HALO: Heterogeneous-Aware LoRA Optimization via Rank Allocation and Client-Aware Projection

Xiaohua Feng
Zhejiang University
Hangzhou, China
fengxiaohua@zju.edu.cn

Yuyuan Li
Hangzhou Dianzi University
Hangzhou, China
y2li@hdu.edu.cn

Jiayuan Fang
Hangzhou Dianzi University
Hangzhou, China
fangjiayuan@hdu.edu.cn

Fengyuan Yu
Zhejiang University
Hangzhou, China
miller_yu@outlook.com

Jiaming Zhang
Zhejiang University
Hangzhou, China
jm.zhang@zju.edu.cn

Li Zhang
Zhejiang University
Hangzhou, China
zhanglizl80@gmail.com

Xiang Liu
OPPO Internet Services System
Shenzhen, China
liuxiang10@oppo.com

Jun Wang*
OPPO Research Institute
Shenzhen, China
junwang.lu@gmail.com

Chaochao Chen
Zhejiang University
Hangzhou, China
zjuccc@zju.edu.cn

Abstract

Federated Learning (FL) with Low-Rank Adaptation (LoRA) enables privacy-preserving collaborative fine-tuning of foundation models. Existing federated LoRA studies often assume homogeneous ranks and lack principled rank allocation, leading to underfitting or overfitting under data and resource heterogeneity. Although later heterogeneous-rank federated LoRA methods relax the homogeneous-rank assumption, they still ignore resource-adaptive rank allocation across clients, while their aggregation and distribution strategies suffer from structural limitations, including rank collapse in naive truncation and the loss of locally relevant update signals caused by objective mismatch in SVD truncation. To address these issues, we propose HALO, a heterogeneous federated LoRA framework that integrates adaptive rank allocation with client-aware aggregation and distribution. Under the neural tangent kernel regime, we derive a sufficient rank condition to guide client-specific rank allocation from local data geometry and resource constraints. We further propose Client-Aware Projection (CAP), which builds a product-summed reference and redistributes rank-compatible balanced LoRA factors through projections onto client-induced subspaces, improving generalization via cross-client information while preserving locally relevant update signals. We show that CAP controls the client-loss-relevant truncation residual under the NTK approximation without additional communication overhead, which stabilizes local training and improves performance. Extensive experiments demonstrate the effectiveness of HALO. Our code is available at https://github.com/BanBan25.

1 Introduction

Foundation Models (FMs) have demonstrated strong performance in generative tasks and are widely deployed in domains such as virtual assistants [13], healthcare [41, 44], and search engines [28].

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

However, adapting pre-trained FMs to downstream tasks typically requires full-parameter fine-tuning, which is computationally prohibitive. To mitigate this issue, Parameter-Efficient Fine-Tuning (PEFT) methods such as prompt tuning [32], adapter tuning [23], and Low-Rank Adaptation (LoRA) [24] have been proposed. LoRA is particularly popular because it freezes the pre-trained parameters and introduces lightweight trainable adapters $\mathbf{A}$ and $\mathbf{B}$, updating weights via $\mathbf{\Delta W} = \mathbf{BA}$, thereby enabling efficient adaptation at significantly reduced cost.

Beyond computational constraints, fine-tuning FMs relies on high-quality data that is often distributed across institutions or devices, making direct aggregation a significant privacy concern [21, 33, 16]. To address this issue, Federated Learning (FL) has been integrated with PEFT, particularly LoRA, by keeping the pre-trained model frozen and deploying tunable LoRA modules on both the server and the clients [50, 6, 15, 30]. By locally updating LoRA parameters and aggregating globally, FL enables collaborative fine-tuning while preserving data privacy.

Prior work on federated LoRA has largely assumed homogeneous settings, where all clients use LoRA modules with identical ranks and average the adapters $\mathbf{A}$ and $\mathbf{B}$ independently [42]. However, this assumption poses two key challenges in practice: **i) Data heterogeneity.** Clients differ in data volume and task complexity. High-rank modules may cause overfitting for clients with limited data, while a small fixed rank restricts representational capacity for complex tasks, introducing structural bias. **ii) Resource heterogeneity.** Clients have imbalanced computational capacities, and uniform rank settings can make resource-constrained clients bottlenecks, reducing overall efficiency. Recent studies have relaxed the homogeneous assumption by allowing heterogeneous LoRA ranks and designing rank-compatible aggregation and distribution strategies, such as naive truncation after partial summation [9, 53] and SVD truncation after product summation [2, 3, 34], as shown in Fig. 1. However, these efforts focus primarily on aggregation and lack theoretical guidance for rank assignment, limiting local representational capacity. Moreover, existing aggregation and distribution strategies still have structural limitations, i.e., NT suffers from rank collapse while SVD suffers from objective mismatch, failing to preserve locally relevant heterogeneous update signals during rank-compatible redistribution. To bridge this gap, we focus on two fundamental questions: **Q1:** How can LoRA ranks be adaptively allocated according to each client's local data geometry and resource constraints? **Q2:** How can heterogeneous LoRA updates be aggregated and distributed while preserving locally relevant update signals under client-specific rank constraints?

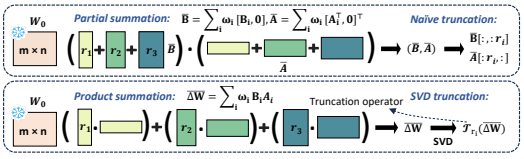


Figure 1: Overview of current federated heterogeneous LoRA aggregation methods; the rightmost part shows the model update at each round.

To address these questions, we propose a heterogeneous federated LoRA framework, `HALO`, which combines adaptive rank allocation with client-aware aggregation and distribution. We analyze client-side LoRA fine-tuning through the Neural Tangent Kernel (NTK) lens, formulating it as empirical risk minimization over a linearized model. Under the NTK regime, we use a dimension-counting argument based on Sard's lemma to derive a sufficient rank condition for each client, which characterizes *how many LoRA degrees of freedom are needed to avoid harmful second-order stationary points in the local surrogate objective.* This condition allows `HALO` to adaptively allocate LoRA ranks according to local data geometry and resource constraints, mitigating overfitting while preserving sufficient representational capacity (addressing **Q1**). For heterogeneous-rank LoRA aggregation and distribution, we identify two structural limitations of existing methods: NT suffers from rank collapse due to repeated compression of higher-rank dimensions, while SVD suffers from objective mismatch because its global spectral truncation can discard directions that are locally relevant to individual clients. We therefore propose Client-Aware Projection (CAP), which first forms a product-summed reference from uploaded LoRA factors and then projects this reference onto each client-induced update subspace before redistribution, preserving locally relevant heterogeneous update signals (addressing **Q2**). We further show that, under the NTK approximation and a client-subspace coverage condition, CAP controls the client-relevant residual induced by redistribution, while incurring no additional communication overhead beyond standard LoRA-factor upload and download.

The main contributions of this work are summarized as follows:

1. We propose HALO, a heterogeneous federated LoRA framework that combines adaptive rank allocation with client-aware aggregation and distribution.
2. We derive an NTK-based sufficient rank condition, providing theoretical guidance for assigning client-specific LoRA ranks under data and resource heterogeneity.
3. We identify rank collapse in NT and objective mismatch in SVD, and propose CAP to redistribute rank-compatible LoRA factors while preserving locally relevant update signals.

2 Related Works

2.1 PEFT in Federated Learning

PEFT techniques have been extensively integrated into FL to reduce communication overhead and improve training efficiency. Common approaches include collaborative training of adapters [7, 39] or prompt vectors [20], as well as utilizing large FMs for few-shot or zero-shot learning [47, 55]. However, each strategy has inherent limitations. Adapter-based collaboration often involves structurally complex and numerous modules, increasing communication and storage costs [54, 5]. Prompt-based approaches suffer from non-linear semantics in prompt vectors, leading to misalignment when averaged across clients and resulting in degraded performance [40, 37]. Few-shot and zero-shot approaches, while reducing reliance on labeled data, are highly sensitive to data distribution and task variation, limiting their effectiveness in Non-Independent and Identically Distributed (Non-IID) federated settings [43, 2]. In contrast, LoRA's low-rank decomposition offers structural simplicity and linearity, enabling efficient cross-client aggregation and improved robustness to data heterogeneity [19, 42]. These advantages have positioned LoRA as the predominant choice for integrating with FL.

2.2 LoRA in Federated Learning

Recent studies have begun exploring the integration of LoRA into FL. [54] assume homogeneous LoRA modules across clients and enabled cross-client parameter sharing via independent averaging of the trainable low-rank matrices $\mathbf{A}$ and $\mathbf{B}$. Subsequent work by [1] examines the effect of LoRA initialization strategies on federated training, while [51] leverage LoRA for personalization under Non-IID data. However, these methods primarily rely on naive averaging of homogeneous modules and thus fall short in addressing the practical challenges of data and resource heterogeneity. To overcome these limitations, recent efforts have explored heterogeneous federated LoRA, allowing clients to adopt varying rank configurations, followed by aggregation through naive truncation after partial summation [9, 53], or SVD truncation after product summation [2, 3, 34], as shown in Fig. 1. While these approaches relax the homogeneous assumption, their rank allocation remains heuristic and lacks theoretical justification. Additionally, the proposed aggregation strategies are not optimal and have theoretical flaws. Motivated by these gaps, we introduce a framework that systematically addresses both adaptive rank allocation and client-aware aggregation and distribution.

3 Preliminaries

In this section, we first establish the basic notations, then formally describe the FL problem, and finally introduce the foundations of LoRA in the federated setting.

Neural network. Let $f_\Theta : \mathcal{X} \to \mathbb{R}^k$ denote a neural network parameterized by Θ (e.g., a Transformer-based model). The input dataset is denoted by $\mathcal{X}$ (e.g., natural language text or images), and the output space is denoted by $\mathbb{R}^k$ (e.g., next-token prediction or image classification softmax logits). Here, k represents the output dimension of f_Θ; when using Mean Squared Error (MSE) loss, k also corresponds to the dimension of the label set $\mathcal{Y}$. We assume the model has been pretrained with parameters Θ_0, and we denote the pretrained model as f_{Θ_0}.

