HoRA: Rethinking Multimodal Low-Rank Adaptation as Implicit Parameterization

Anonymous authors
Paper under double-blind review

Abstract

While multimodal low-rank adaptation (LoRA) has achieved strong empirical success, its explicit parameterization makes adaptation capacity closely tied to the dimensions and design of low-rank factors, making modality-wise parameter allocation less directly controllable once the rank is fixed. To address this limitation, we propose feature hashing-based multimodal low-rank adaptation (HoRA), a novel framework that reformulates multimodal LoRA as an implicit parameterization problem. HoRA constructs low-rank adaptations via feature hashing-based implicit parameterization, generating matrix entries on-the-fly from compact parameter buckets instead of explicitly storing them. By independently controlling the bucket size of each modality, HoRA introduces an additional degree of freedom for fine-grained control over modality-wise adaptation parameters within a fixed-rank formulation. We theoretically characterize the feature hashing-based low-rank parameterization through its unbiased reconstruction and collision-dependent approximation bound under the assumed hashing scheme. We further establish the convergence property of HoRA, with the convergence bound characterized by modality-specific bucket sizes. Extensive experiments across diverse multimodal benchmarks demonstrate that HoRA achieves competitive or superior performance with comparable or fewer trainable parameters. The code is available at `https://anonymous.4open.science/r/HoRA-E512`.

1 Introduction

Recent years have seen rapid advancements in multimodal large language models (MLLM) (Liu et al., 2023; OpenAI, 2023; Zhu et al., 2024) across diverse domains, such as embodied intelligence (Driess et al., 2023; Yang et al., 2025), multimodal retrieval (Jia et al., 2021), and recommendation systems (Lopez-Avila & Du, 2025). Due to the massive parameter scale of large models, full fine-tuning is often impractical in real-world settings, thereby making parameter-efficient fine-tuning (PEFT) increasingly attractive. By modeling parameter updates with low-rank decomposition, low-rank adaptation (LoRA) (Hu et al., 2022; Zhang et al., 2023) enables efficient fine-tuning while freezing the backbone model, and has emerged as a key PEFT approach.

Building upon its success, LoRA has been extended to MLLM to enable effective multimodal fine-tuning, giving rise to multimodal LoRA (Dou et al., 2023; Tian et al., 2024). Existing multimodal LoRA can be broadly categorized into two paradigms: structured multimodal LoRA (Tian et al., 2024; Wei et al., 2025) and Mixture of Experts (MoE)-based multimodal LoRA (Abouelenin et al., 2025; Dou et al., 2023). The former is primarily inspired by extending the LoRA architecture to multimodal settings, typically employing modality-specific low-rank adaptation branches for explicit modeling. Representative works in this line include HydraLoRA (Tian et al., 2024), DoRA (Liu et al., 2024a) and MokA (Wei et al., 2025). In contrast, the latter methods, including LoRAMoE (Dou et al., 2023) and Phi-4-Multimodal (Abouelenin et al., 2025), incorporate an MoE mechanism to dynamically route or activate LoRA modules across different modalities or tasks. These methods inherit the core advantages of LoRA in parameter-efficient fine-tuning, thereby substantially reducing the number of trainable parameters in multimodal large-scale models.

Although existing methods have made notable progress in parameter-efficient multimodal adaptation, existing methods predominantly rely on *explicit low-rank parameterization* to model the adaptation process. Here, explicit low-rank parameterization refers to approximating the update term using

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

the product of two low-dimensional matrices, i.e., $\Delta\mathbf{W} = \mathbf{BA}$. Such explicit factorization ties the number of trainable adaptation parameters to the dimensions of the low-rank factors, making parameter allocation less directly controllable. More importantly, this limitation becomes more pronounced in multimodal settings, where such coupling prevents independent allocation of the adaptation parameter budget across modalities at a fixed rank. This motivates us to explore an alternative parameterization that provides an additional degree of freedom for modality-wise parameter allocation while maintaining the parameter-efficient nature of low-rank adaptation.

Feature hashing (Weinberger et al., 2009; Chen et al., 2015; Kamma et al., 2018) provides an alternative implicit parameterization, where a compact set of trainable parameters can be used to reconstruct a larger parameter space through a deterministic hashing scheme. While NOLA (Abbasi Koohpayegani et al., 2024) explores implicit reparameterization to decouple the overall trainable parameter count from the conventional LoRA rank, HoRA employs feature hashing to parameterize low-rank factors with compact trainable buckets, providing an additional degree of freedom for controlling the adaptation parameter budget without changing the rank. More importantly, in the multimodal setting, separate buckets can then be assigned to different modalities, allowing their adaptation budgets to be independently configured under a fixed rank. From a theoretical perspective, building on standard properties of feature hashing, we characterize the approximation and optimization behavior of HoRA by establishing an unbiased reconstruction property, deriving a collision-dependent approximation bound, and further showing that the hashing-induced gradient bias is controlled by the modality-specific bucket sizes, leading to a convergence bound for the resulting optimization. Our contributions can be summarized as follows:

- We propose HoRA, a feature hashing-based implicit parameterization for multimodal low-rank adaptation, which reconstructs modality-specific adaptation factors from compact parameter buckets rather than explicitly storing the low-rank factors. By independently controlling the bucket size of each modality, HoRA introduces an additional degree of freedom for modality-wise parameter allocation within a fixed-rank formulation.
- We formulate modality-wise adaptation through independently configurable parameter budgets and develop forward and backward computation schemes that realize this allocation without explicitly materializing the full update matrices.
- We establish theoretical guarantees for both the approximation and optimization behavior of HoRA. We establish an unbiased reconstruction property and derive a collision-dependent approximation bound. We further show that the hashing-induced gradient bias is controlled by the modality-specific bucket sizes, leading to a corresponding convergence bound for the optimization process.
- Extensive experiments across multiple multimodal benchmarks demonstrate the effectiveness of HoRA, achieving competitive or superior performance under comparable parameter budgets. The results further demonstrate the flexibility of the proposed parameterization for modality-wise parameter allocation.

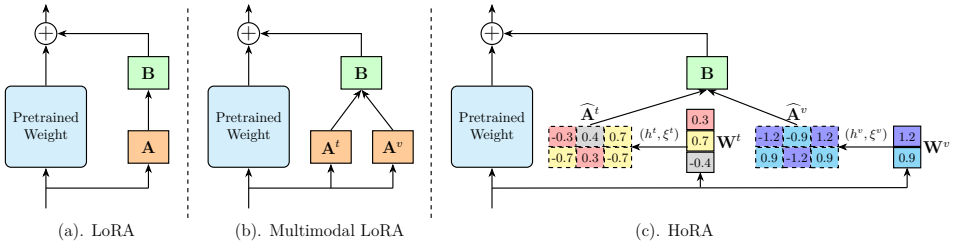
The rest of this paper is organized as follows. Section 2 introduces the main methodology. And Section 3 presents the theoretical analysis for HoRA. Furthermore, Section 4 reports the experimental results. We discuss the related work in Section 5. Finally, we conclude our work in Section 6.

2 METHODOLOGY

In this section, we present the proposed HoRA framework in detail, as shown in Fig. 1. We first revisit multimodal LoRA and its limitations, then introduce our feature hashing-based implicit parameterization method, followed by the corresponding forward and backward propagation algorithms during training procedure.

2.1 PRELIMINARY: MULTIMODAL LOW-RANK ADAPTATION

Low-Rank Adaptation (LoRA) (Hu et al., 2022) avoids full parameter fine-tuning by freezing the pre-trained weights and approximating weight updates with learnable low-rank matrices injected into each layer, as shown in Fig. 1 (a). In particular, LoRA parameterizes the weight update $\Delta\mathbf{W}$ as the



(a). LoRA (b). Multimodal LoRA (c). HoRA

Figure 1: The architecture of LoRA, multimodal LoRA, and HoRA. In HoRA, the modality-specific low-rank factors $\mathbf{A}^o$ are implicitly reconstructed from modality-specific parameter buckets $\mathbf{W}^o$ through hash functions $\{h^o, \xi^o\}$.

product of two low-rank matrices, enabling a low-rank approximation of the full parameter update in a parameter-efficient manner, which can be formulated as follows:

$$\widehat{\mathbf{W}} = \mathbf{W} + \Delta\mathbf{W} = \mathbf{W} + \mathbf{BA}, \tag{1}$$

where $\mathbf{W}, \Delta\mathbf{W} \in \mathbb{R}^{d\times k}$, $\mathbf{B} \in \mathbb{R}^{d\times r}$ and $\mathbf{A} \in \mathbb{R}^{r\times k}$, with $r \ll \min(d, k)$. During fine-tuning procedure, only $\mathbf{A}$ and $\mathbf{B}$ are trained, while the base model remains unchanged.

For multimodal scenarios, a variety of multimodal LoRA methods have been proposed, including structured and MoE-based variants, which predominantly rely on explicit low-rank parameterization. In this work, we instantiate our method under a partial-sharing multimodal LoRA mechanism as shown in Fig. 1 (b), where modality-specific $\mathbf{A}$ matrices are combined with a shared $\mathbf{B}$ matrix. The corresponding updating strategy can be formulated as:

$$\widehat{\mathbf{W}} = \mathbf{W} + \mathbf{BA}^a + \mathbf{BA}^v = \mathbf{W} + \sum\nolimits_{o\in\{a,v\}} \mathbf{BA}^o, \tag{2}$$

where $\mathbf{A}^a$ and $\mathbf{A}^v$ denote the modality-specific low-rank matrix for audio and visual modality, respectively. Here, we employ text and visual for illustration. The formulation can be straightforwardly extended to more than two modalities by introducing a modality-specific bucket for each modality.

2.2 Feature Hashing based Multimodal LoRA

To enable flexible allocation of adaptation capacity without altering the underlying low-rank structure, we introduce feature hashing as an implicit parameterization mechanism. Rather than explicitly storing the entries of the low-rank factor, we represent them using a compact parameter bucket and reconstruct each entry on-the-fly through hash functions.

Formally, we define the index hash function for each modality:

$$\forall o \in \{a, v\}, h^o : \mathbb{N}^+ \times \mathbb{N}^+ \mapsto \{1, 2, \cdots, M^o\}, \tag{3}$$

and the signed hash function:

$$\forall o \in \{a, v\}, \xi^o : \mathbb{N}^+ \times \mathbb{N}^+ \mapsto \{-1, +1\}, \tag{4}$$

where $\mathbb{N}^+$ denotes the set of positive integers, and M^o denotes the dimensional of parameter bucket. These two hash functions are stochastic and mutually independent, and the signed hash function takes values in $\{-1, +1\}$ with equal probability for any given indices $(p, q) \in \mathbb{N}^+ \times \mathbb{N}^+$:

$$P(\xi^o(p, q) = 1) = P(\xi^o(p, q) = -1) = \frac{1}{2}. \tag{5}$$

Given a low-rank factor $\mathbf{A}^o \in \mathbb{R}^{r\times k}$, feature hashing maps its entries into the parameter bucket $W_t^o \in \mathbb{R}^{M^o}$ via these two hash functions:

$$W_t^o = \sum\nolimits_{(p,q):h^o(p,q)=t} \xi^o(p, q) \cdot A_{p,q}^o, \tag{6}$$

This mapping compresses the entries of $\mathbf{A}^o$ into a compact parameter space through hashing-based parameter sharing. In HoRA, however, $\mathbf{A}^o$ is not explicitly stored or optimized. Instead, its entries are implicitly reconstructed from the trainable bucket $\mathbf{W}^o$ on-the-fly as:

$$\widehat{A}_{i,j}^o = \xi^o(i, j) \cdot W_{h^o(i,j)}^o. \tag{7}$$

Thus, each entry of the low-rank factor can be generated when needed without explicitly materializing the full $r \times k$ matrix.

Importantly, the implicit reconstruction retains the same low-rank factor dimensions: $\widehat{\mathbf{A}}^o \in \mathbb{R}^{r\times k}$, while the rank parameter r remains unchanged. Instead, the bucket size M^o provides an additional control over the number of trainable parameters used to represent the adaptation. In multimodal settings, we assign a separate parameter bucket $\mathbf{W}^o$ to each modality and independently configure its size M^o. This allows different modalities to be assigned different adaptation parameter budgets through modality-specific bucket sizes, enabling flexible modality-wise parameter allocation within a fixed-rank formulation.

