting the corresponding model by $f(\cdot; \theta, r)$, where θ includes the shared frozen backbone and LoRA parameters, the local objective for client k is

$$\mathcal{L}_k = \frac{1}{|\mathcal{R}_k|} \sum_{r \in \mathcal{R}_k} \mathcal{L}_{\mathrm{CE}}\big(f(\cdot; \theta, r), \mathcal{D}_k\big), \tag{2}$$

where $\mathcal{L}_{\mathrm{CE}}$ denotes the standard cross-entropy loss.

In practice, each training step consists of one full-rank forward/backward pass and additional sub-rank passes for all $r \in \mathcal{R}_k \setminus \{r_k\}$. This training strategy ensures that each prefix segment is explicitly optimized at the same rank granularity used later during aggregation, encouraging task-relevant signal to concentrate in lower-rank dimensions. As a result, prefix segments remain informative and directly compatible with segment-wise aggregation across heterogeneous clients. The sub-rank passes run sequentially with in-place masking, so no extra adapter tensors are allocated and peak memory is unchanged; the cost is compute time, which grows with the client's rank (Eq. 1), so a client at the lowest rank level performs a single pass (Appendix E.5).

The total local objective is the average over all $|\mathcal{R}_k|$ sub-rank losses. This encourages the prefix dimensions to concentrate the most task-relevant signal, while higher dimensions encode incremental refinements, as depicted in Figure 1. The resulting adapter is thus trained to be usable at every prefix truncation, making it compatible with aggregation across clients of varying ranks.

4.3 Segment-wise Aggregation

Since clients contribute only to rank segments within their capacity, standard averaging across all clients requires aligning heterogeneous adapters, typically via zero-padding lower-rank adapters

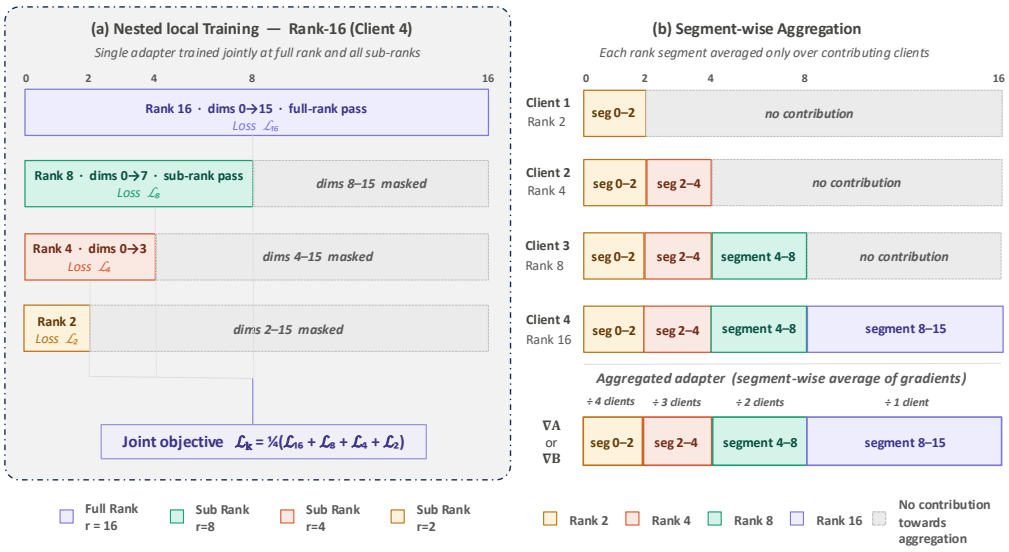


Figure 1: Overview of the proposed nested federated LoRA method using an illustrative example with ranks 2, 4, 8, 16. Left: nested local training optimizes a single adapter at full rank along with multiple prefix sub-ranks, encouraging lower-rank dimensions to capture task-relevant information through a joint objective. Right: segment-wise aggregation in the update space, where each client shares pseudo-gradients $(\nabla A_k, \nabla B_k)$ for its active rank segments. The server performs segment-wise averaging of these gradients over contributing clients and applies the aggregated updates to obtain the global adapter parameters $(A^{\text{global}}, B^{\text{global}})$.

to the maximum rank. This introduces zero-valued contributions in higher-rank segments for low-rank clients, diluting updates from high-rank clients. Instead, we aggregate each rank segment independently in the update space, averaging only over clients whose rank is sufficient to contribute to that segment.

Formally, for the l-th segment $[r_{l-1}, r_l)$, clients share pseudo-gradients ∇A_k and ∇B_k, and the server computes:

$$\nabla A^{\text{global}}_{r_{l-1}:r_l} = \frac{\sum_{k:\, r_k \geq r_l} n_k \, \nabla A^k_{r_{l-1}:r_l}}{\sum_{k:\, r_k \geq r_l} n_k}, \tag{3}$$

and analogously for $\nabla B^{\text{global}}_{r_{l-1}:r_l}$. The aggregated gradients are then applied to obtain A^{global} and B^{global}.

This aligns with training: each segment $[r_{l-1}, r_l)$ is optimized only by clients with rank at least r_l, so the same clients contribute during aggregation, ensuring consistency. Segments are determined solely by the rank levels present among clients; they are not assigned specific content, and the importance analysis in Section 5.3.1 is a post-hoc diagnostic rather than an input to segment construction.

As illustrated in Figure 1, segment 0–2 aggregates over all clients, 2–4 over three, 4–8 over two, and 8–16 over only the highest-rank client, preventing signal dilution while preserving meaningful lower-rank sharing.

PreLort has no method-specific convergence guarantee; Appendix E.1 relates its ingredients to existing results on ordered prefix training and nested sub-networks.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets and Setup. We evaluate on two standard instruction-tuning benchmarks: **Alpaca** (52K examples) (Taori et al., 2023) and **databricks-dolly-15k** (Zhang et al., 2024a), as well as a classification dataset, **20 Newsgroups** (Mitchell, 1997).