Federated learning setting. We consider a standard FL framework consisting of a central server and M clients. The local dataset of the i-th client is denoted by D_i, where $X_i = \{x \mid (x, y) \in D_i\}$, $Y_i = \{y \mid (x, y) \in D_i\}$, represent the input set and label set of the i-th client, respectively. Here, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Furthermore, we define the dataset size of the i-th client be N_i.

LoRA in federated learning. Let $\mathbf{W_0} \in \mathbb{R}^{m \times n}$ denote a weight matrix from the pretrained model. LoRA updates the weight via an additional low-rank decomposition branch: $\mathbf{W} = \mathbf{W_0} + \mathbf{\Delta W} =$

$\mathbf{W_0} + \mathbf{BA}$, where $\mathbf{A} \in \mathbb{R}^{r\times n}$ and $\mathbf{B} \in \mathbb{R}^{m\times r}$ are trainable low-rank matrices. Since $\mathbf{W_0} \subset \Theta_0$, for simplicity, we denote $f_{\mathbf{W}}$ as the model where all parameters except $\mathbf{W}$ are fixed to their pretrained values in Θ_0. Furthermore, let the LoRA module on the i-th client be denoted as $\mathbf{\Delta W_i} = \mathbf{B_i A_i}$ with rank r_i (i.e., $\mathbf{A_i} \in \mathbb{R}^{r_i\times n}$, $\mathbf{B_i} \in \mathbb{R}^{m\times r_i}$). The local training loss on the i-th client can be written as $\hat{\mathcal{L}}_i(\mathbf{\Delta W_i}) = \frac{1}{N_i}\sum_{i=1}^{N_i} \ell(f_{\mathbf{W_0}+\mathbf{\Delta W_i}}(x_i), y_i)$, where $f_{\mathbf{W_0}+\mathbf{\Delta W_i}}$ is the locally fine-tuned model via LoRA, and $\ell(x, y)$ is the loss function. Following [36, 27], we assume that $\ell(x, y)$ is convex, non-negative, and twice differentiable w.r.t. x for any y. Typical examples include Cross-Entropy (CE) loss and MSE loss.

4 Method

In this section, we propose HALO, a federated LoRA framework (Fig. 2) designed to address two unresolved issues in prior work: adaptive LoRA rank allocation and client-aware aggregation-distribution of LoRA updates.

4.1 Adaptive LoRA Rank Allocation

4.1.1 NTK Regime

Under the NTK regime (lazy training), the fine-tuned model on the i-th client can be approximated by its first-order Taylor expansion throughout fine-tuning [26, 8, 46]. Based on this approximation [36], the training loss can be linearized as

$$\hat{\mathcal{L}}(\mathbf{\Delta W_i}) = \frac{1}{N_i}\sum_{i=1}^{N_i} \ell(f_{\mathbf{W_0}}(x_i) + \langle \nabla f_{\mathbf{W_0}}(x_i), \mathbf{\Delta W_i}\rangle, y_i) \approx \hat{\mathcal{L}}(\mathbf{\Delta W_i}). \tag{1}$$

Substituting the low-rank form $\mathbf{\Delta W_i} = \mathbf{B_i A_i}$, Eq. (1) becomes

$$\hat{\mathcal{L}}(\mathbf{B_i A_i}) = \frac{1}{N_i}\sum_{i=1}^{N_i} \ell(f_{\mathbf{W_0}}(x_i) + \langle \mathbf{G}(x_i), \mathbf{B_i A_i}\rangle, y_i), \tag{2}$$

where $\mathbf{G}(x_i) \triangleq \nabla_{\mathbf{W}} f_{\mathbf{W_0}}(x_i)$ denotes the gradient of the pre-trained model with respect to $\mathbf{W}$ at input x_i. For output dimension k, $\mathbf{G}(x_i) \in \mathbb{R}^{k\times m\times n}$ can be regarded as a block-diagonal gradient tensor. LoRA optimization usually adds weight decay [24, 12], which is equivalent to solving the following regularized problem:

$$\min_{\mathbf{B_i},\mathbf{A_i}} \hat{\mathcal{L}}_\lambda(\mathbf{B_i A_i}) = \hat{\mathcal{L}}(\mathbf{B_i A_i}) + \frac{\lambda}{2}\|\mathbf{B_i}\|_F^2 + \frac{\lambda}{2}\|\mathbf{A_i}\|_F^2, \tag{3}$$

where λ is the regularization coefficient and $\|\cdot\|_F$ denotes the Frobenius norm. Since $\hat{\mathcal{L}}_\lambda$ is twice differentiable in $\mathbf{B_i}$ and $\mathbf{A_i}$, prior studies imply that under the NTK regime, both Gradient Descent (GD) [31] and Stochastic Gradient Descent (SGD) [17] converge to a Second-Order Stationary Point (SOSP) of $\hat{\mathcal{L}}_\lambda(\mathbf{B_i A_i})$. Hence, the key question is to determine the rank condition under which these SOSPs correspond to global optima.

4.1.2 Determination of the Rank Range

Under the NTK approximation, the local prediction loss becomes convex with respect to the product variable $\Delta\mathbf{W}_i$. However, the LoRA parameterization itself remains non-convex because $\Delta\mathbf{W}_i$ is represented by two low-rank factors. Therefore, the key question is how large r_i should be so that this factorized surrogate has no harmful SOSPs. To analyze this question, we use a lifted representation of the LoRA factors. Let $\mathbf{Q}_i^\top = \begin{bmatrix}\mathbf{B}_i^\top & \mathbf{A}_i\end{bmatrix} \in \mathbb{R}^{r_i\times(m+n)}$, so that the upper-right block of $\mathbf{Q}_i\mathbf{Q}_i^\top$ equals $\mathbf{B}_i\mathbf{A}_i$. We further consider an arbitrarily small generic positive-semidefinite perturbation on the lifted variable. This perturbation is used only as an analysis device: it removes degenerate stationary points, while its effect on the original factorized objective can be made arbitrarily small.

Theorem 4.1 (Generic rank sufficiency)**.** *Consider the NTK surrogate in Eq. (3), and assume $\lambda > 0$. For client-i, define the NTK feature of the a-th output coordinate $\mathbf{G}_i^{(a)}(x) \triangleq \nabla_{\mathbf{W}} f_{\mathbf{W}_0}^{(a)}(x)$. Let*

$$d_i \triangleq \operatorname{rank}(\mathbf{\Phi}_i) = \operatorname{rank}([\operatorname{vec}(\mathbf{G}_i^{(a)}(x))^\top]_{(x,a)\in D_i\times[k]}). \tag{4}$$

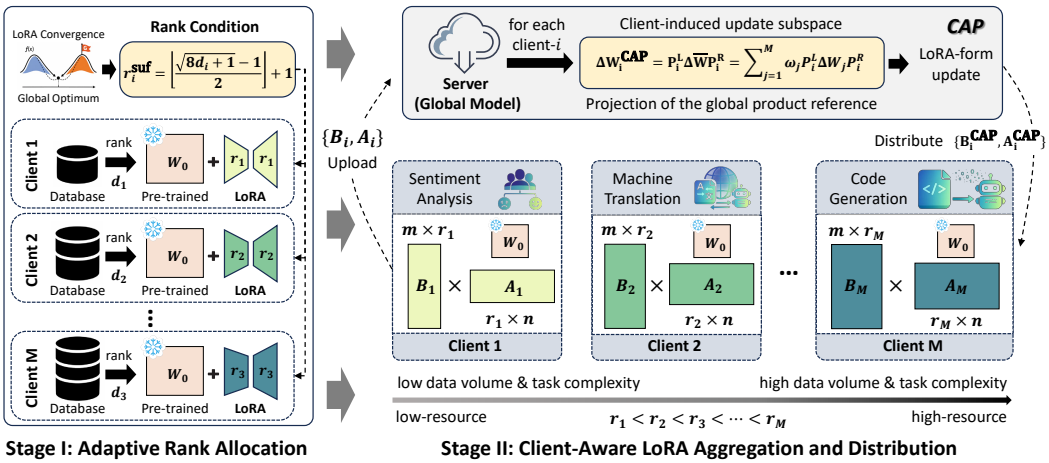


Figure 2: The overall pipeline of HALO. Stage I adaptively assigns each client a LoRA rank based on local data geometry. Stage II aggregates uploaded LoRA factors into a product-summed reference, and applies CAP to redistribute rank-compatible LoRA factors through client-induced projections.

If $r_i(r_i+1)/2 > d_i$, then, for almost every arbitrarily small positive-semidefinite lifted perturbation, every SOSP of the perturbed rank-r_i NTK surrogate is a global minimizer. Moreover, its objective value on the unperturbed rank-r_i surrogate is within an $\mathcal{O}(\varepsilon)$ perturbation error.

The detailed proof is provided in Appendix C. Theorem 4.1 identifies the effective constraint dimension d_i as the key quantity controlling the required LoRA rank. Mathematically, d_i is the rank of the local NTK feature matrix $\mathbf{\Phi}_i$, and therefore measures how many independent linearized directions the local objective imposes on $\Delta\mathbf{W}_i$. A larger d_i means that the client's local objective contains more independent constraints, requiring a larger LoRA rank to represent the corresponding update space. In practical tasks, $\mathbf{\Phi}_i$ can be constructed on a small calibration batch. For classification, each row corresponds to $\mathbf{\Phi}_i$ for a selected example x and output coordinate a. For language modeling, the same construction can be applied to selected token positions and selected logits. Since computing the exact rank of the full NTK feature matrix is expensive, we estimate d_i by the numerical rank or stable rank of this calibration feature matrix. This estimate automatically ignores duplicate, redundant, or nearly collinear local constraints. Given the estimated effective dimension d_i, we assign the smallest rank whose triangular factorization capacity exceeds d_i:

$$r_i^{\mathrm{suf}} \triangleq \min\{r \in \mathbb{N} : \frac{r(r+1)}{2} > d_i\} = \lfloor \frac{\sqrt{8d_i+1}-1}{2} \rfloor + 1. \tag{5}$$

Finally, r_i^{suf} is projected onto the feasible candidate rank set allowed by the client's resource budget.