Notably, our default design applies implicit parameterization to the modality-specific factors $\mathbf{A}^o$, while keeping the shared factor $\mathbf{B}$ explicitly parameterized. This choice preserves the original shared-factor structure of multimodal LoRA and isolates modality-wise capacity allocation to the modality-specific factors.

Based on the above feature hashing-based implicit reconstruction of the low-rank matrix $\mathbf{A}^o$, we next present the forward and backward propagation procedures of HoRA. Specifically, given an input $\mathbf{x} = [\mathbf{x}^a, \mathbf{x}^v]$, the forward computation can be formulated as:

$$\mathbf{h} = \widehat{\mathbf{W}}\mathbf{x} = \sum\nolimits_{o\in\{a,v\}} \mathbf{W}\mathbf{x}^o + \mathbf{B}\mathbf{A}^o\mathbf{x}^o \tag{8}$$

$$= \sum\nolimits_{o\in\{a,v\}} \mathbf{W}\mathbf{x}^o + \left[\mathbf{B}\mathbf{A}^o_{:,1}, \mathbf{B}\mathbf{A}^o_{:,2}, \cdots, \mathbf{B}\mathbf{A}^o_{:,k}\right]\mathbf{x}^o \tag{9}$$

$$= \sum\nolimits_{o\in\{a,v\}} \mathbf{W}\mathbf{x}^o + \left[\mathbf{B}(\xi^o(:,1)\odot \mathbf{W}^o_{h^o(:,1)}), \cdots, \mathbf{B}(\xi^o(:,k)\odot \mathbf{W}^o_{h^o(:,k)})\right]\mathbf{x}^o, \tag{10}$$

where $\mathbf{h}$ denotes the output, $\odot$ indicates the element-wise product, $h^o(:,j)$ and $\xi^o(:,j)$ denote the hash indices and signs associated with the j-th column of the implicit factor, respectively.

We then derive the gradients of the loss function with respect to the parameter buckets during training. Based on Equation (7) and the chain rule, we obtain:

$$\frac{\partial \mathcal{L}}{\partial W^o_k} = \sum_{(i,j):h^o(i,j)=k} \frac{\partial \mathcal{L}}{\partial \widehat{A}^o_{i,j}} \cdot \frac{\partial \widehat{A}^o_{i,j}}{\partial W^o_k} = \sum_{(i,j):h^o(i,j)=k} \xi^o(i,j)\cdot \frac{\partial \mathcal{L}}{\partial \widehat{A}^o_{i,j}}, \tag{11}$$

where $\mathcal{L}$ denotes the loss function. Thus, the gradient of each bucket aggregates the gradients of all factor entries mapped to the same bucket, with the corresponding signed hash coefficients.

Based on the above forward and backward procedures, the complete training and inference pipeline of HoRA can be readily implemented.

3 Theoretical Analysis of Hashing-based Parameterization

In this section, we theoretically characterize the approximation and optimization behavior of HoRA under feature hashing. We first establish the unbiased reconstruction property and quantify the collision-dependent approximation gap. We then characterize how this approximation propagates to the gradient and derive a corresponding convergence bound.

3.1 Approximation Properties of Implicit Parameterization

We begin by characterizing the approximation gap introduced by reconstructing the target low-rank update $\Delta\mathbf{W}$ as $\Delta\widehat{\mathbf{W}}$ via feature hashing (Kamma et al., 2018), and show that the resulting reconstruction is unbiased[1]:

Proposition 3.1. *(Unbiasedness of Feature Hashing-based Reconstruction, Informal) Under some assumptions for index hash functions and signed hash functions, for a matrix $\mathbf{B}$, we have:*

$$\mathbb{E}\left[\Delta\widehat{\mathbf{W}}\right] = \Delta\mathbf{W}, \tag{12}$$

where $\Delta\mathbf{W} = \sum_{o\in\{a,v\}} \mathbf{B}\mathbf{A}^o$ and $\Delta\widehat{\mathbf{W}} = \sum_{o\in\{a,v\}} \mathbf{B}\widehat{\mathbf{A}}^o$.

[1]For brevity, the detailed proof of all theoretical results are deferred to the appendix.

This unbiasedness property shows that the hashed parameterization recovers the target update in expectation, despite the approximation introduced by feature hashing. We next quantify the resulting approximation gap and characterize its dependence on the modality-specific bucket sizes. Specifically, we have the following theorem:

Theorem 3.2. *(Multimodal Approximation Bound, Informal) Under some assumptions for index hash functions and signed hash functions, the approximation bound introduced by feature hashing-based implicit low-rank construction is bounded:*

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] \leq \|\mathbf{B}\|^2 \cdot \sum_{a\in\{a,v\}} \sum_{i,j} \sum_{(p,q)\neq(i,j)} [A^o_{p,q}]^2 P(h^o(p,q) = h^o(i,j)). \tag{13}$$

If $h^o(\cdot)$ is uniform, we have:

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] \leq \sum\nolimits_{o\in\{a,v\}} \|\mathbf{A}^o\|^2\|\mathbf{B}\|^2 \frac{rk-1}{M^o}. \tag{14}$$

Together, these results show that feature hashing provides an unbiased yet controllable approximation of the target update, with the approximation fidelity determined by the modality-specific bucket sizes.

3.2 Optimization Behavior under Feature Hashing-based Parameterization

We further analyze how the approximation introduced by feature hashing propagates to the optimization process, focusing on the resulting gradient bias and convergence behavior. Specifically, we characterize the hashing-induced gradient bias and establish its dependence on the modality-specific bucket sizes.

Formally, we define the full gradient of loss function $\mathcal{L}(\cdot)$ at iteration t as: $g^\star_t = \nabla\mathcal{L}(\Delta\mathbf{W}_t)$. And the gradient under the hashed reconstruction $\Delta\widehat{\mathbf{W}}_t$ is defined as: $g^h_t = \nabla\mathcal{L}(\Delta\widehat{\mathbf{W}}_t)$. The hashing-induced gradient bias is defined as the deviation between the expected hashed gradient and the full gradient:

$$b_t = \mathbb{E}_{\zeta_t}[g^h_t \mid \theta_t] - g^\star_t, \tag{15}$$

where ζ_t and θ_t denote the randomness induced by feature hashing and trainable parameters at iteration t, respectively. Then we have the following Theorem:

Theorem 3.3. *(Hashing-Induced Gradient Bias Bound, Informal) Under some assumptions, the gradient introduced by hashing-based implicit low-rank constrain satisfies:*

$$\mathbb{E}_{\zeta_t}\left\|g^h_t - g^\star_t\right\|^2 \leq \sum\nolimits_{o\in\{a,v\}} \frac{L_g^2 C^o}{M^o} = \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right), \tag{16}$$

where L_g, C^a and C^v are constant.

Consequently, the squared hashing-induced gradient bias satisfies:

$$\|b_t\|^2 \leq \sum\nolimits_{o\in\{a,v\}} \frac{L_g^2 C^o}{M^o} = \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right). \tag{17}$$

During training, the gradient is additionally affected by stochastic sampling. We define g_t as the gradient encountered during training process at iteration t. We further analyze the impact of hashing-induced randomness and sampling-induced randomness on the convergence of the learning algorithm.

Corollary 3.4. *(Convergence of HoRA, Informal) Under some Assumptions, if $T \geq 16L_g^2$, $\{h^o\}_{o\in\{a,v\}}$ are uniform and set the learning rate $\eta = \frac{1}{\sqrt{T}}$, then:*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|g^\star_t\|^2 \leq \frac{4\Delta\mathcal{L} + 2L_g\sigma^2}{\sqrt{T}} + \left(2 + \frac{4L_g}{\sqrt{T}}\right)\delta_M = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right), \tag{18}$$

where $\delta_M = \sum_{a\in\{a,v\}} L_g^2\|\mathbf{A}^o_t\|^2\|\mathbf{B}_t\|^2\frac{rk-1}{M^o}$, $\Delta\mathcal{L} = \mathcal{L}(\theta_0) - \mathcal{L}(\theta^\star)$, σ is a constant.

Thus, the approximation fidelity can be directly controlled by the modality-specific bucket sizes, with larger buckets improving the approximation fidelity by reducing collision effects. This provides a principled basis for flexible modality-wise allocation under a fixed-rank formulation.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Dataset: We select six extensive benchmarks for experiments, i.e., AVE (Tian et al., 2018), MUSIC-AVQA (Li et al., 2022), MMBench (Liu et al., 2024b), POPE (Li et al., 2023b), SEED-Bench (Li et al., 2024), and MME (Fu et al., 2023). Among these benchmarks, the first two involve trimodal audio-visual-text settings, whereas the remaining four focus on visual-text bimodal scenarios. The AVE dataset is a benchmark for audio-visual event localization in unconstrained video. The MUSIC-AVQA dataset is a large-scale benchmark for audio-visual question answering. The MMBench is a benchmark for evaluating vision-language models, featuring diverse questions and a CircularEval strategy for robust and fair assessment. The POPE benchmark evaluates object hallucination in vision-language models via a polling-based query strategy. The SEED-Bench evaluates hierarchical capabilities of multimodal large language models using 19K human-annotated multiple-choice questions. The MME benchmark is a comprehensive benchmark for MLLM covering 14 subtasks spanning perception and cognition.

Baselines: We evaluate HoRA against LoRA (Hu et al., 2022) and several representative multimodal LoRA variants, including structured (HydraLoRA (Tian et al., 2024), DoRA (Liu et al., 2024a), and MokA (Wei et al., 2025)) and MoE-based methods (LoRAMoE (Dou et al., 2023) and Phi-4-Multimodal (Abouelenin et al., 2025)). Despite their architectural differences, all these methods rely on explicit low-rank parameterization, where the capacity is coupled with the chosen rank.

Implementation Details: Following MokA (Wei et al., 2025), we employ a two-stage training pipeline consisting of pretraining and fine-tuning. We use the 7B variants of Qwen2 (Yang et al., 2024) and LLaMA2 (Touvron et al., 2023) as backbone LLMs. Textual inputs are tokenized using the corresponding LLM tokenizer, while visual and audio inputs are encoded by ViT (Dosovitskiy et al., 2021) and BEATs (Chen et al., 2023), respectively, followed by modality-specific Q-Formers and MLPs for alignment with the textual space. During pretraining, we use AudioCaps (Kim et al., 2019) and Video-LLaVA (Lin et al., 2024) for trimodal settings and Video-LLaVA for bimodal settings, while freezing the LLM and updating the Q-Formers and MLPs. During fine-tuning, we use subsets of AVE and MUSIC-AVQA for the corresponding trimodal tasks, while combining LLaVA-Instruct-150K (Liu et al., 2023) with a 12K instruction subset of A-OKVQA (Schwenk et al., 2022) for bimodal settings. We instantiate HoRA based on MokA. We initialize $\mathbf{W}^o$ with Kaiming uniform distribution (He et al., 2016) and $\mathbf{B}$ with zeros, and implement the index and sign hash functions using xxhash. For fair comparison, we set the rank of all low-rank methods to $r = 4$. For bimodal settings, we evaluate the models on MME, MMBench, POPE, and SEED-Bench. More implementation details are provided in the appendix.

4.2 MAIN RESULTS

We report the performance on all datasets in Tab. 1 by comparing with multimodal LoRA baselines. For trimodal setting, we report question-answer accuracy, while for the bimodal setting, we employ the standard metrics provided by the corresponding benchmarks.

From Tab. 1, we can draw the following observations: (1). Existing multimodal LoRA methods generally improve upon vanilla LoRA by introducing modality-aware adaptation structures. However, their adaptation parameters are still explicitly instantiated through low-rank factors, leaving the allocation of modality-specific parameter budgets largely tied to the underlying factorization design. (2). Across the evaluated benchmarks, HoRA achieves competitive or superior performance across different trainable parameter scales. Rather than explicitly storing modality-specific low-rank factors, HoRA reconstructs them from compact parameter buckets through feature hashing. This implicit parameterization introduces modality-specific bucket sizes as an additional degree of freedom for controlling adaptation parameters, enabling finer-grained allocation of parameter budgets across modalities under a fixed-rank formulation. (3). More importantly, the improvement is achieved within the same fixed-rank formulation, where the modality-specific bucket sizes provide an additional degree of freedom for allocating adaptation parameters. This suggests that explicitly increasing the size of low-rank factors is not the only way to improve adaptation capacity, and that implicit parameterization provides a more flexible mechanism for modality-wise parameter allocation.

