LoRA in LoRA: Towards Parameter-Efficient Architecture Expansion for Continual Visual Instruction Tuning

Chang Che[1], Ziqi Wang[1], Pengwan Yang[2*], Cheems Wang[3], Hui Ma[1], Zenglin Shi[1*]

[1]Hefei University of Technology,
[2]University of Amsterdam,
[3]Tsinghua University

Abstract

Continual Visual Instruction Tuning (CVIT) enables Multimodal Large Language Models (MLLMs) to incrementally learn new tasks over time. However, this process is challenged by catastrophic forgetting, where performance on previously learned tasks deteriorates as the model adapts to new ones. A common approach to mitigate forgetting is architecture expansion, which introduces task-specific modules to prevent interference. Yet, existing methods often expand entire layers for each task, leading to significant parameter overhead and poor scalability. To overcome these issues, we introduce LoRA in LoRA (LiLoRA), a highly efficient architecture expansion method tailored for CVIT in MLLMs. LiLoRA shares the LoRA matrix A across tasks to reduce redundancy, applies an additional low-rank decomposition to matrix B to minimize task-specific parameters, and incorporates a cosine-regularized stability loss to preserve consistency in shared representations over time. Extensive experiments on a diverse CVIT benchmark show that LiLoRA consistently achieves superior performance in sequential task learning while significantly improving parameter efficiency compared to existing approaches.

Introduction

Multimodal Large Language Models (MLLMs) (Bai et al. 2025; Liu et al. 2024a; Zhu et al. 2023) represent a significant advancement over traditional Large Language Models (LLMs) (Team et al. 2024; Touvron et al. 2023; Kaddour et al. 2023), enabling the handling of complex vision-language tasks such as visual question answering (VQA) (Chen et al. 2024b; Lee et al. 2024), image captioning (Awadalla et al. 2023; Liu et al. 2023a), and visual reasoning (Huang et al. 2023; Wang et al. 2024b). These models are typically trained using a multi-stage pipeline (Zhu et al. 2023; Liu et al. 2023a; Wang et al. 2024a), where pretraining on large-scale image-text pairs is followed by visual instruction tuning, aligning model outputs with human intent and improving performance on downstream multimodal tasks.

Visual instruction tuning is commonly performed in a static multi-task setting (Dai et al. 2023; Liu et al. 2023a), where all tasks are learned simultaneously using a unified instruction-based format. However, real-world applications increasingly demand that MLLMs continually acquire new capabilities without retraining from scratch. This has led to growing interest in Continual Visual Instruction Tuning (CVIT) (Wang et al. 2024c; Chen et al. 2024a; He et al. 2023), where models incrementally learn new vision-language tasks over time. A major obstacle in this setting is catastrophic forgetting (Zhai et al. 2023; He et al. 2023), in which newly acquired knowledge disrupts or erases information learned from previous tasks.

Most existing CVIT methods adopt static architectures (Chen et al. 2024a; Wang et al. 2024c; Zhao et al. 2025), where the model's structure remains fixed and task-specific routing is used to control parameter sharing. These approaches often incorporate Mixture-of-Experts (MoE) (Lepikhin et al. 2020; Fedus, Zoph, and Shazeer 2022) modules to reduce interference, but struggle to scale as the number of diverse or unrelated tasks grows. Fixed capacity leads to increased competition among tasks, reducing performance and limiting long-term learning. To address these limitations, we explore dynamic architecture expansion, a strategy widely used in general continual learning (CL) that introduces task-specific modules as new tasks arrive. While this method offers isolation between tasks, existing CVIT approaches that adopt it (He et al. 2023) often do so by expanding entire layers of the backbone per task, an approach that quickly becomes inefficient due to significant parameter redundancy and poor scalability in large-scale scenarios.

In this paper, we propose LoRA in LoRA (LiLoRA), a lightweight and scalable architecture expansion method tailored for CVIT in MLLMs. Instead of expanding full layers, LiLoRA builds on Low-Rank Adaptation (LoRA) (Hu et al. 2021), which expresses fine-tuned updates as a product of two low-rank matrices A and B. Through empirical analysis, we observe that the matrix A often converges to similar structures across different tasks. Based on this insight, LiLoRA shares matrix A across all tasks and restricts task-specific adaptation solely to matrix B, significantly reducing redundancy. To further improve parameter efficiency, LiLoRA applies an additional low-rank decomposition to the task-specific matrix B, factorizing it into a set of shared basis matrices and task-specific low-rank ma-

*Corresponding authors: yangpengwan2016@gmail.com, zenglin.shi@hfut.edu.cn

trices. This design allows each task to retain flexibility while keeping the overall parameter growth minimal. However, as learning progresses, the shared basis may drift, causing misalignment with previously learned task-specific representations. To counter this, we introduce a cosine-regularized basis stability loss, which penalizes updates to the shared basis based on cosine similarity with prior states, encouraging stability and knowledge retention.

Our contributions are summarized as follows:

- We propose LiLoRA, a parameter-efficient architecture expansion method for CVIT that shares LoRA components across tasks while preserving adaptability through task-specific low-rank decomposition.
- We introduce a cosine-regularized basis stability loss, which constrains changes to the shared basis and helps retain knowledge over time.
- We perform extensive experiments on the CVIT Benchmark, showing that LiLoRA achieves state-of-the-art performance while maintaining superior parameter efficiency compared to existing methods.

Related Work

CVIT for MLLMs

A wide range of approaches have been proposed to mitigate catastrophic forgetting in CVIT. CoIN (Chen et al. 2024a) applied the token-wise MoE (Liu et al. 2023b; Dou et al. 2023) to selectively activate expert weights for different tokens. Continual LLaVA (Cao et al. 2024) proposed a novel dual-embedding mechanism combined with selective LoRA modules to mitigate forgetting. LLaCA (Qiao et al. 2024) designed a gradient-guided exponential moving average strategy to adapt model weights. Fwd-Prompt (Zheng et al. 2024) leveraged prompt tuning with residual projection to mitigate gradient interference. MR-LoRA (Zhao et al. 2025) proposed a simple method with domain-specific low-rank tuning and pretrained model-based parameter selection. SMoLoRA (Wang et al. 2024c) introduced a separable mixture of low-rank adaptations to address dual forgetting. Although these approaches have demonstrated effectiveness, their static architectures face difficulties in coping with large-scale scenarios. In this paper, we focus on architecture expansion that dynamically adds new parameters to accommodate new tasks.