4.2 Client-Aware LoRA Aggregation and Distribution

To leverage cross-client information and improve generalization, we study the aggregation and distribution problem in federated LoRA with heterogeneous ranks. Since different clients may use different LoRA ranks, their factors cannot be directly averaged or redistributed in a uniform fixed-rank form. In this setting, the server-side aggregate serves as an intermediate carrier of cross-client information, rather than a single deployable global LoRA module. *The goal is therefore to combine local update information across clients and generate rank-compatible LoRA factors for each client.*

4.2.1 Limitations of Current Methods

Existing heterogeneous-rank LoRA aggregation and distribution strategies mainly follow two paradigms: Naive Truncation (NT) after partial summation and SVD Truncation (SVD) after product summation, both of which have notable limitations.

NT suffers from rank collapse. NT zero-pads each local factor to rank $r_{\max}$ before averaging: $\bar{\mathbf{B}} = \sum_i \omega_i[\mathbf{B}_i, \mathbf{0}]$, $\bar{\mathbf{A}} = \sum_i \omega_i[\mathbf{A}_i^\top, \mathbf{0}]^\top$, where ω_i denotes the aggregation weight, and then truncates $\bar{\mathbf{B}}[:, :\, r_i]$ and $\bar{\mathbf{A}}[:\, r_i, :]$ for client-i. Although simple, this introduces a structural bias:

the j-th column of $\bar{\mathbf{B}}$ and the j-th row of $\bar{\mathbf{A}}$ receive contributions only from clients with $r_i \geq j$, so their effective aggregation weight becomes $\tilde{\omega}_j = \sum_{i:r_i \geq j} \omega_i < 1$ for $j > r_{\min}$. Thus, higher-index dimensions are repeatedly compressed toward zero during aggregation. Over T rounds, this compression accumulates, forcing the energy of the redistributed product $\bar{\mathbf{B}}\bar{\mathbf{A}}$ to concentrate on the shared $r_{\min}$-dimensional subspace and potentially reducing its effective rank toward $r_{\min}$.

SVD suffers from objective mismatch. SVD aggregation first forms the product-summed aggregate $\Delta\overline{\mathbf{W}}$ and then sends client-i the leading r_i singular components $\mathcal{T}_{r_i}(\Delta\overline{\mathbf{W}})$, where $\mathcal{T}_{r_i}$ denotes rank-r_i SVD truncation. By the Eckart–Young theorem, this truncation minimizes the Frobenius reconstruction error of $\Delta\overline{\mathbf{W}}$, and thus selects components solely according to global spectral energy.

However, this reconstruction objective is not aligned with local loss reduction. Assume that the loss is L-Lipschitz with respect to the model output. Under the NTK approximation, the discarded residual $\mathbf{E} = \Delta\overline{\mathbf{W}} - \mathcal{T}_{r_i}(\Delta\overline{\mathbf{W}})$ influences client-i through its projection onto the local NTK features:

$$\|\mathbf{E}\|_{i,\mathrm{NTK}}^2 \triangleq \frac{1}{N_i} \sum_{(x,a)\in D_i\times[k]} \langle \mathbf{G}_i^{(a)}(x), \mathbf{E}\rangle^2. \tag{6}$$

Thus, this NTK-relevant residual controls the discrepancy between the local effects of $\Delta\overline{\mathbf{W}}$ and its rank-r_i SVD truncation. Therefore, spectral energy alone cannot determine local usefulness: a low-energy component may be crucial if it aligns with client-i's NTK features, whereas a high-energy component may have little effect on that client's objective. This mismatch becomes more pronounced in federated learning. Components shared by many clients are reinforced during aggregation and tend to acquire large singular values, while client-specific components often remain low-energy even when they encode essential local constraints. As a result, SVD can discard locally relevant directions, especially for clients with larger effective dimensions. These clients receive rank-compatible but poorly aligned LoRA updates, forcing them to relearn the removed information in later rounds and leading to slower convergence and weaker local performance.

4.2.2 Client-Aware Projection for Aggregation and Distribution

Based on the above analysis, an ideal scheme should preserve the product-summed global reference $\Delta\overline{\mathbf{W}}$ while minimizing the effect of the discarded residual on client-i.

Client-induced update subspace. However, the server cannot directly evaluate this residual effect, since it has no access to the local dataset D_i or the local NTK features $\mathbf{G}_i^{(a)}(x)$. It must therefore infer the directions that are locally relevant to client-i from the uploaded information. Under the NTK approximation, client-i updates its LoRA product along directions induced by its local NTK features. Thus, the uploaded product update $\Delta\mathbf{W}_i$ provides a low-rank summary of the directions through which client-i reduces its local loss. This motivates approximating the same product-summed global reference $\Delta\overline{\mathbf{W}}$ within the client-induced update subspace revealed by $\Delta\mathbf{W}_i$, so that the discarded residual is controlled along directions most relevant to client-i.

Projection of the global product reference. Following this principle, we propose Client-Aware Projection (CAP) scheme. CAP keeps $\Delta\overline{\mathbf{W}}$ as the global reference and projects it onto the client-induced update subspace before distributing it to client-i. Concretely, the server extracts this subspace from the compact SVD $\Delta\mathbf{W}_i = \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^\top$, where $\mathbf{S}_i$ contains the nonzero singular values, and $\mathbf{U}_i$ and $\mathbf{V}_i$ span the left and right subspaces of $\Delta\mathbf{W}_i$. Let $\mathbf{P}_i^L = \mathbf{U}_i\mathbf{U}_i^\top$ and $\mathbf{P}_i^R = \mathbf{V}_i\mathbf{V}_i^\top$ be the corresponding projectors. The CAP update for client-i is defined as

$$\Delta\mathbf{W}_i^{\mathrm{CAP}} = \mathbf{P}_i^L\Delta\overline{\mathbf{W}}\mathbf{P}_i^R = \sum_{j=1}^{M} \omega_j \mathbf{P}_i^L\Delta\mathbf{W}_j\mathbf{P}_i^R. \tag{7}$$

For implementation, the server need not explicitly materialize $\Delta\mathbf{W}_i^{\mathrm{CAP}}$. It only computes the core matrix within the client-induced subspace: $\mathbf{M}_i = \mathbf{U}_i^\top\Delta\overline{\mathbf{W}}\mathbf{V}_i = \widetilde{\mathbf{U}}_i\mathbf{\Sigma}_i\widetilde{\mathbf{V}}_i^\top$. The server then sets $\mathbf{B}_i^{\mathrm{CAP}} = \mathbf{U}_i\widetilde{\mathbf{U}}_i\mathbf{\Sigma}_i^{1/2}$ and $\mathbf{A}_i^{\mathrm{CAP}} = \mathbf{\Sigma}_i^{1/2}\widetilde{\mathbf{V}}_i^\top\mathbf{V}_i^\top$, so that $\mathbf{B}_i^{\mathrm{CAP}}\mathbf{A}_i^{\mathrm{CAP}} = \Delta\mathbf{W}_i^{\mathrm{CAP}}$. Therefore, the distributed LoRA factors exactly realize the CAP-projected product. Client-i receives a balanced

LoRA-form update that preserves the product-summed global reference within its client-induced update subspace, without requiring access to local data. Appendix E provides the algorithmic details.

We next formalize why this projection controls the NTK-relevant residual.

Proposition 4.2 (CAP controls NTK residual)**.** *Let $\mathcal{P}_i(\mathbf{Z}) = \mathbf{P}_i^L \mathbf{Z} \mathbf{P}_i^R$. If client-$i$'s NTK features are approximately covered by the client-induced update subspace, i.e.,*

$$\frac{1}{N_i} \sum_{(x,a)\in D_i\times[k]} \left\| \mathbf{G}_i^{(a)}(x) - \mathcal{P}_i(\mathbf{G}_i^{(a)}(x)) \right\|_F^2 \le \epsilon_i^2, \text{ then } \left\| \Delta\overline{\mathbf{W}} - \Delta\mathbf{W}_i^{\mathrm{CAP}} \right\|_{i,\mathrm{NTK}}^2 \le \epsilon_i^2 \left\| \Delta\overline{\mathbf{W}} \right\|_F^2. \quad (8)$$

In particular, if client-i's NTK feature span is contained in the client-induced update subspace, then the projection preserves the first-order effect of $\Delta\overline{\mathbf{W}}$ on client-i under the NTK linearization.

The detailed proof is provided in Appendix D. Proposition 4.2 shows that, when the client-induced update subspace covers the local NTK features of client-i, projecting the product-summed global reference $\Delta\overline{\mathbf{W}}$ onto this subspace controls the effect of the discarded residual on client-i predictions.

5 Experiments

5.1 Experimental Setup

Datasets. Following [9, 2, 48], we select three datasets covering vision and language modalities. For vision, we use CIFAR-100 [29] for image classification ($k = 100$), whose large output dimension makes the adaptive rank allocation predictions most distinguishable. For language, we use GSM8K [11] for mathematical reasoning as an empirical extension to generative tasks, and Commonsense15K [25] for commonsense reasoning, whose evaluation protocol follows [48] for fair comparison. Following [53, 48], all tasks adopt a non-IID Dirichlet split with $\alpha = 0.5$, partitioned by semantic structure to maximize federated heterogeneity: CIFAR-100 by class label, GSM8K by problem difficulty, and Commonsense15K by question domain.

Models. Following [45, 48], we select three open-source pretrained models with diverse architectures and scales. For vision, we use ViT-Base [14] on CIFAR-100 to validate `HALO` on an encoder-only architecture. For language, we use LLaMA3-8B [18] as the primary backbone and Qwen3-14B [49] to examine generalization to a larger scale and different model family. These three models span encoder-only visual and decoder-only language architectures with 86M, 8B, and 14B parameters, enabling a comprehensive evaluation of `HALO`'s generality.

Evaluation metrics. Since heterogeneous-rank federated LoRA does not maintain a single deployable server-side LoRA module, all metrics are computed from the final client-side LoRA modules. Specifically, after training, each client is evaluated using its own rank-compatible LoRA update, and we report the test-sample-weighted average across clients. For CIFAR-100, we report Top-1 and Top-5 Accuracy. For GSM8K, we report Exact-Match Accuracy (EM.) and MATH Accuracy [22] (MATH). For Commonsense15K, we evaluate on HellaSwag [52] (HS.), WinoGrande [38] (WG.), ARC-Challenge [10] (ARC-C.), and PIQA [4], covering situational completion, pronoun resolution, scientific reasoning, and physical commonsense.