Table 1: Performance comparison with multimodal LoRA baselines. The best and second best results are highlighted as **bold** and underline, respectively.

LLM	Method	Audio-Visual-Text				Visual-Text				
		#params	AVE	#params	M-AVQA	#params	MMBench	POPE	SEED-Bench	MME
LLAMA2-7B	LoRA	10.0M	67.91%	10.0M	72.11%	10.0M	50.64%	70.28%	39.71%	908.52
	LoRAMOE	19.2M	73.96%	19.2M	76.69%	12.8M	51.98%	71.15%	39.13%	938.52
	DoRA	11.4M	73.86%	11.4M	76.86%	11.4M	51.31%	71.07%	38.96%	786.47
	HydraLoRA	17.1M	71.82%	17.1M	73.29%	13.6M	47.33%	70.87%	38.81%	774.47
	Phi-4-MM	20.0M	70.90%	20.0M	74.27%	20.0M	51.98%	72.24%	39.91%	988.37
	MokA	19.1M	77.06%	19.1M	77.76%	14.5M	56.02%	77.07%	40.80%	**1138.15**
	HoRA	15.0M	**78.39%**	13.2M	**79.29%**	10.4M	**61.42%**	**80.08%**	**45.64%**	1087.70
QWen2-7B	LoRA	10.1M	70.77%	10.1M	74.40%	10.1M	57.61%	77.04%	49.86%	1117.44
	LoRAMOE	24.9M	75.27%	24.9M	76.84%	16.6M	62.58%	80.82%	52.19%	1220.39
	DoRA	11.5M	74.73%	11.5M	77.16%	11.5M	76.08%	79.90%	57.59%	1222.73
	HydraLoRA	19.7M	73.91%	19.7M	76.04%	14.9M	75.55%	81.02%	58.38%	1182.77
	Phi-4-MM	20.2M	74.63%	20.2M	75.55%	20.2M	64.95%	77.22%	56.38%	1234.08
	MokA	19.2M	77.86%	19.2M	79.17%	14.6M	76.25%	81.83%	**58.59%**	1245.76
	HoRA	13.7M	**80.16%**	12.8M	**79.60%**	14.1M	**76.36%**	**81.98%**	58.43%	**1252.65**

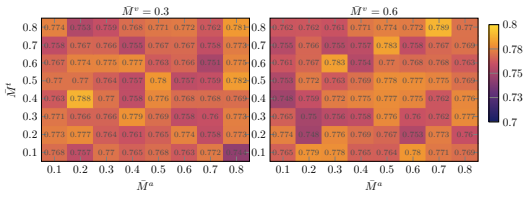


Figure 2: Capacity analysis on AVE dataset.

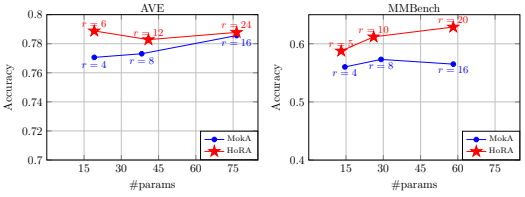


Figure 3: Performance under matched budgets.

4.3 Multimodal Capacity Allocation Analysis via Bucket Size

In HoRA, the bucket size M^o controls the number of trainable parameters allocated to modality o. We therefore explore different combinations of M^a, M^t, and M^v in the trimodal setting and evaluate their effects on model performance. The results are shown in Fig. 2, where $\bar{M}^o = M^o/(rk)$ is used to normalize the bucket size. Although different capacity allocations lead to performance variations, the overall performance remains relatively stable across a broad range of configurations. More importantly, these results demonstrate that HoRA provides an explicit degree of freedom for adjusting modality-wise parameter allocation through M^o, rather than relying solely on the rank to determine the adaptation structure.

4.4 Ablation Study

Comparison under Matched Parameter Budgets: To further examine whether the performance gains of HoRA arise from differences in the number of trainable parameters, we compare HoRA with MokA under multiple matched parameter budgets. For each budget, we adjust the corresponding model configurations to obtain comparable numbers of trainable parameters for the two methods. The resulting performance on AVE and MMBench is shown in Fig. 3. HoRA achieves superior or competitive performance across the examined parameter budgets. These results show that the performance gains of HoRA are maintained when controlling for the number of trainable parameters, indicating that its advantage is not solely attributable to differences in parameter budget.

The Impact of Hashing Mechanism: We evaluate two variants of HoRA by replacing the randomized index mapping with a deterministic mapping, i.e., "Deterministic hash", or removing the signed hashing function ξ^o, i.e., "HoRA w/o ξ^o", respectively. As shown in Fig. 4, both variants underperform HoRA, indicating the importance of jointly employing randomized index mapping and signed hashing. Notably, these modifications also violate the assumptions underlying theoretical analysis, and the corresponding unbiasedness and error guarantees therefore no longer apply.

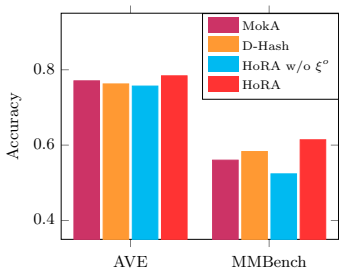


Figure 4: Hashing mechanism.

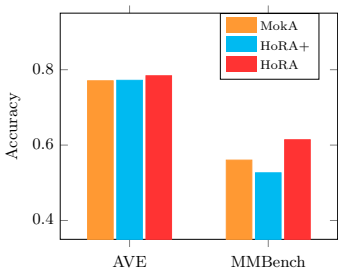


Figure 5: Hashing scope.

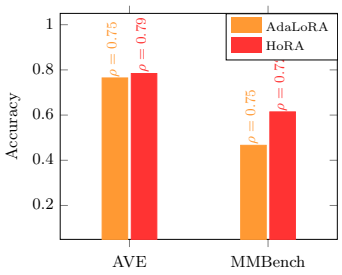


Figure 6: Comp. with AdaLoRA.

Table 2: Hashing across different layers.

Placement	Layers	AVE	MMBench
Full	$0 \sim 31$	78.39%	61.42%
Early	$0 \sim 15$	76.39%	53.65%
Middle	$8 \sim 23$	75.62%	59.16%
Late	$16 \sim 31$	76.14%	52.31%

Table 3: Running overhead on AVE (s).

Stage	MokA	HoRA
Hash setup	N/A	30.47
Param reconstruction	N/A	0.29
Inference	290.78	283.82
Total time	290.78	314.58

The Impact of Hashing Placement: We investigate the placement of hashing from two perspectives. Fig. 5 compares different hashing configurations, including hashing the modality-specific factor $\mathbf{A}$ (HoRA) and both $\mathbf{A}$ and the shared factor $\mathbf{B}$ (HoRA+). Both variants demonstrate the feasibility of incorporating hashing into different parts of the low-rank parameterization. Tab. 2 further evaluates hashing at different network layers. Applying hashing across all layers achieves the best overall performance, whereas restricting it to specific layer ranges generally degrades performance. Therefore, we adopt full-layer hashing as the default setting.

4.5 Further Analysis

Comparison with Parameter Allocation-based Unimodal LoRA: To further examine the parameter allocation flexibility of HoRA, we compare it with AdaLoRA (Zhang et al., 2023), which adjusts adaptation capacity through rank allocation. We extend AdaLoRA to the multimodal setting and compare the two methods under matched effective parameter proportions. As shown in Fig. 6, ρ denotes the proportion of effective parameters, which is determined by the proportion of non-zero singular values for AdaLoRA (Zhang et al., 2023) and by the average normalized bucket size $\bar{M}^o$ for HoRA. Under comparable values of ρ, HoRA consistently achieves better performance, demonstrating the effectiveness of its hashing-based parameter allocation mechanism in the multimodal setting.

Running Overhead Analysis: We further decompose the running overhead of HoRA into hash setup, parameter reconstruction, and inference. As shown in Tab. 3, parameter reconstruction introduces only negligible overhead, while the additional cost mainly comes from the one-time hash setup. Moreover, HoRA achieves comparable inference time to MokA, indicating that the on-the-fly hashed parameterization introduces limited runtime overhead.

The Impact of Regularization: To examine the effect of feature hashing as an implicit regularization mechanism, we compare HoRA with standard L_1 regularization applied to MokA. Specifically, the objective of MokA is augmented as $\mathcal{L}_{\text{MokA}} + \lambda \|\Theta\|_1^2$, where λ and Θ denote the regularization coefficient and the LoRA parameters, respectively. The results are reported in Fig. 7, where ρ denotes the proportion of effective parameters and is defined as the proportion of non-zero parameters for the L_1-regularized MokA. While L_1 regularization can improve the performance of MokA, HoRA achieves better results across different values of ρ. This suggests that the effect of feature hashing extends beyond conventional parameter regularization, providing a distinct mechanism for constructing the low-rank adaptation.

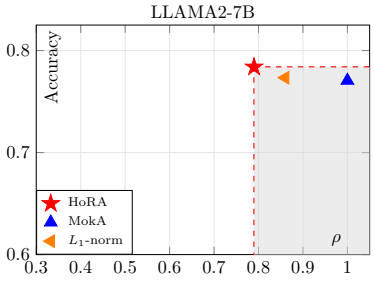


Figure 7: Regularization effect.

5 Related Works

Multimodal Low-Rank Adaptation: LoRA (Hu et al., 2022; Zhang et al., 2023) enables parameter-efficient fine-tuning via low-rank decomposition, and has been naturally extended to multimodal settings (Tian et al., 2024; Liu et al., 2024a; Wei et al., 2025) to model heterogeneous representations and cross-modal interactions. Existing multimodal LoRA methods can be broadly categorized into structured variants (Tian et al., 2024; Wei et al., 2025) based on explicit parameterization and MoE-based variants (Dou et al., 2023; Abouelenin et al., 2025) that enable dynamic cross-modal allocation of adaptation modules. Representative structured multimodal LoRA methods, including HydraLoRA (Tian et al., 2024), DoRA (Liu et al., 2024a), and MokA (Wei et al., 2025), extend LoRA via modality-aware designs. HydraLoRA (Tian et al., 2024) employs multiple parallel low-rank branches to capture modality-specific adaptations, while DoRA (Liu et al., 2024a) decouples direction and magnitude to enhance expressivity under low-rank constraints. MokA (Wei et al., 2025) introduces modular LoRA components with cross-modal interaction mechanisms for flexible knowledge aggregation. In contrast, MoE-based approaches (Dou et al., 2023; Abouelenin et al., 2025) improve adaptability via dynamic routing. LoRAMoE (Dou et al., 2023) integrates LoRA into an MoE framework with gating-based expert selection, and Phi-4-Multimodal (Abouelenin et al., 2025) further scales this paradigm by conditioning expert routing on modality and context, enabling more flexible cross-modal modeling.

Overall, existing multimodal LoRA methods rely on explicit low-rank designs with either fixed structures or routing-based variants, which tightly tie adaptation capacity to the explicit parameterization of low-rank factors and thereby limit its flexible allocation across modalities.

Feature Hashing: Feature hashing (Shi et al., 2009; Weinberger et al., 2009), aka, hashing trick, maps high-dimensional sparse features into a fixed-size space via hash functions, eliminating the need for an explicit feature dictionary. By incorporating randomized signed hashing, it approximately preserves inner products in expectation, enabling efficient and collision-tolerant learning over the compressed space. Feature hashing has become a practical technique for large-scale sparse learning, enabling memory-efficient modeling in tasks such as CTR prediction (Weinberger et al., 2009), collaborative filtering (Weinberger et al., 2009) by eliminating explicit feature dictionaries. It also supports scalable multi-task learning by allowing different tasks to share a common hash space with minimal overhead (Weinberger et al., 2009). This hashing-based parameter sharing mechanism has further been adopted in neural network compression (Chen et al., 2015). Theoretically, feature hashing provides an unbiased estimator of inner products while approximately preserving geometric structure, with variance introduced by collisions remaining well-controlled (Shi et al., 2009). These favorable properties provide a theoretical basis for using randomized parameter sharing mechanisms, suggesting that compact hashed representations can retain essential geometric structure while significantly reducing dimensionality.