Architecture Expansion for CL

Architecture expansion is an effective strategy to mitigate catastrophic forgetting during CL. Existing methods (Yan, Xie, and He 2021; Kim, Ke, and Liu 2022; Douillard et al. 2022; Xie et al. 2024) typically extend the model with additional modules for each task and apply these task-specific weights to learn new tasks. DER (Yan, Xie, and He 2021) added task-specific tokens to achieve task-specialized embeddings through a new task-attention layer. DyTox (Douillard et al. 2022) introduced new learnable feature extractors with the arrival of new classes to incorporate additional feature dimensions. MORE (Kim, Ke, and Liu 2022) and BNCIL (Xie et al. 2024) adopted multi-head classification strategies specialized for different classification tasks. Although these methods can preserve previous knowledge when learning new tasks during CL, when new tasks differ in type from previously seen ones, which is often the case in CVIT, these class-incremental strategies fail to adapt effectively. In this paper, we focus on efficient architecture expansion tailored for CVIT.

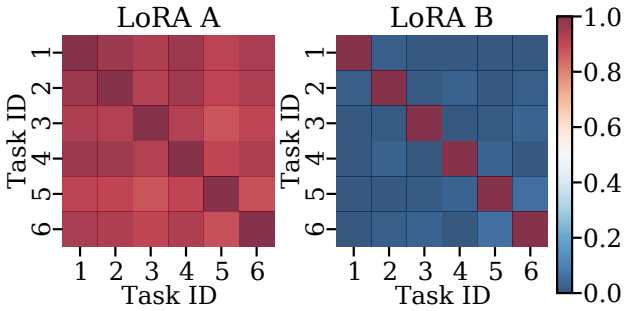

Figure 1: Heatmaps of CKA similarity for LoRA matrices in the linear layers learned by DirLoRA across different tasks. The matrices A exhibit high similarity across tasks, while matrices B show low similarity.

Methodology

In the CVIT setting, the model is presented with a stream of tasks $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_n\}$, where each task τ_t is associated with a dataset $\mathcal{D}_t = \{(X^{\text{ins}}, X^{\text{inputs}}, X^{\text{gt}})\}$, consisting of textual instructions, visual and textual inputs, and ground-truth response. A fundamental challenge in CVIT is catastrophic forgetting: performance on previously learned tasks degrades as the model updates its parameters to learn new ones. This occurs because shared parameters are overwritten, optimizing them for new tasks at the expense of older ones. To mitigate this, we explore the idea of LoRA-based parameter expansion for CVIT. LoRA has been widely adopted to enable parameter-efficient fine-tuning by introducing a pair of trainable low-rank matrices $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$, with $r \ll \min(d, k)$, into linear layers of the model. This method preserves the pretrained weight $W_0 \in \mathbb{R}^{d\times k}$ which remains frozen, while BA serves as the residual weights for adaptation:

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

where matrix B is initialized to zeros, while A is drawn from a standard Gaussian distribution.

A straightforward way to apply the idea of LoRA-based parameter expansion in the CVIT context is Direct LoRA Expansion (DirLoRA), which assigns an independent LoRA module to each task. For a task τ_i, the weight update can be expressed as:

$$\Delta W_i = B_i A_i, \tag{2}$$

where $B_i \in \mathbb{R}^{d\times r}$, $A_i \in \mathbb{R}^{r\times k}$ denote task-specific matrices. While DirLoRA effectively prevents task interference and mitigates forgetting, it introduces substantial parameter

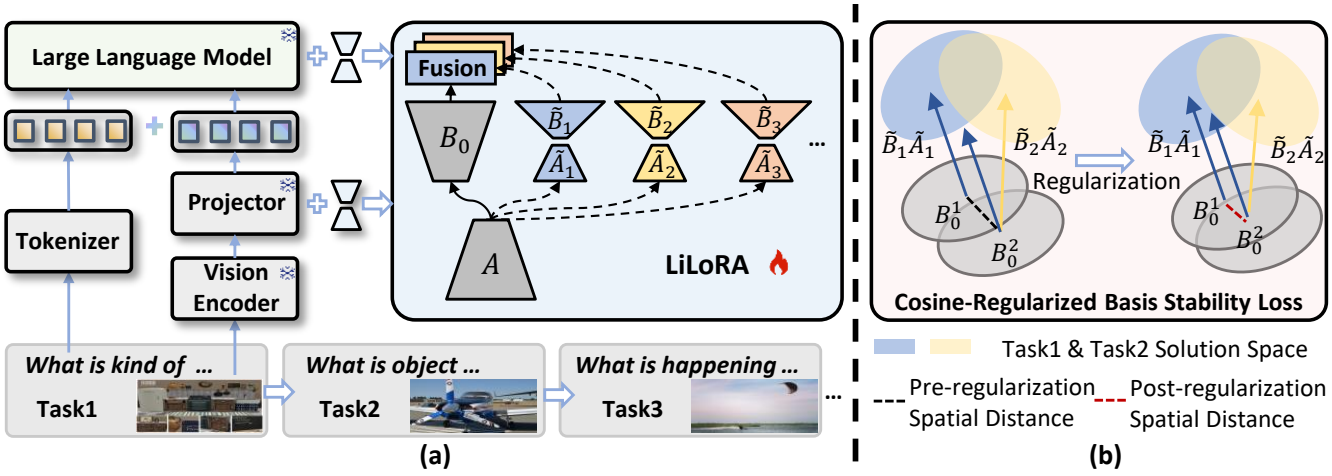

Figure 2: LiLoRA Framework. (a) LiLoRA is an efficient architecture expansion method tailored for CVIT, which freezes the pretrained model weights and injects lightweight trainable parameters into existing MLLMs. Specifically, LiLoRA is initialized with a pair of shared basis matrices and dynamically inserts task-specific low-rank matrices during CVIT. (b) Example of regularization constraining shared basis updates. The gray region represents the shared basis parameter space. The arrows indicate the task-specific residual weight shifts in LiLoRA. When task2 arrives, if the direction of its residual shift exhibits a large angular deviation (i.e., low cosine similarity) from task1, $\mathcal{L}_{reg}$ penalizes large updates to the shared basis, thereby preserving the parameter representations learned from task1.