Baselines. We establish baselines from two perspectives: rank allocation and heterogeneous aggregation. For rank allocation, since no prior work has studied rank assignment in federated LoRA, we construct three naive baselines: i) *Fixed Uniform Rank*, where all clients share the same rank; ii) *Random Rank* [1], where ranks are uniformly sampled from a predefined set; and iii) *Heuristic Rank*, where ranks are proportional to local data volume, i.e., $r_i = r_{\min} + \lfloor \frac{|D_i|}{|D_{\max}|} \times (r_{\max} - r_{\min}) \rfloor$.

For heterogeneous aggregation, we select four representative methods spanning the two paradigms analyzed in Section 4.2: i) For NT, *HetLoRA* [9] pads smaller-rank factors with zeros to the global maximum rank and performs sparsity-weighted averaging in the factor space; ii) For SVD, *FlexLoRA* [2] applies SVD to the server-side weighted average for rank-heterogeneous redistribution, and *raFLoRA* [48] addresses rank collapse in SVD-based aggregation by partitioning by rank level and restricting each partition to its effective contributors.

Experimental details. Due to space limitations, we provide experimental details in Appendix F.1.

Table 1: Overall comparison under heterogeneous ranks. Values are mean ± std over five runs, averaged across final client-side LoRA modules. Shaded rows denote Adaptive rank allocation; **bold** and underline mark the highest and second-highest value per column.

Aggregation	Rank Alloc.	CIFAR-100		GSM8K		Commonsense15K				Avg.↑
		Top-1↑	Top-5↑	EM.↑	MATH↑	HS.↑	WG.↑	ARC-C.↑	PIQA↑	
Centralized	—	91.23±0.34	99.01±0.11	67.83±0.38	46.27±0.44	87.34±0.62	78.43±0.81	78.42±0.44	85.91±0.37	79.31
Naive Truncation (NT)										
HetLoRA	Fixed	81.42±0.43	97.36±0.14	51.78±0.58	31.24±0.52	79.63±0.67	76.47±0.83	55.97±0.61	81.17±0.38	69.38
HetLoRA	Random	84.25±0.37	98.25±0.12	54.06±0.63	33.18±0.47	79.07±0.71	75.05±0.91	55.54±0.55	81.55±0.41	70.12
HetLoRA	Heuristic	84.83±0.29	98.26±0.09	57.32±0.51	31.96±0.39	79.48±0.58	77.26±0.76	56.65±0.48	81.44±0.33	70.90
HetLoRA	Adaptive	87.21±0.44	98.27±0.16	57.93±0.72	37.82±0.63	82.54±0.84	75.21±1.02	62.14±0.79	84.33±0.51	73.18
SVD Truncation (SVD)										
FlexLoRA	Fixed	83.06±0.38	98.04±0.11	51.19±0.54	34.07±0.44	78.66±0.62	74.51±0.87	52.09±0.57	79.97±0.35	68.95
FlexLoRA	Random	84.39±0.42	98.19±0.13	54.00±0.67	33.43±0.51	79.58±0.73	75.29±0.94	56.79±0.63	80.98±0.44	70.33
FlexLoRA	Heuristic	84.21±0.31	98.24±0.10	51.56±0.49	33.76±0.38	79.67±0.55	74.72±0.81	57.11±0.52	81.93±0.37	70.15
FlexLoRA	Adaptive	**88.56±0.47**	98.48±0.17	57.14±0.81	36.52±0.58	83.64±0.91	74.73±1.08	61.92±0.83	83.04±0.53	73.00
raFLoRA	Fixed	84.07±0.36	98.28±0.12	50.18±0.59	33.62±0.48	79.72±0.64	74.82±0.89	54.26±0.58	79.70±0.38	69.33
raFLoRA	Random	85.74±0.41	98.49±0.14	47.61±0.53	31.58±0.42	79.80±0.68	75.69±0.97	57.50±0.61	81.33±0.40	69.72
raFLoRA	Heuristic	84.19±0.33	98.31±0.10	55.12±0.62	32.47±0.45	79.37±0.71	74.82±0.85	55.88±0.54	80.14±0.36	70.04
raFLoRA	Adaptive	87.90±0.52	98.29±0.18	59.13±0.88	36.84±0.67	84.17±0.93	75.92±1.11	62.38±0.87	83.77±0.56	73.55
Ours										
CAP	Fixed	87.54±0.39	98.45±0.13	57.47±0.66	35.24±0.53	83.00±0.74	76.23±0.96	63.77±0.71	82.89±0.46	73.12
CAP	Random	87.03±0.45	98.24±0.15	55.38±0.71	35.87±0.57	82.58±0.78	74.23±1.03	61.87±0.75	82.91±0.49	72.26
CAP	Heuristic	88.19±0.37	98.28±0.11	57.84±0.69	37.51±0.61	83.47±0.82	74.95±0.98	62.84±0.78	83.07±0.43	73.35
CAP	**Adaptive**	88.54±0.48	**98.59±0.16**	**58.37±0.93**	**43.47±0.74**	**85.83±0.97**	**77.43±1.14**	**63.94±0.91**	**84.93±0.58**	**75.33**
HALO vs. raFLoRA+Fixed (SOTA)		+4.47	+0.31	+8.19	+9.85	+6.11	+2.61	+9.68	+5.23	+5.81

5.2 Results

5.2.1 Overall Performance

We decompose federated LoRA training into rank allocation and client-aware aggregation and distribution, and evaluate all combinations on three benchmarks with ViT-Base on CIFAR-100 and LLaMA3-8B on language tasks. Since heterogeneous-rank federated LoRA does not maintain a single deployable server-side LoRA module, we evaluate the final client-side LoRA modules and report the average performance across clients. Results are shown in Tab. 1, where aggregation and distribution methods are grouped with different rank allocation strategies.

- *Adaptive ranks are consistently best.* Within each group, Random occasionally underperforms Fixed due to accidental low-rank assignments, Heuristic consistently improves over Fixed, and Adaptive achieves the highest Avg. in every group. Since each client's required LoRA capacity is governed by the effective dimension of its local NTK problem. Too-small ranks restrict the feasible update space and cause underfitting, whereas oversized ranks introduce redundant degrees of freedom with limited local data. By assigning ranks according to local data geometry, Adaptive improves the quality of local updates, and this benefit transfers across different backbones.
- *CAP gives the strongest redistribution.* CAP yields the highest Avg. under all four rank allocation strategies, with especially clear gains over FlexLoRA and raFLoRA on Commonsense15K, the most heterogeneous benchmark. This is because SVD-based methods minimize global Frobenius reconstruction error and therefore favor high-energy global directions, which may be weakly aligned with the NTK features of a specific client. CAP instead keeps the product-summed reference but projects it onto each client-induced update subspace, so the redistributed factors preserve directions that are locally relevant to the receiving client, enhancing the local performance.
- *CAP+Adaptive performs best.* The full configuration, CAP+Adaptive, outperforms the strongest existing baseline, raFLoRA+Fixed, by 5.81 Avg. points, with particularly strong gains on GSM8K and Commonsense15K. This improvement comes from the complementarity of the two modules: Adaptive rank allocation provides each client with sufficient local update capacity, while CAP uses the resulting client-induced subspace for rank-compatible redistribution.

5.2.2 Robustness and Generalization

For hyperparameter robustness, we vary the number of clients $M \in \{5, 10, 20, 50\}$ and the heterogeneity level $\alpha \in \{1.0, 0.5, 0.3, \text{patho}\}$ on CIFAR-100 and GSM8K, while keeping other settings fixed. Results are shown in Fig. 3. Across different client scales, HALO consistently outperforms raFLoRA and shows smaller variance, indicating stable behavior from cross-silo to larger-scale settings. This robustness comes from the combination of the two modules: adaptive rank allocation

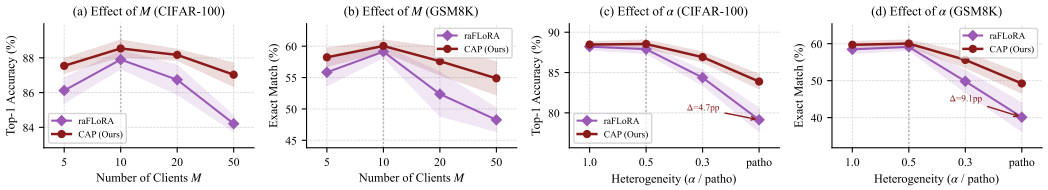


Figure 3: Robustness to client scale and data heterogeneity on CIFAR-100 and GSM8K. Each curve varies one factor while keeping other settings fixed; shaded bands denote ± 1 std over five runs.

adjusts each client's local capacity according to its effective dimension, while CAP redistributes the product-summed reference within each client-induced update subspace, avoiding reliance on a fixed global rank space. As data heterogeneity increases, all methods degrade, but HALO drops more slowly and maintains a larger margin, especially under the pathological split. This is because stronger heterogeneity weakens the alignment between global singular directions and local update directions, while adaptive rank allocation and CAP help capture and preserve these local signals. Overall, HALO is more robust than raFLoRA to client scale and data heterogeneity.

5.2.3 Effectiveness of Rank Allocation and Heterogeneous Signal Preservation

To examine the reasonableness of the sufficient-rank condition, we construct three representative clients with different empirical NTK effective dimensions $d_i = \mathrm{rank}(\Phi_i)$: high-dimensional C_{hi}, medium-dimensional C_{md}, and low-dimensional C_{lo}. We vary their ranks over $r \in \{2, 4, 8, 16, 32, 64\}$, while keeping other clients at their Adaptive ranks, and report the averaged client-side test performance. Results are shown in Fig. 4. Each curve shows a clear three-stage pattern: performance increases rapidly when $r < r_i^{\mathrm{suf}}$, stabilizes around $r \approx r_i^{\mathrm{suf}}$, and becomes flat or slightly declines as r further increases. Moreover, clients with larger effective dimensions reach the stable region at higher ranks, while lower-dimensional clients stabilize earlier. This is because the LoRA rank controls the degrees of freedom for the local NTK-linearized problem: ranks below r_i^{suf} cannot cover the independent client-side constraints, whereas larger ranks mainly introduce redundant parameters under finite local data. We further validate CAP's heterogeneous-signal preservation in Appendix F.2.