6 Conclusion

In this paper, we propose feature hashing-based multimodal LoRA (HoRA), an implicit parameterization framework that reconstructs modality-specific low-rank adaptations from compact parameter buckets through feature hashing. By controlling the bucket size independently for each modality, HoRA provides an additional degree of freedom for modality-wise parameter allocation within a fixed-rank formulation. Theoretically, we characterize the approximation and optimization behavior of HoRA by establishing the unbiasedness of feature hashing-based reconstruction, deriving bounds on the approximation gap and hashing-induced gradient bias, and characterizing the convergence behavior of the resulting optimization process. Extensive experiments across diverse multimodal benchmarks demonstrate the effectiveness of HoRA and its flexibility for modality-wise parameter allocation.

Limitations: Despite its advantages, HoRA has several limitations. First, feature hashing-based parameter sharing inevitably introduces collisions, which may lead to information interference and limit representation fidelity, especially under small bucket sizes. Second, as discussed earlier, how to dynamically or heuristically determine the parameter allocation for each modality remains an important direction for future research.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing and error checking during the manuscript preparation process. We have not used generative AI tools for idea generation, method design, theoretical derivation or experimental design. Additionally, we used generative AI tools for preliminary proofreading and review of the manuscript. We have reviewed all AI-assisted work:

- We used LLMs to check the manuscript for grammatical errors and spelling errors.
- We used LLMs to assist with debugging and correcting experimental code.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve any ethical issues and therefore does not require a specific ethics statement.

REPRODUCIBILITY STATEMENT

We have provided an anonymous code repository for this work, available at: `https://anonymous.4open.science/r/HoRA-E512`. We have also provided hash-function implementation details in Appendix C.1; data processing and data splits in Appendix C.2; and training hyperparameters, random-seed settings, and hardware specifications in Appendix C.3 and Table II.

AUTHOR CONTRIBUTIONS

N/A.

ACKNOWLEDGMENTS

N/A.

REFERENCES

Soroush Abbasi Koohpayegani, KL Navaneet, Parsa Nooralinejad, Soheil Kolouri, and Hamed Pirsiavash. Nola: Compressing lora using linear combination of random basis. In *International Conference on Learning Representations*, volume 2024, pp. 23610–23626, 2024.

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, Dong Chen, Dongdong Chen, Jun-Kun Chen, Weizhu Chen, Yen-Chun Chen, Yi-ling Chen, Qi Dai, Xiyang Dai, Ruchao Fan, Mei Gao, Min Gao, Amit Garg, Abhishek Goswami, Junheng Hao, Amr Hendy, Yuxuan Hu, Xin Jin, Mahmoud Khademi, Dongwoo Kim, Young Jin Kim, Gina Lee, Jinyu Li, Yunsheng Li, Chen Liang, Xihui Lin, Zeqi Lin, Mengchen Liu, Yang Liu, Gilsinia Lopez, Chong Luo, Piyush Madan, Vadim Mazalov, Arindam Mitra, Ali Mousavi, Anh Nguyen, Jing Pan, Daniel Perez-Becker, Jacob Platin, Thomas Portet, Kai Qiu, Bo Ren, Liliang Ren, Sambuddha Roy, Ning Shang, Yelong Shen, Saksham Singhal, Subhojit Som, Xia Song, Tetyana Sych, Praneetha Vaddamanu, Shuohang Wang, Yiming Wang, Zhenghao Wang, Haibin Wu, Haoran Xu, Weijian Xu, Yifan Yang, Ziyi Yang, Donghan Yu, Ishmam Zabir, Jianwen Zhang, Li Lyna Zhang, Yunan Zhang, and Xiren Zhou. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *CoRR*, abs/2503.01743, 2025.

Jean-Philippe Aumasson and Daniel J Bernstein. Siphash: a fast short-input prf. In *ICCI*, pp. 489–508, 2012.

Sanyuan Chen, Yu Wu, Chengyi Wang, Shujie Liu, Daniel Tompkins, Zhuo Chen, Wanxiang Che, Xiangzhan Yu, and Furu Wei. Beats: Audio pre-training with acoustic tokenizers. In *ICML*, Proceedings of Machine Learning Research, pp. 5178–5193. PMLR, 2023.

Wenlin Chen, James T. Wilson, Stephen Tyree, Kilian Q. Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *ICML*, JMLR Workshop and Conference Proceedings, pp. 2285–2294. JMLR.org, 2015.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. Loramoe: Alleviate world knowledge forgetting in large language models via moe-style plugin. *arXiv preprint arXiv:2312.09979*, 2023.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In *ICML*, Proceedings of Machine Learning Research, pp. 8469–8488. PMLR, 2023.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Zhenyu Qiu, Wei Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, and Rongrong Ji. MME: A comprehensive evaluation benchmark for multimodal large language models. *CoRR*, abs/2306.13394, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778. IEEE Computer Society, 2016.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*. OpenReview.net, 2022.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, Proceedings of Machine Learning Research, pp. 4904–4916. PMLR, 2021.

Lior Kamma, Casper Benjamin Freksen, and Kasper Green Larsen. Fully understanding the hashing trick. In *NeurIPS*, pp. 5394–5404, 2018.

Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *NAACL-HLT*, 2019.

Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. Seed-bench: Benchmarking multimodal large language models. In *CVPR*, pp. 13299–13308. IEEE, 2024.

Guangyao Li, Yake Wei, Yapeng Tian, Chenliang Xu, Ji-Rong Wen, and Di Hu. Learning to answer questions in dynamic audio-visual scenarios. In *CVPR*, pp. 19086–19096. IEEE, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, Proceedings of Machine Learning Research, pp. 19730–19742. PMLR, 2023a.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *EMNLP*, pp. 292–305. Association for Computational Linguistics, 2023b.

Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *EMNLP*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *ICML*, Proceedings of Machine Learning Research, pp. 32100–32121. PMLR / OpenReview.net, 2024a.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player? In *ECCV*, Lecture Notes in Computer Science, pp. 216–233. Springer, 2024b.

Alejo Lopez-Avila and Jinhua Du. A survey on large language models in multimodal recommender systems. *CoRR*, abs/2505.09777, 2025.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-okvqa: A benchmark for visual question answering using world knowledge. In *ECCV*, 2022.

Hajime Senuma. mmh3: A python extension for murmurhash3. *JOSS*, 10(105):6124, 2025.

Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alexander J. Smola, Alexander L. Strehl, and Vishy Vishwanathan. Hash kernels. In David A. Van Dyk and Max Welling (eds.), *AIS*, JMLR Proceedings, pp. 496–503. JMLR.org, 2009.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Cheng-Zhong Xu. Hydralora: An asymmetric lora architecture for efficient fine-tuning. In *NeurIPS*, 2024.

Yapeng Tian, Jing Shi, Bochen Li, Zhiyao Duan, and Chenliang Xu. Audio-visual event localization in unconstrained videos. In *ECCV*, Lecture Notes in Computer Science, pp. 252–268. Springer, 2018.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

Yake Wei, Yu Miao, Dongzhan Zhou, and Di Hu. Moka: Multimodal low-rank adaptation for mllms. *CoRR*, abs/2506.05191, 2025.

Kilian Q. Weinberger, Anirban Dasgupta, John Langford, Alexander J. Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *ICML*, volume 382 of *ACM International Conference Proceeding Series*, pp. 1113–1120. ACM, 2009.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *CoRR*, abs/2407.10671, 2024.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *ICML*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *ICLR*. OpenReview.net, 2023.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *ICLR*. OpenReview.net, 2024.

A NOTATION AND DEFINITIONS

In this section, we introduce the notation used throughout the paper and summarize the main symbols. We use boldface lowercase letters like $\mathbf{a}$ to denote vectors. Boldface uppercase letters like $\mathbf{A}$ are used to denote matrix, and the element in the i-th row and j-th column is denoted as $A_{i,j}$. The i-th row of $\mathbf{A}$ is denoted as $\mathbf{A}_{i,:}$ and the j-th column of $\mathbf{A}$ is denoted as $\mathbf{A}_{:,j}$. Furthermore, $P(\cdot)$ denotes the probability of an event occurring. The symbol $\odot$ indicates the element-wise product. $\mathbb{E}_{\xi}[\cdot]$ denotes the expectation of a random variable ξ with respect to its underlying distribution. And $\mathbb{1}(\cdot)$ denotes the indicator function, which is defined as:

$$\mathbb{1}(condition) = \begin{cases} 1, & \text{if } condition \text{ is true,} \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Furthermore, we present the notations we used in this paper in Tab. I.

Table I: Notation description.

Notation	Description
d, k	The dimensions of parameters.
$\mathbf{A}, \mathbf{B}$	Low-rank adaptation matrices.
r	The dimension of LoRA matrices.
$\mathbf{W}$	Parameters of large models.
$\Delta\mathbf{W}$	The weight update.
$\widehat{\mathbf{W}}$	The updated parameters with LoRA strategy.
a, v	Audio modality and visual modality indicator.
$\mathbf{A}^o$	Modality-specific low-rank adaptation matrix.
h^o	Modality-specific index hash function.
ξ^o	Modality-specific signed hash function.
$\mathbf{W}^o$	Modality-specific parameters bucket.
M^o	Bucket size.
$P(\cdot)$	Probability of event.
$\widehat{\mathbf{A}}^o$	Implicit low-rank adaptation matrix.
$\mathbf{x} = [\mathbf{x}^a, \mathbf{x}^v]$	Input example.
$\mathbf{h}$	Output example.
$\mathcal{L}$	Loss function.
$\Delta\widehat{\mathbf{W}}$	The updated parameters with HoRA strategy.
ϵ	The random seed for hash function.
ρ	The proportion of effective parameters.

B PROOFS OF MAIN THEORETICAL RESULTS

In this section, we give the detailed proof of main theoretical results in the main text. Our theoretical analysis consists of two parts: the impact of feature hashing on parameterization and its effect on the training process.

B.1 THE IMPACT OF FEATURE HASHING ON PARAMETERIZATION

We first present the theoretical analysis of the impact of feature hashing on parameterization. We introduce a set of assumptions on the hash functions underlying the feature hashing (Chen et al., 2015) framework. Formally, we make the following assumptions on the independence of the hash functions:

Assumption B.1. *(Independence of Index and Signed Hash Functions) For each modality, the index hash function $h^o(\cdot,\cdot)$ and the signed hash function $\xi^o(\cdot,\cdot)$ are independent.*

Assumption B.2. *(Independence across Modalities) The index hash functions $h^o(\cdot,\cdot)$ and the signed hash functions $\xi^o(\cdot,\cdot)$ are mutually independent across different modalities.*

Based on the above assumptions, we first establish theoretical results in the unimodal setting, laying the groundwork for analyzing approximation bound and capacity allocation in the multimodal case.

Proposition B.3. *(Unimodal Unbiasedness) Under the Assumption B.1 and B.2, for a matrix* $\mathbf{B}$*, we have:*

$$\mathbb{E}_{h^o,\xi^o}\left[\widehat{\mathbf{A}}^o\right] = \mathbf{A}^o, \quad \mathbb{E}_{h^o,\xi^o}\left[\mathbf{B}\widehat{\mathbf{A}}^o\right] = \mathbf{B}\mathbf{A}^o, \tag{20}$$

Proposition B.4. *(Unimodal Approximation Bound) Under the Assumption B.1 and B.2, the gap introduced by approximation is bounded:*

$$\mathbb{E}\left[\|\mathbf{B}\widehat{\mathbf{A}}^o - \mathbf{B}\mathbf{A}^o\|^2\right] \leq \|\mathbf{B}\|^2 \cdot \sum_{i,j} \sum_{(p,q)\neq(i,j)} \left[A^o_{p,q}\right]^2 P(h^o(p,q) = h^o(i,j)) \tag{21}$$

If $h^o(\cdot)$ *is uniform, we have:*

$$\mathbb{E}\left[\|\mathbf{B}\widehat{\mathbf{A}}^o - \mathbf{B}\mathbf{A}^o\|^2\right] \leq \|\mathbf{A}^o\|^2\|\mathbf{B}\|^2\frac{rk-1}{M^o}. \tag{22}$$

We first give the detailed proof of Proposition B.3.