We evaluate both classification and generation performance. Specifically, we report accuracy on MMLU (Dolly, Alpaca) and 20 Newsgroups, and Rouge-L on instruction-tuning benchmarks, as shown in Table 1. We additionally report perplexity to measure token-level likelihood, as shown in Table 2.

We evaluate on two base models: **TinyLlama-1.1B** (Zhang et al., 2024b) and **Qwen2.5-0.5B** (Qwen Team et al., 2025), with a preliminary Qwen2.5-7B (Qwen Team et al., 2025) experiment in Appendix F.3.

Implementation Details. For both TinyLlama-1.1B and Qwen2.5-0.5B, LoRA modules are applied to self-attention layers only following Hu et al. (2022). Local training uses the AdamW optimizer with a learning rate of 3×10^{-4}; rounds, local epochs, and batch sizes are listed in Appendix C. Clients are sampled uniformly at random in each round, and the datasets are uniformly partitioned between clients using a fixed random split. To test robustness to data heterogeneity, we additionally partition 20 Newsgroups across clients with a Dirichlet distribution ($\alpha = 0.1$). Each category's training articles are split across clients with Dir($\alpha = 0.1$) proportions, and client ranks keep the same fixed assignment as in the IID setting, independent of each client's data. All experiments are conducted on NVIDIA RTX A6000 GPUs.

Baselines. We compare against representative heterogeneous federated LoRA methods: **ZeroPad** (Zhang et al., 2024a; Wang et al., 2024), which applies FedAvg with zero-padding to align heterogeneous ranks; **HetLoRA** (Cho et al., 2024), which uses rank self-pruning with sparsity-weighted aggregation; **FLoRA** (Wang et al., 2024), which employs stacking-based aggregation for noise-free updates; and **FlexLoRA** (Bai et al., 2024), which aggregates client updates in the full weight space and redistributes rank-specific approximations via SVD.

We also report a **homogeneous reference curve**, where all clients use a fixed rank $r \in \{1, 2, 4, 8, 16, 32, 64, 128\}$. This provides a reference for understanding the effective rank–performance trade-off under uniform capacity. This powers-of-two grid is a standard log-scale sweep for studying the rank–performance trade-off and is not required by the method.

In the heterogeneous LoRA setting, we follow (Wang et al., 2024) and adopt their rank configuration for the 10-client case, assigning local ranks [64, 32, 16, 16, 8, 8, 4, 4, 4, 4]. For the 5-client setting, we use a similar rank pattern, i.e., [16, 8, 4, 2, 1], to simulate heterogeneous computational resources. Nested training evaluates all valid prefix sub-ranks for each batch.

5.2 MAIN RESULTS

Table 1 presents the main comparison across heterogeneous federated LoRA methods in terms of accuracy and ROUGE-L.

Across both TinyLlama-1.1B and Qwen2.5-0.5B, our method (PreLort) consistently achieves the best performance on MMLU (Dolly, Alpaca) and ROUGE-L metrics. In particular, it outperforms all baselines on both MMLU tasks across models, and achieves the highest ROUGE-L scores on Dolly and Alpaca. On 20 Newsgroups, PreLort remains competitive, achieving the second-best performance while outperforming most baselines. Against FlexLoRA, PreLort achieves higher accuracy in all six model–dataset settings, by 1.02 to 6.98 percentage points. These results indicate that our approach improves instruction-following and generalization performance without sacrificing robustness across tasks. When the Qwen2.5-0.5B/Dolly setting is repeated over multiple seeds, PreLort again has the highest mean accuracy (39.15 ± 0.58), ahead of FLoRA (38.84 ± 0.50) and HetLoRA (38.40 ± 0.57) (Appendix F.1).

Table 1: Evaluation of heterogeneous federated LoRA methods across base models. Accuracy is reported for MMLU (Dolly, Alpaca) and 20 Newsgroups classification, while Rouge-L is reported for instruction-tuning benchmarks (Dolly, Alpaca). Higher is better.

Foundation Model	Method	Accuracy			ROUGE-L	
		MMLU				
		Dolly	Alpaca	20 Newsgroups	Dolly	Alpaca
TinyLlama-1.1B	ZeroPad	29.1	26.72	14.68	27.34	32.1
	HetLoRA	29.9	29.09	17.01	28.13	33.27
	FLoRA	28.42	29.19	**34.08**	28.72	33.21
	FlexLoRA	26.88	27.58	31.33	28.49	32.87
	Nested Aggregation Only	29.63	28.63	29.48	28.16	28.76
	Nested Training Only	28.73	30.21	31.68	26.65	33.2
	PreLort	**33.86**	**30.76**	32.47	**29.34**	**33.79**
Qwen2.5-0.5B	ZeroPad	38.36	33.1	16.95	27.07	30.49
	HetLoRA	38.88	34.59	30.90	27.23	32.47
	FLoRA	38.84	34.51	32.94	28.33	34.22
	FlexLoRA	38.83	34.06	38.23	28.52	34.01
	Nested Aggregation Only	38.32	32.71	**39.93**	28.42	30.6
	Nested Training Only	39.33	34.66	28.22	26.53	34.43
	PreLort	**39.85**	**35.76**	39.52	**28.79**	**35.03**

Table 2: Perplexity comparison of heterogeneous federated LoRA methods on instruction-tuning benchmarks using TinyLlama-1.1B and Qwen2.5-0.5B. Lower is better.