overhead, scaling linearly with the number of tasks, which results in inefficient use of model capacity.

To overcome these limitations, we propose LiLoRA, a more efficient LoRA-based architecture expansion strategy. LiLoRA introduces several key innovations: a shared matrix A across all tasks, a low-rank decomposition of matrix B to further reduce task-specific parameters, and a cosine-regularized stability loss to maintain alignment of the shared components over time. Together, these components enable LiLoRA to preserve performance across tasks while significantly improving parameter efficiency.

LiLoRA

Task-Invariant Matrix A Sharing. To balance parameter efficiency with knowledge retention across sequential tasks, we explore a more effective expansion strategy by investigating the feature representations captured by low-rank matrices. Specifically, we conduct a centered kernel alignment (CKA) (Kornblith et al. 2019) similarity analysis on the LoRA matrices learned by DirLoRA. As shown in Fig. 1, the matrices A learned across tasks exhibit high similarity, suggesting redundant learning. Motivated by this observation, we propose a shared module design in LiLoRA by reusing the matrix A and limiting task-specific adaptation to matrix B. Specifically, we adopt a shared matrix $A \in \mathbb{R}^{r\times k}$ across tasks, while each task τ_i retains its own task-specific matrices $B_i \in \mathbb{R}^{d\times r}$. The weight update for task τ_i is then expressed as:

$$\Delta W_i = B_i A. \tag{3}$$

This design substantially reduces parameter growth while retaining task-level adaptation capacity.

Task-Specific Matrix B Decomposition. Although the matrix B exhibits lower cross-task similarity, it can be further decomposed into a shared basis and task-specific residuals to improve parameter efficiency. Instead of updating the entire matrix B for each task, we apply task-specific expansion only to its residual component. For each task τ_i, we introduce a pair of low-rank task-specific matrices $\tilde{B}_i \in \mathbb{R}^{d\times\tilde{r}}$ and $\tilde{A}_i \in \mathbb{R}^{\tilde{r}\times r}$, with $\tilde{r} < r$. Compared to the original matrix $B_i \in \mathbb{R}^{d\times r}$, the product matrix $\tilde{B}_i\tilde{A}_i$ contains significantly fewer parameters, achieving substantial parameter savings while maintaining expressiveness. Each task's weight is represented as a combination of the shared basis and task-specific matrices:

$$\Delta W_i = (B_0 + \tilde{B}_i\tilde{A}_i)A, \tag{4}$$

where the shared matrices B_0 and A provide a shared basis across tasks, while the task-specific matrices $\tilde{B}_i$ and $\tilde{A}_i$ specialize the knowledge for the particular task τ_i. Since the importance of shared versus task-specific knowledge may vary across tasks, we introduce a learnable fusion coefficient $\alpha \in (0, 1)$ to balance their contributions. The coefficient is initialized as:

$$\alpha \sim \text{Sigmoid}(\mathcal{N}(0, 1)), \tag{5}$$

where $\mathcal{N}(0, 1)$ denotes the standard Gaussian distribution, and Sigmoid$(\cdot)$ ensures $\alpha \in (0, 1)$. During training, α is learned via backpropagation, allowing the model to dynamically balance shared and task-specific knowledge. The updated task-specific weight becomes:

$$\Delta W_i = (\alpha B_0 + (1-\alpha)\tilde{B}_i\tilde{A}_i)A. \tag{6}$$

A higher α encourages reliance on shared knowledge, while a lower value promotes task-specific adaptation. This

adaptive fusion mechanism allows LiLoRA to flexibly tailor its representation to the specific characteristics of each task.

Decomposition Basis Regularization. The decomposition of the task-specific matrix B introduces an issue during the training of task τ_t: updating the shared basis matrix B_0 may interfere with the representations learned for previous tasks $\tau_1, \ldots, \tau_{t-1}$. Although each task retains fixed task-specific matrices $(\tilde{B}_i, \tilde{A}_i)$, modifications to B_0 may affect the composite weights ΔW_i for earlier tasks, potentially causing forgetting of past knowledge.

To mitigate this issue, we introduce a cosine-regularized basis stability loss, which constrains the magnitude of updates to B_0 based on the similarity between task-specific representations. When the new task-specific matrices exhibit low similarity to the previous tasks, the update to B_0 should be restricted to preserve the representations of prior tasks.

Specifically, upon the arrival of a new task τ_t, we compute the cosine similarity between its task-specific matrix product $\tilde{B}_t\tilde{A}_t$ and that of the immediately preceding task $\tilde{B}_{t-1}\tilde{A}_{t-1}$:

$$sim_t = \cos\left(\tilde{B}_t\tilde{A}_t, \tilde{B}_{t-1}\tilde{A}_{t-1}\right), \tag{7}$$

where the value sim_t then serves as an importance score, scaling the permissible extent of the B_0 update. The cosine-regularized basis stability loss is defined as:

$$\mathcal{L}_{reg} = (1 - sim_t) \cdot \left\|B_0^t - B_0^{t-1}\right\|_F^2, \tag{8}$$

where B_0^{t-1} is the value from the previous task, and B_0^t is the current value during task τ_t. As shown in Fig. 2 (b), this loss penalizes large deviations in B_0 when the new task's representation $(\tilde{B}_t\tilde{A}_t)$ is dissimilar to the previous one, thereby enhancing the stability of the shared basis B_0 within the CVIT framework. The overall training procedure for LiLoRA under CVIT is summarized in Algorithm 1.