Proof. The approximation matrix is defined as[2]:

$$\widehat{A}_{i,j} = \xi(i,j)W_{h(i,j)} = \xi(i,j)\sum_{(p,q):h(p,q)=h(i,j)} \xi(p,q)A_{p,q}. \tag{23}$$

We have:

$$\widehat{A}_{i,j} = \xi(i,j)\sum_{(p,q):h(p,q)=h(i,j)} \xi(p,q)A_{p,q} \tag{24}$$

$$= \xi(i,j)\xi(i,j)A_{i,j} + \xi(i,j)\sum_{\substack{h(p,q)=h(i,j)\\(i,j)\neq(p,q)}} \xi(p,q)A_{p,q} \tag{25}$$

$$= A_{i,j} + \xi(i,j)\sum_{\substack{h(p,q)=h(i,j)\\(i,j)\neq(p,q)}} \xi(p,q)A_{p,q} \tag{26}$$

$$= A_{i,j} + \sum_{\substack{h(p,q)=h(i,j)\\(i,j)\neq(p,q)}} \xi(i,j)\xi(p,q)A_{p,q} \tag{27}$$

$$= A_{i,j} + \sum_{(i,j)\neq(p,q)} A_{p,q}\xi(i,j)\xi(p,q)\mathbb{1}(h(p,q) = h(i,j)), \tag{28}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function.

Taking expectation:

$$\mathbb{E}[\widehat{A}_{i,j}] = A_{i,j} + \sum_{(i,j)\neq(p,q)} A_{p,q}\mathbb{E}\left[\xi(i,j)\xi(p,q)\mathbb{1}(h(p,q) = h(i,j))\right] \tag{29}$$

$$= A_{i,j} + \sum_{(i,j)\neq(p,q)} A_{p,q}\mathbb{E}\left[\xi(i,j)\xi(p,q)\right]\mathbb{E}\left[\mathbb{1}(h(p,q) = h(i,j))\right] \tag{30}$$

$$= A_{i,j} + \sum_{(i,j)\neq(p,q)} A_{p,q}\mathbb{E}\left[\xi(i,j)\right]\mathbb{E}\left[\xi(p,q)\right]\mathbb{E}\left[\mathbb{1}(h(p,q) = h(i,j))\right] \tag{31}$$

$$= A_{i,j}. \tag{32}$$

Then we have:

$$\mathbb{E}_{h,\xi}[\widehat{\mathbf{A}}] = \mathbf{A}, \tag{33}$$

$$\mathbb{E}_{h,\xi}[\mathbf{B}\widehat{\mathbf{A}}] = \mathbf{B}\mathbb{E}_{h,\xi}[\widehat{\mathbf{A}}] = \mathbf{B}\mathbf{A}. \tag{34}$$

This completes the proof. □

[2]For simplicity, we omit the superscript o that indicates the modality in the unimodal theoretical analysis.

Then, we give the proof of Proposition B.4.

Proof. We first have:

$$\widehat{A}_{i,j} = A_{i,j} + \sum_{\substack{(p,q)\neq(i,j)\\ h(p,q)=h(i,j)}} \xi(p,q)\xi(i,j)A_{p,q}. \tag{35}$$

Define $\Delta_{ij} = \sum_{\substack{(p,q)\neq(i,j)\\ h(p,q)=h(i,j)}} \xi(p,q)\xi(i,j)A_{p,q}$. We have $\mathbb{E}[\Delta_{ij}] = 0$. Hence we have:

$$Var(\widehat{A}_{i,j}) \tag{36}$$

$$= \mathbb{E}\left[\left(\widehat{A}_{i,j} - \mathbb{E}\left[\widehat{A}_{i,j}\right]\right)^2\right] \tag{37}$$

$$= \mathbb{E}\left[\Delta_{i,j}^2\right] \tag{38}$$

$$= \mathbb{E}\left[\left(\sum_{\substack{(p,q)\neq(i,j)\\ h(p,q)=h(i,j)}} \xi(p,q)\xi(i,j)A_{p,q}\right)^2\right] \tag{39}$$

$$= \sum_{(p,q)\neq(i,j)} \sum_{(p',q')\neq(i,j)} A_{p,q}A_{p',q'}\mathbb{E}\left[\xi(i,j)\xi(p,q)\xi(i,j)\xi(p',q')\mathbb{1}(h(p,q)=h(i,j))\mathbb{1}(h(p',q')=h(i,j))\right] \tag{40}$$

$$= \sum_{(p,q)\neq(i,j)} \sum_{(p',q')\neq(i,j)} A_{p,q}A_{p',q'}\mathbb{E}\left[\xi(p,q)\xi(p',q')\mathbb{1}(h(p,q)=h(i,j))\mathbb{1}(h(p',q')=h(i,j))\right] \tag{41}$$

$$= \sum_{(p,q)\neq(i,j)} \sum_{(p',q')\neq(i,j)} A_{p,q}A_{p',q'}\mathbb{E}\left[\xi(p,q)\xi(p',q')\right]\mathbb{E}\left[\mathbb{1}(h(p,q)=h(i,j))\mathbb{1}(h(p',q')=h(i,j))\right] \tag{42}$$

If $(p,q) \neq (p',q')$, we have:

$$\mathbb{E}\left[\xi(p,q)\xi(p',q')\right] = \mathbb{E}\left[\xi(p,q)\right]\mathbb{E}\left[\xi(p',q')\right] = 0. \tag{43}$$

If $(p,q) = (p',q')$, $\xi(p,q)^2 = 1$, we have:

$$A_{p,q}A_{p',q'}\mathbb{E}\left[\xi(p,q)\xi(p',q')\right]\mathbb{E}\left[\mathbb{1}(h(p,q)=h(i,j))\mathbb{1}(h(p',q')=h(i,j))\right] \tag{44}$$

$$= A_{p,q}^2\mathbb{E}\left[\mathbb{1}(h(p,q)=h(i,j))\right] \tag{45}$$

$$= A_{p,q}^2 P(h(p,q)=h(i,j)) \tag{46}$$

In summary:

$$Var(\widehat{A}_{i,j}) = \sum_{(p,q)\neq(i,j)} A_{p,q}^2 P(h(p,q)=h(i,j)) \tag{47}$$

Consider the error introduced by approximation:

$$\mathbb{E}\left[\|\mathbf{B}\widehat{\mathbf{A}} - \mathbf{B}\mathbf{A}\|^2\right] \leq \mathbb{E}\left[\|\mathbf{B}\|^2\|\widehat{\mathbf{A}} - \mathbf{A}\|^2\right] \tag{48}$$

$$= \|\mathbf{B}\|^2\mathbb{E}\left[\|\widehat{\mathbf{A}} - \mathbf{A}\|^2\right] \tag{49}$$

$$= \|\mathbf{B}\|^2 \sum_{i,j} Var(\widehat{A}_{ij}) \tag{50}$$

$$= \|\mathbf{B}\|^2 \sum_{i,j} \sum_{(p,q)\neq(i,j)} A_{p,q}^2 P(h(p,q)=h(i,j)) \tag{51}$$

If h is uniform, we have:

$$Var(\widehat{A}_{i,j}) = \sum_{(p,q)\neq(i,j)} A_{p,q}^2 P(h(p,q)=h(i,j)) \tag{52}$$

$$= \frac{1}{M}\left(\|\mathbf{A}\|^2 - A_{ij}^2\right) \tag{53}$$

Then we have:

$$\mathbb{E}\left[\|\mathbf{B}\widehat{\mathbf{A}} - \mathbf{B}\mathbf{A}\|^2\right] \leq \|\mathbf{B}\|^2 \sum_{i,j} Var(\widehat{A}_{ij}) \tag{54}$$

$$= \frac{1}{M}\|\mathbf{B}\|^2 \sum_{i,j} \left(\|\mathbf{A}\|^2 - A_{ij}^2\right) \tag{55}$$

$$= \frac{1}{M}\|\mathbf{B}\|^2 \left(rk\|\mathbf{A}\|^2 - \sum_{i,j} A_{ij}^2\right) \tag{56}$$

$$= \frac{1}{M}\|\mathbf{B}\|^2 \left(rk\|\mathbf{A}\|^2 - \|\mathbf{A}\|^2\right) \tag{57}$$

$$= \|\mathbf{A}\|^2\|\mathbf{B}\|^2 \frac{rk-1}{M} \tag{58}$$

This completes the proof. □

Building upon the unimodal analysis, we now present theoretical results under the multimodal setting, focusing on the multimodal approximation bound.

Proposition B.5. *(Multimodal Unbiasedness, formal) Under the Assumption B.1 and B.2, for a matrix* $\mathbf{B}$*, we have:*

$$\mathbb{E}\left[\Delta\widehat{\mathbf{W}}\right] = \Delta\mathbf{W}, \tag{59}$$

where $\Delta\mathbf{W} = \sum_{o\in\{a,v\}} \mathbf{B}\mathbf{A}^o$ *and* $\Delta\widehat{\mathbf{W}} = \sum_{o\in\{a,v\}} \mathbf{B}\widehat{\mathbf{A}}^o$.

Theorem B.6. *(Multimodal Approximation Bound, formal) Under the Assumption B.1 and B.2, the gap introduced by multimodal approximation is bounded:*

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] \leq \|\mathbf{B}\|^2 \cdot \sum_{a\in\{a,v\}} \sum_{i,j} \sum_{(p,q)\neq(i,j)} \left[A_{p,q}^o\right]^2 P(h^o(p,q) = h^o(i,j)) \tag{60}$$

If $h^o(\cdot)$ *is uniform, we have:*

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] \leq \sum_{o\in\{a,v\}} \|\mathbf{A}^o\|^2\|\mathbf{B}\|^2 \frac{rk-1}{M^o}. \tag{61}$$

We then give the detailed proof of multimodal theoretic results. We first give the proof of Theorem B.5.

Proof. The definition of $\Delta\widehat{\mathbf{W}}$ is formed as:

$$\Delta\widehat{\mathbf{W}} = \sum\nolimits_{o\in\{a,v\}} \mathbf{B}\widehat{\mathbf{A}}^o \tag{62}$$

Taking expectation, we have:

$$\mathbb{E}\left[\Delta\widehat{\mathbf{W}}\right] = \mathbb{E}\left[\sum\nolimits_{o\in\{a,v\}} \mathbf{B}\widehat{\mathbf{A}}^o\right] \tag{63}$$

$$= \sum\nolimits_{o\in\{a,v\}} \mathbf{B}\mathbb{E}\left[\widehat{\mathbf{A}}^o\right] \tag{64}$$

$$= \sum\nolimits_{o\in\{a,v\}} \mathbf{B}\mathbf{A}^o \tag{65}$$

$$= \Delta\mathbf{W}. \tag{66}$$

This completes the proof. □

Then we give the proof of Theorem B.6.