Method	ZeroPad	HetLoRA	FLoRA	FlexLoRA	Nested Agg. Only	Nested Train. Only	PreLort
TinyLlama 1.1B							
Dolly	4.5902	4.5589	4.4407	4.4700	4.9115	4.5492	**4.4030**
Alpaca	2.6822	2.6822	**2.6230**	2.6500	3.1084	2.6699	2.6379
NewsGroup	7.1240	6.9611	6.5561	6.5400	6.6924	6.6105	**6.4899**
Qwen-2.5-0.5B							
Dolly	7.0482	7.0276	6.8025	**6.7000**	7.6927	6.9159	6.8156
Alpaca	2.8185	2.7983	2.6774	3.3800	2.6695	3.4447	**2.6378**
NewsGroup	11.6069	11.5412	11.4055	**11.0000**	11.9582	11.3023	11.2229

Table 2 reports perplexity on instruction-tuning datasets. PreLort achieves the lowest perplexity in three of the six settings (TinyLlama on Dolly and 20 Newsgroups, Qwen2.5 on Alpaca), the second lowest in two, and the third lowest on Qwen2.5/Dolly, where FlexLoRA and FLoRA are lower. Compared with FlexLoRA, PreLort has lower perplexity in four of the six settings. Compared to FLoRA, which benefits from stacking-based aggregation that increases effective rank, PreLort achieves comparable or better perplexity while maintaining a fixed parameter budget, avoiding growth in adapter size as the number of clients increases (Wang et al., 2024).

Capacity and communication relative to FLoRA. FLoRA is not capacity- or communication-matched to the other methods. It stacks all participating adapters each round, giving a per-round update of rank $\sum_{k \in S_t} r_k$ (160 in our 10-client setting, 31 in the 5-client setting) and an accumulated rank of up to $\min\big(m, n, \sum_{t=1}^{T} \sum_{k \in S_t} r_k\big)$ after T rounds, and each client downloads the stacked adapter. PreLort keeps a persistent adapter bounded by the largest client rank, so its per-client download does not grow with the number of clients (Appendix E.3). FLoRA's stronger results in some settings may therefore partly reflect its larger capacity; Appendix F.2 includes a variant restricted to a per-round effective rank of 64.

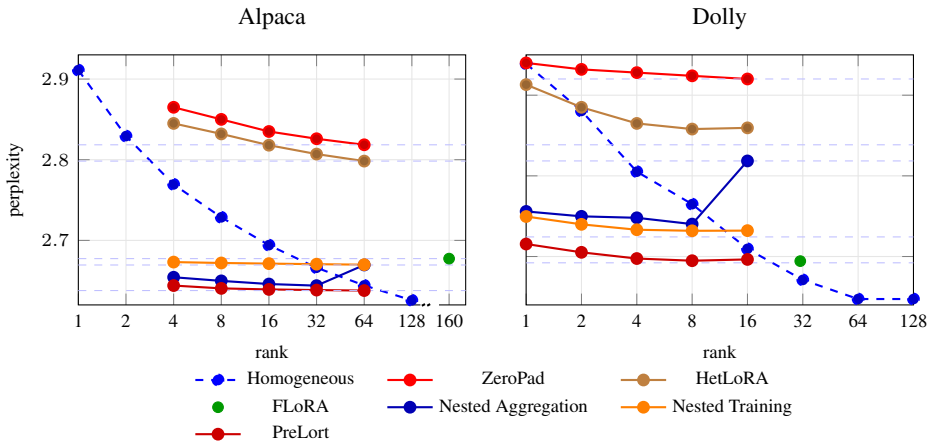


Figure 2: Comparison of homogeneous and heterogeneous methods on Alpaca and Dolly. For heterogeneous methods, each curve shows performance under prefix rank truncation: the rightmost point corresponds to the full trained rank, and points to the left reflect evaluation at progressively smaller prefix sub-ranks of the same trained adapter. FLoRA is shown as a single point at its per-round effective rank (160 for 10 client setting, 31 for 5 client setting), as it merges adapters into the backbone after each round and does not maintain a persistent adapter amenable to truncation

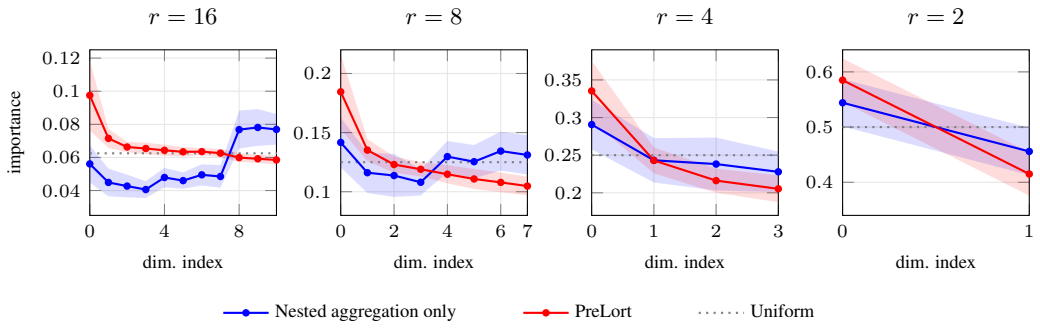


Figure 3: Per-dimension importance across rank budgets. Nested aggregation only and our method are shown for ranks 16, 8, 4, and 2. Shaded regions denote one standard deviation, and the dotted line denotes the uniform importance baseline $1/r$.

Non-IID data. Under a Dirichlet ($\alpha = 0.1$) partition of 20 Newsgroups over three seeds, PreLort achieves the best mean accuracy and perplexity on both backbones, improving accuracy by 1.84 (Qwen2.5-0.5B) and 3.28 (TinyLlama-1.1B) points over capacity-controlled FLoRA, and by 4.28 and 16.74 points over HetLoRA (Appendix F.2).

Overall, the results demonstrate that (i) segment-wise aggregation effectively utilizes heterogeneous rank capacity without dilution, and (ii) the nested training objective is critical for aligning representations across ranks. The ablation (Nested Aggregation Only and Nested Training Only) confirms that aggregation alone is insufficient, and that performance gains arise from the combination of both components.

5.3 Analysis

5.3.1 Per-Dimension Importance

To better understand how nested training affects the internal structure of the adapter, we analyze the per-dimension importance of each rank index i. We define importance as the product of the Frobenius norm of the i-th row of A and the i-th column of B, normalized across all dimensions. This provides a proxy for the contribution of each rank component to the overall update $\Delta W = BA$.