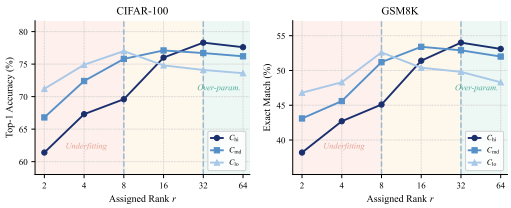


Figure 4: Rank–performance curves for representative clients. Dashed lines mark sufficient ranks.

5.2.4 Communication and Computation Efficiency

We analyze communication overhead, computation cost, and convergence speed on CIFAR-100 and GSM8K. All methods use Adaptive rank allocation, so the comparison isolates the efficiency of different aggregation and distribution mechanisms. For communication, we report the trade-off between cumulative communication cost and performance, where the cumulative cost at round T is the total upload and download volume across all clients and rounds [45, 48]. Results are shown in Fig. 5. CAP has the same communication scale as mainstream efficient baselines because it only changes the server-side construction of redistributed LoRA factors: clients still upload and download rank-compatible LoRA factors without transmitting extra statistics, gradients, or local data. Its Pareto advantage therefore comes from more effective redistribution rather than additional communication. By projecting the product-summed reference onto each client-induced update subspace, CAP produces client-aligned LoRA factors that achieve higher performance under the same communication budget, or reach the same target performance with fewer rounds. Computation and convergence efficiency are reported in Appendix F.3.

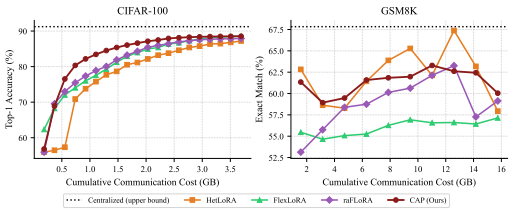


Figure 5: Performance–communication trade-off. Markers denote fixed epoch intervals.

6 Conclusion

In this work, we addressed two fundamental challenges in heterogeneous federated LoRA fine-tuning: how to assign client-specific ranks and how to redistribute heterogeneous LoRA updates without losing locally relevant information. We proposed HALO, a unified framework that combines NTK-guided adaptive rank allocation with CAP. Experiments across vision and language tasks show that HALO consistently outperforms existing heterogeneous LoRA baselines, converges faster, and remains robust under varying client scales, data heterogeneity levels, and model architectures. More broadly, our results suggest that federated PEFT should move beyond fixed global adapter spaces and instead treat rank capacity and update geometry as client-dependent quantities. This opens a path toward more scalable and personalized federated fine-tuning of foundation models in realistic deployments, where clients differ substantially in data distribution, task complexity, and computational resources. We hope this work provides a theoretical basis and a strong empirical baseline for future studies on heterogeneity-aware federated adaptation.

Acknowledgements

This work was supported in part by the R&D Program of Zhejiang (No. 2026C02A1246) and the National Natural Science Foundation of China (No. 62522217).

References

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

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

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

[4] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

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

[6] Chaochao Chen, Xiaohua Feng, Yuyuan Li, Lingjuan Lyu, Jun Zhou, Xiaolin Zheng, and Jianwei Yin. Integration of large language models and federated learning. *Patterns*, 5(12), 2024.

[7] Jinyu Chen, Wenchao Xu, Song Guo, Junxiao Wang, Jie Zhang, and Haozhao Wang. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022.

[8] Zixiang Chen, Yuan Cao, Quanquan Gu, and Tong Zhang. A generalized neural tangent kernel analysis for two-layer neural networks. *Advances in Neural Information Processing Systems*, 33:13363–13373, 2020.

[9] Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. 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, 2024.

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

[11] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[12] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

[13] Xin Luna Dong, Seungwhan Moon, Yifan Ethan Xu, Kshitiz Malik, and Zhou Yu. Towards next-generation intelligent assistants leveraging llm techniques. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5792–5793, 2023.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[15] Tianyu Du, Shouling Ji, Lujia Shen, Yao Zhang, Jinfeng Li, Jie Shi, Chengfang Fang, Jianwei Yin, Raheem Beyah, and Ting Wang. Cert-rnn: Towards certifying the robustness of recurrent neural networks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021.

[16] Jiawei Ge, Jiuxin Cao, Xuelin Zhu, Xinyu Zhang, Chang Liu, Kun Wang, and Bo Liu. Consistencies are all you need for semi-supervised vision-language tracking. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 1895–1904, 2024.

[17] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842. PMLR, 2015.

[18] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[19] Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. *arXiv preprint arXiv:2410.01463*, 2024.

[20] Tao Guo, Song Guo, and Junxiao Wang. Pfedprompt: Learning personalized prompt for vision-language models in federated learning. In *Proceedings of the ACM Web Conference 2023*, pages 1364–1374, 2023.

[21] Yexiao He, Ziyao Wang, Zheyu Shen, Guoheng Sun, Yucong Dai, Yongkai Wu, Hongyi Wang, and Ang Li. Shed: Shapley-based automated dataset refinement for instruction fine-tuning. *Advances in Neural Information Processing Systems*, 37:99382–99403, 2024.

[22] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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

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

[25] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 5254–5276, 2023.

[26] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[27] Uijeong Jang, Jason D Lee, and Ernest K Ryu. Lora training in the ntk regime has no spurious local minima. In *International Conference on Machine Learning*, pages 21306–21328. PMLR, 2024.

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

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[30] Guangchen Lan, Dong-Jun Han, Abolfazl Hashemi, Vaneet Aggarwal, and Christopher Brinton. Asynchronous federated reinforcement learning with policy gradient updates: Algorithm design and convergence analysis. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[31] Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on learning theory*, pages 1246–1257. PMLR, 2016.

[32] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 3045–3059, 2021.

[33] Fei Liu, Yiming Yao, Ping Guo, Zhiyuan Yang, Xi Lin, Zhe Zhao, Xialiang Tong, Kun Mao, Zhichao Lu, Zhenkun Wang, et al. A systematic survey on large language models for algorithm design. *ACM Computing Surveys*, 58(8):1–32, 2026.

[34] Muqing Liu, Chongjie Si, and Yuheng Jia. Flexlora: Entropy-guided flexible low-rank adaptation. *arXiv preprint arXiv:2601.22905*, 2026.

[35] Shuo Lu, Yingsheng Wang, Lijun Sheng, Lingxiao He, Aihua Zheng, and Jian Liang. Out-of-distribution detection: A task-oriented survey of recent advances. *ACM Computing Surveys*, 58(2):1–39, 2025.

[36] Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, pages 23610–23641. PMLR, 2023.

[37] Thu Hang Phung, Duong M Nguyen, Thanh Trung Huynh, Quoc Viet Hung Nguyen, Trong Nghia Hoang, and Phi Le Nguyen. Federated prompt-tuning with heterogeneous and incomplete multimodal client data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3936–3946, 2025.

[38] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[39] Aliaksandra Shysheya, John Bronskill, Massimiliano Patacchiola, Sebastian Nowozin, and Richard E Turner. Fit: Parameter efficient few-shot transfer learning for personalized and federated image classification. *arXiv preprint arXiv:2206.08671*, 2022.

[40] Fatema Siddika, Md Anwar Hossen, J Pablo Muñoz, Tanya Roosta, Anuj Sharma, and Ali Jannesari. Fedreft: Federated representation fine-tuning with all-but-me aggregation. *arXiv preprint arXiv:2508.20295*, 2025.

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

[42] Raghav Singhal, Kaustubh Ponkshe, and Praneeth Vepakomma. Fedex-lora: Exact aggregation for federated and efficient fine-tuning of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1316–1336, 2025.

[43] Jingwei Sun, Ziyue Xu, Hongxu Yin, Dong Yang, Daguang Xu, Yudong Liu, Zhixu Du, Yiran Chen, and Holger R Roth. Fedbpt: Efficient federated black-box prompt tuning for large language models. In *International Conference on Machine Learning*, pages 47159–47173. PMLR, 2024.

[44] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023.

[45] 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. *Advances in Neural Information Processing Systems*, 37:22513–22533, 2024.

[46] Alexander Wei, Wei Hu, and Jacob Steinhardt. More than a toy: Random matrix models predict how real-world neural representations generalize. In *International conference on machine learning*, pages 23549–23588. PMLR, 2022.

[47] Mitchell Wortsman, Suchin Gururangan, Shen Li, Ali Farhadi, Ludwig Schmidt, Michael Rabbat, and Ari S Morcos. lo-fi: distributed fine-tuning without communication. *arXiv preprint arXiv:2210.11948*, 2022.

[48] Fei Wu, Jia Hu, Geyong Min, and Shiqiang Wang. Preventing rank collapse in federated low-rank adaptation with client heterogeneity. *arXiv preprint arXiv:2602.13486*, 2026.

[49] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

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

[52] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 4791–4800, 2019.

[53] Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the federatedgpt: Federated instruction tuning. In *ICASSP 2024-2024 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6915–6919. IEEE, 2024.

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

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

A Limitations and Broader Impacts

HALO adapts both LoRA rank capacity and redistribution geometry to client-specific heterogeneity. Its current design relies on empirical NTK features to estimate local effective dimensions and uses uploaded LoRA products to construct client-induced update subspaces. Although these estimates are stable in our experiments, future work can improve their efficiency and robustness for larger models, longer-context tasks, and more resource-constrained devices. These extensions mainly concern estimation quality and scalability, without changing the core HALO framework. More broadly, HALO can reduce redundant computation for low-resource clients while preserving sufficient capacity for clients with more complex data, making federated fine-tuning more practical in heterogeneous environments. This may benefit privacy-sensitive applications such as on-device personalization, healthcare collaboration, and enterprise model adaptation, where data cannot be centralized. However, HALO does not replace standard privacy and safety mechanisms; practical deployments should still consider secure aggregation, differential privacy when needed, access control, fairness auditing, and domain-specific safety evaluation. Overall, HALO offers a scalable and heterogeneity-aware direction for federated parameter-efficient fine-tuning, supporting more efficient, accessible, and personalized foundation model adaptation.