Proof. Define the error introduced by approximation:

$$\mathbf{E}^o = \mathbf{B}(\widehat{\mathbf{A}}^o - \mathbf{A}^o) \tag{67}$$

Based on Assumption B.2, we have:

$$\mathbb{E}\left\langle \mathbf{E}^a, \mathbf{E}^a \right\rangle = 0,\ \mathbb{E}\left\langle \mathbf{E}^a, \mathbf{E}^v \right\rangle = 0,\ \mathbb{E}\left\langle \mathbf{E}^a, \mathbf{E}^v \right\rangle = 0. \tag{68}$$

Then we have:

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] = \mathbb{E}\left[\|\sum_{o\in\{a,v\}} \mathbf{B}(\widehat{\mathbf{A}}^o - \mathbf{A})\|^2\right] \tag{69}$$

$$= \mathbb{E}\left[\|\sum_{o\in\{a,v\}} \mathbf{E}^o\|^2\right] \tag{70}$$

$$= \mathbb{E}\|\mathbf{E}^a\|^2 + 2\mathbb{E}\left\langle \mathbf{E}^a, \mathbf{E}^v \right\rangle + \mathbb{E}\|\mathbf{E}^v\|^2 \tag{71}$$

$$= \sum_{o\in\{a,v\}} \mathbb{E}\|\mathbf{E}^o\|^2 \tag{72}$$

$$\leq \|\mathbf{B}\|^2 \cdot \sum_{a\in\{a,v\}} \sum_{i,j} \sum_{(p,q)\neq(i,j)} [A^o_{p,q}]^2 P(h^o(p,q) = h^o(i,j)) \tag{73}$$

If $\{h^o\}_{o\in\{a,v\}}$ are uniform, we have:

$$\mathbb{E}\left[\|\Delta\widehat{\mathbf{W}} - \Delta\mathbf{W}\|^2\right] \leq \sum\nolimits_{a\in\{a,v\}} \|\mathbf{A}^o\|^2\|\mathbf{B}\|^2 \frac{rk-1}{M^o}. \tag{74}$$

This completes the proof. □

B.2 The Impact of Feature Hashing on the Training Process

We next analyze the impact of introducing feature hashing on the training process. For convenience, we redefine several variables used in this analysis. Given the loss function $\mathcal{L}(\Delta\mathbf{W})$, its gradient at t-th iteration is defined as:

$$g_t^\star = \nabla\mathcal{L}(\Delta\mathbf{W}_t). \tag{75}$$

$g_t^\star$ is also referred to as the full gradient. During training, HoRA may be subject to two sources of randomness: the randomness introduced by feature hashing and the randomness arising from stochastic sampling. We denote the former as:

$$\forall o \in \{a, v\}, \zeta^o = \{h^o, \xi^o\}, \tag{76}$$

$$\zeta = \{\zeta^a, \zeta^v\}. \tag{77}$$

Accordingly, we redefine the update variables in HoRA as functions of the random variable ζ, denoted by:

$$\Delta\widehat{\mathbf{W}}(\zeta) = \sum_{o\in\{a,v\}} \mathbf{B}\mathbf{A}^o(\zeta^o). \tag{78}$$

For notational simplicity, we denote $\Delta\widehat{\mathbf{W}}(\zeta)$ simply as $\Delta\mathbf{W}_\zeta$ hereafter. The gradient at t-th iteration after applying feature hashing is defined as:

$$g_t^h = \nabla\mathcal{L}(\Delta\mathbf{W}_{\zeta,t}). \tag{79}$$

We first analyze the discrepancy between the full gradient $g_t^\star$ and the gradient affected by feature hashing, i.e., g_t^h. We define the hashing-induced gradient bias as:

$$b_t = \mathbb{E}_{\zeta_t}\left[g_t^h \mid \theta_t\right] - g_t^\star, \tag{80}$$

where θ_t denotes the trainable parameters of HoRA at iteration t.

Furthermore, to facilitate this analysis, we make the following assumption on the objective function:

Assumption B.7. *(Smoothness) We assume that the objective function $\mathcal{L}(\cdot)$ is L-smooth:*

$$\|\nabla\mathcal{L}(\theta_1) - \nabla\mathcal{L}(\theta_2)\| \leq L_g\|\theta_1 - \theta_2\|, \tag{81}$$

where L_g is a constant.

Then we obtain the following theorem:

Theorem B.8. *(Hashing-Induced Gradient Bias Bound, formal) Under Assumptions B.1, B.2, and B.7, if the index hash functions are uniform over their corresponding buckets, the gradient introduced by hashing-based implicit low-rank constrain satisfies:*

$$\mathbb{E}\left[\left\|g_t^\star - g_t^h\right\|^2\right] \leq \sum\nolimits_{o\in\{a,v\}} L_g^2\|\mathbf{A}_t^o\|^2\|\mathbf{B}_t\|^2\frac{rk-1}{M^o} = \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right), \tag{82}$$

where L_g, C^a and C^v are constant.

Consequently, the squared hashing-induced gradient bias satisfies:

$$\|b_t\|^2 \leq \sum\nolimits_{o\in\{a,v\}} L_g^2\|\mathbf{A}_t^o\|^2\|\mathbf{B}_t\|^2\frac{rk-1}{M^o} = \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right). \tag{83}$$

Proof. Since $\mathcal{L}$ is L-smooth:

$$\left\|g_t^\star - g_t^h\right\| = \left\|\nabla\mathcal{L}\left(\Delta\mathbf{W}_{\zeta,t}\right) - \nabla\mathcal{L}\left(\Delta\mathbf{W}_t\right)\right\| \leq L_g\left\|\Delta\mathbf{W}_{\zeta,t} - \Delta\mathbf{W}_t\right\|. \tag{84}$$

Taking conditional expectation for random variable ζ_t, we have:

$$\mathbb{E}_{\zeta_t}\left[\left\|g_t^\star - g_t^h\right\|^2 \mid \theta_t\right] \leq \mathbb{E}_{\zeta_t}\left[L_g^2\left\|\Delta\mathbf{W}_{\zeta,t} - \Delta\mathbf{W}_t\right\|^2\right], \tag{85}$$

$$\leq \sum\nolimits_{a\in\{a,v\}} L_g^2\|\mathbf{A}^o\|^2\|\mathbf{B}\|^2\frac{rk-1}{M^o}. \tag{86}$$

According to the definition of b_t, we have:

$$b_t = \mathbb{E}_{\zeta_t}\left[g_t^h \mid \theta_t\right] - g_t^\star \tag{87}$$

$$= \mathbb{E}_{\zeta_t}\left[g_t^h - g_t^\star \mid \theta_t\right] \tag{88}$$

By Jensen's inequality, we have:

$$\|b_t\|^2 = \left\|\mathbb{E}_{\zeta_t}\left[g_t^h - g_t^\star \mid \theta_t\right]\right\|^2 \tag{89}$$

$$\leq \mathbb{E}_{\zeta_t}\left\|\left[g_t^h - g_t^\star \mid \theta_t\right]\right\|^2 \tag{90}$$

$$\leq \sum\nolimits_{a\in\{a,v\}} L_g^2\|\mathbf{A}^o\|^2\|\mathbf{B}\|^2\frac{rk-1}{M^o}. \tag{91}$$

This completes the proof. □

We next analyze the impact of feature hashing on the convergence of the algorithm during the actual training process. We define the gradient during training at iteration t as:

$$g_t = \nabla\mathcal{L}(\Delta\mathbf{W}_{\zeta,t}; \mathcal{B}_t), \tag{92}$$

where $\mathcal{B}_t$ denotes the mini-batch via stochastic sampling. The actual gradient g_t during training is affected by both hashing-induced gradient bias b_t and the zero-mean stochastic error arising from the two stochastic processes, namely feature hashing and stochastic sampling. The latter is defined as:

$$\varepsilon_t = g_t - \mathbb{E}\left[g_t \mid \theta_t\right]. \tag{93}$$

For ε_t, we have:

$$\mathbb{E}_{\zeta_t,\mathcal{B}_t}\left[\varepsilon_t\right] = 0, \tag{94}$$

and

$$\mathbb{E}_{\zeta_t,\mathcal{B}_t}\left[\|\varepsilon_t\|^2 \mid \theta_t\right] \leq \sigma^2. \tag{95}$$

Then we have the following gradient decomposition:

$$g_t = \underbrace{g_t^\star}_{\text{full gradient}} + \underbrace{b_t}_{\text{hashing-induced bias}} + \underbrace{\varepsilon_t}_{\text{stochastic error}}. \tag{96}$$

Furthermore, to analyze the impact of feature hashing on the training process, we make the following assumption:

Assumption B.9. *(Low Bounded Objective) We assume that that the optimal value of $\mathcal{L}(\theta)$ is bounded. That is, there exists a $\theta^\star$, where $\mathcal{L}(\theta^\star) > -\infty$:*

$$\mathcal{L}(\theta) \geq \mathcal{L}(\theta^\star), \tag{97}$$

where θ denotes the parameters of $\mathcal{L}(\cdot)$.

Then we obtain the following corollary:

Corollary B.10. *(Convergence of HoRA, formal) Suppose Assumptions B.1, B.2, B.7, B.9 hold and $T \geq 16L_g^2$, if $\{h^o\}_{o\in\{a,v\}}$ are uniform and set the learning rate $\eta = \frac{1}{\sqrt{T}}$, then:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla\mathcal{L}(\theta_t)\|^2 \leq \frac{4\Delta\mathcal{L} + 2L_g\sigma^2}{\sqrt{T}} + \left(2 + \frac{4L_g}{\sqrt{T}}\right)\delta_M = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}\left(\frac{1}{M^a} + \frac{1}{M^v}\right), \tag{98}$$

where $\delta_M = \sum_{a\in\{a,v\}} L_g^2\|\mathbf{A}_t^o\|^2\|\mathbf{B}_t\|^2\frac{rk-1}{M^o}$, $\Delta\mathcal{L} = \mathcal{L}(\theta_0) - \mathcal{L}(\theta^\star)$.

Proof. During training process at t-th iteration, the updating rule can be formulated as:

$$\theta_{t+1} = \theta_t - \eta g_t, \tag{99}$$

where θ_t denotes the trainable parameters, η is the learning rate. Based on Theorem B.8, we have:

$$\mathbb{E}\left[\|b_t\|^2\right] \leq \delta_M, \tag{100}$$

where $\delta_M = \sum_{a\in\{a,v\}} L_g^2\|\mathbf{A}_t^o\|^2\|\mathbf{B}_t\|^2\frac{rk-1}{M^o}$.

Since $\mathcal{L}(\cdot)$ is L_g-smooth, we have:

$$\mathcal{L}_{t+1} \leq \mathcal{L}_t + \langle\nabla\mathcal{L}_t, \theta_{t+1} - \theta_t\rangle + \frac{L_g}{2}\|\theta_{t+1} - \theta_t\|^2, \tag{101}$$

where we use $\mathcal{L}_t$ to denote $\mathcal{L}(\theta_t)$ for brevity. Based on updating rule in Equation (99), we have:

$$\mathcal{L}_{t+1} \leq \mathcal{L}_t - \eta\langle g_t^\star, g_t\rangle + \frac{L_g\eta^2}{2}\|g_t\|^2. \tag{102}$$

Taking the conditional expectation with respect to θ_t, and combining the definition of b_t, we have:

$$\begin{aligned}
\mathbb{E}\left[\langle g_t^\star, g_t\rangle \mid \theta_t\right] &= \langle g_t^\star, \mathbb{E}\left[g_t \mid \theta_t\right]\rangle &(103)\\
&= \langle g_t^\star, g_t^\star + b_t\rangle &(104)\\
&= \|g_t^\star\|^2 + \langle g_t^\star, b_t\rangle. &(105)
\end{aligned}$$

By Young's inequality, we have:

$$\langle g_t^\star, b_t\rangle \geq -\frac{1}{2}\|g_t^\star\|^2 - \frac{1}{2}\|b_t\|^2. \tag{106}$$

Therefore, we have:

$$\begin{aligned}
\mathbb{E}\left[\langle g_t^\star, g_t\rangle \mid \theta_t\right] &= \|g_t^\star\|^2 + \langle g_t^\star, b_t\rangle &(107)\\
&\geq \frac{1}{2}\|g_t^\star\|^2 - \frac{1}{2}\|b_t\|^2. &(108)\\
-\eta\mathbb{E}\left[\langle g_t^\star, g_t\rangle \mid \theta_t\right] &\leq -\frac{\eta}{2}\|g_t^\star\|^2 + \frac{\eta}{2}\|b_t\|^2. &(109)
\end{aligned}$$