As shown in Figure 3, without nested training, importance is much less concentrated: it stays closer to uniform at small ranks and, at $r = 16$, even shifts toward the higher dimensions. In contrast, our approach produces a strongly structured distribution, where importance is concentrated in the leading dimensions. This behavior suggests that the adapter learns an implicit prioritization over rank components. We read this as an empirically observed concentration of importance, consistent with the asymmetric optimization pressure described in Section 4.1, rather than a guaranteed semantic partition of information across ranks. Such a structure is particularly beneficial for nested aggregation. Low-rank clients contribute to these high-impact dimensions, while higher-rank clients provide incremental refinements, leading to more stable and effective updates.

5.3.2 Rank Robustness

Figure 2 further illustrates the rank–performance trade-off. It shows the performance of each method across different effective rank levels. The homogeneous reference improves steadily as rank increases, as expected. In contrast, our method remains stable across the lower-rank regime and has lower perplexity than the other heterogeneous methods that keep a persistent adapter on both Alpaca and Dolly; FLoRA, at a per-round effective rank of 31, is slightly lower on Dolly.

Appendix E.4 describes how FLoRA, which does not keep a persistent adapter, is placed in Figure 2; the other heterogeneous curves truncate the same final aggregated adapter to each prefix rank.

The *Nested Aggregation only* baseline exhibits unstable behavior, particularly at higher ranks, indicating that aggregation alone is insufficient. Without nested training, importance is not aligned with the aggregation structure: higher-rank dimensions can dominate locally but are supported by only a subset of clients during aggregation. As illustrated in Figure 3, these dimensions exhibit high importance despite limited support, leading to overfitting and instability.

In contrast, our nested training encourages a hierarchical ordering of importance, concentrating task-relevant signal in shared prefix dimensions. This acts as an implicit regularizer and yields stable performance across rank levels.

5.4 Limitations

The effectiveness of the proposed nested training strategy depends on sufficient local optimization steps; with limited local epochs, the model may not fully learn well-aligned low-rank prefix representations. Nested training also increases local compute time by about 40% (peak memory is unchanged), and our single-GPU simulation does not capture distributed round time or stragglers. PreLort is not a privacy mechanism, and most of our experiments use IID partitions and backbones below 1.5B parameters. Appendix E.5 discusses these and further limitations in detail.

6 Conclusion

We introduced PreLort, a federated LoRA fine-tuning method designed for rank-heterogeneous clients. The core idea is to make heterogeneous ranks compatible not just at aggregation time, but during training itself. To this end, nested training optimizes each adapter under multiple prefix truncations, encouraging task-relevant information to concentrate in shared low-rank dimensions. Segment-wise aggregation then averages each rank segment only over clients that actually contribute to it, avoiding zero-padding dilution while preserving useful higher-rank capacity.

Across Alpaca, Dolly, and 20 Newsgroups with TinyLlama-1.1B and Qwen2.5-0.5B, PreLort consistently improves accuracy and ROUGE-L over heterogeneous LoRA baselines, while achieving lower or comparable perplexity; its advantage holds under non-IID client data. The ablation and per-dimension analysis further show that aggregation alone is not enough: the gains come from coupling segment-wise aggregation with nested training.

Overall, our results show that rank heterogeneity in federated LoRA is best handled by jointly structuring how adapters are trained and how their updates are aggregated.

AI USE STATEMENT

We used generative AI tools to rephrase text and correct grammar, to help draft and revise some passages, to assist with LaTeX syntax and reference formatting, and to help create diagrams. We did not use generative AI tools to generate research ideas, implement the method, design or run experiments, or analyse results. The authors reviewed and verified all AI-assisted content and take full responsibility for the final content of this work.

ETHICS STATEMENT

This work proposes a training and aggregation method for federated LoRA fine-tuning with heterogeneous adapter ranks. A potential benefit is that clients with limited compute or memory can take part in federated adaptation with smaller adapters while still benefiting from updates contributed by better-resourced clients. The method does not provide privacy guarantees: clients share pseudo-gradients of their adapter parameters, which can leak information about local data through reconstruction or inference attacks (Zhu et al., 2019; Nasr et al., 2019). Deployments that handle sensitive data should combine it with mechanisms such as differential privacy (McMahan et al., 2018) or secure aggregation (Bonawitz et al., 2017). Models fine-tuned with PreLort also inherit biases present in the pretrained backbones and in the fine-tuning datasets. All models and datasets we use are publicly available.

REPRODUCIBILITY STATEMENT

Algorithms 1 and 2 specify local training and aggregation, Appendix C lists all hyperparameters, rounds, and local epochs, and Section 5.1 describes the data partitions, including the non-IID construction. All models and datasets are publicly available. We will release our code upon acceptance.

REFERENCES

Samiul Alam, Luyang Liu, Ming Yan, and Mi Zhang. Fedrolex: Model-heterogeneous federated learning with rolling sub-model extraction. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=OtxyysUdBE.

Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. Federated fine-tuning of large language models under heterogeneous tasks and client resources. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=gkOzoHBXUw.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.1. URL https://aclanthology.org/2022.acl-short.1/.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pp. 1175–1191, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349468. doi: 10.1145/3133956.3133982. URL https://doi.org/10.1145/3133956.3133982.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once for all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/pdf/1908.09791.pdf.

Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. Heterogeneous LoRA for federated fine-tuning of on-device foundation models. In Yaser Al-Onaizan, Mohit Bansal, and

Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 12903–12913, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.717. URL https://aclanthology.org/2024.emnlp-main.717/.

Enmao Diao, Jie Ding, and Vahid Tarokh. Hetero{fl}: Computation and communication efficient federated learning for heterogeneous clients. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=TNkPBBYFkXg.