B Licenses for Existing Assets

This work uses existing public datasets, benchmarks, and pretrained models only for research and evaluation purposes. For pretrained models, we use the ViT-Base checkpoint google/vit-base-patch16-224-in21k under the Apache-2.0 license, Meta-Llama-3-8B under the Meta Llama 3 Community License, and Qwen/Qwen3-14B under the Apache-2.0 license. For datasets and benchmarks, we use the CIFAR-100 Python dataset from the official University of Toronto CIFAR source, GSM8K under the MIT license, MMLU under the MIT license, AI2 ARC/ARC-Challenge under the CC BY-SA 4.0 license, WinoGrande under the CC-BY license, PIQA under the Academic Free License v3.0, and HellaSwag under the MIT license. For Commonsense15K, we use the curated 15K subset of Commonsense170K from the LLM-Adapters setting [25, 35]. The LLM-Adapters data release is provided under the Open Data Commons Attribution License (ODC-By). Since the ODC-By license covers database rights and does not override rights in individual contents, we also follow the terms of the underlying source benchmarks. We do not redistribute third-party datasets, original benchmark data, or pretrained model weights. Any released code, configuration files, data split indices, or trained LoRA adapters will include appropriate attribution, copyright notices, license information, and usage guidelines for the corresponding third-party assets.

C Proofs of the Local Rank-Sufficiency Condition

We prove Theorem 4.1. Throughout this section, let

$$\Delta \mathbf{W}_i = \mathbf{B}_i \mathbf{A}_i, \quad \mathbf{Q}_i = \begin{bmatrix} \mathbf{B}_i \\ \mathbf{A}_i^\top \end{bmatrix} \in \mathbb{R}^{(m+n)\times r_i},$$

so that the upper-right block of $\mathbf{Q}_i \mathbf{Q}_i^\top$ equals $\mathbf{B}_i \mathbf{A}_i$. Let $p = m + n$. For $\mathbf{S} \in \mathbb{R}^{m\times n}$, define

$$\mathscr{H}(\mathbf{S}) = \begin{bmatrix} \mathbf{0} & \mathbf{S} \\ \mathbf{S}^\top & \mathbf{0} \end{bmatrix} \in \mathbb{S}^p.$$

The unperturbed and perturbed factorized NTK objectives are

$$\mathcal{J}_i(\mathbf{Q}_i) = \hat{\mathcal{L}}_i(\mathbf{B}_i \mathbf{A}_i) + \frac{\lambda}{2} \|\mathbf{Q}_i\|_F^2, \tag{9}$$

$$\mathcal{J}_{i,\mathbf{P}_i}(\mathbf{Q}_i) = \mathcal{J}_i(\mathbf{Q}_i) + \frac{1}{2} \left\langle \mathbf{P}_i, \mathbf{Q}_i \mathbf{Q}_i^\top \right\rangle, \tag{10}$$

where $\mathbf{P}_i \succeq 0$. We assume $\lambda > 0$ in this rank-sufficiency analysis. Hence $\lambda \mathbf{I}_p + \mathbf{P}_i \succ 0$ for every $\mathbf{P}_i \succeq 0$.

Let

$$\mathbf{\Phi}_i = \left[\operatorname{vec}\left(\mathbf{G}_i^{(a)}(x) \right)^\top \right]_{(x,a)\in D_i \times [k]}, \quad d_i = \operatorname{rank}(\mathbf{\Phi}_i),$$

and equivalently define

$$\mathcal{S}_i = \operatorname{span}\left\{\mathbf{G}_i^{(a)}(x) : (x,a) \in D_i \times [k]\right\} \subseteq \mathbb{R}^{m\times n}, \quad \dim(\mathcal{S}_i) = d_i.$$

Lemma C.1 (Rank-deficient SOSPs are globally optimal)**.** *If $\widehat{\mathbf{Q}}_i$ is a rank-deficient SOSP of $\mathcal{J}_{i,\mathbf{P}_i}$, then it is a global minimizer of $\mathcal{J}_{i,\mathbf{P}_i}$ over $\mathbb{R}^{p\times r_i}$.*

Proof. For $\mathbf{X} \in \mathbb{S}_+^p$, let $\mathbf{X}_{12}$ denote its upper-right block and define

$$G(\mathbf{X}) = \frac{1}{2}\langle \lambda \mathbf{I}_p + \mathbf{P}_i, \mathbf{X}\rangle, \quad H(\mathbf{X}) = \hat{\mathcal{L}}_i(\mathbf{X}_{12}).$$

Then

$$G(\mathbf{Q}_i\mathbf{Q}_i^\top) + H(\mathbf{Q}_i\mathbf{Q}_i^\top) = \mathcal{J}_{i,\mathbf{P}_i}(\mathbf{Q}_i).$$

The function G is convex and twice differentiable. Since $\lambda\mathbf{I}_p + \mathbf{P}_i \succ 0$, if $\mu = \lambda_{\min}(\lambda\mathbf{I}_p + \mathbf{P}_i) > 0$, then for every $\mathbf{X} \succeq 0$,

$$G(\mathbf{X}) \geq \frac{\mu}{2}\operatorname{tr}(\mathbf{X}) \geq \frac{\mu}{2}\|\mathbf{X}\|_F.$$

Thus G has compact level sets. Moreover, since $\hat{\mathcal{L}}_i$ is convex and continuous in $\Delta\mathbf{W}_i$ and $\mathbf{X} \mapsto \mathbf{X}_{12}$ is linear, H is proper, convex, and lower semicontinuous.

By the structured factorization theorem of Haeffele and Vidal, any rank-deficient SOSP of $G(\mathbf{Q}_i\mathbf{Q}_i^\top) + H(\mathbf{Q}_i\mathbf{Q}_i^\top)$ induces a global minimizer of $G + H$ over $\mathbb{S}_+^p$. Hence, for every $\mathbf{Q}_i \in \mathbb{R}^{p\times r_i}$,

$$\mathcal{J}_{i,\mathbf{P}_i}(\widehat{\mathbf{Q}}_i) \leq \mathcal{J}_{i,\mathbf{P}_i}(\mathbf{Q}_i),$$

which proves the claim. □

Lemma C.2 (Generic absence of full-rank stationary points)**.** *Assume*

$$\frac{r_i(r_i+1)}{2} > d_i.$$

Then, for almost every $\mathbf{P}_i \in \mathbb{S}^p$, the perturbed objective $\mathcal{J}_{i,\mathbf{P}_i}$ has no full-column-rank first-order stationary point. Consequently, the same holds for almost every $\mathbf{P}_i$ in any small set

$$\{\mathbf{P}_i \in \mathbb{S}_+^p : \|\mathbf{P}_i\|_F \leq \varepsilon\}$$

with respect to the restricted Lebesgue measure.

Proof. If $r_i > p$, no matrix in $\mathbb{R}^{p\times r_i}$ has full column rank, so the claim is trivial. We therefore assume $r_i \leq p$.

Let $\widehat{\mathbf{Q}}_i$ be a first-order stationary point and set

$$\mathbf{S}_i = \nabla_{\Delta\mathbf{W}_i}\hat{\mathcal{L}}_i(\widehat{\mathbf{B}}_i\widehat{\mathbf{A}}_i).$$

By the NTK linearization and the chain rule,

$$\mathbf{S}_i = \sum_{(x,a)\in D_i\times[k]} v_{x,a}\mathbf{G}_i^{(a)}(x) \in \mathcal{S}_i$$

for suitable coefficients $v_{x,a}$. Differentiating Eq. (10) with respect to $\mathbf{Q}_i$ gives

$$\nabla_{\mathbf{Q}_i}\mathcal{J}_{i,\mathbf{P}_i}(\widehat{\mathbf{Q}}_i) = \big(\mathscr{H}(\mathbf{S}_i) + \lambda\mathbf{I}_p + \mathbf{P}_i\big)\widehat{\mathbf{Q}}_i = \mathbf{0}.$$

If $\widehat{\mathbf{Q}}_i$ has full column rank r_i, then

$$\dim\ker\left(\mathscr{H}(\mathbf{S}_i) + \lambda\mathbf{I}_p + \mathbf{P}_i\right) \geq r_i.$$

For $s = r_i, \ldots, p$, define

$$\mathcal{M}_s = \{\mathbf{Y} \in \mathbb{S}^p : \dim\ker(\mathbf{Y}) = s\}.$$

Strictly, $\mathcal{M}_s$ is a finite union of smooth embedded manifolds, corresponding to the possible inertias of the nonzero spectrum. Each component has dimension

$$\dim(\mathcal{M}_s) = \frac{p(p+1)}{2} - \frac{s(s+1)}{2}.$$

Indeed, one chooses the s-dimensional kernel subspace and then a nonsingular symmetric form on its $(p-s)$-dimensional orthogonal complement.

Next define

$$\mathcal{Z}_i = \{\mathscr{H}(\mathbf{S}) : \mathbf{S} \in \mathcal{S}_i\} \subseteq \mathbb{S}^p.$$

Since $\mathscr{H}$ is injective,

$$\dim(\mathcal{Z}_i) = \dim(\mathcal{S}_i) = d_i.$$

Thus, if a full-rank stationary point exists, then for some $s \in \{r_i, \dots, p\}$,

$$\mathbf{P}_i \in \mathcal{M}_s - \lambda \mathbf{I}_p - \mathcal{Z}_i.$$

For each component of $\mathcal{M}_s$, we have

$$\dim(\mathcal{M}_s) + \dim(\mathcal{Z}_i) = \frac{p(p+1)}{2} - \frac{s(s+1)}{2} + d_i \le \frac{p(p+1)}{2} - \frac{r_i(r_i+1)}{2} + d_i < \frac{p(p+1)}{2}.$$

Therefore, by the standard Sard-type dimension argument, each set $\mathcal{M}_s - \lambda \mathbf{I}_p - \mathcal{Z}_i$ has Lebesgue measure zero in $\mathbb{S}^p$. Taking the finite union over all $s = r_i, \dots, p$ and all inertia components still gives a measure-zero set. Hence, outside this null set, no full-column-rank stationary point exists.