Next, since $g_t = g_t^\star + b_t + \varepsilon_t$ and $\mathbb{E}[\varepsilon_t \mid \theta_t] = 0$, we have:

$$\mathbb{E}\left[\|g_t\|^2 \mid \theta_t\right] = \|g_t^\star + b_t\|^2 + \|\varepsilon_t\|^2 \tag{110}$$

$$\leq 2\|g_t^\star\|^2 + 2\|b_t\|^2 + \sigma^2 \tag{111}$$

$$= 2\|g_t^\star\|^2 + 2\|b_t\|^2 + \sigma^2. \tag{112}$$

Taking expectation for Equation (102) and combining Equation (109), (112), (100), we have:

$$\mathbb{E}\left[\mathcal{L}_{t+1} \mid \theta_t\right] \leq \mathbb{E}\left[\mathcal{L}_t \mid \theta_t\right] - \eta\mathbb{E}\left[\langle g_t^\star, g_t\rangle \mid \theta_t\right] + \frac{L_g\eta^2}{2}\mathbb{E}\left[\|g_t\|^2 \mid \theta_t\right] \tag{113}$$

$$\leq \mathbb{E}\left[\mathcal{L}_t \mid \theta_t\right] - \frac{\eta}{2}\|g_t^\star\|^2 + \frac{\eta}{2}\|b_t\|^2 + L_g\eta^2\|g_t^\star\|^2 + 2\|b_t\|^2 + \frac{L_g\eta^2}{2}\sigma^2 \tag{114}$$

$$= \mathbb{E}\left[\mathcal{L}_t \mid \theta_t\right] - \left(\frac{\eta}{2} - L_g\eta^2\right)\|g_t^\star\|^2 + \left(\frac{\eta}{2} + L_g\eta^2\right)\|b_t\|^2 + \frac{L_g\eta^2}{2}\sigma^2 \tag{115}$$

$$\leq \mathbb{E}\left[\mathcal{L}_t \mid \theta_t\right] - \left(\frac{\eta}{2} - L_g\eta^2\right)\|g_t^\star\|^2 + \left(\frac{\eta}{2} + L_g\eta^2\right)\delta_M + \frac{L_g\eta^2}{2}\sigma^2. \tag{116}$$

By setting: $\eta = \min\left\{\frac{1}{4L_g}, \frac{1}{\sqrt{T}}\right\}$, we obtain:

$$\frac{\eta}{2} - L_g\eta^2 \geq \frac{\eta}{4}. \tag{117}$$

Then we have:

$$\frac{\eta}{4}\mathbb{E}\left[\|g_t^\star\|^2\right] \leq \mathbb{E}\left[\mathcal{L}_t \mid \theta_t\right] - \mathbb{E}\left[\mathcal{L}_{t+1} \mid \theta_t\right] + \left(\frac{\eta}{2} + L_g\eta^2\right)\delta_M + \frac{L_g\eta^2}{2}\sigma^2. \tag{118}$$

By summing Equation (118) from $t = 0, \cdots, T-1$, we obtain:

$$\frac{\eta}{4}\sum_{t=0}^{T-1}\mathbb{E}\left[\|g_t^\star\|^2\right] \leq \mathcal{L}_0 - \mathbb{E}\left[\mathcal{L}_T\right] + T\left(\frac{\eta}{2} + L_g\eta^2\right)\delta_M + \frac{TL_g\eta^2}{2}\sigma^2. \tag{119}$$

Since: $\mathcal{L}(\theta_T) \geq \mathcal{L}(\theta^\star)$, we have:

$$\frac{\eta}{4}\sum_{t=0}^{T-1}\mathbb{E}\left[\|g_t^\star\|^2\right] \leq \mathcal{L}_0 - \mathcal{L}^\star + T\left(\frac{\eta}{2} + L_g\eta^2\right)\delta_M + \frac{TL_g\eta^2}{2}\sigma^2. \tag{120}$$

Then we obtain:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|g_t^\star\|^2\right] \leq \frac{4(\mathcal{L}_0 - \mathcal{L}^\star)}{\eta T} + (2 + 4L_g\eta)\,\delta_M + 2L_g\eta\sigma^2. \tag{121}$$

Combining the setting of learning rate η, we have:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|g_t^\star\|^2\right] \leq \begin{cases} \dfrac{4(\mathcal{L}_0 - \mathcal{L}^\star) + 2L_g\sigma^2}{\sqrt{T}} + \left(2 + \dfrac{4L_g}{\sqrt{T}}\right)\delta_M, & \text{if } T \geq 16L_g^2, \\ \dfrac{16L_g(\mathcal{L}_0 - \mathcal{L}^\star)}{T} + 3\delta_M + \dfrac{1}{2}\sigma^2. & \text{otherwise.} \end{cases} \tag{122}$$

This proves the stated convergence bound. □

C Additional Implementation Details

C.1 Hash Function Implementation Details

The index hash function and signed hash function play a critical role in HoRA. Here we provide their implementation details. We adopt the xxhash library[3] by Google and construct the hash keys

[3] https://code.google.com/p/xxhash/

based on the weight indices and layer identifiers, where the layer information is used to ensure the independent hash function across different layers. The specific implementation is shown below. With this implementation, we ensure both the individual independence of the index and signed hash functions, as well as their mutual independence.

```python
import xxhash

# definition of index hash function
def index_hash_func(w_dim, row_idx, col_idx, layer_id, hash_seed):
    hash_key = f'{row_idx}_{col_idx}idxW_M{layer_id}'
    hashed_index = xxhash.xxh32(hash_key, seed=hash_seed).intdigest() % w_dim
    return hashed_index

# definition of signed hash function
def signed_hash_func(row_idx, col_idx, layer_id, hash_seed, target_dtype):
    hash_key = f'{row_idx}_{col_idx}xiW_M{layer_id}'
    binary_val = xxhash.xxh32(hash_key, seed=hash_seed).intdigest() % 2
    signed_tensor = binary_val.float().add(1).mul(2).add(-3).to(target_dtype)
    return signed_tensor
```

We further provide the details of the hash functions implementation with SipHash and MurmurHash3. Specifically, SipHash processes the input message in 64-bit blocks, iteratively mixing them into a 256-bit internal state (composed of four 64-bit words) using a sequence of structured add-rotate-xor (ARX) rounds, known as SipRounds, followed by a finalization phase to produce a 64-bit output. The implementation is shown below.

```python
import siphashc

# helper function to derive a 16-byte key from seed
def seed_to_key(seed):
    return int(seed).to_bytes(16, byteorder="little", signed=False)

# definition of index hash function
def index_hash_func(w_dim, row_idx, col_idx, layer_id, hash_seed):
    hash_key = f'{row_idx}_{col_idx}idxW_M{layer_id}'
    key = seed_to_key(hash_seed)
    data = hash_key.encode("utf-8")
    hashed_index = siphashc.siphash(key, data) % w_dim
    return hashed_index

# definition of signed hash function
def signed_hash_func(row_idx, col_idx, layer_id, hash_seed, target_dtype):
    hash_key = f'{row_idx}_{col_idx}xiW_M{layer_id}'
    key = seed_to_key(hash_seed)
    data = hash_key.encode("utf-8")
    binary_val = siphashc.siphash(key, data) % 2
    signed_tensor = binary_val.float().add(1).mul(2).add(-3).to(target_dtype)
    return signed_tensor
```

MurmurHash3 processes input data in fixed-size blocks, applying a sequence of mixing operations-including multiplications with carefully chosen constants, bit rotations, and XOR operations-to achieve uniform distribution and strong avalanche behavior in the resulting hash values. The implementation is shown below.

```python
import mmh3

# definition of index hash function
def index_hash_func(w_dim, row_idx, col_idx, layer_id, hash_seed):
    hash_key = f'{row_idx}_{col_idx}idxW_M{layer_id}'
    hashed_index = mmh3.hash(hash_key, hash_seed, signed=False) % w_dim
    return hashed_index

# definition of signed hash function
def signed_hash_func(row_idx, col_idx, layer_id, hash_seed, target_dtype):
    hash_key = f'{row_idx}_{col_idx}xiW_M{layer_id}'
    binary_val = mmh3.hash(hash_key, hash_seed, signed=False) % 2
    signed_tensor = binary_val.float().add(1).mul(2).add(-3).to(target_dtype)
    return signed_tensor
```

C.2 Data Splitting

Here, we provide more details on the data partitioning strategy during training. The AVE dataset contains 4,143 videos covering 28 event categories. For this dataset, we divide each video into

10 temporal segments, and for each segment, we extract one video frame and the corresponding one-second audio segment, forming 10 aligned audio-video pairs. These pairs are then aligned with second-level event annotations, resulting in 4,020 labeled segments for testing. Then we utilize 3,339 samples for training and 402 samples for testing. The MUSIC-AVQA dataset comprises over 45K question-answer pairs across diverse templates. For MUSIC-AVQA dataset, we uniformly sample 10 video frames from each video, and extract 10 audio segments using sliding window with a duration of 2 seconds and a stride of 6 seconds, forming aligned audio-visual pairs. For bimodal setting, we utilize 169,712 samples to construct training set for a combined dataset comprising LLAVA-Instruction-150K and a 12K instruction subset from A-OKVQA following the setting of MokA (Wei et al., 2025). For each sample in the combined dataset, we convert multi-turn conversations into independent single-turn human-to-GPT pairs and resize the image to 224×224, yielding standardized training pairs. For MME dataset, we adopt the perception subset of the MME benchmark, consisting of 10 subtasks designed to assess visual-text perception capabilities. The model is then evaluated on MMBench, MME, POPE and SEED-Bench.

C.3 Parameter Settings

We provide detailed setting of experiments. Following the setting of MokA (Wei et al., 2025), we employ a two-stage experimental pipeline consisting of pretraining and fine-tuning. In terms of the overall architecture, we build upon a large language model (LLM) backbone and incorporate a Q-Former (Li et al., 2023a) to process visual and audio modalities. Specifically, we adopt the 7B variants of QWen2 (Yang et al., 2024) and LLAMA2 (Touvron et al., 2023) as the backbone LLMs. For the textual modality, inputs are first tokenized using the corresponding tokenizer of each LLM. For the visual modality, we employ a ViT (Dosovitskiy et al., 2021) encoder, followed by a visual Q-Former and an MLP projection layer to align visual representations with the textual space. The audio modality is processed in a similar manner: an audio encoder BEATs (Chen et al., 2023) is first applied, followed by an audio Q-Former and an MLP projection for modality alignment. Finally, all tokens from all modalities are concatenated and fed into the LLM as input. For trimodal setting, we utilize AudioCaps (Kim et al., 2019) and Video-LLAVA (Lin et al., 2024) as training set during pretraining stage, freezing the LLM while updating the Q-Former and MLP projection layers. And the number of epoch is set to be 1. The batch size and learning rate are set to be 16 and 10^{-4}, respectively. For bimodal datasets, we utilize Video-LLAVA (Lin et al., 2024) as training set for pretraining. The number of epoch is set to be 1. The batch size and learning rate are set to be 16 and 10^{-4}, respectively. During fine-tuning stage, we adopt the subset of AVE and MUSIC-AVQA datasets as training set for AVE and MUSIC-AVQA datasets, respectively. While for bimodal setting, we adopt a combined dataset comprising LLAVA-Instruction-150K (Liu et al., 2023) and a 12K instruction subset from A-OKVQA (Schwenk et al., 2022) as training set. And then we perform evaluation on MME, MMBench, POPE, and SEED-Bench benchmarks. At this stage, we primarily optimize the parameters introduced in HoRA, namely the per-layer parameter buckets $\mathbf{W}^o$ and the low-rank matrices $\mathbf{B}$. Following standard practices, $\mathbf{W}^o$ is initialized via Kaiming uniform distribution (He et al., 2016) based on the uncompressed feature dimension, and $\mathbf{B}$ is initialized with zeros. Furthermore, for trimodal setting, the number of epochs, batch size, and learning rate are set to be 3, 8, and 10^{-4}, respectively. In the bimodal setting, these hyper-parameters are adjusted to 2, 32, and 10^{-4}. For HoRA, we implement the index and signed hash function via xxhash provided by Google. Further details and code are provided in appendix. Since our method can be integrated with existing explicit low-rank parameterization approaches, we adopt MokA as the base method to implement HoRA. To ensure fair comparison, we set the rank of the low-rank matrices in HoRA to be the same as that of the baseline methods, i.e., $r = 4$. All experiments are perform on a computing cluster with 16 RTX 4090 GPUs, and all models are implemented with PyTorch.

For ease of reference, we summarize the key parameters involved in our HoRA and report their values used in practice, as shown in Tab. II.

C.4 Extension of AdaLoRA to the Multimodal Setting

Unlike HoRA, AdaLoRA (Zhang et al., 2023) leverages an SVD-based parameterization and dynamically re-allocates rank budget across layers according to importance scores. Essentially, AdaLoRA and HoRA operate on different granularity levels of capacity allocation, i.e., layer-wise versus modality-wise, making them complementary rather than mutually exclusive. To further demonstrate

Table II: Parameter setting in HoRA.