Samuel Horváth, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Donald Lane. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=4fLr7H5D_eT.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Chao Du, Tianyu Pang, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic loRA composition. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=lyRpY2bpBh.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14(1–2):1–210, June 2021. ISSN 1935-8237. doi: 10.1561/2200000083. URL https://doi.org/10.1561/2200000083.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=9njZa1fm35.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th*

Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Shih-yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=3d5CIRG1n2.

Michael S Matena and Colin Raffel. Merging models with fisher-weighted averaging. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=LSKlp_aceOC.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJ0hF1Z0b.

Yiqun Mei, Pengfei Guo, Mo Zhou, and Vishal Patel. Resource-adaptive federated learning with all-in-one neural composition. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=wfel7CjOYk.

Tom Mitchell. Twenty Newsgroups. UCI Machine Learning Repository, 1997. DOI: https://doi.org/10.24432/C5C323.

Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 739–753, 2019. doi: 10.1109/SP.2019.00065.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 487–503, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.39. URL https://aclanthology.org/2021.eacl-main.39/.

Qwen Team, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Raghav Singhal, Kaustubh Ponkshe, and Praneeth Vepakomma. FedEx-LoRA: Exact aggregation for federated and efficient fine-tuning of large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1316–1336, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.67. URL https://aclanthology.org/2025.acl-long.67/.

Zichen Tang, Junlin Huang, Rudan Yan, Yuxin Wang, Zhenheng Tang, Shaohuai Shi, Amelie Chi Zhou, and Xiaowen Chu. Bandwidth-aware and overlap-weighted compression for communication-efficient federated learning. In *ICPP*, pp. 866–875, 2024. URL https://doi.org/10.1145/3673038.3673142.

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

Nurbek Tastan and Karthik Nandakumar. A framework for double-blind federated adaptation of foundation models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 923–933, October 2025.

Nurbek Tastan, Samuel Horváth, and Karthik Nandakumar. Aequa: Fair model rewards in collaborative learning via slimmable networks. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Tw81RElDpe.

Nurbek Tastan, Stefanos Laskaridis, Martin Takáč, Karthik Nandakumar, and Samuel Horváth. LoFT: Low-rank adaptation that behaves like full fine-tuning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=86P3sb1dpr.

Hui-Po Wang, Sebastian U Stich, Yang He, and Mario Fritz. Progfed: Effective, communication, and computation efficient federated learning by progressive training, 2022. URL https://openreview.net/forum?id=Gpp1dfvZYYH.

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. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=TcCorXxNJQ.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xtaX3WyCj1.

Jiahui Yu and Thomas Huang. Autoslim: Towards one-shot architecture search for channel numbers, 2020. URL https://openreview.net/forum?id=H1gz_nNYDS.

Jiahui Yu and Thomas S. Huang. Universally slimmable networks and improved training techniques. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1803–1811, 2019. URL https://api.semanticscholar.org/CorpusID:76660361.

Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1gMCsAqY7.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=fq0NaiU8Ex.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Yufan Zhou, Guoyin Wang, and Yiran Chen. Towards building the federated gpt: Federated instruction tuning, 2024a. URL https://arxiv.org/abs/2305.05644.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model, 2024b. URL https://arxiv.org/abs/2401.02385.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Ziyu Zhao, Tao Shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, and Fei Wu. Merging loRAs like playing LEGO: Pushing the modularity of loRA to extremes through rank-wise clustering. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=j6fsbpAllN.

Ligeng Zhu, Zhijian Liu, and Song Han. *Deep leakage from gradients*. Curran Associates Inc., Red Hook, NY, USA, 2019.

A ALGORITHM

Algorithm 1 PreLort: Nested Local Training

Require: Client k with rank r_k, local dataset $\mathcal{D}_k$, global rank set $\mathcal{R} = \{r_1 < r_2 < \cdots < r_L\}$, adapter parameters $(\mathbf{A}_k, \mathbf{B}_k) \in \mathbb{R}^{r_k \times n} \times \mathbb{R}^{m \times r_k}$, learning rate η, local epochs E
Ensure: Updated adapter parameters $(\mathbf{A}_k, \mathbf{B}_k)$, pseudo-gradients $(\nabla\mathbf{A}_k, \nabla\mathbf{B}_k)$

$\mathcal{R}_k \leftarrow \{r \in \mathcal{R} \mid r \le r_k\}$ ▷ Active training ranks for client k
for $e = 1$ to E **do**
  for each minibatch $\mathcal{B} \subseteq \mathcal{D}_k$ **do**
    $\mathcal{L}_k \leftarrow 0$
    for each $r \in \mathcal{R}_k$ **do**
      Construct truncated adapter: $\mathbf{A}_k^{(r)} \leftarrow \mathbf{A}_k[:r, :],\ \mathbf{B}_k^{(r)} \leftarrow \mathbf{B}_k[:, :r]$
      $\mathcal{L}_k \leftarrow \mathcal{L}_k + \mathcal{L}_{\mathrm{CE}}(f(\cdot;\, \theta,\, r),\, \mathcal{B})$
    end for
    $\mathcal{L}_k \leftarrow \frac{1}{|\mathcal{R}_k|}\,\mathcal{L}_k$ ▷ Average over sub-rank losses
    $(\mathbf{A}_k, \mathbf{B}_k) \leftarrow (\mathbf{A}_k, \mathbf{B}_k) - \eta\,\nabla_{(\mathbf{A}_k, \mathbf{B}_k)}\,\mathcal{L}_k$
  end for
end for
Compute pseudo-gradients: $\nabla\mathbf{A}_k \leftarrow \mathbf{A}_k^{(0)} - \mathbf{A}_k,\ \nabla\mathbf{B}_k \leftarrow \mathbf{B}_k^{(0)} - \mathbf{B}_k$
return $(\nabla\mathbf{A}_k, \nabla\mathbf{B}_k)$