Algorithm 1: Training of LiLoRA

Input: Dataset $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_t\}$, Pretrained model M, fusion coefficient α, regularization weight λ
Output: Shared basis matrices B_0 and A, Task-specific matrices $\{\tilde{B}_t, \tilde{A}_t\}_{t=1}^T$

```
Freeze pretrained model M
for each Dataset D_t do
    if t = 1 then
        Initialize B_0, A
    end if
    Initialize B̃_t, Ã_t
    for each batch in D_t do
        Compute weight ΔW_t using Eq. 6
        Compute autoregressive loss L_task on the batch
        if t > 1 then
            Compute regularization loss L_reg using Eq. 8
        else
            L_reg ← 0
        end if
        Minimize L_task + λL_reg
        Update B_0, A, B̃_t, Ã_t, α
    end for
end for
return B_0, A, {B̃_t, Ã_t}_{t=1}^T
```

Experiments

Datasets and Evaluation Metrics

Datasets. The datasets used in our experiments are from the CVIT Benchmark (Wang et al. 2024c), which includes six instruction datasets covering visual question answering (VQA) (Chen et al. 2024b; Lee et al. 2024), image classification (Huang et al. 2023; Wang et al. 2024b), and image captioning (Awadalla et al. 2023; Liu et al. 2023a) tasks. Specifically, the benchmark consists of ScienceQA (Lu et al. 2022), TextVQA (Singh et al. 2019), Flickr30k (Plummer et al. 2015), ImageNet (Deng et al. 2009), GQA (Hudson and Manning 2019), and VQAv2 (Goyal et al. 2017).

AP and MAP. To evaluate overall performance at each learning stage, we compute Average Performance (AP) and Mean Average Performance (MAP). Specifically, let $a_{k,j}$ denote the accuracy on the j-th task after training on the k-th task. These metrics are defined as:

$$\mathrm{AP}_k = \frac{1}{k}\sum_{j=1}^{k} a_{k,j}, \quad \mathrm{MAP}_k = \frac{1}{k}\sum_{i=1}^{k} \mathrm{AP}_i. \tag{9}$$

BWT. To quantify the degree of forgetting, we employ Backward Transfer (BWT) (Lin et al. 2022). It is defined as:

$$\mathrm{BWT}_k = \frac{1}{k-1}\sum_{j=1}^{k-1}(a_{k,j} - a_{j,j}). \tag{10}$$

MIF. We adopt the evaluation metric Mean Instruction Following (MIF) (Wang et al. 2024c) to evaluate the model's instruction-following consistency. MIF is defined as:

$$\mathrm{MIF}_k = \frac{1}{k}\sum_{j=1}^{k}\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{B}_j(o_i^j)\right), \tag{11}$$

where $\mathcal{B}()$ is a binary function that returns 1 if the model output o_i^j satisfies the instruction format of the j-th task, and 0 otherwise. n denotes the number of evaluation samples.

Baseline Methods

We compare our method with a comprehensive set of baselines to highlight its superior performance. **SeqLoRA** sequentially fine-tunes the model using a single shared LoRA module across tasks. **DoRA** (Liu et al. 2024b) and **C-LoRA**(Smith et al. 2023) serve as enhanced variants of LoRA designed for fine-tuning. We also include classical CL approaches such as **EWC** which constrains updates on the importance of parameters to previous tasks (Kirkpatrick et al. 2017), and **Replay** (Chaudhry et al. 2019), which stores or generates past samples to replay during new task training. In addition, we evaluate several methods tailored for CVIT, including **MoeLoRA** (Liu et al. 2024b), **EWC+TIR**, and **Eproj** (He et al. 2023). Notably, **Eproj**

	Method	Accuracy on Each Task						Overall Results			
		ScienceQA	TextVQA	Flickr30k	ImageNet	GQA	VQAv2	**AP ↑**	**MAP ↑**	**BWT ↑**	**MIF ↑**
Single-type	Zero-shot	52.72	2.95	52.64	22.10	2.73	0.65	22.30	-	-	17.84
	DirLoRA*	83.75	60.66	164.20	96.71	58.55	64.93	88.13	90.18	0.00	98.41
	SeqLoRA	55.31	50.22	33.89	22.73	50.52	64.61	46.21	57.41	-48.10	78.35
	DoRA	51.26	46.36	36.41	28.24	45.29	56.87	44.07	65.03	-31.12	78.59
	MoeLoRA	55.01	48.87	32.04	22.00	50.03	63.64	45.27	56.16	-48.05	79.97
	C-LoRA	57.25	38.70	56.50	25.27	42.89	54.06	45.78	57.04	-19.58	65.84
	Replay	75.61	47.58	31.97	35.84	48.51	58.67	49.70	69.78	-22.71	82.06
	EWC	57.04	50.02	32.96	22.85	50.16	64.54	46.26	56.19	-49.71	78.90
	EWC+TIR	72.22	44.78	34.54	25.98	46.86	58.73	47.19	67.21	-25.64	81.62
	Eproj	65.29	52.87	148.19	39.45	28.06	57.86	65.29	73.53	-14.02	89.81
	SMoLoRA	77.36	58.29	151.99	95.35	51.96	**65.71**	83.44	84.85	-3.23	97.79
	LiLoRA	**77.88**	**58.83**	**152.93**	**96.02**	**58.28**	65.33	**84.88**	**87.70**	**-3.13**	**98.24**
Five-type	Zero-shot	51.85	5.11	44.05	20.34	2.37	1.16	20.81	-	-	19.45
	DirLoRA*	83.85	60.51	164.66	96.71	57.93	64.90	88.09	91.49	0.00	98.31
	SeqLoRA	59.21	50.80	20.99	20.30	49.98	64.41	44.28	53.75	-48.73	79.47
	DoRA	52.03	47.37	27.97	26.18	46.05	57.33	42.82	56.24	-34.11	78.30
	MoeLoRA	58.09	53.30	22.82	22.61	51.80	65.15	45.63	54.88	-49.94	78.19
	C-LoRA	55.58	38.64	59.05	22.81	40.93	51.65	44.78	52.37	-18.83	70.01
	Replay	66.06	47.78	24.21	25.66	46.53	58.59	44.81	66.68	-26.88	80.38
	EWC	53.60	49.07	20.38	20.48	50.11	64.63	43.10	53.94	-52.47	78.18
	EWC+TIR	66.94	45.76	29.49	21.68	46.90	58.80	44.93	64.51	-26.38	80.56
	Eproj	63.45	53.18	151.41	20.63	45.30	57.32	65.22	72.10	-14.43	89.93
	SMoLoRA	**80.50**	58.30	146.63	94.28	52.42	**65.96**	83.02	85.05	-6.50	98.12
	LiLoRA	78.38	**59.14**	**155.26**	**95.82**	**56.27**	64.74	**84.94**	**87.43**	**-1.94**	**98.16**