Stage	Setting	Parameter	Value
Pretraining	Trimodal	Training set	AudioCaps, Video-LLAVA
		Batch size	16
		Learning rate	10^{-4}
		Epoch	1
	Bimodal	Training set	Video-LLAVA
		Batch size	16
		Learning rate	10^{-4}
		Epoch	1
Fine-tuning	Trimodal	Training set	AVE/MUSIC-AVQA
		$\bar{M}^v$	0.6/0.6 for LLAMA2-7B, 0.8/0.6 for QWen2-7B
		$\bar{M}^t$	0.8/0.5 for LLAMA2-7B, 0.5/0.5 for QWen2-7B
		$\bar{M}^a$	0.7/0.6 for LLAMA2-7B, 0.5/0.5 for QWen2-7B
		Batch size	8
		Learning rate	10^{-4}
		Epoch	3
	Bimodal	Training set	LLAVA-Instruction-150K and A-OKVQA-12k
		$\bar{M}^v$	0.6 for LLAMA2-7B, 0.94 for QWen2-7B
		$\bar{M}^t$	0.5 for LLAMA2-7B, 0.94 for QWen2-7B
		Batch size	32
		Learning rate	10^{-4}
		Epoch	2

the effectiveness of HoRA, we extend AdaLoRA to the multimodal setting. Specifically, we adapt two key components of AdaLoRA to the multimodal setting, namely SVD-based adaptation and global budget scheduler. We utilize text-visual modalities for illustration.

SVD-based Adaptation for Multimodal Setting: To extend the AdaLoRA to the multimodal setting, we apply the SVD-based parameterization across all modalities, which can be formulated as:

$$\mathbf{h} = \widehat{\mathbf{W}}\mathbf{x} = \mathbf{W}\mathbf{x} + \mathbf{B}^a\mathbf{A}^a\mathbf{x}^a + \mathbf{B}^v\mathbf{A}^v\mathbf{x}^v \tag{123}$$

$$= \mathbf{W}\mathbf{x} + \mathbf{P}^a\mathbf{\Lambda}^a\mathbf{Q}^a\mathbf{x}^a + \mathbf{P}^v\mathbf{\Lambda}^v\mathbf{Q}^v\mathbf{x}^v \tag{124}$$

$$= \mathbf{W}\mathbf{x} + \sum\nolimits_{o\in\{a,v\}} \mathbf{P}^o\mathbf{\Lambda}^o\mathbf{Q}^o\mathbf{x}^o, \tag{125}$$

where $\mathbf{P}^o$, $\mathbf{\Lambda}^o$, $\mathbf{Q}^o$ are obtained by SVD-based parameterization proposed by AdaLoRA (Zhang et al., 2023). Since AdaLoRA decomposes the product of the low-rank matrices $\mathbf{A}$ and $\mathbf{B}$, it cannot be directly applied to partial-sharing multimodal LoRA frameworks, making it less flexible than HoRA.

Global Budget Scheduler for Multimodal Setting: AdaLoRA achieves layer-wise parameter allocation via importance-driven rank adaptation. Building upon this idea, we further extend the allocation mechanism to the modality dimension, enabling cross-modal capacity adaptation. By calculating importance scores globally, the model automatically balances priorities across modalities. We present the comparison between AdaLoRA and multimodal AdaLoRA in Table III. This architectural extension enables the model to adapt to multimodal tasks while dynamically allocating higher rank capacity to the most significant modalities.

D Additional Experimental Results

D.1 Multimodal Capacity Allocation Analysis via Bucket Size

In this section, we provide a supplementary experiment for the capacity analysis on AVE dataset with $M^v = 0.9$. As shown in Figure I, the results demonstrate that the HoRA maintains stable performance across different capacity allocation hyper-parameters.

Table III: Comparison between AdaLoRA and Multimodal AdaLoRA.

Feature	**AdaLoRA**	**Multimodal AdaLoRA**
Adapter Structure	1 SVD Triplet per Linear	3 Independent SVD Triplets per Linear
Routing Mechanism	Dense Activation	Hard Modality Routing (Token-level)
Initial Parameters	$r \times (d_{in} + d_{out} + 1)$	$3 \times r \times (d_{in} + d_{out} + 1)$
Budget Scope	Across layers and modules	Across layers, modules, and modalities

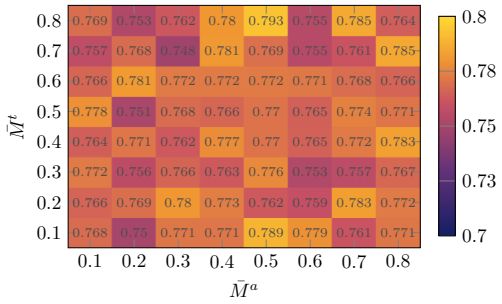


Figure I: Additional capacity analysis on AVE dataset for $M^v = 0.9$.

Additionally, we further supplement the analysis of capacity allocation via bucket size on the MMBench, POPE, SEED-Bench, and MME datasets. The results are shown in Figure II. From the experimental results, we arrive at consistent conclusion: HoRA not only improves performance but also enables controllable modality-wise capacity allocation.

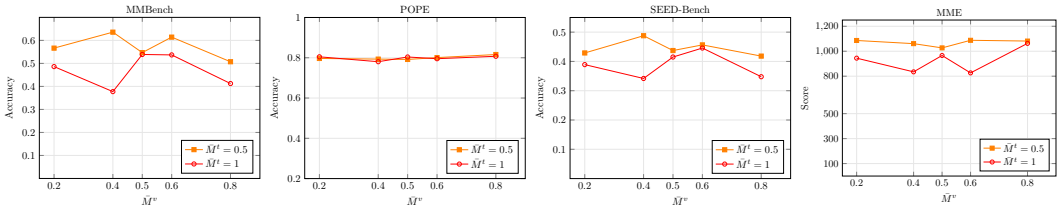


Figure II: Multimodal capacity allocation analysis on MMBench, POPE, SEED-Bench, and MME datasets.

D.2 The Robustness of HoRA

To remove the impact of randomness, we run the experiments with different random seeds. And the performance is reported in Tab. IV. We can find that HoRA exhibits controlled variation across different random seeds, with the observed fluctuations remaining within an acceptable range.

Table IV: Robustness analysis.

Seed	AVE	
	QWen2-7B	LLAMA2-7B
$\epsilon = 0$	79.55%	78.88%
$\epsilon = 1$	81.42%	77.99%
$\epsilon = 2$	79.90%	77.94%
$\epsilon = 3$	79.75%	78.76%
Avg. accuracy	80.16%$\pm$0.74%	78.39%$\pm$0.43%

D.3 Comparison with Unimodal LoRA Method

In Section 4.5, we compare HoRA with the multimodal variant of AdaLoRA. Since it is difficult to precisely match the effective parameter ratio ρ between HoRA and AdaLoRA (Zhang et al., 2023), and due to space limitation, we report in the main text the comparison under similar ρ values. Here, we provide more comprehensive results on AVE dataset by comparing the two methods across a range of ρ values for HoRA. As shown in Figure III, HoRA consistently achieves better performance while maintaining a comparable proportion of effective parameters.

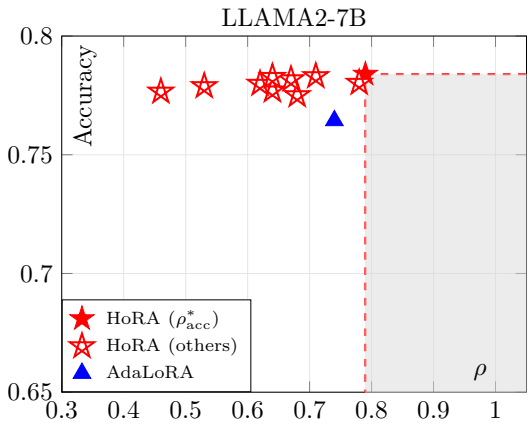
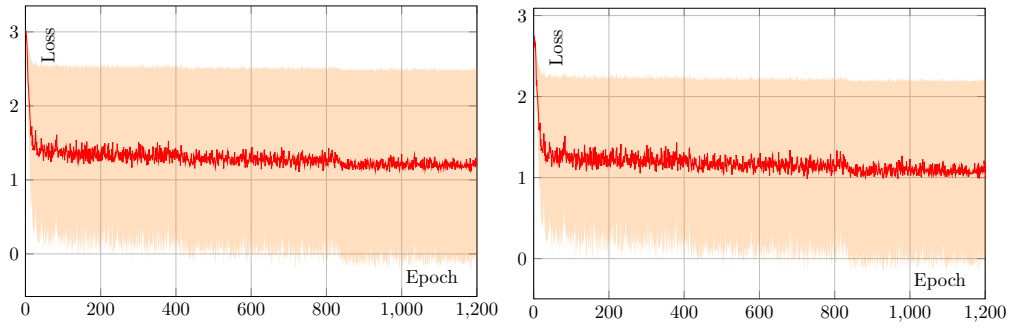


Figure III: Comparison with AdaLoRA.

D.4 Loss Visualization

To further validate the convergence of our method, we report the training loss during fine-tuning on the AVE dataset. The loss trends for architectures based on LLAMA2-7B and QWen2-7B are shown in Figure IV (a) and (b), respectively. The loss in Figure IV demonstrates that our method converges effectively.



(a). LLAMA2-7B. (b). QWen2-7B.

Figure IV: Loss visualization on AVE dataset.

D.5 Parameter Efficiency

Our method implicitly parameterizes the low-rank matrices, thereby achieving a form of LoRA parameter compression. Here, we present a comparison of different methods in terms of performance and LoRA parameter efficiency. Specifically, in Figure V, we report the comparison between different methods in terms of model performance and the actual number of parameters in multimodal LoRA under the trimodal and bimodal settings, respectively. For HoRA, the total number of LoRA parameters consists of those in the low-rank matrices $\mathbf{B}$ as well as those in the implicit parameter buckets $\mathbf{W}^o$. For multimodal LoRA baselines, the parameter scale of the LoRA components

varies due to differences in the underlying architectures. From Figure V, HoRA achieves the best performance while using fewer parameters than most competing approaches.

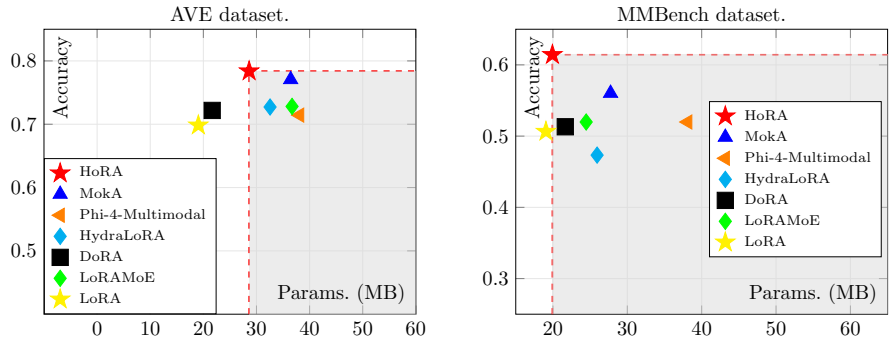


Figure V: Parameter efficiency analysis.

D.6 Impact of Rank

We further study the impact of rank r for HoRA. The results on AVE dataset are reported in Table V. From the results in the Table V, we observe that the performance of MokA improves to some extent as the rank increases. In contrast, HoRA is relatively insensitive to the rank, which is likely due to its ability to decouple model capacity from the rank parameter.

Table V: The impact of rank r in AVE dataset.

Method	$r = 4$	$r = 8$	$r = 16$
MokA	77.06%	77.31%	78.56%
HoRA	78.39%	78.16%	78.83%

D.7 Impact of Hashing Function

The hash function is a core component of HoRA. In this section, we investigate the impact of different hash functions on the performance of HoRA. Specifically, we select three commonly used hash functions, i.e., xxhash, SipHash (Aumasson & Bernstein, 2012), and MurmurHash3 (Senuma, 2025), for comparison. In Tab. VI, we present a comparison on the AVE dataset using LLAMA2-7B as the backbone. The results in Tab. VI show that different hash functions achieve comparable performance, while xxhash yields the best performance. Therefore, we adopt xxhash to implement hash functions $\{h^o, \xi^o\}$ for HoRA.

Table VI: Ablation study for hash function.

Hash function	Accuracy
SipHash	77.71%
MurmurHash3	78.13%
xxhash	**78.39%**