Algorithm 2 PreLort: Segment-wise Aggregation

Require: Pseudo-gradients $\{(\nabla\mathbf{A}_k, \nabla\mathbf{B}_k)\}_{k\in\mathcal{K}}$ from participating clients, client ranks $\{r_k\}_{k\in\mathcal{K}}$, dataset sizes $\{n_k\}_{k\in\mathcal{K}}$, rank levels $\mathcal{R} = \{r_1 < r_2 < \cdots < r_L\}$ with $r_0 = 0$, global adapter $(\mathbf{A}^{\mathrm{global}}, \mathbf{B}^{\mathrm{global}})$
Ensure: Updated global adapter $(\mathbf{A}^{\mathrm{global}}, \mathbf{B}^{\mathrm{global}})$

for $l = 1$ to L **do**
  $\mathcal{K}_l \leftarrow \{k \in \mathcal{K} \mid r_k \ge r_l\}$ ▷ Clients contributing to segment l
  $N_l \leftarrow \sum_{k\in\mathcal{K}_l} n_k$
  $\nabla\mathbf{A}^{\mathrm{global}}_{r_{l-1}:r_l} \leftarrow \dfrac{1}{N_l}\sum_{k\in\mathcal{K}_l} n_k\,\nabla\mathbf{A}_k[r_{l-1}:r_l,\, :]$
  $\nabla\mathbf{B}^{\mathrm{global}}_{r_{l-1}:r_l} \leftarrow \dfrac{1}{N_l}\sum_{k\in\mathcal{K}_l} n_k\,\nabla\mathbf{B}_k[:,\, r_{l-1}:r_l]$
end for
Assemble $\nabla\mathbf{A}^{\mathrm{global}}$ and $\nabla\mathbf{B}^{\mathrm{global}}$ from all segments
$\mathbf{A}^{\mathrm{global}} \leftarrow \mathbf{A}^{\mathrm{global}} - \nabla\mathbf{A}^{\mathrm{global}}$
$\mathbf{B}^{\mathrm{global}} \leftarrow \mathbf{B}^{\mathrm{global}} - \nabla\mathbf{B}^{\mathrm{global}}$
return $(\mathbf{A}^{\mathrm{global}}, \mathbf{B}^{\mathrm{global}})$

B ENVIRONMENTS, DATASETS, AND METRIC

Computer Resources. All experiments were run on NVIDIA RTX A6000 GPUs.

Dolly dataset. The Dolly dataset is an open-source dataset containing 15k text samples generated by Databricks employees. Topics covered include brainstorming, classification, closed QA, generation, information extraction, open QA, and summarization.

Alpaca dataset. The Alpaca dataset contains 52K instruction-following data samples used for fine-tuning language models. The dataset is designed to be diverse enough for fine-tuning LLMs.

20 Newsgroup dataset. The 20 Newsgroup dataset is a widely used benchmark collection of approximately 20,000 news documents partitioned across 20 different categories. It is commonly used for text classification tasks.

Evaluation Metrics. We evaluate model performance using three metrics:

- **Accuracy:** Each model is fine-tuned on the Dolly, Alpaca, and 20 Newsgroup datasets respectively, and evaluated on the corresponding held-out test set.
- **ROUGE-L:** Measured on the held-out test sets for the Dolly and Alpaca datasets to assess the quality of generated text.
- **Perplexity:** Also computed on the held-out test sets for Dolly and Alpaca to evaluate language modeling performance.

C Hyperparameter Details

In all our experiments, the batch size is 128 and the micro batch size is 16. We explored various learning rates (5×10^{-3}, 1×10^{-4}, 3×10^{-4}, 1×10^{-5}, 5×10^{-5}) and found that 3×10^{-4} yielded the best overall performance. Table 3 summarizes the communication rounds and local epochs used for each experiment.

Table 3: Communication rounds and local epochs per experiment. Rounds denotes the number of federated communication rounds, and Epochs denotes the number of local fine-tuning epochs per round.

Foundation	Dataset	Rounds	Epochs
TinyLlama	Dolly	10	3
	Alpaca	10	3
	20 Newsgroup	15	1
Qwen2.5-0.5B	Dolly	10	3
	Alpaca	10	3
	20 Newsgroup	15	1

D Perplexity Convergence

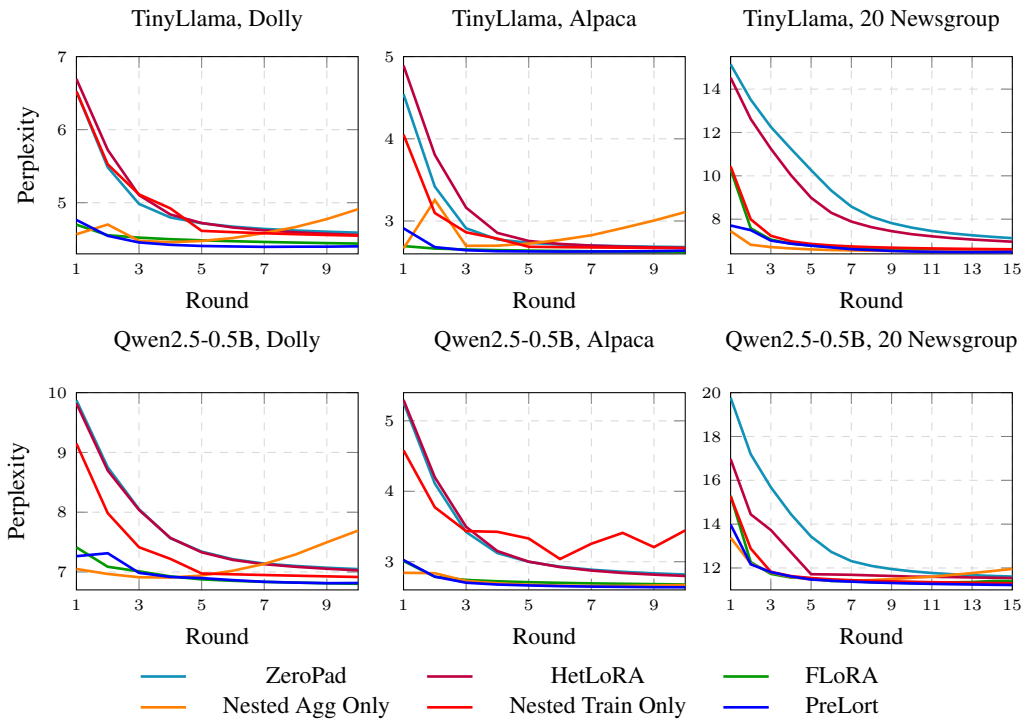


Figure 4: Perplexity convergence across communication rounds for TinyLlama and Qwen2.5-0.5B on Dolly, Alpaca, and 20 Newsgroup datasets.