Table 1: The evaluation results (%) for continual visual instruction tuning on the CVIT Benchmark (Wang et al. 2024c) after training on the final task. *: The performance of DirLoRA serve as an upper-bound for CVIT. LiLoRA consistently maintains high performance across both Single-type and Five-type settings.

is an architecture expansion method that extends projection layers based on task similarity. Furthermore, we compare with **SMoLoRA** (Wang et al. 2024c), a recent state-of-the-art method tailored for CVIT that introduces a separable mixture of low-rank adaptations to address dual forgetting. To provide performance bounds, we include **DirLoRA**, which assigns an independent LoRA module for each task as an upper-bound reference, and **Zero-shot**, which evaluates the pre-trained model without any fine-tuning as a lower-bound reference.

Implementation Details

We adopt the pre-trained first-stage **LLaVA-v1.5-7B** (Liu et al. 2023a) as the base model, without any instruction tuning. The LiLoRA adapters are inserted into the FeedForward Network (FFN) layers of the LLM, as well as into the projection layer between the LLM and the vision encoder. The rank of the shared matrices r in LiLoRA is initialized to 128, while task-specific $\tilde{r}$ is set to half of r. We employ the Adam optimizer with a learning rate of 2×10^{-5} and a batch size of 64. All tasks are trained for only one epoch.

Main Results

We evaluate LiLoRA on the CVIT Benchmark under two settings, *Single-type instruction* and *Five-type instruction*. After training on the last task, VQAv2, as shown in Table 1, LiLoRA consistently outperforms all baseline methods. Compared to the recent state-of-the-art method SMoLoRA under the *Single-type instruction* setting, our approach achieves improvements of +1.44% in AP, +2.85% in MAP, and +0.10% in BWT, respectively. In terms of instruction-following ability, our approach also brings a +0.45% improvement. In comparison to traditional CL approaches such as EWC and Replay, LiLoRA achieves significantly superior performance. Notably, when compared with the upper-bound DirLoRA, LiLoRA shows a competitive performance across all results. Consistently, under the *Five-type instruction* setting, LiLoRA achieves improvements of +1.92% in AP, +2.38% in MAP, +4.56% in BWT and +0.04% in MIF, respectively. These results show that LiLoRA maintains superior performance compared to other baselines, highlighting its robustness in handling more complex instruction data.

Ablation Study

In this section, we perform a series of ablation studies to examine the importance of each component in LiLoRA. Specifically, we evaluate the impact of: sharing the LoRA matrix A, decomposing the matrix B, and incorporating the regularization loss $\mathcal{L}_{reg}$ to stabilize the learned basis. To provide a comprehensive baseline, we also include a DirLoRA setting, where none of these components are applied. As shown in Table 2, solely sharing matrix A achieves

Component			Overall Performance				Efficiency	
Share A	**Decompose B**	L_{reg}	**AP** ↑	**MAP** ↑	**BWT** ↑	**MIF** ↑	**TP** ↓	**EP** ↓
–	–	–	88.13	90.18	0.00	98.41	2143.9	357.3
✓			85.38	88.40	-2.87	98.39	1,250.6	178.7
✓	✓		73.99	83.19	-16.14	93.60	985.1	104.6
✓	✓	✓	84.88	87.70	-3.13	98.24	985.1	104.6

Table 2: Ablation study on components of LiLoRA and efficiency analysis under the Single-type setting. The **TP** and **EP** represent the number of total and each task-specific expansion parameters, respectively, with values given in MB.

r	$\tilde{r}$	**AP** ↑	**MAP** ↑	**BWT** ↑	**MIF** ↑
128	64 ($r/2$)	84.88	87.70	-3.13	98.24
	32 ($r/4$)	83.87	86.70	-2.15	97.82
	16 ($r/8$)	83.83	86.42	-1.55	97.94
64	32 ($r/2$)	84.31	87.47	-2.28	97.93
	16 ($r/4$)	83.67	86.76	-2.51	97.63
	8 ($r/8$)	83.63	86.17	-1.67	97.97

Table 3: Further analysis on the rank of shared basis (r) and task-specific matrices ($\tilde{r}$).