Intersecting this null set with $\{\mathbf{P}_i \in \mathbb{S}^p_+ : \|\mathbf{P}_i\|_F \le \varepsilon\}$ preserves measure zero under the restricted Lebesgue measure. This proves the claim for arbitrarily small positive-semidefinite perturbations. □

Proof of Theorem 4.1. Assume

$$\frac{r_i(r_i+1)}{2} > d_i.$$

By Lemma C.2, for almost every arbitrarily small positive-semidefinite perturbation $\mathbf{P}_i$, every SOSP of $\mathcal{J}_{i,\mathbf{P}_i}$ is rank-deficient. By Lemma C.1, every such SOSP is a global minimizer of the perturbed rank-r_i NTK surrogate.

It remains to control the perturbation error. Since $\lambda > 0$, $\mathcal{J}_i$ is coercive and continuous, hence admits a global minimizer. Let

$$\mathbf{Q}_i^\star \in \arg\min_{\mathbf{Q}_i \in \mathbb{R}^{p \times r_i}} \mathcal{J}_i(\mathbf{Q}_i),$$

and let $\widehat{\mathbf{Q}}_i$ be any SOSP of $\mathcal{J}_{i,\mathbf{P}_i}$. For almost every admissible $\mathbf{P}_i$, the first part gives

$$\mathcal{J}_{i,\mathbf{P}_i}(\widehat{\mathbf{Q}}_i) \le \mathcal{J}_{i,\mathbf{P}_i}(\mathbf{Q}_i^\star).$$

If $\|\mathbf{P}_i\|_F \le \varepsilon$, then using $\mathbf{P}_i \succeq 0$, Cauchy–Schwarz, and $\|\mathbf{Q}_i^\star \mathbf{Q}_i^{\star\top}\|_F \le \|\mathbf{Q}_i^\star\|_F^2$, we obtain

$$\begin{aligned}
\mathcal{J}_i(\widehat{\mathbf{Q}}_i) &\le \mathcal{J}_{i,\mathbf{P}_i}(\widehat{\mathbf{Q}}_i) \\
&\le \mathcal{J}_{i,\mathbf{P}_i}(\mathbf{Q}_i^\star) \\
&= \mathcal{J}_i(\mathbf{Q}_i^\star) + \frac{1}{2}\left\langle \mathbf{P}_i, \mathbf{Q}_i^\star \mathbf{Q}_i^{\star\top} \right\rangle \\
&\le \min_{\mathbf{Q}_i} \mathcal{J}_i(\mathbf{Q}_i) + \frac{1}{2}\|\mathbf{P}_i\|_F \|\mathbf{Q}_i^\star \mathbf{Q}_i^{\star\top}\|_F \\
&\le \min_{\mathbf{Q}_i} \mathcal{J}_i(\mathbf{Q}_i) + \frac{\varepsilon}{2}\|\mathbf{Q}_i^\star\|_F^2.
\end{aligned}$$

Thus the objective value of the perturbed SOSP on the unperturbed rank-r_i surrogate is within

$$\frac{\varepsilon}{2}\|\mathbf{Q}_i^\star\|_F^2 = \mathcal{O}(\varepsilon)$$

of the unperturbed optimum. This proves Theorem 4.1. □

D Proof of CAP Control over the NTK Residual

Proposition D.1 (CAP controls NTK residual)**.** *Let $\mathcal{P}_i(\mathbf{Z}) = \mathbf{P}_i^L \mathbf{Z} \mathbf{P}_i^R$ and $\Delta \mathbf{W}_i^{\mathrm{CAP}} = \mathcal{P}_i(\Delta \overline{\mathbf{W}})$. If client-$i$'s NTK features are approximately covered by the client-induced update subspace, i.e.,*

$$\frac{1}{N_i} \sum_{(x,a) \in D_i \times [k]} \left\| \mathbf{G}_i^{(a)}(x) - \mathcal{P}_i(\mathbf{G}_i^{(a)}(x)) \right\|_F^2 \le \epsilon_i^2,$$

then,

$$\left\| \Delta \overline{\mathbf{W}} - \Delta \mathbf{W}_i^{\mathrm{CAP}} \right\|_{i,\mathrm{NTK}}^2 \le \epsilon_i^2 \left\| \Delta \overline{\mathbf{W}} \right\|_F^2 .$$

In particular, if client-i's NTK feature span is contained in the client-induced update subspace, then the projection preserves the first-order effect of $\Delta \overline{\mathbf{W}}$ on client-i under the NTK linearization.

Proof. Let

$$\mathbf{E}_i = \Delta \overline{\mathbf{W}} - \Delta \mathbf{W}_i^{\mathrm{CAP}} = \Delta \overline{\mathbf{W}} - \mathcal{P}_i(\Delta \overline{\mathbf{W}}) = (\mathcal{I} - \mathcal{P}_i)(\Delta \overline{\mathbf{W}}),$$

where $\mathcal{I}$ denotes the identity operator on the matrix space equipped with the Frobenius inner product.

We first verify that $\mathcal{P}_i$ is an orthogonal projector under the Frobenius inner product. Since $\mathbf{P}_i^L = \mathbf{U}_i \mathbf{U}_i^\top$ and $\mathbf{P}_i^R = \mathbf{V}_i \mathbf{V}_i^\top$ are orthogonal projectors, they satisfy $(\mathbf{P}_i^L)^2 = \mathbf{P}_i^L$, $(\mathbf{P}_i^R)^2 = \mathbf{P}_i^R$, $(\mathbf{P}_i^L)^\top = \mathbf{P}_i^L$, and $(\mathbf{P}_i^R)^\top = \mathbf{P}_i^R$. Therefore, for any matrix $\mathbf{Z}$,

$$\mathcal{P}_i(\mathcal{P}_i(\mathbf{Z})) = \mathbf{P}_i^L (\mathbf{P}_i^L \mathbf{Z} \mathbf{P}_i^R) \mathbf{P}_i^R = \mathbf{P}_i^L \mathbf{Z} \mathbf{P}_i^R = \mathcal{P}_i(\mathbf{Z}),$$

so $\mathcal{P}_i$ is idempotent. Moreover, for any matrices $\mathbf{X}$ and $\mathbf{Z}$ of the same size,

$$\begin{aligned} \langle \mathbf{X}, \mathcal{P}_i(\mathbf{Z}) \rangle_F &= \operatorname{tr}\left(\mathbf{X}^\top \mathbf{P}_i^L \mathbf{Z} \mathbf{P}_i^R \right) \\ &= \operatorname{tr}\left(\mathbf{P}_i^R \mathbf{X}^\top \mathbf{P}_i^L \mathbf{Z} \right) \\ &= \operatorname{tr}\left((\mathbf{P}_i^L \mathbf{X} \mathbf{P}_i^R)^\top \mathbf{Z} \right) \\ &= \langle \mathcal{P}_i(\mathbf{X}), \mathbf{Z} \rangle_F . \end{aligned}$$

Thus, $\mathcal{P}_i$ is self-adjoint. Hence $\mathcal{P}_i$ is an orthogonal projector, and $\mathcal{I} - \mathcal{P}_i$ is also self-adjoint.

Now fix any local NTK feature $\mathbf{G}_i^{(a)}(x)$. By the definition of $\mathbf{E}_i$ and the self-adjointness of $\mathcal{I} - \mathcal{P}_i$,

$$\begin{aligned} \left\langle \mathbf{G}_i^{(a)}(x), \mathbf{E}_i \right\rangle_F &= \left\langle \mathbf{G}_i^{(a)}(x), (\mathcal{I} - \mathcal{P}_i)(\Delta \overline{\mathbf{W}}) \right\rangle_F \\ &= \left\langle (\mathcal{I} - \mathcal{P}_i)(\mathbf{G}_i^{(a)}(x)), \Delta \overline{\mathbf{W}} \right\rangle_F . \end{aligned}$$

Applying the Cauchy–Schwarz inequality gives

$$\left\langle \mathbf{G}_i^{(a)}(x), \mathbf{E}_i \right\rangle_F^2 \le \left\| (\mathcal{I} - \mathcal{P}_i)(\mathbf{G}_i^{(a)}(x)) \right\|_F^2 \left\| \Delta \overline{\mathbf{W}} \right\|_F^2 .$$

Since

$$(\mathcal{I} - \mathcal{P}_i)(\mathbf{G}_i^{(a)}(x)) = \mathbf{G}_i^{(a)}(x) - \mathcal{P}_i(\mathbf{G}_i^{(a)}(x)),$$

averaging the above inequality over $(x, a) \in D_i \times [k]$ yields

$$\begin{aligned} \|\mathbf{E}_i\|_{i,\mathrm{NTK}}^2 &= \frac{1}{N_i} \sum_{(x,a) \in D_i \times [k]} \left\langle \mathbf{G}_i^{(a)}(x), \mathbf{E}_i \right\rangle_F^2 \\ &\le \frac{1}{N_i} \sum_{(x,a) \in D_i \times [k]} \left\| \mathbf{G}_i^{(a)}(x) - \mathcal{P}_i(\mathbf{G}_i^{(a)}(x)) \right\|_F^2 \left\| \Delta \overline{\mathbf{W}} \right\|_F^2 \\ &\le \epsilon_i^2 \left\| \Delta \overline{\mathbf{W}} \right\|_F^2 . \end{aligned}$$

Taking square roots on both sides, and using $\epsilon_i \ge 0$, gives

$$\left\| \Delta \overline{\mathbf{W}} - \Delta \mathbf{W}_i^{\mathrm{CAP}} \right\|_{i,\mathrm{NTK}} = \|\mathbf{E}_i\|_{i,\mathrm{NTK}} \le \epsilon_i \left\| \Delta \overline{\mathbf{W}} \right\|_F .$$

It remains to justify the exact preservation statement. If client-i's NTK feature span is contained in the client-induced update subspace, then

$$\mathcal{P}_i(\mathbf{G}_i^{(a)}(x)) = \mathbf{G}_i^{(a)}(x)$$

for all $(x, a) \in D_i \times [k]$. Hence the coverage error is zero, and the above bound gives $\|\mathbf{E}_i\|_{i,\mathrm{NTK}} = 0$. By the definition of $\|\cdot\|_{i,\mathrm{NTK}}$, this implies

$$\left\langle \mathbf{G}_i^{(a)}(x), \Delta\overline{\mathbf{W}} - \Delta\mathbf{W}_i^{\mathrm{CAP}} \right\rangle_F = 0$$

for every $(x, a) \in D_i \times [k]$. Under the NTK linearization, the first-order change of output coordinate a at input x induced by an update $\Delta\mathbf{W}$ is $\langle \mathbf{G}_i^{(a)}(x), \Delta\mathbf{W}\rangle_F$. Therefore, $\Delta\mathbf{W}_i^{\mathrm{CAP}}$ and $\Delta\overline{\mathbf{W}}$ induce the same first-order output change on all client-i samples and output coordinates. This completes the proof. □

E Algorithm Procedure

By combining adaptive rank allocation with Client-Aware Projection (CAP), the complete HALO procedure is summarized as follows.