Figure 4 shows the perplexity convergence of all methods across communication rounds on both models and all three datasets. PreLort consistently achieves competitive or lower final perplexity compared to baselines, though the visual gap is diminished due to the high initial perplexity of ZeroPad and HetLoRA dominating the y-axis scale. Nested Aggregation Only diverges over rounds on most settings, confirming that segment-wise aggregation alone is insufficient without the corresponding local training strategy. Similarly, Nested Training Only exhibits poor or sometimes unstable convergence, suggesting that segment-wise local training must be coupled with the corresponding aggregation strategy to achieve stable performance.

E Extended Discussion

E.1 Theoretical Context

PreLort does not come with a method-specific convergence or optimality guarantee, but its ingredients connect to existing results. FjORD shows that, for linear models, ordered prefix training recovers the optimal truncated SVD solution, i.e., each prefix is the best approximation of its rank in ℓ_2 distance (Horváth et al., 2021). Aequa shows that restricting training to nested sub-networks yields a smoothness constant no larger than that of the full model, so the standard Local-SGD convergence guarantee carries over (Tastan et al., 2025). Extending these results to segment-wise aggregation is left for future work.

E.2 Model Merging

Merging methods combine separately fine-tuned models or adapters. Task Arithmetic (Ilharco et al., 2023) adds task vectors; TIES-Merging (Yadav et al., 2023) and DARE (Yu et al., 2024) reduce interference by trimming, sign resolution, or random dropping of update entries; LoraHub (Huang et al., 2023) and LoRA-LEGO (Zhao et al., 2025) compose LoRA modules across tasks. In the update space, Task Arithmetic reduces to a weighted sum of client updates, which FLoRA's stacking already represents exactly. TIES and DARE are not invariant to LoRA reparameterization, since the signs and magnitudes of the factors change under $(B, A) \mapsto (BQ, Q^{-1}A)$; applying them here would require materializing each dense update $\Delta W_k = B_k A_k$, merging in the dense space, and recompressing with SVD. LoraHub and LoRA-LEGO target post-hoc, often cross-task composition rather than iterative federated optimization. We therefore do not use these methods as direct baselines (Appendix E.5).

E.3 Capacity and Communication Relative to FLoRA

FLoRA is not strictly capacity- or communication-matched to the other methods. In each round it stacks the adapters of all participating clients S_t, giving a per-round update of rank $\sum_{k \in S_t} r_k$ (160 in our 10-client setting and 31 in the 5-client setting). Because this update is merged into the backbone and fresh adapters are initialized in the next round, the accumulated update after T rounds can have rank up to $\min\big(m, n, \sum_{t=1}^{T} \sum_{k \in S_t} r_k\big)$. Client uploads are comparable across methods, since each client sends its own rank-r_k adapter, but each FLoRA client downloads the stacked global adapter of rank $\sum_{k \in S_t} r_k$. PreLort instead keeps a persistent adapter whose rank is bounded by the largest client rank (64 in the 10-client setting); total traffic still grows with the number of participating clients. FLoRA's stronger results in some settings may therefore partly reflect its larger effective capacity. To separate the two effects, Appendix F.2 includes a FLoRA variant whose per-round effective rank is restricted to 64.

E.4 Evaluation Protocol for Figure 2

For **FLoRA**, adapter updates are merged into the frozen backbone after each communication round, and fresh adapters are initialized in subsequent rounds. As a result, there is no persistent aggregated adapter at the end of training, and we report a single perplexity value corresponding to the final merged model. We place FLoRA at its per-round effective rank, the sum of client ranks: 160 for the Alpaca (10-client) setting and 31 for the Dolly (5-client) setting; its accumulated rank over rounds can be larger (Appendix E.3).

In contrast, methods such as ZeroPad, HetLoRA, Nested Aggregation, and PreLort maintain an explicit aggregated adapter after each round of aggregation on the server. This allows us to evaluate rank robustness by truncating the final aggregated adapter to prefix ranks at evaluation time. The reported curves for these methods therefore reflect the performance of the same trained adapter under different truncation levels.

E.5 Detailed Limitations

Nested training adds computation because of the extra forward/backward passes per batch. In end-to-end wall-clock time on our single-GPU setup, PreLort takes 32.7% (20 Newsgroups), 39.6% (Dolly), and 46.6% (Alpaca) longer than the average of the baselines, about 39.6% on average (Appendix F.4). Peak memory is unchanged because sub-rank passes reuse the adapter through in-place masking, and the overhead grows with a client's rank, so the lowest-rank clients incur none. Training only a subset of the prefix ranks below a client's rank could reduce this cost; we have not evaluated it.

Our experiments simulate federated training on a single GPU with synchronous rounds, so the wall-clock numbers measure local computation rather than distributed round time. In a real deployment, faster clients wait for the slowest participating client, and the local overhead becomes higher round latency. Segment-wise aggregation averages each segment over the participating clients that hold it and is therefore compatible with partial participation, but we have not evaluated asynchronous training or stale updates.