	AP ↑	**MAP** ↑	**BWT** ↑	**MIF** ↑
$\alpha = 1$	46.21	57.41	-48.10	78.35
$\alpha = 0$	47.01	66.73	-13.45	75.53
$\alpha = 0.5$	81.84	87.17	-6.42	97.22
Learnable α (ours)	**84.88**	**87.70**	**-3.13**	**98.24**

Table 4: Further analysis on the hyperparameter α.

competitive performance, with 85.38% AP, 88.40% MAP, -2.87% BWT, and 98.39% MIF. However, when the matrix B is decomposed without applying $\mathcal{L}_{reg}$, the performance degrades significantly. This degradation can be attributed to the shared basis becoming less aligned with previously learned task-specific matrices over time. With the incorporation of $\mathcal{L}_{reg}$, LiLoRA substantially recovers performance, achieving 84.88% AP, 87.70% MAP, -3.13% BWT, and 98.24% MIF, close to the results of solely sharing matrix A. These results demonstrate the effectiveness of $\mathcal{L}_{\text{reg}}$ in preserving the stability of the shared basis.

For efficiency analysis, we focus on the number of expansion parameters, both in total and task-specific terms. Although DirLoRA achieves strong performance, it suffers from the highest parameter cost, with a total parameter count of 2,143.9MB and 357.3MB for each task. In contrast, LiLoRA with all components enabled (shared A, decomposed B, and $\mathcal{L}_{\text{reg}}$), reduces the number of total parameters to 985.1MB and the number of parameters for each task to 104.6MB, achieving a substantial reduction of 54% in total parameters and greater savings of 70% in each task overhead compared to DirLoRA. Furthermore, during inference, LiLoRA can be fully merged into the pretrained weights, introducing no extra computational overhead. These results demonstrate that our approach maintains competitive performance while significantly reducing the cost of architecture expansion in CVIT.

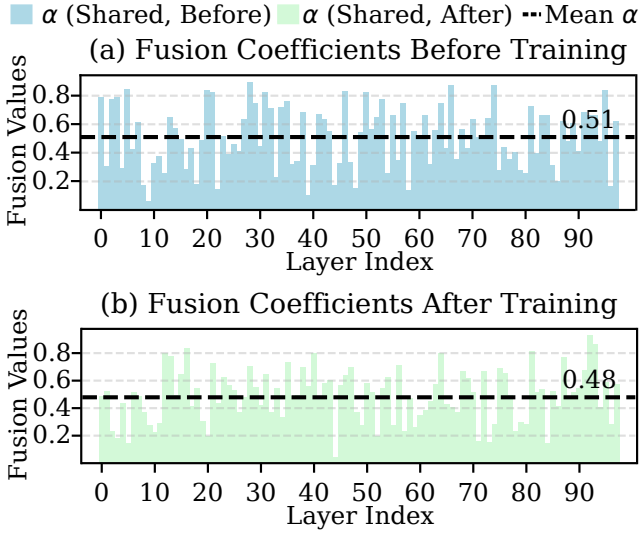

Figure 3: The value of fusion coefficients before and after training on the ScienceQA dataset. (a) shows the initial distribution of fusion values at each layer, while (b) presents the updated values after training.

Further Analysis

Effect of the Rank in LiLoRA. To further investigate the impact of hyperparameters in LiLoRA, we explore how the rank of the shared basis (r) and the task-specific matrices ($\tilde{r}$) influence the overall performance. As shown in Table 3, we conduct experiments under the *Single-type instruction* setting with two values of r (64 and 128), and for each, we evaluate $\tilde{r}$ across $\{r/2, r/4, r/8\}$. The results show that LiLoRA maintains consistently strong performance across a broad range of configurations, even when $\tilde{r}$ is reduced to as low as $r/8$. Although a higher value of r leads to modest gains in certain settings, the performance gap across different $\tilde{r}$ values remains minor, which highlights the robustness of LiLoRA to variations in rank design. These results demonstrate the flexibility of LiLoRA in selecting rank configurations without significant performance degradation, making it well-suited for real-world scenarios with computational or memory constraints.

Effect of the Hyperparameter α. To examine the impact of the hyperparameter α in LiLoRA, we conduct ablation experiments under four settings. First, we fix α to 1, activating only the shared component. Secondly, we set α to 0,

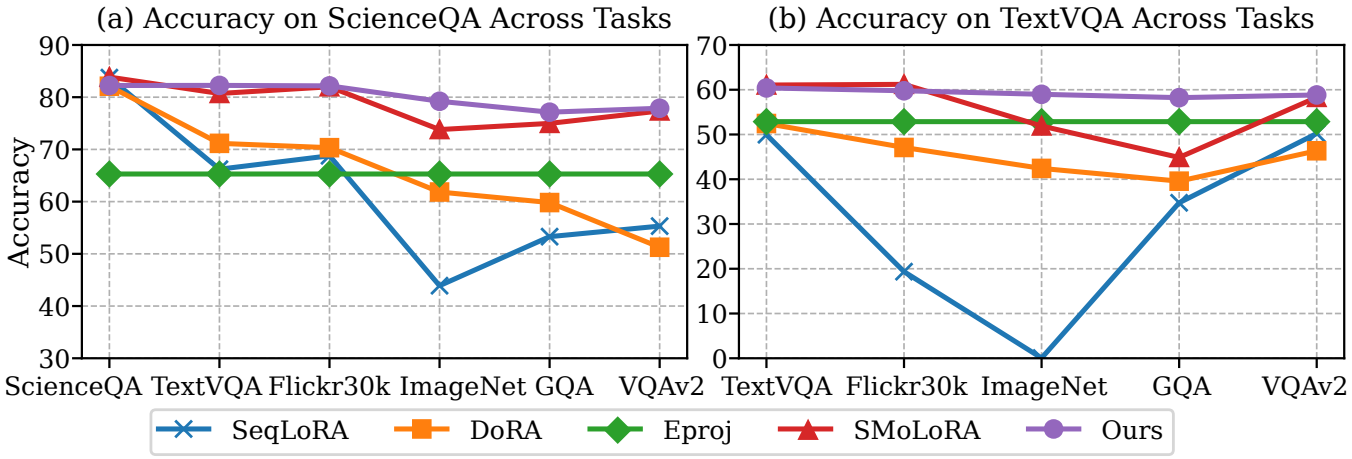

Figure 4: Accuracy (%) curves of ScienceQA and TextVQA during CVIT across sequential tasks. Our method consistently outperforms other methods.

enabling only the task-specific component. The third setting assigns a fixed value of 0.5 to α, balancing the contributions of both components equally. Lastly, in our proposed approach, α is a learnable parameter and dynamically optimized during training to adapt to task-specific requirements. The results in Table 4 show that both the $\alpha = 0$ and $\alpha = 1$ settings lead to significantly worse performance, indicating that relying solely on either shared or task-specific components is insufficient. The equal weighting ($\alpha = 0.5$) shows better results, demonstrating the benefit of combining shared and task-specific weights. Notably, the learnable α achieves the best performance across all metrics, validating the effectiveness of learnable fusion in LiLoRA.