Algorithm 1 HALO

Require: Pretrained model f_{W_0}; M clients with datasets $\{D_i\}_{i=1}^M$; client weights $\{\omega_i\}$; regularization λ; local steps τ; local step size η_1.

Initialization: Server broadcasts frozen W_0 to all clients.
for each client $i \in [M]$ **in parallel do**
  Compute adaptive rank r_i by Eq. (5), and initialize local LoRA factors $(\mathbf{B}_i^{(0,0)}, \mathbf{A}_i^{(0,0)})$.
end for
for round $t = 0, 1, \ldots, T-1$ **do**
  // — Phase 1: Local LoRA Training —
  for each client $i \in [M]$ **in parallel do**
    Starting from $(\mathbf{B}_i^{(t,0)}, \mathbf{A}_i^{(t,0)})$, run τ SGD steps on Eq. (3) and obtain $(\mathbf{B}_i^{(t)}, \mathbf{A}_i^{(t)})$.
    Upload $(\mathbf{B}_i^{(t)}, \mathbf{A}_i^{(t)})$ to the server, where $\Delta\mathbf{W}_i^{(t)} = \mathbf{B}_i^{(t)}\mathbf{A}_i^{(t)}$.
  end for
  // — Phase 2: Product-Summed Reference —
  Server forms the product-summed reference $\Delta\overline{\mathbf{W}}^{(t)} = \sum_{j=1}^M \omega_j \mathbf{B}_j^{(t)}\mathbf{A}_j^{(t)}$.
  // — Phase 3: Client-Aware Projection —
  for each client $i \in [M]$ **do**
    Compute the compact SVD $\Delta\mathbf{W}_i^{(t)} = \mathbf{U}_i^{(t)}\mathbf{S}_i^{(t)}\mathbf{V}_i^{(t)\top}$.
    Compute the projected core $\mathbf{M}_i^{(t)} = \mathbf{U}_i^{(t)\top}\Delta\overline{\mathbf{W}}^{(t)}\mathbf{V}_i^{(t)}$.
    Factorize $\mathbf{M}_i^{(t)} = \widetilde{\mathbf{U}}_i^{(t)}\mathbf{\Sigma}_i^{(t)}\widetilde{\mathbf{V}}_i^{(t)\top}$ and construct balanced CAP factors $(\widetilde{\mathbf{B}}_i^{(t)}, \widetilde{\mathbf{A}}_i^{(t)})$.
    Send the rank-r_i factors $(\widetilde{\mathbf{B}}_i^{(t)}, \widetilde{\mathbf{A}}_i^{(t)})$ to client i.
  end for
  // — Phase 4: Client-Specific Initialization —
  for each client $i \in [M]$ **in parallel do**
    Set $(\mathbf{B}_i^{(t+1,0)}, \mathbf{A}_i^{(t+1,0)}) \leftarrow (\widetilde{\mathbf{B}}_i^{(t)}, \widetilde{\mathbf{A}}_i^{(t)})$.
  end for
end for

Ensure: Trained rank-adaptive client-side LoRA modules $\{\mathbf{B}_i^{(T,0)}, \mathbf{A}_i^{(T,0)}\}_{i=1}^M$.

F More Experimental Details and Results

F.1 Experimental Details

Following [53, 45, 2], all experiments use a cross-silo setting with 10 clients and 100% participation to evaluate aggregation quality without client sampling noise. Each client runs one local epoch per

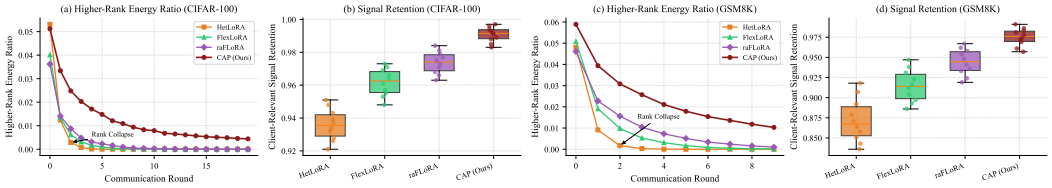


Figure 6: Higher-rank energy ratio over communication rounds (a)(c) and client-relevant signal retention after redistribution (b)(d), on CIFAR-100 with ViT-Base and GSM8K with LLaMA3-8B respectively. All methods use Adaptive rank allocation.

round with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 0.01). The LoRA scaling factor α equals the rank for unit scaling, applied to all attention projection layers. Vision tasks run for 100 rounds with learning rate 5×10^{-4} (linear decay) and batch size 64; language tasks run for 20 rounds with 3×10^{-4} and batch size 128. All experiments are repeated five times; mean and standard deviation are reported. Following [45, 48], the candidate rank sets are $\{4, 8, 16, 32, 64\}$ for language and $\{8, 16, 32, 48, 64\}$ for vision tasks. All experiments were conducted on four NVIDIA H800 GPUs.

F.2 Rank Collapse Prevention and Heterogeneous Signal Preservation

We examine whether heterogeneous-rank aggregation suppresses higher-rank client signals over training by tracking the higher-rank energy ratio of redistributed LoRA products [48] and the client-relevant signal retention across rounds. Results are shown in Fig. 6. HetLoRA exhibits the sharpest rank collapse, driving the energy ratio to near-zero within the first four rounds on CIFAR-100 and three rounds on GSM8K. FlexLoRA and raFLoRA decay more slowly but still lose substantial energy, as global SVD truncation favors aggregate spectral directions and discards client-specific components. CAP stabilizes at 4.4×10^{-3} on CIFAR-100 and 1.0×10^{-2} on GSM8K, remaining one to three orders of magnitude above all baselines at convergence, by projecting the product-summed reference onto each client-induced update subspace rather than applying one-shot global truncation. The retention analysis (Fig. 6 (b)(d)) further confirms that the preserved energy is locally useful: CAP achieves the highest retention on both benchmarks, with a larger advantage on GSM8K (gap $\approx$0.109) than on CIFAR-100 (gap $\approx$0.054), reflecting that client-subspace projection becomes increasingly beneficial as task and data heterogeneity grow.

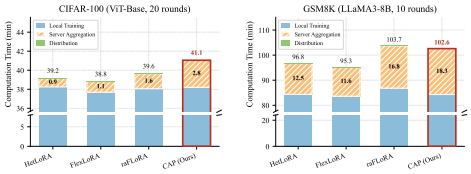


Figure 7: Runtime breakdown on GSM8K. The broken y-axis highlights aggregation and distribution costs; labels show total runtime.

Method	Rounds to Target↓		Final Performance	
	CIFAR-100	GSM8K	Top-1 (%)↑	EM. (%)↑
HetLoRA	20	—	87.21±0.44	57.93±0.72
FlexLoRA	15	—	88.56±0.47	57.14±0.81
raFLoRA	15	5	87.90±0.52	59.13±0.88
CAP (Ours)	**11**	**4**	**88.59**±0.48	**60.04**±0.93
Centralized	—	—	91.23±0.34	67.83±0.38

Table 2: Convergence under Adaptive ranks. Targets: CIFAR-100 Top-1 $\geq 87\%$, GSM8K EM. $\geq 60\%$; "—" denotes target not reached within the allotted rounds.

F.3 Computation and Convergence Efficiency

For computation and convergence efficiency, we compare per-round runtime and the number of rounds required to reach the same performance level, as shown in Fig. 7 and Tab. 2, respectively. Runtime differences mainly come from server-side aggregation and redistribution. On CIFAR-100, CAP takes about 41.1 minutes per round, close to raFLoRA (39.6 minutes) and FlexLoRA (38.8 minutes). On GSM8K, the three methods are similarly comparable: CAP requires 102.6 minutes, raFLoRA 103.7 minutes, and FlexLoRA 95.3 minutes. This is because CAP does not introduce extra client-side optimization or communication; its additional server-side cost only comes from extracting client-induced subspaces from local LoRA products and factorizing the small projected core matrices used for balanced redistribution. Since these operations are performed on low-rank factors or $r_i \times r_i$ core matrices, the overhead remains modest compared with local fine-tuning.

In terms of convergence, CAP reaches the CIFAR-100 target in only 11 rounds, compared with 15 for both raFLoRA and FlexLoRA, and also converges earlier than all baselines on GSM8K (4 rounds vs. 5 for raFLoRA; HetLoRA and FlexLoRA do not reach the target within the allotted rounds). This faster convergence comes from more effective cross-client information transfer. SVD-based redistribution may discard low-energy but locally relevant directions, forcing clients to relearn them in later rounds. CAP instead projects the product-summed reference onto each client-induced update subspace, so each client receives a rank-compatible update better aligned with its local geometry. As a result, CAP reduces cross-round information loss and offsets its modest server-side cost in end-to-end runtime.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: See the Abstract and Introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: See Appendix A.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: See Section 4, Appendix C, and Appendix D.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: See the Section 5 and Appendix F.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: See the anonymous repository link in the Abstract.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: See Appendix F.1.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: See Table 1.

   Guidelines:

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

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: See Appendix F.1.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: All research presented in this paper complies with the NeurIPS Code of Ethics in all respects.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: See Section 6 and Appendix A.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: See citations and Appendix B.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.