PreLort is not a privacy mechanism. It avoids sharing raw data, but its pseudo-gradients may leak information as in standard FL (Zhu et al., 2019; Nasr et al., 2019), and we do not evaluate privacy attacks or combine the method with defenses such as differential privacy (McMahan et al., 2018) or secure aggregation (Bonawitz et al., 2017).

Finally, we do not include an adapted merging baseline (e.g., dense TIES or DARE followed by SVD recompression), and PreLort has no method-specific convergence guarantee. Apart from the 20 Newsgroups experiment in Appendix F.2, all results use IID partitions, and its three-seed comparison covers only HetLoRA and capacity-controlled FLoRA. Apart from the preliminary Qwen2.5-7B experiment in Appendix F.3, which compares all methods for a single seed and only FlexLoRA over three seeds, all backbones are below 1.5B parameters.

F Additional Results

F.1 Multi-Seed Results on Qwen2.5-0.5B / Dolly

Table 4 repeats the Qwen2.5-0.5B/Dolly setting of Table 1 over three seeds. Table 1 reports a single run, so its values differ from these means; PreLort has the highest accuracy in both.

Table 4: MMLU accuracy (%) on Qwen2.5-0.5B/Dolly, mean $\pm$ std over three seeds.

Method	**Accuracy**
ZeroPad	36.71 ± 2.25
HetLoRA	38.40 ± 0.57
FLoRA	38.84 ± 0.50
Nested Aggregation Only	36.95 ± 1.12
Nested Training Only	38.21 ± 0.64
FlexLoRA	38.79 ± 0.61
PreLort	$\mathbf{39.15 \pm 0.58}$

F.2 Non-IID Client Data

We partition 20 Newsgroups across the 10 clients with a Dirichlet distribution ($\alpha = 0.1$) and repeat each run over three seeds (Table 5). The three-seed runs cover HetLoRA and FLoRA restricted to a per-round effective rank of 64, which matches PreLort's capacity. PreLort achieves the best mean accuracy and perplexity on both backbones. On Qwen2.5-0.5B, it improves accuracy by 4.28

points over HetLoRA and by 1.84 points over capacity-controlled FLoRA; on TinyLlama-1.1B, the corresponding gains are 16.74 and 3.28 points.

Table 5: Non-IID 20 Newsgroups (Dirichlet $\alpha = 0.1$, 10 clients), mean $\pm$ std over three seeds. Further baselines, including FlexLoRA, are reported for a single seed in Table 6.

Backbone	Method	Accuracy (%) ↑	Perplexity ↓
Qwen2.5-0.5B	HetLoRA	19.92 ± 2.90	11.5337 ± 0.0824
	FLoRA (rank 64)	22.36 ± 1.11	11.3314 ± 0.0783
	PreLort	$\mathbf{24.20 \pm 1.01}$	$\mathbf{11.1876 \pm 0.0890}$
TinyLlama-1.1B	HetLoRA	18.46 ± 2.35	6.8925 ± 0.0571
	FLoRA (rank 64)	31.92 ± 0.88	6.5886 ± 0.0438
	PreLort	$\mathbf{35.20 \pm 0.60}$	$\mathbf{6.5170 \pm 0.0168}$

Table 6 reports a single-seed run of the non-IID 20 Newsgroups setting (Dirichlet $\alpha = 0.1$) on Qwen2.5-0.5B with all baselines, including FLoRA with and without the capacity restriction. Unconstrained FLoRA (per-round effective rank 160) obtains the highest accuracy and lowest perplexity in this run; restricted to a per-round effective rank of 64, the maximum client rank, it falls below PreLort on both metrics.

Table 6: Non-IID 20 Newsgroups (Dirichlet $\alpha = 0.1$), Qwen2.5-0.5B, single seed.

Method	Perplexity ↓	Accuracy (%) ↑
ZeroPad	11.44	18.55
HetLoRA	11.49	22.48
FLoRA (per-round effective rank 160)	11.32	24.97
FLoRA (per-round effective rank 64)	11.50	22.06
FlexLoRA	11.39	22.59
PreLort	11.37	23.04

F.3 Preliminary Results on Qwen2.5-7B

We run Qwen2.5-7B on 20 Newsgroups in non-IID setting with the 10-client rank configuration [64, 32, 16, 16, 8, 8, 4, 4, 4, 4]. These results are preliminary. Table 7 compares all methods for a single seed after two communication rounds: PreLort obtains the highest accuracy and ties FlexLoRA for the lowest perplexity. Table 8 compares PreLort with FlexLoRA over three seeds after two communication rounds: PreLort reaches $50.51 \pm 2.49\%$ accuracy versus $45.46 \pm 2.32\%$ for FlexLoRA, with slightly lower perplexity.

Table 7: Qwen2.5-7B on 20 Newsgroups, single seed, two communication rounds.

Method	Accuracy (%) ↑	Perplexity ↓
ZeroPad	28.3	8.27
HetLoRA	29.2	8.17
FLoRA	47.7	7.44
FlexLoRA	53.3	**7.40**
PreLort	**54.0**	**7.40**

Table 8: Qwen2.5-7B on 20 Newsgroups, mean $\pm$ std over three seeds.

Method	Accuracy (%) ↑	Perplexity ↓
FlexLoRA	45.46 ± 2.32	7.4144 ± 0.0115
PreLort	$\mathbf{50.51 \pm 2.49}$	$\mathbf{7.4029 \pm 0.0033}$

F.4 Wall-Clock Overhead of Nested Training

Table 9 reports the additional end-to-end training time of PreLort relative to the average total training time of the baselines. The absolute difference grows with dataset size. The extra time comes from the sub-rank passes in the nested objective (Eq. 2); peak memory is unchanged.

Table 9: Additional end-to-end training time of PreLort over the baseline average on a single GPU across all baselines.

Dataset	**Additional time (h)**	**Increase**
20 Newsgroups	+1.24	+32.7%
Dolly	+2.10	+39.6%
Alpaca	+3.10	+46.6%
Average	≈2.15	≈39.6%