Furthermore, Fig. 3 illustrates the distributions of fusion coefficients across different layers for the ScienceQA dataset, both before and after training. The dashed lines indicate the mean of α, which decreases after training, suggesting an increasing reliance on the task-specific components. Notably, the α values vary significantly across layers, indicating that the contributions of shared and task-specific components are not uniform throughout the model. These findings further highlight that LiLoRA can dynamically adjust its dependence on shared versus task-specific components when handling diverse tasks.

Stability Across Tasks. To comprehensively evaluate LiLoRA's performance during CVIT, we analyze the task accuracy trends by plotting the representative accuracy curves for two datasets: ScienceQA and TextVQA. These curves illustrate the model's performance after training on each sequential task, offering a clear depiction of each method's ability to retain previously learned knowledge while learning a new task. As shown in Fig. 4, the accuracy curve of LiLoRA consistently remains at the top across the entire training sequence for both datasets. In contrast, other methods exhibit a clear downward trend, indicating performance degradation caused by catastrophic forgetting. These findings further highlight the strong stability of our approach compared to competing methods.

Methods	AP ↑	MAP ↑	BWT ↑	MIF ↑
DirLoRA	67.55	74.14	0.00	93.81
SeqLoRA	45.41	53.39	-6.42	70.91
Eproj	58.87	61.21	-2.47	90.12
LiLoRA	**64.63**	**68.30**	**-0.94**	**92.94**

Table 5: The evaluation results (%) on **Qwen2-VL-2B**.

Cross-Model Generalizability. To validate the generalization ability of our approach across various MLLMs, we further evaluate the performance of LiLoRA on Qwen2-VL-2B (Wang et al. 2024a) under the *Single-type instruction* setting, comparing it with DirLoRA, SeqLoRA, and Eproj. As shown in Table 5, although Qwen2-VL exhibits a lower degree of forgetting compared to LLaVA, our method still achieves overall performance improvements over SeqLoRA and Eproj, approaching the performance of DirLoRA. These results highlight the robustness of LiLoRA in enhancing performance across diverse models.

Conclusion

In this paper, we present LiLoRA, a novel and efficient architecture expansion method for CVIT in MLLMs. Motivated by our observation that LoRA matrices A tend to converge to similar representations across tasks, we propose to share matrix A globally and restrict task-specific adaptation solely to matrix B, significantly reducing redundancy. To further minimize the parameter footprint, we decompose matrix B into a shared basis and smaller task-specific low-rank matrices. To stabilize the shared basis during CVIT, we introduce a cosine-regularized basis stability loss, which helps maintain alignment with previously learned components and mitigates representational drift. Extensive experiments on the CVIT benchmark demonstrate that LiLoRA not only achieves strong performance across sequential tasks but also offers substantial improvements in parameter efficiency over existing approaches.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (NO. 62472138, No. 62502145), the Anhui Provincial Natural Science Foundation (No. 2408085QF188) and the Fundamental Research Funds for the Central Universities (No. JZ2025HGTA0162, NO.JZ2025HGQA0134)

References

Awadalla, A.; Gao, I.; Gardner, J.; Hessel, J.; Hanafy, Y.; Zhu, W.; Marathe, K.; Bitton, Y.; Gadre, S.; Sagawa, S.; et al. 2023. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*.

Cao, M.; Liu, Y.; Liu, Y.; Wang, T.; Dong, J.; Ding, H.; Zhang, X.; Reid, I.; and Liang, X. 2024. Continual LLaVA: Continual Instruction Tuning in Large Vision-Language Models. *arXiv preprint arXiv:2411.02564*.

Chaudhry, A.; Rohrbach, M.; Elhoseiny, M.; Ajanthan, T.; Dokania, P. K.; Torr, P. H. S.; and Ranzato, M. 2019. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*.

Chen, C.; Zhu, J.; Luo, X.; Shen, H.; Gao, L.; and Song, J. 2024a. CoIN: A Benchmark of Continual Instruction tuNing for Multimodel Large Language Model. *arXiv preprint arXiv:2403.08350*.

Chen, D.; Chen, R.; Zhang, S.; Liu, Y.; Wang, Y.; Zhou, H.; Zhang, Q.; Wan, Y.; Zhou, P.; and Sun, L. 2024b. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. *arXiv preprint arXiv:2402.04788*.

Dai, W.; Li, J.; Li, D.; Tiong, A.; Zhao, J.; Wang, W.; Li, B.; Fung, P.; and Hoi, S. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. arxiv 2023. *arXiv preprint arXiv:2305.06500*, 2.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255.

Dou, S.; Zhou, E.; Liu, Y.; Gao, S.; Zhao, J.; Shen, W.; Zhou, Y.; Xi, Z.; Wang, X.; Fan, X.; et al. 2023. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7).

Douillard, A.; Rame, A.; Couairon, G.; and Cord, M. 2022. DyTox: Transformers for Continual Learning with Dynamic Token Expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9285–9295.

Fedus, W.; Zoph, B.; and Shazeer, N. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120): 1–39.

Goyal, Y.; Khot, T.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6904–6913.

He, J.; Guo, H.; Tang, M.; and Wang, J. 2023. Continual instruction tuning for large multimodal models. *arXiv preprint arXiv:2311.16206*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Huang, S.; Dong, L.; Wang, W.; Hao, Y.; Singhal, S.; Ma, S.; Lv, T.; Cui, L.; Mohammed, O. K.; Patra, B.; Liu, Q.; Aggarwal, K.; Chi, Z.; Bjorck, J.; Chaudhary, V.; Som, S.; Song, X.; and Wei, F. 2023. Language Is Not All You Need: Aligning Perception with Language Models. *arXiv preprint arXiv:2302.14045*.

Hudson, D. A.; and Manning, C. D. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6700–6709.

Kaddour, J.; Harris, J.; Mozes, M.; Bradley, H.; Raileanu, R.; and McHardy, R. 2023. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*.

Kim, G.; Ke, Z.; and Liu, B. 2022. A Multi-Head Model for Continual Learning via Out-of-Distribution Replay. *arXiv preprint arXiv:2208.09734*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Kornblith, S.; Norouzi, M.; Lee, H.; and Hinton, G. 2019. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*.

Lee, J.; Cha, S.; Lee, Y.; and Yang, C. 2024. Visual question answering instruction: Unlocking multimodal large language model to domain-specific visual multitasks. *arXiv preprint arXiv:2402.08360*.

Lepikhin, D.; Lee, H.; Xu, Y.; Chen, D.; Firat, O.; Huang, Y.; Krikun, M.; Shazeer, N.; and Chen, Z. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Lin, S.; Yang, L.; Fan, D.; and Zhang, J. 2022. Beyond not-forgetting: Continual learning with backward knowledge transfer. *Advances in Neural Information Processing Systems*, 35: 16165–16177.

Liu, H.; Li, C.; Li, Y.; and Lee, Y. J. 2024a. Improved Baselines with Visual Instruction Tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 26296–26306.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023a. Visual instruction tuning. In *NeurIPS*, 34892–34916.

Liu, Q.; Wu, X.; Zhao, X.; Zhu, Y.; Xu, D.; Tian, F.; and Zheng, Y. 2023b. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *arXiv preprint arXiv:2310.18339*.

Liu, S.-Y.; Wang, C.-Y.; Yin, H.; Molchanov, P.; Wang, Y.-C. F.; Cheng, K.-T.; and Chen, M.-H. 2024b. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*.

Lu, P.; Mishra, S.; Xia, T.; Qiu, L.; Chang, K.-W.; Zhu, S.-C.; Tafjord, O.; Clark, P.; and Kalyan, A. 2022. Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering. In *NeurIPS*, 2507–2521.

Plummer, B. A.; Wang, L.; Cervantes, C. M.; Caicedo, J. C.; Hockenmaier, J.; and Lazebnik, S. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2641–2649.

Qiao, J.; Zhang, Z.; Tan, X.; Qu, Y.; Ding, S.; and Xie, Y. 2024. Large Continual Instruction Assistant. *arXiv preprint arXiv:2410.10868*.

Singh, A.; Natarajan, V.; Shah, M.; Jiang, Y.; Chen, X.; Batra, D.; Parikh, D.; and Rohrbach, M. 2019. Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8317–8326.

Smith, J. S.; Hsu, Y.-C.; Zhang, L.; Hua, T.; Kira, Z.; Shen, Y.; and Jin, H. 2023. Continual diffusion: Continual customization of text-to-image diffusion with c-lora. *arXiv preprint arXiv:2304.06027*.

Team, G.; Georgiev, P.; Lei, V. I.; Burnell, R.; Bai, L.; Gulati, A.; Tanzer, G.; Vincent, D.; Pan, Z.; Wang, S.; et al. 2024. Gemini 1.5: Unlocking Multimodal Understanding Across Millions of Tokens of Context. *arXiv preprint arXiv:2403.05530*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, P.; Bai, S.; Tan, S.; Wang, S.; Fan, Z.; Bai, J.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Fan, Y.; Dang, K.; Du, M.; Ren, X.; Men, R.; Liu, D.; Zhou, C.; Zhou, J.; and Lin, J. 2024a. Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. *arXiv preprint arXiv:2408.15262*.

Wang, Y.; Chen, W.; Han, X.; Lin, X.; Zhao, H.; Liu, Y.; Zhai, B.; Yuan, J.; You, Q.; and Yang, H. 2024b. Exploring the reasoning abilities of multimodal large language models (mllms): A comprehensive survey on emerging trends in multimodal reasoning. *arXiv preprint arXiv:2401.06805*.

Wang, Z.; Che, C.; Wang, Q.; Li, Y.; Shi, Z.; and Wang, M. 2024c. SMoLoRA: Exploring and Defying Dual Catastrophic Forgetting in Continual Visual Instruction Tuning. *arXiv preprint arXiv:2411.13949*.

Xie, X.; Qiu, Y.; Lin, R.; Zheng, W.; and Wang, R. 2024. Class Incremental Learning with Task-Specific Batch Normalization and Out-of-Distribution Detection. *arXiv preprint arXiv:2411.00430*.

Yan, S.; Xie, J.; and He, X. 2021. DER: Dynamically Expandable Representation for Class Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3014–3023.

Zhai, Y.; Tong, S.; Li, X.; Cai, M.; Qu, Q.; Lee, Y. J.; and Ma, Y. 2023. Investigating the catastrophic forgetting in multimodal large language models. *arXiv preprint arXiv:2309.10313*.

Zhao, H.; Zhu, F.; Wang, R.; Meng, G.; and Zhang, Z. 2025. MLLM-CL: Continual Learning for Multimodal Large Language Models. *arXiv preprint arXiv:2506.05453*.

Zheng, J.; Ma, Q.; Liu, Z.; Wu, B.; and Feng, H. 2024. Beyond Anti-Forgetting: Multimodal Continual Instruction Tuning with Positive Forward Transfer. *arXiv preprint arXiv:2401.09181*.

Zhu, D.; Chen, J.; Shen, X.; Li, X.; and Elhoseiny, M. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.